

Commenced Publication in 1973

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

Lancaster University, UK

Takeo Kanade

Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler

University of Surrey, Guildford, UK

Jon M. Kleinberg

Cornell University, Ithaca, NY, USA

Friedemann Mattern

ETH Zurich, Switzerland

John C. Mitchell

Stanford University, CA, USA

Moni Naor

Weizmann Institute of Science, Rehovot, Israel

Oscar Nierstrasz

University of Bern, Switzerland

C. Pandu Rangan

Indian Institute of Technology, Madras, India

Bernhard Steffen

University of Dortmund, Germany

Madhu Sudan

Massachusetts Institute of Technology, MA, USA

Demetri Terzopoulos

University of California, Los Angeles, CA, USA

Doug Tygar

University of California, Berkeley, CA, USA

Moshe Y. Vardi

Rice University, Houston, TX, USA

Gerhard Weikum

Max-Planck Institute of Computer Science, Saarbruecken, Germany

Bin Xiao Laurence T. Yang
Jianhua Ma Christian Muller-Schloer
Yu Hua (Eds.)

Autonomic and Trusted Computing

4th International Conference, ATC 2007
Hong Kong, China, July 11-13, 2007
Proceedings

Volume Editors

Bin Xiao

The Hong Kong Polytechnic University, Hong Kong, China

E-mail: csbxiao@comp.polyu.edu.hk

Laurence T. Yang

St. Francis Xavier University, Antigonish, NS, B2G 2W5, Canada

E-mail: lyang@stfx.ca

Jianhua Ma

Hosei University, Tokyo 184-8584, Japan

E-mail: jianhuamah@gmail.com

Christian Muller-Schloer

University of Hannover, Institute of Systems Engineering, Germany

E-mail: cms@sra.uni-hannover.de

Yu Hua

Huazhong University of Science and Technology, Wuhan, 430074, China

E-mail: csyhua@gmail.com

Library of Congress Control Number: 2007930223

CR Subject Classification (1998): D.2, C.2, D.1.3, D.4, E.3, H.4, K.6

LNCS Sublibrary: SL 2 – Programming and Software Engineering

ISSN 0302-9743

ISBN-10 3-540-73546-1 Springer Berlin Heidelberg New York

ISBN-13 978-3-540-73546-5 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

Springer is a part of Springer Science+Business Media

springer.com

© Springer-Verlag Berlin Heidelberg 2007

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper SPIN: 12088591 06/3180 5 4 3 2 1 0

Preface

This volume contains the proceedings of ATC 2007, the 4th International Conference on Autonomic and Trusted Computing: Bringing Safe, Self-x and Organic Computing Systems into Reality. The conference was held in Hong Kong, during July 11-13, 2007. ATC 2007 is a successor of the 1st International Workshop on Trusted and Autonomic Ubiquitous and Embedded Systems (TAUES 2005, Japan, December 2005), the International Workshop on Trusted and Autonomic Computing Systems (TACS 2006, Austria, April 2006), and the 3rd International Conference on Autonomic and Trusted Computing (ATC 2006, Three Gorges, China, September 2006).

Computing systems including hardware, software, communication and networks are growing towards an ever-increasing scale and heterogeneity, becoming overly complex. Such complexity is getting even more critical with the ubiquitous permeation of embedded devices and other pervasive systems. To cope with the growing and ubiquitous complexity, autonomic computing (AC) focuses on self-manageable computing and communication systems that exhibit self-awareness, self-configuration, self-optimization, self-healing, self-protection and other self-x operations to the maximum extent even without human intervention or guidance. Organic computing (OC) additionally emphasizes natural-analogue concepts like self-organization and controlled emergence.

Any autonomic or organic system must be trustworthy to avoid the risk of losing control and to retain confidence that the system will not fail. Trust and/or distrust relationships on the Internet and in pervasive infrastructures are key factors to enable dynamic interaction and cooperation of various users, systems and services. Trusted/trustworthy computing (TC) aims at making computing and communication systems as well as services available, predictable, traceable, controllable, assessable, sustainable, dependable, persist-able, security/privacy protect-able, etc. A series of grand challenges exist to achieve practical self-manageable autonomic systems with truly trustworthy services. ATC 2007 addressed the most innovative research and development in these challenging areas and included all technical aspects related to autonomic/organic computing (AC/OC) and trusted computing (TC).

The ATC 2007 conference provided a forum for engineers and scientists in academia, industry and government to exchange ideas and experiences in developing AC/TC theory and models, architectures and systems, components and modules, communication and services, tools and interfaces, services and applications. There was a large number of paper submissions (223), representing 25 countries and regions, not only from Asia and the Pacific, but also from Europe, and North and South America. All submissions were reviewed by at least three Program Committee members or external reviewers. It was extremely difficult to select the presentations for the conference because there were so many

excellent and interesting submissions. In order to allocate as many papers as possible and keep the high quality of the conference, we finally decided to accept 55 papers for presentations, which reflected a 24.6% acceptance rate. We believe that all of these papers and topics not only provided novel ideas, new results, work in progress and state-of-the-art techniques in this field, but also stimulated the future research activities in the area of autonomic and trusted computing. In addition to the refereed papers the proceedings include Hartmut Schmeck's keynote addressing "Remarks on Self-Organization and Trust in Organic Computing Systems".

Organization of conferences with a large number of submissions requires a lot of hard work and dedication from many people. We would like to take this opportunity to thank numerous people whose work made this conference possible and ensured its high quality. We wish to thank the authors of submitted papers, as they contributed to the conference technical program. We wish to express our deepest gratitude to the Program (Vice) Chairs, Mazin Yousif, Omer F. Rana, Xiaobo Zhou, Wolfgang Reif, Dimitris Nikolopoulos and Silvia Giordano for their hard work and commitment to quality when helping with paper selection. We would also like to thank all Program Committee members and external reviewers for their excellent job in the paper review process, the Advisory Committee for their continuous advice, and Stephen S. Yau for organizing a panel on "Future Trends of Autonomic and Ubiquitous Computing." We are also indebted to the Publicity Chairs for advertising the conference, to Lin Chen and other people from the Local Organizing Committee for managing registration and other conference organization-related tasks, and to the Department of Computing, Hong Kong Polytechnic University for hosting the conference. We are also grateful to Tony Li Xu and Liu Yang for their hard work in managing the conference Web site and the conference management system.

July 2007

Bin Xiao
Laurence T. Yang
Jianhua Ma
Christian Muller-Schloer
Yu Hua

Organization

Executive Committee

General Chairs	Stephen S. Yau, Arizona State University, USA Christian Muller-Schloer, University of Hannover, Germany Laurence T. Yang, St. Francis Xavier University, Canada
Program Chairs	Bin Xiao, Hong Kong Polytechnic University, Hong Kong Mazin Yousif, Intel, USA Omer F. Rana, Cardiff University, UK
Program Vice Chairs	Xiaobo Zhou, University of Colorado at Colorado Springs, USA Wolfgang Reif, University of Augsburg, Germany Dimitris Nikolopoulos, Virginia Tech, USA Silvia Giordano, University of Applied Science, Switzerland
Steering Committee	Jianhua Ma (Chair), Hosei University, Japan Laurence T. Yang (Chair), St. Francis Xavier University, Canada Hai Jin, Huazhong University of Science and Technology, China Jeffrey J.P. Tsai, University of Illinois at Chicago, USA Theo Ungerer, University of Augsburg, Germany
International Advisory Committee	Makoto Amamiya, Kyushu University, Japan Jiannong Cao, Hong Kong Polytechnic University, Hong Kong Chin-Chen Chang, Feng Chia University, Taiwan Jingde Cheng, Saitama University, Japan Zhong Chen, Peking University, China Petre Dini, Cisco Systems, USA Tadashi Dohi, Hiroshima University, Japan Salim Hariri, University of Arizona, USA Jadwiga Indulska, University of Queensland, Australia Janusz Kacprzyk, Polish Academy of Sciences, Poland Sy-Yen Kuo, National Taiwan University, Taiwan David Ogle, IBM, USA

	Manish Parashar, Rutgers University, USA
	Franz J. Rammig, University of Paderborn, Germany
	Kouichi Sakurai, Kyushu University, Japan
	A Min Tjoa, Vienna University of Technology, Austria
	Kishor S. Trivedi, Duke University, USA
	Xinmei Wang, Xidian University, China
Publicity Chairs	Willy Susilo, University of Wollongong, Australia
	Xinwen Fu, Dakota State University, USA
	Xiaoyuan Gu, Technical University of Braunschweig, Germany
	Deqing Zou, Huazhong University of Science and Technology, China
	Noria Foukia, Otago University of Otago, New Zealand
International Liaison Chairs	Benno Overeinder, Vrije University, The Netherlands
	Jean-Marc Seigneur, University of Geneva, Switzerland
	Yuanshun Dai, Indiana University-Purdue University, USA
	Noriaki Yoshikai, Nihon University, Japan
Publication Chairs	Tony Li Xu, St. Francis Xavier University, Canada
	Xiaolin (Andy) Li, Oklahoma State University, USA
Award Chairs	Yi Mu, University of Wollongong, Australia
	Roy Sterritt, University of Ulster at Jordanstown, UK
	Chunming Rong, University of Stavanger, Norway
Panel Chairs	Huaglori Tianfield, Glasgow Caledonian University, UK
	Zhen Liu, IBM Research Center, USA
Financial Chair	Lin Chen, Hong Kong Polytechnic University, Hong Kong
Web Administration Chairs	Tony Li Xu, St. Francis Xavier University, Canada
	Liu Yang, St. Francis Xavier University, Canada
Local Arrangement Chairs	Zhijun Wang, Hong Kong Polytechnic University, Hong Kong
	Zili Shao, Hong Kong Polytechnic University, Hong Kong
	Kang Ying Allan Wong, Hong Kong Polytechnic University, Hong Kong

Program Committee

Emmanuelle Anceaume	IRISA, France
Taleb Bendiab	Liverpool John Moores University, UK
Raouf Boutaba	University of Waterloo, Canada
Manfred Broy	Technical University of Munich, Germany
Roy Campbell	UIUC, USA
Valentina Casola	University “Federico II” of Naples, Italy
Guanling Chen	University of Massachusetts Lowell, USA
Wei Chen	Nanjing University of Posts and Telecommunications, China
Rodrigo de Mello	University of Sao Paulo, Brazil
Falko Dressler	University of Erlangen, Germany
Hakan Duman	British Telecom, UK
Schahram Dustdar	Vienna University of Technology, Austria
Fabian E. Bustamante	Northwestern University, USA
Torsten Eymann	University of Bayreuth, Germany
Pascal Felber	University of Neuchatel, Switzerland
Sachin Garg	Avaya Labs, USA
Swapna Gokhale	University of Connecticut, USA
Panagiotis Hadjidakis	University of Ioannina, Greece
Dong-Guk Han	ETRI, Korea
Naohiro Hayashibara	Tokyo Denki University, Japan
Yanxiang He	Wuhan University, China
Xubin He	Tennessee Technological University, USA
Tian He	University of Minnesota, USA
M. S. P. Hernandez	Universidad Politecnica de Madrid, Spain
Valerie Issarny	INRIA, France
Brendan Jennings	The Waterford Institute of Technology, Ireland
Young-Sik Jeong	Wonkwang University, Korea
Hiroaki Kikuchi	Tokai University, Japan
Dan Jong Kim	Michigan State University, USA
Seungjoo Kim	Sungkyunkwan University, Korea
Bastian Koller	HLRS Stuttgart, Germany
Fabio Kon	University of Sao Paulo, Brazil
Michael Lagoudakis	Technical University of Crete, Greece
Fang-Yie Leu	Tunghai University, Taiwan
Kuan-Ching Li	Providence University, Taiwan
Zhitang Li	Huazhong University of Science and Technology, China
Maozhen Li	Brunel University, UK
Alex Zhaoyu Liu	University of North Carolina at Charlotte, USA
Zakaria Maamar	Zayed University, UAE
Mark Manulis	Ruhr-University of Bochum, Germany
Antonino Mazzeo	Second University of Naples, Italy
Geyong Min	University of Bradford, UK

Maurice Mulvenna	University of Ulster, UK
Daniel Olmedilla	I3C Hannover, Germany
Symeon Papavassiliou	National Technical University of Athens, Greece
Jeong-Hyun Park	ETRI, Korea
Andrea Passarella	IIT-CNR, Italy
Maria S Perez	UPM, Spain
Fernando Pedone	University of Lugano (USI), Switzerland
Christian Perez	IRISA, France
Xiao Qin	New Mexico Institute of Mining and Technology, USA
Khaled Ragab	University of Tokyo, Japan
Indrajit Ray	Colorado State University, USA
Joon S. Park	Syracuse University, USA
S. Masoud Sadjadi	Florida International University, USA
Hartmut Schmeck	University of Karlsruhe, Germany
Wolfgang Schreiner	Johannes Kepler University Linz, Austria
Xipeng Shen	College of William and Mary, USA
Weisong Shi	Wayne State University, USA
Chi-Sheng (Daniel) Shih	National Taiwan University, Taiwan
Michael Smirnov	Fraunhofer Institute FOKUS, Germany
Henrik Stormer	University of Fribourg, Switzerland
Yuqing Sun	Shandong University, China
Tsuyoshi Takagi	Future University - Hakodate, Japan
Kenichi Takahashi	Institute of Systems and Information Technologies, Japan
Juergen Teich	University of Erlangen-Nuremburg, Germany
Johnson Thomas	Oklahoma State University, USA
Eli Tilevich	Virginia Tech, USA
Jordi Torres	Barcelona Supercomputer Centre, Spain
Peter Triantafyllou	University of Patras, Greece
Can Turker	Swiss Federal Institute of Technology, Zurich, Switzerland
Johan van de Groenen-daal	Intel Corporation, USA
Javier Vazquez	UPC Barcelona, Spain
Daniel Veit	University of Karlsruhe, Germany
Umberto Villano	University of Sannio, Italy
Yong Wang	University of Northern British Columbia, Canada
Weichao Wang	University of Kansas, USA
Chuan-Kun Wu	Chinese Academy of Sciences, China
Liudong Xing	University of Massachusetts Dartmouth, USA
Naixue Xiong	JAIST, Japan
Qiuliang Xu	Shandong University, China
Lili Yang	Loughborough University, UK

George Yee	National Research Council, Canada
Meng Yu	Monmouth University, USA
Bo Yu	Fudan University, China
Huanguo Zhang	Wuhan University, China
Liqiang Zhang	Indiana University South Bend, USA
Ning Zhang	University of Manchester, UK
Albert Zomaya	University of Sydney, Australia

Additional Reviewers

Flora Amato	Sebastian Hudert	Juam Martinez
Chang Ho Jung	Alfredo Goldman	Werner Streitberger
Thomas Kuhn	Manos Kapritsos	Dwi Anoraganingrum
Tobias Limmer	Francisco Reverbel	David Villegas
Rodrigo Mello	Maozhen Li	Christoph Niemann
Byung-Gil Lee	Wolfgang Schreiner	Raphael Y. Camargo
William Fitzgerald	Angelos Lenis	Daniel Hamburg
Jimmy McGibney	Vassilis Chatzigiannakis	Ioannis Rexakis
Abdalkarim Awad	Johnson Thomas	Sakis Moralis
Luigi Coppolino	Lili Yang	Vassiliki Pouli
Kyuhyung Cho	Michael Glass	Li Ou
Bastian Koller	Martin Streubuehr	Rudolf Roth
Selim Kalayci	Pavan Balaji	Jung Yeon Hwang
Christian Prehofer	Thilo Streichert	Diana Goehringer
Seog Chung Seo	Fang-Yie Leu	Dmitrij Kissler
Alexander Amelin	Nicolas Schiper	Holger Ruckdeschel
Steven Davy	Theoni Pitoura	Yolanda Becerra
Michael Fahrmaier	Nikos Ntarmos	Andreas Loupasakis
Seungjoo Kim	Mauro Iacono	Vaide Zuikeviciute
Isabel Dietrich	Andrew Allen	Dany Guevara
Daniel Schall	Ioannis Aekaterinidis	Massimiliano Rak
Uwe Zdun	Riky Subrata	Johan van de
Stefan Koenig	Tim Schmidt	Groenendaal
Raimund Matros	Yali Wu	
Michael Meisinger	Martin Feilkas	

Table of Contents

Keynote Speech

An Intelligent Home System as a Development and Test Platform for Ubiquitous Computing	1
<i>Keith C.C. Chan</i>	

Remarks on Self-organization and Trust in Organic Computing Systems	2
<i>Hartmut Schmeck</i>	

Cryptography and Signatures

ZigBee Security Using Identity-Based Cryptography	3
<i>Son Thanh Nguyen and Chunming Rong</i>	

Efficient Identity-Based Signcryption Scheme for Multiple Receivers	13
<i>Yong Yu, Bo Yang, Xinyi Huang, and Mingwu Zhang</i>	

Identity-Based Proxy Signature from Pairings	22
<i>Wei Wu, Yi Mu, Willy Susilo, Jennifer Seberry, and Xinyi Huang</i>	

Cryptanalysis of BGW Broadcast Encryption Schemes for DVD Content Protection	32
<i>Qianhong Wu, Willy Susilo, Yi Mu, and Bo Qin</i>	

A Digital Signature Mechanism and Authentication Scheme for Group Communication in Grid	42
<i>Yunfa Li, Hai Jin, Deqing Zou, Jieyun Chen, and Zongfen Han</i>	

Cryptanalysis of Server-Aided RSA Key Generation Protocols at MADNES 2005	52
<i>Fanyu Kong, Jia Yu, Baodong Qin, and Daxing Li</i>	

Autonomic Computing and Services

Service-Context Knowledge-Based Solution for Autonomic Adaptation	61
<i>Marcel Cremene and Michel Riveill</i>	

Middleware Based Context Management for the Component-Based Pervasive Computing	71
<i>Di Zheng, Jun Wang, Yan Jia, Wei-Hong Han, and Peng Zou</i>	

Building Autonomic and Secure Service Oriented Architectures with MAWeS	82
<i>Valentina Casola, Emilio Pasquale Mancini, Nicola Mazzocca, Massimiliano Rak, and Umberto Villano</i>	
Biology as Inspiration Towards a Novel Service Life-Cycle	94
<i>David Linner, Heiko Pfeffer, Ilja Radusch, and Stephan Steglich</i>	
Design of Service-Based Systems with Adaptive Tradeoff Between Security and Service Delay	103
<i>Stephen S. Yau, Min Yan, and Dazhi Huang</i>	

Secure and Trusted Computing

Provably Secure Identity-Based Threshold Unsigncryption Scheme	114
<i>Bo Yang, Yong Yu, Fagen Li, and Ying Sun</i>	
Final Fantasy – Securing On-Line Gaming with Trusted Computing	123
<i>Shane Balfe and Anish Mohammed</i>	
An Efficient and Secure Rights Sharing Method for DRM System Against Replay Attack.....	135
<i>Donghyun Choi, Yunho Lee, Hogab Kang, Seungjoo Kim, and Dongho Won</i>	
Establishing Trust Between Mail Servers to Improve Spam Filtering	146
<i>Jimmy McGibney and Dmitri Botvich</i>	

Autonomic Models and Architectures

An Architecture for Self-healing Autonomous Object Groups	156
<i>Hein Meling</i>	
A Generic and Modular System Architecture for Trustworthy, Autonomous Applications	169
<i>George Brancovici and Christian Müller-Schloer</i>	
Cooperative Component Testing Architecture in Collaborating Network Environment	179
<i>Gaeil An and Joon S. Park</i>	
An Approach to a Trustworthy System Architecture Using Virtualization	191
<i>Frederic Stumpf, Michael Benz, Martin Hermanowski, and Claudia Eckert</i>	

Trusted Models and Systems

CuboidTrust: A Global Reputation-Based Trust Model in Peer-to-Peer Networks	203
<i>Ruichuan Chen, Xuan Zhao, Liyong Tang, Jianbin Hu, and Zhong Chen</i>	

A Trust Evolution Model for P2P Networks	216
<i>Yuan Wang, Ye Tao, Ping Yu, Feng Xu, and Jian Lü</i>	
An Adaptive Trust Control Model for a Trustworthy Component Software Platform	226
<i>Zheng Yan and Christian Prehofer</i>	
Towards Trustworthy Resource Selection: A Fuzzy Reputation Aggregation Approach	239
<i>Chunmei Gui, Quanyuan Wu, and Huaimin Wang</i>	

Intrusion Detection

An Adaptive Spreading Activation Approach to Combating the Front-Peer Attack in Trust and Reputation System	249
<i>Yufeng Wang, Yoshiaki Hori, and Kouichi Sakurai</i>	
Research on Cost-Sensitive Learning in One-Class Anomaly Detection Algorithms	259
<i>Jun Luo, Li Ding, Zhisong Pan, Guiqiang Ni, and Guyu Hu</i>	
Improved and Trustworthy Detection Scheme with Low Complexity in VBLAST System	269
<i>So-Young Yeo, Myung-Sun Baek, and Hyoung-Kyu Song</i>	
Stepping-Stone Detection Via Request-Response Traffic Analysis	276
<i>Shou-Husan Stephen Huang, Robert Lychev, and Jianhua Yang</i>	
SPA Countermeasure Based on Unsigned Left-to-Right Recodings	286
<i>Sung-Kyoung Kim, Dong-Guk Han, Ho Won Kim, Kyo IL Chung, and Jongin Lim</i>	

Access Control

A New One-Way Isolation File-Access Method at the Granularity of a Disk-Block	296
<i>Wenyuan Kuang, Yaoxue Zhang, Li Wei, Nan Xia, Guangbin Xu, and Yuezhi Zhou</i>	
Novel Remote User Authentication Scheme Using Bilinear Pairings	306
<i>Chen Yang, Wenping Ma, and Xinmei Wang</i>	
On the Homonymous Role in Role-Based Discretionary Access Control	313
<i>Kai Ouyang, Xiaowen Chu, Yixin Jiang, Hsiao-Hwa Chen, and Jiangchuan Liu</i>	
Ontology Based Hybrid Access Control for Automatic Interoperation . . .	323
<i>Yuqing Sun, Peng Pan, Ho-fung Leung, and Bin Shi</i>	

Recoverable Tamper Proofing Technique for Image Authentication Using Irregular Sampling Coding.....	333
<i>Kuo Lung Hung and Chin-Chen Chang</i>	

Trusted Computing and Communications

A Decomposition Strategy Based Trusted Computing Method for Cooperative Control Problem Faced with Communication Constraints	344
<i>Shieh-Shing Lin</i>	
Formal Analysis of Secure Bootstrap in Trusted Computing	352
<i>Shuyi Chen, Yingyou Wen, and Hong Zhao</i>	
Calculating Trust Using Aggregation Rules in Social Networks	361
<i>Sanguk Noh</i>	
Enhancing Grid Security Using Trusted Virtualization	372
<i>Hans Löhr, HariGovind V. Ramasamy, Ahmad-Reza Sadeghi, Stefan Schulz, Matthias Schunter, and Christian Stübke</i>	
A Wearable System for Outdoor Running Workout State Recognition and Course Provision	385
<i>Katsuhiro Takata, Masataka Tanaka, Jianhua Ma, Runhe Huang, Bernady O. Apduhan, and Norio Shiratori</i>	

Key Management

Malicious Participants in Group Key Exchange: Key Control and Contributiveness in the Shadow of Trust	395
<i>Emmanuel Bresson and Mark Manulis</i>	
Efficient Implementation of the Keyed-Hash Message Authentication Code Based on SHA-1 Algorithm for Mobile Trusted Computing	410
<i>Mooseop Kim, Youngse Kim, Jaechol Ryou, and Sungik Jun</i>	
A Secure DRM Framework for User's Domain and Key Management ...	420
<i>Jinheung Lee, Sanggon Lee, and Sanguk Shin</i>	
A Secret-Key Exponential Key Agreement Protocol with Smart Cards	430
<i>Eun-Jun Yoon and Kee-Young Yoo</i>	
Key Establishment Scheme for Sensor Networks with Low Communication Cost	441
<i>Yong Ho Kim, Hwaseong Lee, Jong Hyuk Park, Laurence T. Yang, and Dong Hoon Lee</i>	

Worm Detection and Data Security

A Worm Containment Model Based on Neighbor-Alarm	449
<i>Jianming Fu, Binglan Chen, and Huanguo Zhang</i>	
A Distributed Self-healing Data Store	458
<i>Wolfgang Trumler, Jörg Ehrig, Andreas Pietzowski, Benjamin Satzger, and Theo Ungerer</i>	
Malicious Codes Detection Based on Ensemble Learning	468
<i>Boyun Zhang, Jianping Yin, Jingbo Hao, Dingxing Zhang, and Shulin Wang</i>	
Generating Simplified Regular Expression Signatures for Polymorphic Worms	478
<i>Yong Tang, Xicheng Lu, and Bin Xiao</i>	

Secured Services and Applications

AAA for Spontaneous Roaming Agreements in Heterogeneous Wireless Networks	489
<i>Zhi (Judy) Fu, Minho Shin, John C. Strassner, Nitin Jain, Vishnu Ram, and William A. Arbaugh</i>	
A Prediction-Based Fair Replication Algorithm in Structured P2P Systems	499
<i>Xianshu Zhu, Dafang Zhang, Wenjia Li, and Kun Huang</i>	
TransCom: A Virtual Disk Based Self-management System	509
<i>Li Wei, Yaoxue Zhang, and Yuezhi Zhou</i>	
Defending Against Jamming Attacks in Wireless Local Area Networks	519
<i>Wei Chen, Danwei Chen, Guozi Sun, and Yingzhou Zhang</i>	

Fault-Tolerant Systems

Schedulability Analysis of the Fault-Tolerant Hard Real-Time Tasks with Limited Priority Levels	529
<i>Jun Li, Fumin Yang, Gang Tu, Wanhua Cao, and Yansheng Lu</i>	
A Property-Based Technique for Tolerating Faults in Bloom Filters for Deep Packet Inspection	539
<i>Yoon-Hwa Choi and Myeong-Hyeon Lee</i>	
A Fuzzy Logic Approach for Secure and Fault Tolerant Grid Job Scheduling	549
<i>Congfeng Jiang, Cheng Wang, Xiaohu Liu, and Yinghui Zhao</i>	

An Enhanced DGIDE Platform for Intrusion Detection	559
<i>Fang-Yie Leu, Fuu-Cheng Jiang, Ming-Chang Li, and Jia-Chun Lin</i>	
Author Index	569

An Intelligent Home System as a Development and Test Platform for Ubiquitous Computing

Keith C.C. Chan

Department of Computing
The Hong Kong Polytechnic University
Hung Hom, Kowloon
Hong Kong
cskcchan@comp.polyu.edu.hk

Abstract. Ubiquitous Computing is concerned with the thorough integration of information processing into everyday objects and activities. As such, someone engaging in ubiquitous computing should be able to enjoy the many benefits it is supposed to bring about at home. Since 2000, a team at the Department of Computing of The Hong Kong Polytechnic University has been developing an Ubiquitous Intelligent Home (UIH) that can demonstrate how a user can interact with “computers” at home in such a way that the user does not have to be aware that he or she is doing so. The UIH consists of four interconnecting networks: an appliance network, a furniture network, a telehealth network, and a security network. Each of these networks is made up of both hardware and software that are designed and developed to try to achieve the kind of ideal ubiquitous computing environment – one that is made up of small, inexpensive, robust networked processing devices, distributed at all scales throughout everyday life. The UIH project has so far been “pervasive” not only in terms of its potential applications but also in terms of the researchers involved. Throughout its development, we have involved researchers in almost all areas of computing including those working on wireless sensor networking, sensor data management, data stream processing, RFID, embedded systems design, distributed processing, artificial intelligence, agent theory, speech recognition, image and video analysis, signal processing, data mining, computational intelligence, Chinese computing, data mining and machine learning, text mining, information retrieval, gesture recognition, biometrics, text-to-speech processing, software engineering, etc. In this talk, we will give the details of the UIH.

Remarks on Self-organization and Trust in Organic Computing Systems

Hartmut Schmeck

Institute AIFB, Karlsruhe Institute of Technology, Germany
Hartmut.Schmeck@kit.edu

Abstract. The vision of Organic Computing postulates the advent of multitudes of services provided by collections of intelligent devices by means of self-organized cooperation. Due to their large numbers and their versatile interactions in potentially unlimited networks, it will be unfeasible to explicitly control the behavior of these (partially mobile) devices and their services. Therefore, they will have to respond autonomously in an intelligent way to changing parameters in their environment in order to guarantee appropriate degrees of behavioral robustness and flexibility. Because of these life-like properties, they are called Organic Computing systems. Apparently, the behavior of adaptive, self-organized systems might be hard to predict. At the same time, these systems will have to be trustworthy to be accepted by human users, otherwise, their potential benefits would not be exploited. A necessary prerequisite for establishing and maintaining trust will be the possibility to influence the behavior of Organic Computing systems in a controlled way whenever the system is moving into behavioral regions that are viewed to be unacceptable by human users or by their current execution environment. Hence, an important facet of Organic Computing is the presence of controlled self-organization, enabled by appropriately designed observer/controller mechanisms and methods of data analysis. The talk will elaborate on the state of the art in the area of Organic Computing and, in particular, will focus on possibilities and problems for the engineering of trustworthy organic systems.

ZigBee Security Using Identity-Based Cryptography

Son Thanh Nguyen and Chunming Rong

Department of Electrical Engineering and Computer Science,
University of Stavanger, Norway
{son.t.nguyen, chunming.rong}@uis.no

Abstract. ZigBee is a specification defining a set of protocols and architecture for monitoring and control networks. With the advantages of high availability, low cost and low power consumption, ZigBee is ideal for both residential and industrial settings. This paper proposes an application of identity-based cryptography for security in ZigBee networks. The proposal enhances the security features in ZigBee networks while reduces the number of required keys.

1 Introduction

ZigBee is a wireless specification defining a suite of protocols and architecture for monitoring, sensing and control applications. Based on IEEE 802.15.4 wireless standard [11], which characterizes physical and MAC layers, ZigBee defines the additional network, security and application specifications.

Designed for low power consumption, low data rate applications, ZigBee is a mesh, self-organizing network of sensing devices to report the environmental conditions in a reliable, accurate and timely manner. With the advantages of high available, inexpensive, and low resource requirement mesh networks, ZigBee is ideal for industrial and commercial applications such as monitoring and control of manufacturing processes, automatic lighting and heating systems, smoke and fire detection, and medical sensing and monitoring, or residential applications like home automation and security.

As shown in ZigBee specification [23], security is inherent in ZigBee protocol architecture. Current ZigBee security uses Advanced Encryption Standard, AES, for encryption. In addition, ZigBee security specification also details other mechanisms to enhance security in ZigBee networks.

Since ZigBee security utilizes symmetric security scheme, it does not take full advantage of public-key cryptography strong points such as digital signature and session key distribution without the intervention of the key distribution center. Due to the strict security requirements in some critical monitoring and control applications, the capability for the communicating ZigBee devices to authenticate is required. Some recent works have proposed to use public-key cryptography for ZigBee security ([2], [3]).

Identity-based cryptography [17], based on public-key cryptography [7], has advantages over its counterpart by eliminating the public key directory. It uses the

identity of each entity as a public key and thus, can simplify the public key distribution process.

In this paper, we propose a solution to use identity-based cryptography for ZigBee network security. The proposal helps to reduce overhead, resource requirements, and complexity of the system, while still keeping the network secured. We also make a comparison between our proposal and current ZigBee security as well as the other proposal of using public-key cryptography for security in ZigBee network.

The rest of this paper is organized as follow. Section 2 summaries security issues in ZigBee network. Section 3 briefly introduces identity-based cryptography. Section 4 proposes an idea of using identity-based cryptography for securing ZigBee applications. Section 5 is for critical analysis. Section 6 is for conclusions.

2 Security in ZigBee Network

ZigBee utilizes a symmetric cryptography scheme with 128-bit AES algorithm, a strong and secure encryption method approved by National Institute for Standards and Technology (NIST, [14]) to provide data transmission confidentiality. Beside encryption, ZigBee security also implements freshness counter to prevent replay attack and message integrity check to prevent message modification. ZigBee also supports authentication capability.

Encryption could be done at network or device level. Network encryption uses a network key, shared among all devices in the network. This method protects against outsider attack with little resource requirement at devices. For better protection, device encryption is used with unique link keys shared between each pair of devices. This method protects against both insider and outsider attacks but requires more resources, especially at ZigBee network trust center.

There are three key types in ZigBee. A master key is a long-term security key between two devices. It could be set up manually, by factory pre-installed, or by sending to corresponding devices via wireless interfaces. A master key is used for delivery of network and link encryption keys. The second key type is link key, which provides security on a specific link between two devices. Finally, a network key provides security in the network. Both link and network keys can be factory pre-installed or set up using key management with the assistance of master keys. Link and network keys can also be updated periodically.

In a ZigBee network, there exists a trust center, which is responsible for membership management and encryption key distribution. The trust center can operate in two modes: residential and commercial mode.

Fig.1 represents the number of keys in residential and commercial modes. In this figure, node A is the ZigBee network coordinator, also the network trust center. Node B, C and D are the network routers, while node E, F, G, and H are the network end devices. In residential mode, only one network encryption key K_N is used. All devices using this shared key to encrypt and decrypt information. This approach requires little resource from the trust center. However, it does not prevent from insider attack as mentioned above. The commercial mode, meanwhile, offers higher security and is suitable for critical applications. This approach requires more keys and consumes more bandwidth. K_N is the shared network key, which is similar to that of the

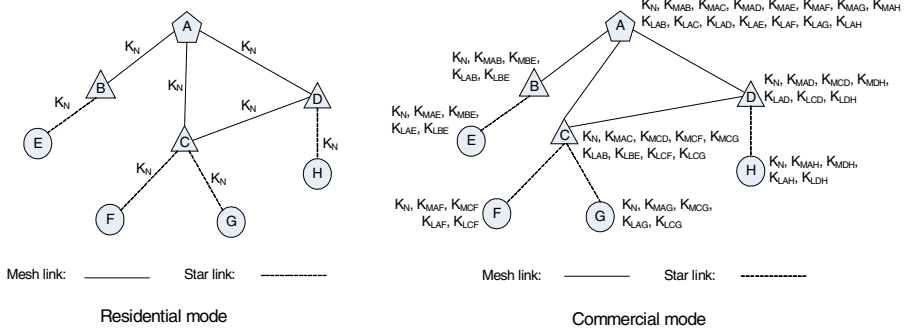


Fig. 1. Keys in residential and commercial mode

residential mode, K_{MXY} is the master key shared between nodes X and Y, and K_{LXY} is the link encryption key between node X and Y.

Although current ZigBee security specification offers strong method to protect information exchange between devices by using AES, it still exposes some restrictions.

The first restriction is the number of keys used in the network. The residential mode is resource-optimized since it uses one key. However, it is not very secure since it cannot prevent from insider attack. The commercial mode is secure but involves a large number of keys (master keys, network keys, link keys). In a full mesh network, if each pair of devices share a master key, then the number of master keys in commercial mode is $O(n^2)$, where n is the number of network devices.

The second restriction of current ZigBee security model is due to its design nature. As based on symmetric key cryptography, this model does not provide digital signature/non-repudiation capability. Though in control and monitoring activities, dealing mostly with network devices, this requirement is not as strict as in our daily life when dealing with human beings, it is still necessary for critical applications. The requirement, however, can be done by using public-key cryptography. There are some recent proposals about using public-key cryptography in ZigBee security in particular ([2], [3]) and wireless sensor networks in general [1].

3 Identity-Based Cryptography

Proposed by Shamir [17] in 1984, identity-based cryptography is based on public-key cryptography, using public and private key pairs to encrypt and decrypt information. However, in this new paradigm, users' identity information, such as unique names or email addresses can be used as public keys for encryption and/or signature verification. Identity-based cryptography, therefore, reduces the complexity and management cost of public-key cryptography system by eliminating the need for key directory. For some applications, the key generation center (private key generator, or PKG) can even be closed after issuing private keys for all network users [17].

The working principle of identity-based cryptography is similar to that of public-key cryptography. For encryption scheme, the sender uses the receiver's identification as the encryption key to encrypt the message. The receiver, having contacted with the PKG, obtained the corresponding decryption key (private key). The receiver then uses this private key to decrypt the message.

The encryption/decryption working principle can be described Fig.2(a) below. In this figure, a master key, *Master-key*, is generated and kept by the PKG while a public parameter, *Params*, is sent to all the interested parties. *ID* and Pr_{ID} are the identity of an entity and the decryption key associated with that identity, respectively.

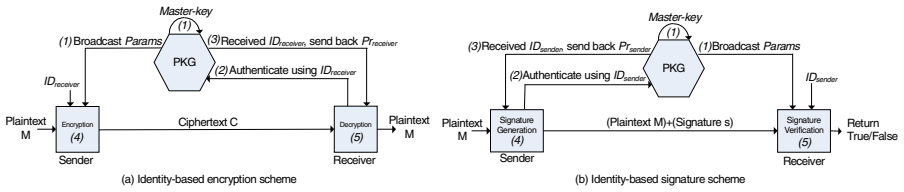


Fig. 2. Identity-based encryption and signature scheme

In the signature generation/verification scheme, the sender uses his private key, Pr_{sender} to sign a message and creates a signature s . He then appends this signature with the intended message and sends to the receiver. The receiver, using the sender's ID and the public parameters to derive the sender's public key, and uses this key to verify the signature. Fig.2(b) represents the identity-based signature scheme.

Though Shamir proposed the framework for identity-based cryptography in 1984, only until 2001, when Boneh and Franklin [4] and Cocks [6] proposed solutions to complete identity-based encryption scheme, this identity-based cryptosystem is fully functional and becomes feasible. The advantages of identity-based cryptography over public-key cryptography are as follow:

- There is no need for keeping public key directory, as the public keys are receivers' identity information widely known by other parties.
- Secure communications can be done even before the recipients obtain the private keys from PKG. In this case, the recipient, having received the encrypted messages, will contact with PKG to get a private key associated with its identity information to decrypt the message.

In spite of having advantages over public-key cryptography, identity-based cryptography has some inherent problems. The key escrow problem happens when the PKG, by keeping the master key, can sign or decrypt any user messages. Though the proposal of multiple PKG ([9], [16]) could help to solve this problem, it imposes a burden on users since they have to authenticate themselves with difference PKG for private key. Another problem associated with identity-based cryptography is key revocation. If a private key is compromised, then there is a problem whether the owner needs to change his identity information corresponding to that private key. This requirement seems to be impractical if the identity information is his representation in real life (name, office numbers or email

addresses). Email spamming when using identity-based encryption [20] is also a problem to be taken care of.

Though identity-based cryptography has both strong and weak points compared to public-key cryptography, it could be applicable in some environments, e.g. a closed group where PKG is trusted. Thus, the use of identity-based cryptography has a promising future.

4 Identity-Based Cryptography for Security in ZigBee Network

When using identity-based cryptography for security in ZigBee, each network device uses some information to identify and distinguish itself from other devices. The device will use that identification as its public key for secure information exchange with other devices.

A device's identity should be self-explanatory, meaning, it helps to differentiate a specific device from the others. This is similar to our real life situation when people can recognize one person based on his self-explanatory identities, e.g. full names, corporate email address etc., but not other not-very-clear identities, e.g. social security number. In a ZigBee network, once a device can be differentiated from other devices based on its identity, this identity can be used as part of the encryption key and the key exchange is simply identity exchange.

We propose to use the descriptive identity, meaning the identity will represent the function of the device in the network. If there is more than one device with the same function, the identity will include additional information to differentiate those devices, e.g. location or serial information. If devices come from different domains, the identity also includes domain descriptions. In addition, to increase security, the identity may optionally have attached time-stamp. In general, the identity information is in the form:

$$ID = \{device\_description@domain\} || time-stamp\}$$

When a device wants to send the information to other devices, it will first obtain the other devices identity information, and then uses this information as the public key to encrypt the information. Each device will also be granted a private key associated with its public identity information and will use that key to decrypt information destined to it. The ZigBee network trust center will play a role of the PKG. The working principle is similar to that of general identity-based cryptography model, as follow:

- Step 1: the trust center generates a master key, *Master-key*, and broadcasts public parameter, *Params*, to all network devices.
- Step 2: a device joins a ZigBee network and authenticates with the ZigBee trust center using its identity information, ID_{device} . The membership approval can be done manually or with hierarchical management.
- Step 3: being approved, a device is granted a private key, Pr_{ID} , associated with its identity information.

- Step 4: other devices, by communicating with this device, obtain its identity information, ID_{device} , and use this information to encrypt message destined to this device. The sender may also sign the message using its private key, Pr_{sender} .
- Step 5: the encrypted message passes through multiple hops to reach the destination. The message is secured on its way since only the designated device can decrypt and read the information using its appropriate private key, Pr_{device} .

In step 2, membership approval can be done in either of the following ways:

- Manual approval: the network operator approves the device's membership and installs the private key for the device manually. When a device first joins a network, the operator assigns identification together with the associated private key for it.
- Approval with hierarchical management: if the device belongs to another trusted domain, the trust center can approve the membership for that device.

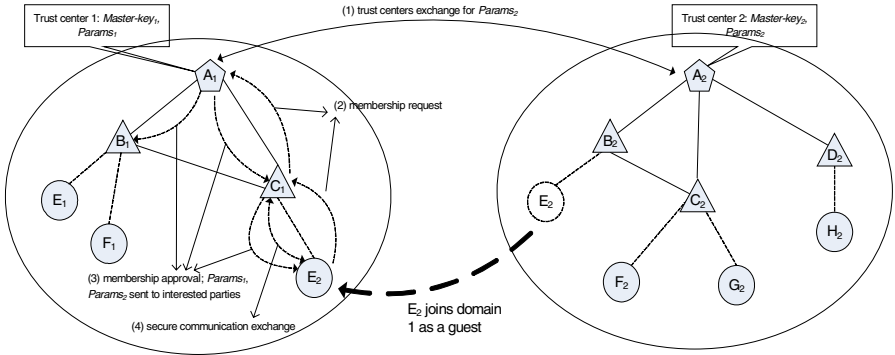


Fig. 3. Membership approval with hierarchical management

Fig.3 illustrates membership approval with hierarchical management. In this scenario, there are two ZigBee domains, e.g. two partnering companies, trusting each other. Device E_2 joins domain 1 as a guest device. Since E_2 uses its private key, Pr_{E_2} , as the decryption key and public parameter $Params_2$ as part of the encryption key, E_2 cannot communicate with other devices in domain 1 using $Params_1$ as part of the public encryption key. In order to communicate in the new domain, E_2 needs to obtain $Params_1$. Similarly, other devices in this domain need to obtain $Params_2$. This requirement is satisfied by the following procedure.

1. Two domain trust centers A_1 , A_2 , exchange public parameters. In this case, A_2 sends $Params_2$ to A_1 . The process can be done in advance or on request of E_2 .
2. E_2 communicates with its adjacent devices (e.g. C_1) and presents its ID. Since E_2 is a foreign device, its ID is forwarded to A_1 for membership approval.
3. Having checked in its database, A_1 knows that E_2 belongs to its trusted domain 2. A_1 then approves the membership and sends $Params_1$ to E_2 , while sending $Params_2$ to other interested parties in its domain.

4. Communication between E_2 and other devices in domain 1 can be done using two public parameters, $Params_1$, $Params_2$, and other identity information.

Once $Params_2$ has been sent to A_1 and is subsequently updated to interested members in domain 1, if a new member from domain 2 joins domain 1 (e.g. F_2), it can just present its ID and obtains $Params_1$ before starting a secure communications.

To guarantee the security in the network, we urge that membership approval is only done by either of the methods above.

Fig.4 shows an example home automation in which identity-based cryptography is applied. In this context, when a TV is turn on, the home audio system should be turn off and the light should be reduce by 50%. The ZigBee device associated with the TV senses that the TV is on and informs the trust center about its status. The trust center will request the light and audio system to change their status subsequently. The information sent is encrypted in the form of $E(S, K, M)$, in which E is an encryption function, S is the signature of the sending device, K is the encryption key (i.e. the recipient's identity), M is the content of the message (i.e. commands, parameters, status). For the simplicity, we assume K is the name of a device. We also assume the communication is done with the help of the trust center. In reality, the communication can be exchanged directly between devices.

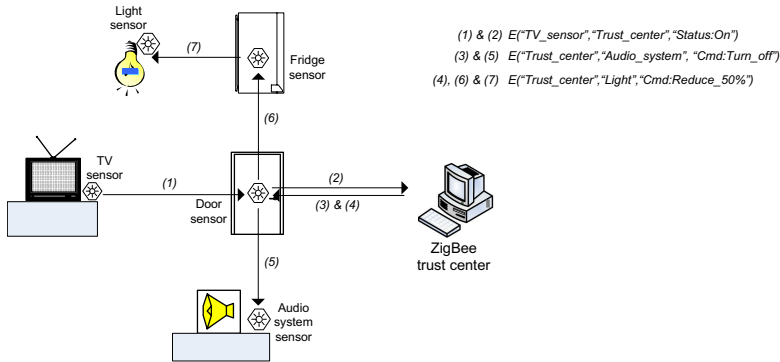


Fig. 4. Home automation example

The TV sensor sends a status message to the trust center, as follow:

$E("TV\_sensor", "Trust\_center", "Status:On")$

On the way to the trust center, these messages can be received and forwarded by other devices, e.g. door sensor, but only the trust center can decrypt these messages and execute the appropriate commands. In response, the trust center issues two commands

$E("Trust\_center", "Light", "Cmd:Reduce\_50\%")$

$E("Trust\_center", "Audio\_system", "Cmd:Turn\_off")$

for the light and audio system respectively. Though these messages are received and relayed by other devices, only the appropriate devices can decrypt and execute the proper commands.

5 Critical Analysis

Network operators can decide as many kinds of identity information as required and stick that information with corresponding devices. To simplify the working environments and to optimize resource usages, we recommend using only one identity information for each device.

Though the use of identity-based cryptography is more resource intensive than symmetric cryptography, we believe that with the use of elliptic curve cryptography ([12], [13], [18]), ZigBee devices can tolerate computational overhead associated with identity-based cryptography

Identity-based cryptography vs. current ZigBee security

The use of identity-based cryptography for security in ZigBee networks provides advantages over the current ZigBee security proposal.

The first advantage is the reduction in the number of keys in network. With a network of n devices, the number of keys in identity-based cryptography is $O(n)$, as each device needs a pair of public/private keys. The number of master keys in current ZigBee security is much larger. With a full mesh network of n devices, the possible number of master keys is $O(n^2)$.

The second advantage of identity-based cryptography over ZigBee's symmetric cryptography is the digital signature/non-repudiation capacity, which is very important for critical applications.

Current ZigBee security proponents may argue that implementing security with public-key cryptography or identity-based cryptography requires a heavy labor work at the installation phase. This amount of work is, however, roughly the same as that of current symmetric key implementation when setting up the master keys for each pair of node. Although in current ZigBee security scheme, master keys can be sent from the trust center to the corresponding devices through the air, this method is not recommended as those keys may be compromised on transmission.

Identity-based cryptography vs. public-key cryptography

In comparison with public-key cryptography, the use of identity-based cryptography also has advantages. Since identity-based cryptography can use device identity information as the encryption key, the trust center does not need to keep the key directory as in normal public-key cryptography scheme and thus, reduces the resource requirement at the trust center. In addition, the encryption key in identity-based cryptography is solely based on identity information and thus, helps to avoid the use of certificates in the trust center.

The inherent key escrow problem in identity-based cryptography can be lessening here as all the devices entrust the trust center and the operator can easily protect the trust center from being compromised.

Table 1. below draws comparisons among different ZigBee security approaches.

Table 1. Comparison of different schemes used for ZigBee security

	Current ZigBee Security	Public-key Cryptography	Identity-based Cryptography
Number of keys in the network ⁽¹⁾ (Full mesh)	$O(n^2)$	$O(n)$	$O(n)$
Digital signature and non-repudiation	No	Yes	Yes
Key directory ⁽²⁾	Yes, at each node and/or at trust center	Possibly, at the trust center	No
Key escrow ⁽³⁾	Yes	No	Yes
Availability of encryption key ⁽⁴⁾	No	No	Yes

(1) This only compares the number of keys needed for session key set up process. This is the number of master key (in ZigBee security), the number of public/private key pairs (in public-key cryptography), and the number of identity information/private key pairs (in identity-based cryptography)

(2) This is the requirement of key store. In current ZigBee security, it is the store of master key, in public-key cryptography, it is the store of public keys at the trust center.

(3) The capability of the trust center to impersonate other nodes

(4) The capability of encrypting a message and sending to a recipient even when that recipient has not obtained its decryption key

Note: (3) and (4) are characteristics of identity-based cryptography. However, in ZigBee applications, these two requirements can be lessening

6 Conclusion

In this paper, we propose an idea to apply identity-based cryptography for providing security in ZigBee networks. The paper presents some features of ZigBee network and its current security issues. It then summarizes some characteristics of identity-based cryptography and proposes an idea to use identity-based cryptography for providing security in ZigBee networks.

Compare to the symmetric key scheme, the use of identity-based cryptography has all the advantages of public-key cryptography, which is more efficient in term of key management. It also helps to provide identification and non-repudiation, which is very important for critical applications.

Compare to public-key cryptography, identity-based cryptography is simpler in key management and thus, optimized for use in resource-sensitive environment like ZigBee.

References

1. Blaser, M.: Enabling Security in Industrial Wireless Sensor Networks. Industrial Embedded Systems (2006)
2. Blaser, M.: Industrial-strength Security for ZigBee: The Case for Public-key Cryptography. Embedded Computing Design (2005)
3. Blaser, M.: Securing ZigBee: Building Robust, Reliable Sensor Networks. Portable Design (2006)
4. Boneh, D., Franklin, M.: Identity-based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
5. Boyen, X.: Multipurpose Identity-based Signcryption: A Swiss Army Knife for Identity-based Cryptography. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 382–398. Springer, Heidelberg (2003)
6. Cocks, C.: An Identity-based Encryption Scheme Based on Quadratic Residues. In: Proceeding of 8th IMA International Conference on Cryptography and Coding, pp. 26–28 (2001)
7. Diffie, W., Hellman, M.: New Direction in Cryptography. IEEE Transactions on Information Theory 22, 644–654 (1976)
8. Frey, G., Muller, M., Ruck, H.: The Tate Pairing and the Discrete Algorithm Applied to Elliptic Curve Cryptosystems. IEEE Transactions on Information Theory 45, 1717–1718 (1999)
9. Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Secure Distributed Key Generation for Discrete-Log Based Cryptosystems. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 295–310. Springer, Heidelberg (1999)
10. Hegland, A.M., Winjum, E., Mjolsnes, S.F., Rong, C., Kure, O., Spilling, P.: A Survey of Key Management in Ad Hoc Networks. IEEE Communications Surveys and Tutorial, 3rd Quarter, pp. 48–66 (2006)
11. IEEE Std 802.15.4-2003. Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (LR-WPANs), IEEE Computer Society (2003)
12. Kobitz, N.: Elliptic Curve Cryptosystems. Mathematics of Computation 48, 203–209 (1987)
13. Menezes, A., Okamoto, T., Vanstone, S.: Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field. IEEE Transactions on Information Theory 39, 1639–1646 (1993)
14. National Institute of Standards and Technology. <http://www.nist.gov>
15. Reddy, J.: ZigBee Security Specification Overview. ZigBee Open House presentation, Hong Kong (2005)
16. Shamir, A.: How to Share a Secret. Communications of the ACM 22, 612–613 (1979)
17. Shamir, A.: Identity-based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
18. Silverman, J.: The Arithmetic of Elliptic Curve. Springer, Heidelberg (1986)
19. Stallings, W.: Cryptography and Network Security: Principles and Practices, 3rd edn. Prentice-Hall, Englewood Cliffs (2003)
20. Veigner, C., Rong, C.: On Email Spamming Under the Shadow of Large Scale Use of Identity-Based Encryption. In: Yang, L.T., Jin, H., Ma, J., Ungerer, T. (eds.) ATC 2006. LNCS, vol. 4158, pp. 521–530. Springer, Heidelberg (2006)
21. Wheeler, A.: ZigBee Wireless Networks for Industrial Systems. White Paper, <http://www.microcontroller.com>
22. ZigBee Alliance. <http://www.zigbee.org>
23. ZigBee Alliance. ZigBee Specification 1.1 (2005)

Efficient Identity-Based Signcryption Scheme for Multiple Receivers^{*}

Yong Yu¹, Bo Yang², Xinyi Huang³, and Mingwu Zhang²

¹ National Key Lab. of ISN,
Xidian University, Xi'an, 710071, P.R. China
yuyong@mail.xidian.edu.cn

² College of information,
South China Agricultural University, Guangzhou, 510642, P.R. China
{byang,zmingwu}@scau.edu.cn

³ School of Computer Science and Software Engineering,
University of Wollongong Wollongong, NSW 2522, Australia
xyhuang81@gmail.com

Abstract. Signcryption is a cryptographic primitive that performs signature and encryption simultaneously. In this paper, we propose an efficient identity based signcryption scheme for multiple receivers which needs only one pairing computation to signcrypt a message for n different receivers and can provide confidentiality and authenticity simultaneously. This scheme turns out to be more efficient than all others proposed so far. We prove its security in the random oracle model. Specifically, we prove its semantic security under the hardness of Bilinear Diffie-Hellman (BDH) problem and its unforgeability under the Computational Diffie-Hellman (CDH) assumption.

1 Introduction

Identity based(ID-based) cryptosystem was introduced by Shamir [1] in 1984. Its main idea is that public keys can be derived from arbitrary strings while private keys can be generated by the trusted Private Key Generator(PKG). This removes the need for senders to look up the receiver's public key before sending out an encrypted message. ID-based cryptography is supposed to provide a more convenient alternative to conventional public key infrastructure.

Signcryption, first proposed by Zheng [2], is a cryptographic primitive that performs signature and encryption simultaneously, at a lower computational costs and communication overheads than the signature-then-encryption approach. Followed by the first constructions given in [2], a number of new schemes and improvements have been proposed [3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]. In [8], Malone-Lee gave the first ID-based signcryption scheme. Libert and Quisquater [9] pointed out that Malone-Lee's scheme [8] is not semantically secure and proposed three

^{*} This work was supported by the National Natural Science Foundation of China under Grants No. 60372046 and 60573043.

provably secure ID-based signcryption schemes. However, the properties of public verifiability and forward security are mutually exclusive in their schemes. To overcome this weakness, Chow *et al.* [10] designed an ID-based signcryption scheme that provides both public verifiability and forward security. In [11], Boyen presented an ID-based signcryption scheme that provides not only public verifiability and forward security but also ciphertext unlinkability and anonymity. In [12], Chen and Malone-Lee improved Boyen's scheme in efficiency and proposed the most efficient signcryption scheme to date.

All of the above schemes consist of only a single receiver. In practice, broadcasting a message to multiple users in a secure and authenticated manner is an important facility for a group of people who are jointly working on the same project to communicate with one another. In [14], Zheng proposed a signcryption scheme for multiple recipients (multi-recipient signcryption). The basic idea is to use two types of keys: the first type consists of only a single randomly chosen key (a message-encryption key) and the second type of keys include a key chosen independently at random for each recipient (called a recipient specific key). The message-encryption key is used to encrypt a message with a private key cipher, while a recipient specific key is used to encrypt the message-encryption key. Duan *et al.* [15] proposed an efficient multi-receiver ID-based signcryption based on Yuen *et al.*'s [17] signcryption scheme recently.

Our Contributions: In this paper, we propose an efficient multi-receiver identity based signcryption scheme which needs only one pairing computation to signcrypt a message for n different receivers and can provide confidentiality and authenticity simultaneously. We compare our scheme with several multi-receiver constructions from the efficiency point of view and argue that our scheme is more efficient than other known constructions. We prove its semantic security under the hardness of Bilinear Diffie-Hellman (BDH) problem and its unforgeability under the Computational Diffie-Hellman (CDH) assumption.

Roadmap: The rest of this paper is organized as follows. Section 2 presents the basic concepts of bilinear pairing and the hard problems underlying our proposed scheme. Section 3 gives the syntax and security notions of multi-receiver ID-based signcryption schemes. We describe our multi-receiver ID-based Signcryption Scheme and prove its security in Section 4, and compare its efficiency to other known constructions in Section 5. We draw our conclusion in Section 6.

2 Preliminaries

2.1 Bilinear Pairings and Related Computational Problems

Let G_1 be a cyclic additive group, whose order is a prime q , and G_2 be a cyclic multiplicative group of the same order. Let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be a mapping with the following properties:

1. *Bilinearity:* $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$ for all $P, Q \in \mathbb{G}_1, a, b \in \mathbb{Z}_q$.
2. *Non-degeneracy:* There exists $P, Q \in \mathbb{G}_1$ such that $\hat{e}(P, Q) \neq 1_{\mathbb{G}_2}$.

3. *Computability*: There exists an efficient algorithm to compute $\hat{e}(P, Q)$ for all $P, Q \in \mathbb{G}_1$.

We refer the readers to [20] for more details about the constructions for such groups and bilinear pairings.

Now we recall some candidate hard problems that provide underling assumptions from pairings that will be used later.

1. **Bilinear Diffie-Hellman problem (BDHP)**. Given $(P, aP, bP, cP) \in \mathbb{G}_1$ for unknown $a, b, c \in \mathbb{Z}_q^*$, to compute $\hat{e}(P, P)^{abc}$.
The advantage of any probabilistic polynomial time (PPT) algorithm $\mathcal{A}$ in solving BDH problem in $\mathbb{G}_1$ is defined as

$$Adv_{\mathcal{A}}^{BDH} = Pr[\mathcal{A}(P, aP, bP, cP) = \hat{e}(P, P)^{abc} : a, b, c \in \mathbb{Z}_q^*].$$

BDH assumption: For any PPT algorithm $\mathcal{A}$, $Adv_{\mathcal{A}}^{BDH}$ is negligible.

2. **Computational Diffie-Hellman problem (CDHP)**. Given $(P, aP, bP) \in \mathbb{G}_1$ for unknown $a, b \in \mathbb{Z}_q^*$, to compute abP .
The advantage of any PPT algorithm $\mathcal{A}$ in solving CDH problem is defined as $Adv_{\mathcal{A}}^{CDH} = Pr[\mathcal{A}(P, aP, bP) = abP : a, b \in \mathbb{Z}_q^*]$.

CDH assumption: For any PPT algorithm $\mathcal{A}$, $Adv_{\mathcal{A}}^{CDH}$ is negligible.

3 Review of Definition of Multi-receiver ID-Based Signcryption

We omit this section for the page limitation. Reader can contact the author for full version of the this paper if needed.

4 The Proposed Scheme and Security Results

4.1 The Scheme

In this section, we present an efficient ID-based signcryption scheme for multiple receivers based on the bilinear pairing. Our scheme is motivated by Chen's efficient ID-Based signcryption scheme [12]. The following shows the details of the scheme.

- **Setup**: Given a security parameter k , the PKG chooses bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2)$ of prime order q . Let P be a generator of $\mathbb{G}_1$. Choose a master key $s \in \mathbb{Z}_q^*$ and $R \in \mathbb{G}_1^*$, compute $P_{pub} = sP$ and $\theta = \hat{e}(R, P_{pub})$. Choose hash functions $H_0 : \{0, 1\}^* \rightarrow \mathbb{G}_1$, $H_1 : \mathbb{G}_1 \times \{0, 1\}^* \rightarrow \mathbb{Z}_q^*$ and $H_2 : \mathbb{G}_2 \rightarrow \{0, 1\}^n$. Here n is the number of bits of a message to be signed and encrypted. The public parameters are $\text{params} = (\mathbb{G}_1, \mathbb{G}_2, P, P_{pub}, R, \theta, \hat{e}, H_0, H_1, H_2)$.
- **Keygen**: To extract the private key for user U with identity ID_U , compute the public key $Q_U = H_0(ID_U)$ and the secret key $D_U = sQ_U$.

- **Signcrypt:** Suppose Alice whose identity is ID_A wants to signcrypt a message m to n different receivers $ID_1, \dots, ID_n$.
 1. Alice chooses $r \in \mathbb{Z}_q^*$, computes $X = rQ_A$, $h_1 = H_1(X, m)$ and $Z = (r + h_1)D_A$.
 2. Alice computes $U = rP$, $\omega = \hat{e}(Z, P)$, $c = m \oplus H_2(\omega)$ and $W = \theta^r \omega$.
Then for $i = 1, \dots, n$, Alice computes $T_i = rH_1(ID_i) + rR$.

The ciphertext is set to be $\sigma = (c, U, X, W, T_1, \dots, T_n, \mathcal{L})$, where $\mathcal{L}$ is a label that contains information about how “ T_i ” is associated with each receiver.
- **De-signcrypt:** Each receiver ID_i can use his or her private key to decrypt σ .
 1. Assume the private key of ID_i is D_i . ID_i computes $\omega' = W\hat{e}(U, D_i)\hat{e}(P_{pub}, T_i)^{-1}$ and $m' = c \oplus H_2(\omega')$.
 2. ID_i computes $Q_A = H_0(ID_A)$ and $h'_1 = H_1(X, m')$ and compare if $\omega' = \hat{e}(P_{pub}, X + h'_1 Q_A)$ holds.

Output m' if the above verifications are true. Otherwise, return $\perp$.

Consistence: It is easy to see that the above De-signcrypt algorithm is consistent. Indeed, if σ is a valid ciphertext, then

$$\begin{aligned}
 \omega' &= W\hat{e}(U, D_i)\hat{e}(P_{pub}, T_i)^{-1} \\
 &= \theta^r \omega \hat{e}(rP, sH_1(ID_i))\hat{e}(sP, rH_1(ID_i) + rR)^{-1} \\
 &= \hat{e}(R, P_{pub})^r \omega \hat{e}(rP, sH_1(ID_i))\hat{e}(sP, rH_1(ID_i))^{-1} \hat{e}(P_{pub}, rR)^{-1} \\
 &= \hat{e}(R, P_{pub})^r \omega \hat{e}(R, P_{pub})^{-r} \\
 &= \omega
 \end{aligned}$$

Therefore, $m' = c \oplus H_2(\omega') = c \oplus H_2(\omega) = m$ and $h'_1 = H_1(X, m') = H_1(X, m) = h_1$. Furthermore,

$$\hat{e}(P_{pub}, X + h_1 Q_A) = \hat{e}(P_{pub}, rQ_A + h_1 Q_A) = \hat{e}(P, (r + h_1)D_A) = \hat{e}(Z, P) = \omega.$$

4.2 Security Results

Confidentiality: The following theorem shows that our scheme is IND-sMIBSC-CCA assuming the BDH problem is hard.

Theorem 1. *In the random oracle model, assume there is an IND-sMIBSC-CCA adversary called $\mathcal{A}$ that is able to distinguish ciphertext during the game of Definition 2 with an advantage ϵ when asking at most q_{H_0} identity hashing queries, at most q_{H_1} H_1 queries, at most q_{H_2} H_2 queries, q_K key extraction queries, q_S signcryption queries and q_D De-signcryption queries. Then, there exists a distinguisher $\mathcal{C}$ that can solve the BDH problem with an advantage*

$$Adv(\mathcal{C})^{BDHP(G_1, P)} > \frac{\epsilon(2^k - q_D q_{H_2}) - q_D q_{H_2}}{2^{k+1}}$$

Proof: Our proof is motivated by [16]. Assume $\mathcal{C}$ receives a random instance (P, aP, bP, cP, h) of the BDH problem. His goal is to decide whether $h = \hat{e}(P, P)^{abc}$ or not. $\mathcal{C}$ will run $\mathcal{A}$ as a subroutine and act as $\mathcal{A}$'s challenger in the IND-sMIBSC-CCA game. During the game, $\mathcal{A}$ will consult $\mathcal{C}$ for answers to the random oracles H_0, H_1 , and H_2 . Roughly speaking, these answers are randomly generated, but to maintain the consistency and to avoid collision, $\mathcal{C}$ keeps three lists L_0, L_1 and L_2 to store the answers. We assume that $\mathcal{A}$ will ask for $H_0(ID)$ before ID is used in any key extraction and de-signcryption queries. At the beginning of the game, $\mathcal{A}$ outputs target multiple identities $(ID_1^*, \dots, ID_n^*)$. $\mathcal{C}$ gives $\mathcal{A}$ the system parameters with $R = bP$ and $P_{pub} = cP$. Note that c is unknown to $\mathcal{C}$ and this value simulates the master key of the PKG in the game.

- **H_0 queries:** On a $H_0(ID_j)$ query, $\mathcal{C}$ checks if there exists (ID_j, t_j, Q_j) in L_0 . If such a tuple is found, $\mathcal{C}$ answers Q_j , otherwise he follows the steps below.
 - If $ID_j = ID_i^*$ for some $i \in [1, n]$, choose t_j^* from Z_q^* randomly and compute $Q_j = t_j^*P - R$.
 - Else choose t_j^* from Z_q^* randomly and compute $Q_j = t_j^*P$.
 - Put (ID_j, t_j, Q_j) into L_0 and return Q_j as the answer.
- **H_1 queries:** For a query $H_1(X_j, m_j)$, $\mathcal{C}$ checks if there exists (X_j, m_j, h_{1j}) in L_1 . If such a tuple is found, $\mathcal{C}$ answers h_{1j} . Otherwise he chooses h_1 from Z_q^* randomly, returns it as an answer to the query and puts the tuple (X_j, m_j, h_1) into L_1 .
- **H_2 queries:** On a $H_2(\omega_j)$ query, $\mathcal{C}$ searches a pair (ω_j, h_{2j}) in the list L_2 . If such a pair is found, $\mathcal{C}$ answers h_{2j} . Otherwise he answers $\mathcal{A}$ by a random binary sequence h_{2j} from $\{0, 1\}^n$ such that no entry $(\cdot, h_{2j})$ exists in L_2 (in order to avoid collisions on H_2) and puts the pair (ω_j, h_{2j}) into L_2 .
- **Key extraction queries:** Upon receiving a key extraction query on ID_j ($ID_j \neq ID_i^*$ for $i = 1, \dots, n$), $\mathcal{C}$ recovers (ID_j, t_j, Q_j) in L_0 and returns $D_j = t_j P_{pub}$ as the answer.
- **Signcryption queries:** Upon receiving a signcryption query on a plaintext m , a sender's identity ID_i and n receivers ID_{R_i} ($i = 1, \dots, n$), $\mathcal{C}$ firstly checks if $ID_i \neq ID_j^*$ ($j = 1, \dots, n$). If $ID_i \neq ID_j^*$ ($j = 1, \dots, n$), $\mathcal{C}$ can get ID_i 's private key by key extraction query, then, $\mathcal{C}$ can generate signcryption ciphertext on m in the normal way. Otherwise, $\mathcal{C}$ chooses $r', h_1, t_R \in \mathbb{Z}_q^*$ at random, computes $U = r'P - h_1(cP)$, $X = (t_j^* - t_R)U$ and checks if L_1 already contains a tuple (X, m, h'_1) with $h'_1 \neq h_1$. In this case, $\mathcal{C}$ repeats the process with another r', h_1, t_R until finding a tuple (X, m, h_1) whose first two elements do not figure in a tuple of L_1 . $\mathcal{C}$ adds the admissible tuple into L_1 . Then $\mathcal{C}$ computes $R = t_R P$ and $W = \hat{e}(t_R U + r'P, cP)$. $\mathcal{C}$ also retrieves (ID_{R_i}, t_{R_i}) from L_0 and computes $T_i = (t_{R_i} + t_R)U$. Finally, $\mathcal{C}$ proceeds as in the normal signcryption process to produce the desired ciphertext.
- **De-signcryption queries:** When $\mathcal{A}$ submits a ciphertext $\sigma = (c, U, X, W, T_1, \dots, T_n, \mathcal{L})$ together with an sender's identity ID_A both chosen by $\mathcal{A}$ and asks for the result of $\text{de-signcrypt}(\sigma, ID)$. If $ID \neq ID_i^*$ for $i = 1, \dots, n$, $\mathcal{C}$ searches L_0 to find (ID, t, Q) , then he computes $D_{ID} = tP_{pub}$. $\mathcal{C}$ can

de-signcrypt the ciphertext σ in the normal way with this private key. If $ID = ID_i^*$ for some $i \in [1, n]$, $\mathcal{C}$ searches all the combinations (ID_i, m_i, X) such that $(m_i, X, h_{1i}) \in L_1$, $(\omega_i, h_{2i}, \omega_i) \in L_2$ for some h_{1i}, h_{2i}, ω_i under the constraint that $\omega_i = \hat{e}(P_{pub}, X + h_{1i}Q_A)$ and $m_i = h_{2i} \oplus c$. If no such tuples are found, the $\perp$ symbol is returned to $\mathcal{A}$. Otherwise, m_i is returned.

At the end of the first stage, $\mathcal{A}$ outputs two equal-length plaintexts m_0, m_1 together with an arbitrary sender's identity ID_A on which he wishes to be challenged. $\mathcal{A}$ requires a challenge ciphertext built under the receivers' identities $(ID_1^*, \dots, ID_n^*)$. $\mathcal{C}$ chooses a random bit $b \in \{0, 1\}$ and signcrypts message m_b . To do so, he sets $U^* = aP$ and searches L_0 to get t_i^* that corresponds to ID_i^* and computes $T_i^* = t_i^*(aP)$ for $i = 1, \dots, n$. $\mathcal{C}$ chooses Z^* from G_1^* and $(X, h_1) \in L_1$ and checks if $\hat{e}(P, Z^*) = \hat{e}(P_{pub}, X + h_1Q_A)$. If the equation does not hold, $\mathcal{C}$ repeats the process until finding a satisfying Z^* and h_1 . Then $\mathcal{C}$ computes $\omega^* = \hat{e}(Z^*, P)$, $W^* = h\omega^*$ where h is the candidate of BDH problem and $c^* = m_b \oplus H_2(\omega^*)$. Finally, $\mathcal{C}$ creates a label $\mathcal{L}^*$ and sends the ciphertext $\sigma^* = (c^*, U^*, X, W^*, T_1^*, \dots, T_n^*, \mathcal{L}^*)$ to $\mathcal{A}$.

$\mathcal{A}$ performs new queries as in above. However, it can not ask the de-signcrypt query of the challenge ciphertext σ^* with the private key of any target identity nor query the de-signcrypt oracle on an identity a ciphertext σ' which is only different from σ^* in the receiver information part. At the end of the simulation, $\mathcal{A}$ outputs a bit b' for which he believes the relation $\sigma^* = \mathbf{Signcrypt}(m_{b'}, D_{ID_A}, (ID_1^*, \dots, ID_n^*))$ holds. At this moment, if $b = b'$, $\mathcal{C}$ outputs 1 as the answer of BDH problem because at this time $\theta^r = \hat{e}(R, P_{pub})^r = \hat{e}(bP, cP)^r = \hat{e}(bP, cP)^a = \hat{e}(P, P)^{abc}$ as a solution of the BDH problem, otherwise $\mathcal{C}$ outputs 0 indicating that $h \neq \hat{e}(P, P)^{abc}$.

Now we have to access $\mathcal{C}$'s probability of success. The only case that the simulation is not perfect is that a valid ciphertext is rejected in a de-signcrypt query. It is easy to see that for every pair (ω_i, h_{2i}) in L_2 , there is exactly one h_{1i} of elements in the range of oracle H_1 providing a valid ciphertext. The probability to reject a valid ciphertext is thus not greater than $q_{H_2}/2^k$. So we have

$$Adv(\mathcal{C}) > \frac{\epsilon + 1}{2} \left(1 - \frac{q_D q_{H_2}}{2^k}\right) - \frac{1}{2} = \frac{\epsilon(2^k - q_D q_{H_2}) - q_D q_{H_2}}{2^{k+1}}.$$

Theorem 2. *In the random oracle model, if a forger $\mathcal{F}$ has non-negligible advantage $\epsilon \geq 10n(q_S + 1)(q_S + q_{H_1})/q + q_D q_{H_2}/2^k$ against the ESUF-sMIBSC-CMA security of our scheme when performing q_S signcrypt queries, q_D de-signcrypt queries and q_{H_i} queries to oracles H_i (for $i = 0, 1, 2$), then there is a algorithm $\mathcal{B}$ that can solve the CDH problem with probability $\epsilon_{\mathcal{B}} \geq 1/9$.*

5 Efficiency

We firstly compare the major computational costs and communication costs of our scheme with those of the obvious construction of multi-receiver ID-based

signcryption scheme that simply re-signcrypts a message n times using an ID-based signcryption scheme. Here we employ Chen and Malone-Lee's improved ID-based signcryption scheme [12] as an example. To signcrypt a message m for n different receivers, our scheme only needs a pairing operation, $n + 3$ scalar multiplications in $\mathbb{G}_1$, n additions in $\mathbb{G}_1$ and 1 exponentiation in $\mathbb{G}_2$. The size of the ciphertext is $(n + 2)|\mathbb{G}_1| + |\mathbb{G}_2| + |m|$. On the other hand, re-signcrypting a message n times using Chen and Malone-Lee's improved ID-based signcryption scheme [12] needs n pairing operations, $2n$ scalar multiplications in $\mathbb{G}_1$. The size of the ciphertext is $2n|\mathbb{G}_1| + n|ID| + n|m|$. The comparisons are summarized in Table 1. Since the pairing is the most expensive computation, our scheme is more efficient than re-signcrypting a message n times using an ID-based signcryption scheme. Compared with the scheme in [15], our scheme is a little more efficient since in the de-signcryption phase, our scheme needs one less pairing operation. We compare the two schemes by giving the computational costs respectively in Table 2 and Table 3. Note that both our scheme and the scheme proposed in [15] are for multiple receivers, therefore, when there are n receivers and all of them need to de-signcrypt a ciphertext, $4n$ pairing computations are needed using the scheme in [15] while only $3n$ pairings in our scheme. When the number of the receiver is becoming large, our scheme is more efficient.

Table 1. Efficiency comparisons

Scheme	Pairing	Add in $\mathbb{G}_1$	Scalar Mul in $\mathbb{G}_1$	Exp in $\mathbb{G}_2$	Ciphertext size
Re-signcryption	n	0	$2n$	0	$2n \mathbb{G}_1 + n ID + n m $
Our Scheme	1	n	$n+3$	1	$(n + 2) \mathbb{G}_1 + \mathbb{G}_2 + m $

Table 2. The computational cost of the scheme in [17]

Scheme in [17]	Mul in $\mathbb{G}_1$	Exponential in $\mathbb{G}_1$	Inverse in $\mathbb{G}_1$	Inverse in $\mathbb{G}_2$	Pairing
Signcrypt	$n+2$	$n+4$	0	0	1
De-signcrypt	1	0	1	1	4

Table 3. The computational cost of our scheme

Our Scheme	Add in $\mathbb{G}_1$	Scalar Mul in $\mathbb{G}_1$	Mul in $\mathbb{G}_2$	Inverse in $\mathbb{G}_2$	Pairing
Signcrypt	n	$n+3$	1	0	1
De-signcrypt	1	1	2	1	3

6 Conclusion

We have proposed an efficient ID-based signcryption scheme for multiple receivers, which needs only one pairing computation to signcrypt a message for n different receivers. We proved that the scheme is secure against adaptive ciphertext attack in the random oracle model assuming the Decisional Bilinear

Diffie-Hellman problem is untractable. Further work is on the way to construct efficient ID-based signcryption schemes without random oracles.

References

1. Shamir, A.: Identity-based cryptosystem and signature scheme. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 120–126. Springer, Heidelberg (1985)
2. Zheng, Y.: Digital signcryption or how to achieve $\text{cost}(\text{signature} \ \& \ \text{encryption}) \ll \text{cost}(\text{signature}) + \text{cost}(\text{encryption})$. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 165–179. Springer, Heidelberg (1997)
3. Petersen, H., Michels, M.: Cryptanalysis and improvement of signcryption schemes. IEE proceedings-Computers and Digital Techniques 145(2), 149–151 (1998)
4. Bao, F., Deng, R.H.: A signcryption scheme with signature directly verifiable by public key. In: Imai, H., Zheng, Y. (eds.) PKC 1998. LNCS, vol. 1431, pp. 55–59. Springer, Heidelberg (1998)
5. Zheng, Y., Imai, H.: How to construct efficient signcryption schemes on elliptic curves. Information Processing Letters 68(5), 227–233 (1998)
6. Malone-Lee, J., Mao, W.: Two birds one stone: signcryption using RSA. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 211–226. Springer, Heidelberg (2003)
7. Jung, H.Y., Lee, D.H., Lim, J.I., Chang, K.S.: New DSA-verifiable signcryption schemes. Information Security Application-WISA 2001, pp. 463–475 (2001)
8. Malone-Lee, J.: Identity based signcryption. Cryptology ePrint Archive. Report 2002/098 (2002)
9. Libert, B., Quisquator, J.J.: A new identity based signcryption scheme from pairings. In: 2003 IEEE information theory workshop. Paris, France, pp. 155–158 (2003)
10. Chow, S.S.M., Yiu, S.M., Hui, L.C.K., Chow, K.P.: Efficient forward and provably secure ID-based signcryption scheme with public verifiability and public ciphertext authenticity. In: Lim, J.-I., Lee, D.-H. (eds.) ICISC 2003. LNCS, vol. 2971, pp. 352–369. Springer, Heidelberg (2004)
11. Boyen, X.: Multipurpose identity based signcryption: a swiss army knife for identity based cryptography. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 383–399. Springer, Heidelberg (2003)
12. Chen, L., Malone-Lee, J.: Improved identity-based signcryption. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 362–379. Springer, Heidelberg (2005)
13. Barreto, P.S.L.M., Libert, B., McCullagh, N., Quisquater, J.J.: Efficient and provably-secure identity based signatures and signcryption from bilinear maps. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 515–532. Springer, Heidelberg (2005)
14. Zheng, Y.: Signcryption and its applications in efficient public key solutions. In: Okamoto, E. (ed.) ISW 1997. LNCS, vol. 1396, pp. 291–312. Springer, Heidelberg (1998)
15. Duan, S., Cao, Z.: Efficient and provably secure multi-receiver identity-based signcryption. In: Batten, L.M., Safavi-Naini, R. (eds.) ACISP 2006. LNCS, vol. 4058, pp. 195–206. Springer, Heidelberg (2006)
16. Baek, J., Safavi-Naini, R., Susilo, W.: Efficient multi-receiver identity based encryption and its application to broad encryption. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 380–397. Springer, Heidelberg (2005)

17. Yuen, T.H., Wei, V.K.: Fast and proven secure blind identity based signcryption from pairings. In: Menezes, A.J. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 305–322. Springer, Heidelberg (2005)
18. Miyaji, A., Nakabayashi, M., Takano, S.: New explicit conditions of elliptic curve trace for FR-reduction. IEICE Transactions on Fundamentals E84-A(5), 1234–1243 (2001)
19. Barreto, P.S.L.M., Kim, H.Y., Lynn, B., Scott, M.: Efficient algorithms for pairing-based cryptosystems. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 354–368. Springer, Heidelberg (2002)
20. Boneh, D., Franklin, M.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
21. Smart, N.P., Vercauteren, F.: On computable isomorphisms in efficient pairing based systems. Cryptology ePrint Archive, Report 2005/116 (2005)
<http://eprint.iacr.org/2005/116>
22. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
23. Boneh, D., Boyen, X.: Efficient selective-ID secure identity based encryption without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004)

Identity-Based Proxy Signature from Pairings

Wei Wu, Yi Mu, Willy Susilo, Jennifer Seberry, and Xinyi Huang

Centre for Computer and Information Security Research

School of Computer Science & Software Engineering

University of Wollongong, Australia

weiwu81@gmail.com, {ymu, wsusilo, j.seberry, xh068}@uow.edu.au

Abstract. A proxy signature scheme allows an entity to delegate its signing capability to another entity (proxy) in such a way that the proxy can sign messages on behalf of the delegator. Proxy signatures have found numerous practical applications such as distributed systems, mobile agent applications, etc. Recently, Xu, Zhang and Feng proposed the first formal models of identity based proxy signature. Unfortunately, their model does not capture the notion of adaptively chosen message and chosen identity attacker in identity based system. In this paper, we redefine the security models of identity based proxy signature to capture the most stringent attacks against adaptively chosen message and chosen identity attacker. We also propose a new provably secure identity based proxy signature scheme whose security is based on the hardness of Computational Diffie-Hellman problem in the random oracle model.

1 Introduction

Traditional public-key cryptography (PKC) has many applications; however, PKC seems less attractable in distributed and ad hoc systems, since the requirement of public-key infrastructure prevents its applications in this field. The notion of identity-based cryptosystem was introduced by Shamir in his seminal paper in [22]. The main essence of identity-based cryptosystem is to remove the need of certification of the public keys. The public key of each party is obtained from his/her public identity, such as the IP address in the ad hoc system, which can uniquely identify the party. Since its introduction in [22], many identity based schemes have been proposed (e.g., [2,15,20,25]).

On the other hand, permanent connections between customers and servers in this kind of system are unnecessary and infeasible. In order to ensure service availability to the customers distributed in the whole networks, the server must delegate his rights to some other parties in the systems, such as the mobile agents. This way, replication can be achieved and there is no need to count on a single server. A proxy signature scheme is a variation of the standard signature schemes, in which an original signer (say, Alice) can delegate his signing right to another signer, called the proxy signer (say, Bob), for signing messages. The notion of proxy signature was introduced by Mambo, Usuda and Okamoto [17]. Since then, proxy signature schemes have attracted a considerable amount of interest from the cryptographic research community. Based on the delegation type,

there are three types of proxy signatures: *full delegation*, *partial delegation*, and *delegation by warrant*. In the full delegation system, Alice's secret key is given to Bob directly so that Bob can have the same signing capability as Alice. In practice, such schemes are obviously impractical and insecure. In a partial delegation proxy signature scheme, a proxy signer possesses a key, called private proxy key, which is different from Alice's private key. Hence, proxy signatures generated by using the proxy private key are different from Alice's signatures. However, in such schemes, the messages a proxy signer can sign are *not* limited. This weakness is eliminated in delegation by a warrant that specifies what kinds of messages are delegated. Here, the original signer uses the signing algorithm of a standard signature scheme and its secret key to sign a warrant and generate a signature on the warrant which is called as delegation. The proxy signer uses the delegation and his secret key to create a proxy signature on behalf of the original signer. According to whether the original signer can generate a valid proxy signature or not, proxy signatures can be classified into *proxy-unprotected* and *proxy-protected* schemes. In a proxy-protected scheme only the proxy signer can generate proxy signatures, while in a proxy-unprotected scheme either the proxy signer or the original signer can generate proxy signatures. In many applications, proxy-protected schemes are required to avoid the potential disputes between the original signer and the proxy signer. Though there exist many proxy signature schemes, most of them are insecure [14,11,13,18,19,23]. Recently, based on the work of [4,16], Xu, Zhang and Feng formalized the notion of security for ID-based proxy signature schemes and proposed a scheme based on the bilinear pairings [26]. However, as we will show later, the model defined in their paper does not capture the definitions of adaptively chosen message and chosen identity attacker in identity based system.

Our contribution

Firstly, we redefine the security notion of ID-based proxy signature schemes to capture the most stringent attacks in this model, namely the adaptively chosen message and chosen identity attacks. Compared with the model proposed in [26], our model captures a stronger security notion of the proxy signature by allowing the adversaries to behave more adaptively in oracle accessing. The adversary can freely choose the identities of the original signer and the proxy signer. We proceed by proposing a new identity based proxy signature scheme whose security is based on the hardness of the Computational Diffie-Hellman problem in the random oracle model. Compared with the scheme proposed in [26], the new proposed scheme enjoys less operation cost.

Roadmap

The rest of this paper is arranged as follows. In next section, we provide the preliminaries of our scheme including bilinear pairings and security assumptions. In Section 3, we describe the formal models of our ID-based proxy signature scheme. We present our ID-based proxy signature scheme with its security analysis in Section 4. Finally, we conclude our paper in Section 5.

2 Preliminaries

2.1 Bilinear Pairing

Let $\mathbb{G}_1$ be a cyclic additive group of prime order q , P is the generator of $\mathbb{G}_1$, $\mathbb{G}_T$ denotes a multiplicative group of the same order. Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ be a bilinear pairing with the following properties:

- e is bilinear: $e(aP, bP) = e(P, P)^{ab}$, for all $a, b \in \mathbb{Z}_q$.
- e is non-degenerate: $e(P, P) \neq 1_{\mathbb{G}_T}$.
- e is efficiently computable.

We say $\mathbb{G}_1$ is a bilinear group if there exists a group $\mathbb{G}_T$, and a bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ as above, and e , and the group action in $\mathbb{G}_1$ and $\mathbb{G}_T$ can be computed efficiently. Using the bilinear pairing on certain elliptic curves over a finite field of characteristic, the elements in $\mathbb{G}_1$ can have short representation [3,9].

2.2 Security Definitions

Definition 1. Computational Diffie-Hellman (CDH) on $\mathbb{G}_1$,

Given $P, aP, bP \in \mathbb{G}_1$, for some unknown $a, b \in_R \mathbb{Z}_q$, compute $abP \in \mathbb{G}_1$.

The success probability of an algorithm $\mathcal{A}$ in solving the CDH problem on $\mathbb{G}_1$ is denoted as $\text{Succ}_{\mathcal{A}, \mathbb{G}_1}^{CDH} = \Pr[\mathcal{A}(P, aP, bP) = abP : a, b \in_R \mathbb{Z}_q]$.

Definition 2. Computational Diffie-Hellman (CDH) Assumption on $\mathbb{G}_1$,

Given $P, aP, bP \in_R \mathbb{G}_1$, for some unknown $a, b \in \mathbb{Z}_q$, $\text{Succ}_{\mathcal{A}, \mathbb{G}_1}^{CDH}$ is negligible for any polynomially bounded algorithm.

3 Formal Models of ID-Based Proxy Signatures

Let Alice denote the original signer and Bob the proxy signer. The ID-based proxy signature scheme consists of the following algorithms: ParaGen, KeyExtract, StandardSign, StandardVer, DelegationGen, ProxySign and ProxyVer.

1. **ParaGen:** Taking as input the system security parameter ℓ , this algorithm outputs system's parameters **Para** and the system's master key s . That is: $(\text{Para}, s) \leftarrow \text{ParaGen}(\ell)$.
2. **KeyExtract:** Taking as input system's parameter **Para** and an identity ID_i where $i \in \{a, b\}$ denotes the identities of Alice and Bob, respectively, this algorithm generates a secret key sk_{ID_i} for them. That is: $sk_{ID_i} \leftarrow \text{KeyExtract}(\text{Para}, ID_i, s)$.
3. **StandardSign:** Input system's parameter **Para**, the signer's secret key sk_{ID} and the message M to be signed, this algorithm outputs the standard signature σ_S . That is: $\sigma_S \leftarrow \text{StandardSign}(\text{Para}, M, sk_{ID})$.

4. **StandardVer**: Input system's parameter Para , the signer's identity ID , the signed message M and the standard signature σ_S , this algorithm outputs **True** if σ_S is a valid standard signing of the message M under the identity ID and outputs $\perp$ otherwise. That is: $\{\text{True}, \perp\} \leftarrow \text{StandardVer}(\text{Para}, ID, M, \sigma_S)$.
5. **DelegationGen**: Input system's parameter Para , the original signer's secret key sk_{ID_a} and the warrant W to be signed, this algorithm uses the **StandardSign** algorithm to generate the delegation σ_w . That is: $\sigma_w \leftarrow \text{DelegationGen}(\text{Para}, W, sk_{ID_a})$.
6. **ProxySign**: Input system's parameter Para , the warrant W , the delegation σ_w , the secret key sk_{ID_b} of the proxy signer and the message M to be signed, this algorithm generates the proxy signature σ . That is: $\sigma \leftarrow \text{ProxySign}(\text{Para}, W, \sigma_w, sk_{ID_b}, M)$.
7. **ProxyVer**: Input system's parameter Para , original signer's identity ID_a , proxy signer's identity ID_b , the warrant W , the signed message M and the signature σ , this algorithm outputs **True** if σ is a valid proxy signature of the message M and outputs $\perp$ otherwise. That is: $\{\text{True}, \perp\} \leftarrow \text{ProxyVer}(\text{Para}, ID_a, ID_b, W, M, \sigma)$.

3.1 Security Models

In [26], Xu, Zhang and Feng proposed the first formal security model of identity-based proxy signature. Actually, this model is a variation of the security model in the traditional public key system which is proposed in [4,16]. While their model provides some properties that an identity-based proxy signature schemes should capture, there are two weaknesses of the models defined in [26].

1. The attacker $\mathcal{A}$'s target identity ID_1 is given to $\mathcal{A}$ before $\mathcal{A}$ submits queries to the challenger. However, we normally allow $\mathcal{A}$ to choose the target identity adaptively after he received responses of all the queries.
2. When $\mathcal{A}$ outputs a forgery signature $(m, psign)$ under the proxy signing key skp (which is defined in their scheme), with original signer ID_1 and the proxy signer ID_i , $\mathcal{A}$ cannot request the proxy signatures of other messages under this proxy signing key. However, we normally allow $\mathcal{A}$ to obtain the signatures of the message m' under this signing key with the only restriction that $m \neq m'$.

Similarly to the model defined in [10], we divide the potential adversary into the following three types:

1. **Type I**: This type adversary $\mathcal{A}_I$ only has the public keys (identities) of Alice and Bob.
2. **Type II**: This type adversary $\mathcal{A}_{II}$ has the public keys (identities) of Alice and Bob, and also can have the secret key of the proxy signer Bob.
3. **Type III**: This type adversary $\mathcal{A}_{III}$ has the public keys (identities) of Alice and Bob, and also can have the secret key of the original signer Alice.

One can find that if an ID-based proxy signature scheme is secure against Type II (or Type III) adversary, the scheme is also secure against Type I adversary. We

note the above classification helps to make the security model clearer; therefore, we will use this classification to improve the security model proposed in [26]. In a warrant based proxy signature, the delegation is the original signer's standard signature on the warrant which contains information regarding the proxy signer such as the proxy signer's ID, a period of validity, the restriction on the class of messages for which the warrant is valid. Therefore, this kind of proxy signature can prevent the misuse of the delegation. Here after, we only focus on the unforgeability of the proxy signature.

Existential unforgeability under adaptive $\mathcal{A}_{II}$ Adversary

Roughly speaking, a valid ID-based proxy signature σ of the message M under the warrant W shows that the original signer agrees on this warrant and has signed this warrant. Therefore, even the adversary can obtain the secret key of the proxy signer, he cannot create a valid ID-based proxy signature under the warrant W if he does not obtain the delegation of this warrant. It is defined using the following game between the challenger $\mathcal{C}$ and a type II adversary $\mathcal{A}_{II}$:

- **Setup:** $\mathcal{C}$ runs the ParaGen algorithm to obtain system's parameter para and the master key s .
- **KeyExtract queries:** Given an identity ID , $\mathcal{C}$ returns the private key sk_{ID} corresponding to ID .
- **StandSign queries:** $\mathcal{A}_{II}$ can request the signature of M under the identity ID . In response, $\mathcal{C}$ runs the StandSign algorithm to obtain σ_S and returns σ_S to the adversary $\mathcal{A}_{II}$. Especially, $\mathcal{A}_{II}$ can request the original signer ID_A 's delegation (that is the standard signature) on (W, ID_A, ID_B) where W denotes the warrant, ID_B denotes the proxy signer's identity and W, ID_A, ID_B are chosen by $\mathcal{A}_{II}$ adaptively. In response, $\mathcal{C}$ runs the StandSign algorithm to sign the message (W, ID_A, ID_B) to generate σ_W . Then $\mathcal{C}$ returns σ_W to the adversary $\mathcal{A}_{II}$.
- **ProxySign queries:** Proceeding adaptively, $\mathcal{A}_{II}$ can request the proxy signature of (W, M, ID_A, ID_B) where W is the warrant, M is the message to be signed, ID_A is the original signer's identity and ID_B is the proxy signer's identity. In response, $\mathcal{C}$ firstly runs the KeyExtract algorithm to obtain the secret keys of the original signer and proxy signer, respectively. Then $\mathcal{C}$ runs the StandSign algorithm to sign the message (W, ID_A, ID_B) and generates the delegation σ_W . At last, $\mathcal{C}$ runs the ProxySign algorithm and generates the proxy signature σ . Then $\mathcal{C}$ returns σ to the adversary $\mathcal{A}_{II}$ as response.
- **Output:** Finally, $\mathcal{A}_{II}$ outputs $(M^*, W_f, \overline{ID}_A, \overline{ID}_B, \sigma^*)$ where $\overline{ID}_A$ is the identity of original signer, $\overline{ID}_B$ is the identity of proxy signer, W_f is the warrant, M^* is the message and σ^* is the signature which satisfy that:
 1. $\overline{ID}_A$ has not been requested as one of the KeyExtract queries.
 2. $(W_f, \overline{ID}_A, \overline{ID}_B)$ has not been requested as one of the StandSign queries.
 3. $(M^*, W_f, \overline{ID}_A, \overline{ID}_B)$ has not been requested as one of the ProxySign queries.
 4. σ^* is a valid ID-based proxy signature of the message m^* under the warrant W_f , the original signer $\overline{ID}_A$ and the proxy signer $\overline{ID}_B$.

Remark: Compared with the model defined in [26], an important refinement is that we allow $\mathcal{A}_{II}$ to adaptively submit the ProxySign queries under the warrant whose delegation is unknown to him. The only restrictions are that when $\mathcal{A}_{II}$ outputs the forgery $(M^*, W_f, \overline{ID}_A, \overline{ID}_B, \sigma^*)$, he cannot submit $\overline{ID}_A$ as one of the KeyExtract queries, or $(W_f, \overline{ID}_A, \overline{ID}_B)$ as one of the StandSign queries, or submit $(M^*, W_f, \overline{ID}_A, \overline{ID}_B, \sigma^*)$ as one of the ProxySign queries. However, he can even submit $\overline{ID}_B$ to the KeyExtract queries, $(W'_f, \overline{ID}_A, \overline{ID}_B)$ to the StandardSign queries where $W'_f \neq W_f$ and $(M', W_f, \overline{ID}_A, \overline{ID}_B)$ to the ProxySign queries where $M' \neq M^*$.

The success probability of an algorithm $\mathcal{A}_{II}$ wins the above game is defined as $\text{Succ}_{\mathcal{A}_{II}}$.

Definition 3. We say a type II adversary $\mathcal{A}_{II}$ can $(t, q_H, q_{KE}, q_S, q_{PS}, \varepsilon)$ break a proxy signature scheme if $\mathcal{A}_{II}$ runs in time at most t , $\mathcal{A}_{II}$ makes at most q_H queries to the hash functions, at most q_{KE} KeyExtract queries, at most q_S StandardSign queries and at most q_{PS} ProxySign queries, and $\text{Succ}_{\mathcal{A}_{II}}$ is at least ε .

Existential unforgeability under adaptive $\mathcal{A}_{III}$ adversary

Roughly speaking, this property states that only the proxy signer can create a valid proxy signature, even the original signer can not. Given a valid ID-based proxy signature, the proxy signer cannot deny the fact that he has signed the message. The existential unforgeability of a proxy signature scheme under a type III attacker requires that it is difficult for the original signer to output a valid proxy signature of a message M^* which has not been signed by the proxy signer. It is defined using the games as same as those games between $\mathcal{A}_{II}$ and $\mathcal{C}$. After all the queries,

- **Output:** Finally, $\mathcal{A}_{III}$ outputs $(M^*, W_f, \overline{ID}_A, \overline{ID}_B, \sigma^*)$ where $\overline{ID}_A$ is the identity of original signer, $\overline{ID}_B$ is the identity of proxy signature, W_f is the warrant, M^* is the message to be signed and σ^* is the ID-based proxy signature which satisfy that:
 1. $\overline{ID}_B$ has not been requested as one of the KeyExtract queries.
 2. $(M^*, W_f, \overline{ID}_A, \overline{ID}_B, \sigma^*)$ has not been requested as one of the ProxySign queries.
 3. σ^* is a valid ID-based proxy signature of the message m^* under the warrant W_f , the original signer $\overline{ID}_A$ and the proxy signer $\overline{ID}_B$.

The success probability of an algorithm $\mathcal{A}_{III}$ wins the above game is defined as $\text{Succ}_{\mathcal{A}_{III}}$.

Definition 4. We say a type III adversary $\mathcal{A}_{III}$ can $(t, q_H, q_{KE}, q_S, q_{PS}, \varepsilon)$ break a proxy signature scheme if $\mathcal{A}_{III}$ runs in time at most t , $\mathcal{A}_{III}$ makes at most q_H queries to the hash functions, at most q_{KE} KeyExtract queries, at most q_S StandardSign queries and at most q_{PS} ProxySign queries, and $\text{Succ}_{\mathcal{A}_{III}}$ is at least ε .

4 Proposed ID-Based Proxy Signature Scheme

In this section, we present our construction of ID-based proxy signature scheme. The scheme consists of the following algorithms.

1. **ParaGen**: Input the system's parameter ℓ , this algorithm generates a bilinear group $\mathbb{G}_1$ of prime order q ($q \geq 2^\ell$) such that CDH problem is hard in $\mathbb{G}_1$. Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ be the bilinear pairing. The generator of $\mathbb{G}_1$ is P . Pick a random master key $s \in \mathbb{Z}_q^*$ and set $P_{pub} = sP$. It also chooses three distinct secure hash functions $H_0, H_1, H_2 : \{0, 1\}^* \rightarrow \mathbb{G}_1$. Then the system's parameter is: $\text{Para} = \{\ell, \mathbb{G}_1, \mathbb{G}_T, q, e, P, P_{pub}, H_0, H_1, H_2\}$.
2. **KeyExtract**: Given a user's identity ID , compute $H_0(ID) \in \mathbb{G}_1$ and $sk_{ID} = sH_0(ID)$.
3. **StandardSign**: Let M be the message to be signed, the standard signature is generated as: $\sigma_S = (sk_{ID} + rH_1(M), rP)$ where $r \in_R \mathbb{Z}_q^*$.
4. **StandardVer**: Given the identity ID of the signer, the message M and a signature σ_S , verify whether $e(\sigma_S, P) \stackrel{?}{=} e(H_0(ID), P_{pub})e(H_1(M), rP)$.
5. **DelegationGen**: Let W be the warrant to be signed by the original signer Alice with the identity ID_A who wants to delegate his signing rights to Bob with the identity ID_B , the delegation is generated as: $\sigma_W = (sk_{ID_A} + r_A H_1(W, ID_A, ID_B), r_A P)$ where $r_A \in_R \mathbb{Z}_q^*$. Then Alice sends the warrant W and delegation σ_W to the proxy signer Bob.
6. **ProxySign**: Given the secret key sk_{ID_B} , the delegation $\sigma_W = (sk_{ID_A} + r_A H_1(W, ID_A, ID_B), r_A P)$ of the warrant W and a message M , the proxy signer chooses $r_B \in_R \mathbb{Z}_q^*$ and computes $\sigma = (\sigma_1, \sigma_2, \sigma_3)$ where $\sigma_1 = sk_{ID_A} + r_A H_1(W, ID_A, ID_B) + sk_{ID_B} + r_B H_2(M, W, ID_A, ID_B)$, $\sigma_2 = r_A P$, $\sigma_3 = r_B P$.
7. **ProxyVer**: Given the identities (ID_A, ID_B) of original signer and proxy signer, a warrant $W \in \{0, 1\}^*$, a message $M \in \{0, 1\}^*$, and a signature $\sigma = (\sigma_1, \sigma_2, \sigma_3)$, verify whether $e(\sigma_1, P) \stackrel{?}{=} e(H_0(ID_A), P_{pub})e(H_0(ID_B), P_{pub})e(H_1(W, ID_A, ID_B), \sigma_2)e(H_2(M, W, ID_A, ID_B), \sigma_3)$. If the equality holds the result is **True**; otherwise the result is $\perp$.

4.1 Unforgeability Against Type II Adversary

Theorem 1. *If there exists a type II adversary $\mathcal{A}_{II}$ who can $(t, q_H, q_{KE}, q_S, q_{PS}, \varepsilon)$ break the proposed proxy signature scheme then there exists another algorithm $\mathcal{B}$ who can use $\mathcal{A}_{II}$ to solve an instance of the CDH problem in $\mathbb{G}_1$ with probability*

$$\text{Succ}_{\mathcal{B}, \mathbb{G}_1}^{CDH} \geq \left(\frac{3}{q_{KE} + q_S + q_{PS} + 1} \right)^3 \left(1 - \frac{3}{q_{KE} + q_S + q_{PS} + 4} \right)^{q_{KE} + q_S + q_{PS} + 4} \varepsilon$$

in time $t + c_1(q_H + q_{KE} + 3q_S + 9q_{PS}) + c_2(q_H + 2q_S + 8q_{PS})$. Here c_1, c_2 are two constants that depends on $\mathbb{G}_1$.

Proof. We are forced to omit it due to page limitation.

4.2 Unforgeability Against Type III Adversary

Theorem 2. *If there exists a type III adversary $\mathcal{A}_{III}$ who can $(t, q_H, q_{KE}, q_S, q_{PS}, \varepsilon)$ break the proposed proxy signature scheme then there exists another algorithm $\mathcal{B}$ who can use $\mathcal{A}_{III}$ to solve an instance of the CDH problem in $\mathbb{G}_1$ with probability*

$$\text{Succ}_{\mathcal{B}, \mathbb{G}_1}^{CDH} \geq \left(\frac{3}{q_{KE} + q_S + q_{PS} + 1}\right)^3 \left(1 - \frac{3}{q_{KE} + q_S + q_{PS} + 4}\right)^{q_{KE} + q_S + q_{PS} + 4} \varepsilon$$

in time $t + c_1(q_H + q_{KE} + 3q_S + 9q_{PS}) + c_2(q_H + 2q_S + 8q_{PS})$. Here c_1, c_2 are two constants that depends on $\mathbb{G}_1$.

Proof. The proof is completely similar to the proof of Theorem 1.

4.3 Efficiency Analysis

In this section we compare our scheme with Xu *et al.*'s scheme [26] in the sense of signature length and operation cost of verification. The two schemes require the same operation cost in the delegation and proxy sign algorithms. In the following table, the notion $|\mathbb{G}_1|$ denotes the bit length of an element in $\mathbb{G}_1$.

Table 1. Comparison between Xu *et al.*'s scheme [26] and our scheme

Scheme	Signature Length	Pairings in Verification	exp. in $\mathbb{G}_2$
Xu <i>et al.</i> 's scheme	$3 \mathbb{G}_1 $	4 (2 can be precomputed)	1
Our scheme	$3 \mathbb{G}_1 $	4 (2 can be precomputed)	0

5 Conclusion

In this paper, we improve the security models of identity based proxy signature defined by Xu, Zhang and Feng [26] by allowing adversaries to behave more adaptively in oracle accessing. We then propose a new identity based proxy signature which is secure against adaptively chosen message and chosen identity attacker. Compared with the scheme proposed in [26], the new proposed scheme enjoys less operation cost, and hence, it outperforms the existing schemes in the literature. The security of the proposed scheme is based on the hardness of the Computational Diffie-Hellman problem in the random oracle model.

References

1. Boneh, D., Boyen, X.: Short Signatures without Random Oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)

2. Boneh, D., Franklin, M.: Identity-based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
3. Boneh, D., Lynn, B., Shacham, H.: Short Signatures from the Weil Pairing. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 514–532. Springer, Heidelberg (2001)
4. Boldyreva, A., Palacio, A., Warinschi, B.: Secure Proxy Signature Scheme for Delegation of Signing Rights. In: IACR ePrint Archive (2003) available at <http://eprint.iacr.org/2003/096/>
5. Bellare, M., Rogaway, P.: The Exact Security of Digital Signatures - How to Sign with RSA and Rabin. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 399–416. Springer, Heidelberg (1995)
6. Cheon, J.H.: Security Analysis of the Strong Diffie-Hellman Problem. In: Vaude- nay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 1–11. Springer, Heidelberg (2006)
7. Canetti, R., Goldreich, O., Halevi, S.: The Random Oracle Methodology, revisited. In: Proceedings of the 30th Annual Symposium on the Theory of Computing- STOC'98, pp. 209–218 (1998)
8. Goldwasser, S., Micali, S., Rivest, R.: A Digital Signature Scheme Secure Against Adaptively Chosen Message Attacks. SIAM Journal on Computing 17(2), 281–308 (1988)
9. Galbraith, S.D., Paterson, K.G., Smart, N.P.: Pairings for Cryptographers. In: IACR ePrint Archive (2006) available at <http://eprint.iacr.org/2006/165/>
10. Huang, X., Mu, Y., Susilo, W., Zhang, F., Chen, X.: A Short Proxy Signature Scheme: Efficient Authentication in the Ubiquitous World. In: Enokido, T., Yan, L., Xiao, B., Kim, D., Dai, Y., Yang, L.T. (eds.) Embedded and Ubiquitous Computing – EUC 2005 Workshops. LNCS, vol. 3823, pp. 480–489. Springer, Heidelberg (2005)
11. Lee, J.-Y., Cheon, J.H., Kim, S.: An Analysis of Proxy Signatures: Is a Secure Channel Necessary? In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 68–79. Springer, Heidelberg (2003)
12. Lee, B., Kim, H., Kim, K.: Strong Proxy Signature and Its Applications. In: Proc of SCIS'01, pp. 603–608 (2001)
13. Lee, B., Kim, H., Kim, K.: Secure Mobile Agent Using Strong Nondesignated Proxy Signature. In: Varadharajan, V., Mu, Y. (eds.) ACISP 2001. LNCS, vol. 2119, pp. 474–486. Springer, Heidelberg (2001)
14. Kim, S., Park, S., Won, D.: Proxy Signatures, revisited. In: Han, Y., Quing, S. (eds.) ICICS 1997. LNCS, vol. 1334, pp. 223–232. Springer, Heidelberg (1997)
15. Hess, F.: Efficient Identity Based Signature Schemes Based on Pairings. In: Nyberg, K., Heys, H.M. (eds.) SAC 2002. LNCS, vol. 2595, pp. 310–324. Springer, Heidelberg (2003)
16. Malkin, T., Obana, S., Yung, M.: The Hierarchy of Key Evolving Signatures and a Characterization of Proxy Signatures. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 306–322. Springer, Heidelberg (2004)
17. Mambo, M., Usuda, K., Okamoto, E.: Proxy Signature: Delegation Of the Power to Sign Messages. IEICE Trans. Fundamentals E79-A(9), 1338–1353 (1996)
18. Okamoto, T., Inomata, A., Okamoto, E.: A Proposal of Short Proxy Signature Using Pairing. In: International Conference on Information Technology (ITCC 2005), pp. 631–635. IEEE Computer Society Press, Los Alamitos (2005)
19. Okamoto, T., Tada, M., Okamoto, E.: Extended Proxy Signatures for Smart Cards. In: Zheng, Y., Mambo, M. (eds.) ISW 1999. LNCS, vol. 1729, pp. 247–258. Springer, Heidelberg (1999)

20. Paterson, K.G., Schuldt, J.C.N.: Efficient Identity-based Signatures Secure in the Standard Model. In: Batten, L.M., Safavi-Naini, R. (eds.) ACISP 2006. LNCS, vol. 4058, pp. 207–222. Springer, Heidelberg (2006)
21. Park, H.-U., Lee, I.-Y.A.: Digital Nominative Proxy Signature Scheme for Mobile Communications. In: Qing, S., Okamoto, T., Zhou, J. (eds.) ICICS 2001. LNCS, vol. 2229, pp. 451–455. Springer, Heidelberg (2001)
22. Shamir, A.: Identity-based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
23. Wang, G., Bao, F., Zhou, J., Deng, R.H.: Security Analysis of Some Proxy Signatures. In: Lim, J.-I., Lee, D.-H. (eds.) ICISC 2003. LNCS, vol. 2971, pp. 305–319. Springer, Heidelberg (2004)
24. Zhang, F., Safavi-Naini, R., Susilo, W.: An Efficient Signature Scheme from Bilinear Pairings and Its Applications. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 277–290. Springer, Heidelberg (2004)
25. Zhang, F., Susilo, W., Mu, Y.: Identity-Based Partial Message Recovery Signatures (or How to Shorten ID-Based Signatures). In: Patrick, A.S., Yung, M. (eds.) FC 2005. LNCS, vol. 3570, pp. 45–56. Springer, Heidelberg (2005)
26. Xu, J., Zhang, Z., Feng, D.: ID-based Proxy Signature Using Bilinear Pairings. In: Chen, G., Pan, Y., Guo, M., Lu, J. (eds.) Parallel and Distributed Processing and Applications - ISPA 2005 Workshops. LNCS, vol. 3759, pp. 359–367. Springer, Heidelberg (2005)

Cryptanalysis of BGW Broadcast Encryption Schemes for DVD Content Protection^{*}

Qianhong Wu¹, Willy Susilo¹, Yi Mu¹, and Bo Qin^{2,3}

¹ Centre for Computer and Information Security Research
University of Wollongong, Wollongong NSW 2522, Australia
{qhw,wsusilo,ymu}@uow.edu.au

² National Key Laboratory of Integrated Service Networks
Xidian University, Xi'an, China

³ Department of Mathematics, School of Science,
Xi'an University of Technology, Xi'an, China
qinboo@xaut.edu.cn

Abstract. Security systems should not only be correctly devised but also be correctly used. In Crypto 2005, Boneh, Gentry and Waters (BGW) proposed two efficient broadcast encryption schemes proven secure in their security definition. They also suggested for a number of applications of their schemes including satellite TV subscription services and DVD content protections. In contrast to this suggestion, we show that any legitimate decoder(s) can collude with the revoked decoders to produce exponentially many equivalent decryption keys, and moreover, this activity cannot be traced by the dealer. Our results remind of abuse that their schemes are *not* suitable for the satellite TV subscription services or DVD content protection applications, although their schemes may be applicable in trusted environments such as conference key distribution.

1 Introduction

A broadcast encryption scheme [6] allows one to broadcast a confidential message to some subset S of n users who are listening to a broadcast channel. The sender has an encryption key and each receiver is equipped with a decryption (private) key, such that the sender can encrypt a message and broadcast the ciphertext in which any authorized receivers can decrypt the ciphertext. However, even if all users outside of S collude, they can obtain no information about the contents of the broadcast. Broadcast encryption schemes have wide applications such as conference key distribution, online web broadcasting, pay-per-view systems and DVD content protection, in which the system end broadcasts messages to a set of privileged receivers through a broadcast channel. The broadcast content is protected via the encryption scheme, and only legitimate users who possess the information (i.e. the decryption keys) can access the content.

^{*} This work is supported by ARC Discovery Grant of Australia under Grant No. DP0557493 and National Natural Science Foundation of China under Grant No. 60473027.

Consider a situation where a content supplier distributes digital contents to its subscribers via a broadcast channel. To protect the data from eavesdropping, the content supplier encrypts the data and broadcasts the ciphertext such that only its subscribers can decrypt the ciphertext. The content supplier gives each subscriber a decoder (i.e. a decoding box) for decrypting the ciphertext. Each decoder consists of a tailored key and a decryption program. These keys can be embedded in the software or in tamper-resistant hardware devices such as smart cards. Nonetheless, experience shows that adversaries attack broadcast encryption systems in a variety of different ways. Their attacks may be on the hardware that stores cryptographic keys, e.g., when they extract keys from a compliant device to develop a pirate device such as the DeSCC software that circumvents the content scrambling system [2]. Existing tamper-resistant hardware is vulnerable to a variety of attacks [1]. Thus, authorized users can essentially extract decryption keys from a legitimate software or hardware decoder. These users can then circumvent the security of the system by employing the compromised keys to generate new decryption keys to be distributed to unauthorized users and sell the pirate decoders for profits.

Motivated by the commercial profits, the malicious subscriber may modify the private key and the decryption program in the pirate decoder to avoid his identity exposure. Furthermore, some *traitors* may collaboratively create new private keys and decryption programs. Hence, when a broadcast encryption scheme is used in commerce applications such as pay TV or DVD content protection, it must prevent such pirate attacks, in the following sense: after obtaining the pirate decoder, the content supplier can extract *at least* one of the subscriber. In this paper, we refer the authorized users who illegally extract and distribute decryption keys as *traitors*, and the unauthorized users who unfairly obtain the keys as *pirates*. The illegal decoder software or hardware devices created by the traitors are pirate decoders.

Fiat and Naor [6] were the first to formally explore broadcast encryption. They presented a solution which was secure against a collusion of t users and has ciphertext size of $O(t \log_2 t \log n)$. Naor *et al.* [11] presented a fully collusion secure broadcast encryption system that is efficient for broadcasting to all, but a small set of revoked users. Their scheme is useful for content protection where broadcasts will be sent to all, but a small set of receivers whose keys have been compromised. Their scheme can be used to encrypt up to $n-r$ users with a header size of $O(r)$ elements and private keys of size $O(\log_2 n)$. Further improvements [8,7] reduce the private key size to $O(\log n)$. Dodis and Fazio [5] extend the NNL (subtree difference) method into a public key broadcast system for a small size public key.

In Crypto 2005, Boneh, Gentry and Water (BGW) [4] proposed two efficient broadcast encryption schemes proven secure in their security definition. These schemes were suggested for a number of applications including satellite TV subscription services and DVD content protection. In this paper, we show that any legit decoder(s) collude with the revoked decoders can produce exponentially many equivalent decryption keys. Moreover, given these equivalent keys,

the dealer cannot find any one of the inner and outsider decoders who have been involved in the pirate behaviors (and hence, *no* tracing mechanism is possible). Hence, their broadcast encryption schemes are not suitable for satellite TV subscription services or DVD content protection. They are only suitable for applications of trusted environment such as conference key distribution.

2 Preliminaries

2.1 Bilinear Maps

The BGW broadcast encryption schemes are based on bilinear pairings. We briefly review the necessary facts about bilinear maps and bilinear map groups. We follow the standard notation [10, 9, 3]. Let G and G_1 be two (multiplicative) cyclic groups of prime order p , g be a generator of G and $e : G \times G \rightarrow G_1$ be a bilinear map with the following properties:

1. Bilinear: for all $u, v \in G$ and $a, b \in \mathbb{Z}$, we have $e(u^a, v^b) = e(u, v)^{ab}$.
2. Non-degenerate: $e(g, g) \neq 1$.

We say that G is a bilinear group if the group action in G can be computed efficiently and there exists a group G_1 and an efficiently computable bilinear map $e : G \times G \rightarrow G_1$ as above. Note that $e(\cdot, \cdot)$ is symmetric since $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

2.2 Broadcast Encryption Systems

We briefly recall the definition of broadcast encryption in [4]. For simplicity, a broadcast encryption scheme is defined as a key encapsulation mechanism. A broadcast encryption system consists of three randomized algorithms:

Setup(n). Takes as input the number of receivers n . It outputs n private keys $d_1, \dots, d_n$ and a public key PK .

Encrypt(S, PK). Takes as input a subset $S \subseteq \{1, \dots, n\}$, and a public key PK . It outputs a pair (Hdr, K) where Hdr is called the header and $K \in \mathcal{K}$ is a message encryption key (or a session key). We will often refer to Hdr as the broadcast ciphertext.

Let M be a message to be broadcast to the set S and let CM be the encryption of M under the symmetric key K . The broadcast to users in S consists of (S, Hdr, CM) . The pair (S, Hdr) is often called the full header and CM is often called the broadcast body.

Decrypt(S, i, d_i, Hdr, PK). Takes as input a subset $S \subseteq \{1, \dots, n\}$, a user index $i \in \{1, \dots, n\}$ and the private key d_i for user i , a header Hdr , and the public key PK . If $i \in S$, then the algorithm outputs the message encryption key $K \in \mathcal{K}$. The key K can then be used to decrypt the broadcast body CM and obtain the message body M .

As usual, the system must be correct, i.e., for all $S \subseteq \{1, \dots, n\}$ and all $i \in S$, if $(PK, (d_1, \dots, d_n)) \leftarrow \text{Setup}(n)$ and $(Hdr, K) \leftarrow \text{Encrypt}(S, PK)$ then $\text{Decrypt}(S, i, d_i, Hdr, PK) = K$.

The most important security notion of a broadcast encryption scheme is the IND-CCA (indistinguishability against a chosen message attack) security. In this notion, the attacker chooses a subset of $\{1, \dots, n\}$ and the dealer produces the public parameters and sends them to the attacker. The attacker is also given the decryption keys except its chosen subset and allowed to query the broadcast encryption oracle and decryption oracles where the intended receivers are a subset of its chosen subset. Then the attacker chooses two messages and the dealer will encrypt one of them to the attacker's chosen subset. After new queries to the broadcast encryption oracle and decryption oracles (note the challenge ciphertext is not allowed to be queried) where the intended receivers are a subset of its chosen subset, the attacker has to answer which message has been encrypted. A broadcast encryption is said to be IND-CCA *secure* if any polynomial time attacker can only win the above game with negligible probability. The two BGW broadcast encryption schemes have been proven secure in this definition.

2.3 Equivalent Key Attack on Broadcast Encryption

Informally, the IND-CCA security states that, given the keys of all the revoked users of a broadcast system, one *cannot* extract any information from a broadcast ciphertext. This security notion is sufficient in trust settings such as conference key distribution. However, when broadcast encryptions are used for pay TV or DVD content protection, the above IND-CCA security is not sufficient. In this scenario, the attacker may mount more powerful attacks by colluding with legit (inner) users provided that anyone of the inner (legit) or outside (revoked) malicious users cannot be found by the broadcaster and the dealer who setups the broadcast encryption system. If such an attack is successful, then the attacker can produce a pirate decoder and sell it at a lower price while the attacker cannot be found (traced). We call such an attack as an *equivalent key attack*.

Now we describe the equivalent key attack on broadcast encryption as follows. The attacker takes input the decryption keys of a number of legit users S' and the revoked users, it outputs a decryption key dk' and a description of a polynomial time algorithm $\mathcal{D}'(\cdot)$ satisfying the following conditions:

1. There exists a set S'' , for any S and Hdr , it holds that $\mathcal{D}'(S, dk', Hdr, PK) = K$ if $S' \subseteq S'' \subseteq S$.
2. Given the pirate decoder as a black box, any polynomial time algorithm cannot find $i \in S'$ with probability non-negligibly more than $1/|S''|$, where $|S''|$ is cardinality of the set S'' .

Now, the above attacker can be served as a pirate decoder of the broadcast encryption when it is used in Pay TV or DVD content protection.

3 Review BGW Broadcast Encryption Schemes

In Crypto 2005, Boneh *et al.* proposed two very efficient broadcast encryption schemes and suggested the applications of their schemes for protecting Pay TV and DVD content protection. In this section, we briefly review their schemes.

3.1 Basic BGW Broadcast Encryption

The basic BGW broadcast encryption system is for n users where the ciphertexts and private keys are of constant size, but the public key size grows linearly in the number of users.

Setup(n): Let G be a bilinear group of prime order p . The algorithm first picks a random generator $g \in G$ and a random $\alpha \in \mathbb{Z}_p$. It computes $g_i = g^{\alpha^i} \in G$ for $i = 1, 2, \dots, n, n+2, \dots, 2n$. Next, it picks a random $\gamma \in \mathbb{Z}_p$ and sets $v = g^\gamma \in G$. The public key is:

$$PK = (g, g_1, \dots, g_n, g_{n+2}, \dots, g_{2n}, v) \in G^{2n+1}$$

The private key for user $i \in \{1, \dots, n\}$ is set as $d_i = g_i^\gamma \in G$. Note that $d_i = v^{\gamma^i}$. The algorithm outputs the public key PK and the n private keys $d_1, \dots, d_n$.

Encrypt(S, PK): Pick a random r in $\mathbb{Z}_p$ and set $K = e(g_{n+1}, g)^r \in G$. The value $e(g_{n+1}, g)$ can be computed as $e(g_n, g_1)$. Next, set

$$Hdr = (g^r, (v \prod_{j \in S} g_{n+1-j})^r) \in G^2$$

and output the pair (Hdr, K) .

Decrypt(S, i, d_i, Hdr, PK): Let $Hdr = (C_0, C_1)$ and recall that $d_i \in G$. Then, output

$$K = e(g_i, C_1) / e(d_i \prod_{j \in S, j \neq i} g_{n+1-j+i}, C_0)$$

A private key is only one group element in G and the ciphertext, Hdr , is only two group elements. $e(g_{n+1}, g)$ can be precomputed. The system is able to broadcast to any subset of users. Hence, the BGW scheme is very efficient.

3.2 General BGW Broadcast Encryption

Next, we recall the general BGW broadcast encryption system [4]. Their idea is to run A parallel instances of the system in the previous section where each instance can broadcast to at most $B \ll n$ users. As a result they can handle as many as $n = AB$ users. The performance was improved by sharing information between the A instances. In particular, all instances share the same public key values $g, g_1, \dots, g_B, g_{B+2}, \dots, g_{2B}$.

Their generalized system trades off the public key size for ciphertext size. Setting $B = n$ gives the system of the previous section. However, setting $B = \lfloor \sqrt{n} \rfloor$ gives a system where both public key and ciphertext size are about $\sqrt{n}$ elements. Note that either way, the private key is always just one group element. Let B be a fixed positive integer. The B -broadcast encryption system works as follows:

Setup $_B(n)$: The algorithm will set up $A = \lceil \frac{n}{B} \rceil$ instances of the scheme. Let G be a bilinear group of prime order p . The algorithm first picks a random generator $g \in G$ and a random $\alpha \in \mathbb{Z}_p$. It computes $g_i = g^{(\alpha^i)} \in G$ for $i = 1, 2, \dots, B, B+2, \dots, 2B$. Next, it picks random $\gamma_1, \dots, \gamma_A \in \mathbb{Z}_p$ and sets $v_1 = g^{\gamma_1}, \dots, v_A = g^{\gamma_A} \in G$. The public key is:

$$PK = (g, g_1, \dots, g_B, g_{B+2}, \dots, g_{2B}, v_1, \dots, v_A) \in G^{2B+A}$$

The private key for user $i \in \{1, \dots, n\}$ is defined as follows: write i as $i = (a-1)B + b$ for some $1 \leq a \leq A$ and $1 \leq b \leq B$ (i.e., $a = \lceil \frac{i}{B} \rceil$ and $b = i \bmod B$). Set the private key for user i as:

$$d_i = g_b^{\gamma_a} \in G \text{ (note that } d_i = v_a^{\alpha^b} \text{)}$$

The algorithm outputs the public key PK and the n private keys $d_1, \dots, d_n$.

Encrypt (S, PK) : For each $\ell = 1, \dots, A$, define the subsets $\hat{S}_\ell$ and S_ℓ as

$$\begin{aligned} \hat{S}_\ell &= S \cap \{(\ell-1)B + 1, \dots, \ell B\} \text{ and} \\ S_\ell &= \{x - \ell B + B \mid x \in \hat{S}_\ell\} \subseteq \{1, \dots, B\} \end{aligned}$$

In other words, $\hat{S}_\ell$ contains all users in S that fall in the ℓ -th interval of length B and S_ℓ contains the indices of those users relative to the beginning of the interval. Pick a random t in $\mathbb{Z}_p$ and set $K = e(g_{B+1}, g)^t \in G$. As before, the value $e(g_{B+1}, g)$ can be computed as $e(g_B, g_1)$. Set

$$Hdr = (g^t, (v_1 \prod_{j \in S_1} g_{B+1-j})^t, \dots, (v_A \prod_{j \in S_A} g_{B+1-j})^t) \in G^{A+1}$$

Output the pair (Hdr, K) . Note that Hdr contains $A+1$ elements.

Decrypt (S, i, d_i, Hdr, PK) : Let $Hdr = (C_0, C_1, \dots, C_A)$ and recall that $d_i \in G$. Write i as $i = (a-1)B + b$ for some $1 \leq a \leq A$ and $1 \leq b \leq B$. Then

$$K = e(g_b, C_a) / e(d_i \prod_{\substack{j \neq b \\ j \in S_a}} g_{B+1-j+b}, C_0)$$

Verifying that the decryption algorithm works correctly is analogous to the calculation in the previous section. When $B = n$ then $A = 1$, it is the system of the previous section.

4 Equivalent Key Attacks on BGW Broadcast Encryption Schemes

In this section, we present the equivalent key attacks on the BGW broadcast encryption schemes.

4.1 Equivalent Key Attack on Basic BGW Scheme

Let inner users $\mathbb{T} = \{i_1, \dots, i_t\} \subset S \subseteq \{1, \dots, n\}$ and revoked users $\mathbb{R} = \{i_{t+1}, \dots, i_{t+k}\} \subset \{1, \dots, n\}$ collude to produce a pirate decoder, where $t > 1$, and $i_{t+1}, \dots, i_{t+k} \notin S$. They work as follows.

Equivalent Key Generation: For $i \in \mathbb{S}$, randomly choose $s_i \leftarrow \mathbb{Z}_p$ such that $\sum_{i \in \mathbb{T}} s_i = 1$. For $i \notin \mathbb{S}$, set $s_i = 1$. Compute

$$\bar{g} = \prod_{i \in \mathbb{T} \cup \mathbb{R}} g_i^{s_i}, \bar{d} = \prod_{i \in \mathbb{T} \cup \mathbb{R}} d_i^{s_i}, h_j = (\prod_{i \in \mathbb{T} \cup \mathbb{R}}^{i \neq j} g_{n+1-j+i})^{s_j}.$$

Equivalent Decryption: Initialize a set $S'' = \{1, \dots, n\}$ and set a threshold T where $t + k \ll T \ll n$. Upon receiving a ciphertext $Hdr = (C_0, C_1)$, the set of receivers S , update $S'' \leftarrow S'' \cap S$. If the cardinality of S'' is less than T , output *error*. Else output

$$K = e(\bar{g}, c_1) / e(\bar{d} \prod_{j \in S} h_j, c_0).$$

We show that equivalent decryption algorithm is correct.

$$\begin{aligned} e(\bar{g}, c_1) &= e(\prod_{i \in \mathbb{T} \cup \mathbb{R}} g_i^{s_i}, c_1) \\ &= e(g^{\sum_{i \in \mathbb{T} \cup \mathbb{R}} s_i \alpha^i}, (v \prod_{j \in S} g_{n+1-j})^r) \\ &= e(g^{\sum_{i \in \mathbb{T} \cup \mathbb{R}} s_i \alpha^i}, g^{r\gamma} g^{r \sum_{j \in S} \alpha^{n+1-j}}) \\ &= e(g^{\gamma \sum_{i \in \mathbb{T} \cup \mathbb{R}} s_i \alpha^i}, g)^r e(g^{\sum_{\ell=1}^k \alpha^{i_\ell} s_{i_\ell}}, g^{\sum_{j \in S} \alpha^{n+1-j}})^r \\ &= e(g^{\gamma \sum_{i \in \mathbb{T} \cup \mathbb{R}} s_i \alpha^i}, g)^r e(g^{\sum_{\ell=1}^k \alpha^{i_\ell} s_{i_\ell} \sum_{j \in S} \alpha^{n+1-j}}, g)^r \\ &= e(g^{\gamma \sum_{i \in \mathbb{T} \cup \mathbb{R}} s_i \alpha^i}, g)^r e(g^{\sum_{i \in \mathbb{T} \cup \mathbb{R}} s_i (\sum_{j \in S} \alpha^{n+1-j+i})}, g)^r \\ &= e(g^{\gamma \sum_{i \in \mathbb{T} \cup \mathbb{R}} s_i \alpha^i}, g)^r e(g^{\sum_{j \in S} \sum_{i \in \mathbb{T}} s_i \alpha^{n+1-j+i}}, g)^r e(g^{\sum_{j \in S} \sum_{i \in \mathbb{R}} s_i \alpha^{n+1-j+i}}, g)^r \\ e(\bar{d} \prod_{j \in S} h_j, c_0) &= e(\bar{d}, g^r) e(\prod_{j \in S} h_j, g^r) \\ &= e(\prod_{i \in \mathbb{T} \cup \mathbb{R}} d_i^{s_i}, g)^r e(\prod_{j \in S} (\prod_{i \in \mathbb{T} \cup \mathbb{R}}^{i \neq j} g_{n+1-j+i})^{s_j}, g)^r \\ &= e(\prod_{i \in \mathbb{T} \cup \mathbb{R}} g_i^{\gamma s_i}, g)^r e(\prod_{j \in S} (g^{\sum_{i \in \mathbb{T} \cup \mathbb{R}} s_i \alpha^{n+1-j+i}}), g)^r \\ &= e(\prod_{i \in \mathbb{T} \cup \mathbb{R}} g^{\gamma \alpha^i s_i}, g)^r e(g^{\sum_{j \in S} \sum_{i \in \mathbb{T}} s_i \alpha^{n+1-j+i}}, g)^r \\ &= e(g^{\gamma \sum_{i \in \mathbb{T} \cup \mathbb{R}} s_i \alpha^i}, g)^r e(g^{\sum_{j \in S} \sum_{i \in \mathbb{T}} s_i \alpha^{n+1-j+i}}, g)^r e(g^{\sum_{j \in S} \sum_{i \in \mathbb{R}} s_i \alpha^{n+1-j+i}}, g)^r \\ &= e(g^{\gamma \sum_{i \in \mathbb{T} \cup \mathbb{R}} s_i \alpha^i}, g)^r e(g^{\sum_{j \in S} \sum_{i \in \mathbb{T}} s_i \alpha^{n+1-j+i}}, g)^r e(g^{\sum_{j \in S} \sum_{i \in \mathbb{R}} s_i \alpha^{n+1-j+i}}, g)^r \end{aligned}$$

It follows that

$$\begin{aligned} e(\bar{g}, c_1) / e(\bar{d} \prod_{j \in S} h_j, c_0) &= e(g^{\sum_{i \in \mathbb{T}} s_i \alpha^{n+1}}, g)^r \\ &= e(g^{\alpha^{n+1}}, g)^{r \sum_{i \in \mathbb{T}} s_i} = e(g^{\alpha^{n+1}}, g)^r = K. \end{aligned}$$

Hence, the pirate decryption is valid. For simplicity, we denote the above pirate decoding algorithm by

$$\mathcal{D}'(S, \mathbb{T} \cup \mathbb{R}, dk(\mathbb{T} \cup \mathbb{R}, PK), Hdr(S, PK, K), PK),$$

where S is the set of legit receivers for the ciphertext $Hdr(S, PK, K)$ of the plaintext K under the public key PK . $\mathbb{T} \subseteq S$ is the set of traitors and $\mathbb{R}$ is the set of some revoked users. $dk(\mathbb{T} \cup \mathbb{R}, PK)$ is the equivalent decryption key corresponding to public key PK and the colluding users $\mathbb{T} \cup \mathbb{R}$.

Now, we show that the pirate decoder *cannot* be traced. We treat the tracing algorithm as a black box. From the pirate decoding algorithm, since the pirate decoder stops to work if the cardinality of S'' is less than T , any tracing algorithm can at most find a set $S'' \subseteq A$ with cardinality T , where $\mathbb{T} \cup \mathbb{R} \subset S''$ and $t + k \ll T$. The tracing algorithm cannot determine the colluding users $\mathbb{T} \cup \mathbb{R}$ including the legit users and the revoked users or determine a malicious user in $\mathbb{T} \cup \mathbb{R}$ with probability significantly more than $\frac{1}{T}$. Hence, the pirate decoder cannot be traced.

4.2 Equivalent Key Attack on General BGW Scheme

Note that the general BGW broadcast encryption can be viewed as A small basic broadcast encryptions: for $1 \leq a \leq A$, the B users share the same ciphertext (C_a, C_0) and the public key

$$PK_a = \{g, g_1, \dots, g_B, g_{B+2}, \dots, g_{2B}, V_a\} \in G^{2B+1}.$$

So we can realize the pirate attack on the general BGW broadcast encryption directly running the basic attack in the above subsection.

For $1 \leq a \leq A$, let inner users $\mathbb{T}_a = \{(a-1)B + b_1, \dots, (a-1)B + b_{t_a}\} \subset S \subseteq \{1, \dots, n\}$ and revoked users $\mathbb{R}_a = \{(a-1)B + b_{t_a+1}, \dots, (a-1)B + b_{k_a}\} \subset \{1, \dots, n\}$ collude to produce a pirate decoder, where $\mathbb{T}_a \neq \emptyset$ for at least one row $a \in \{1, \dots, A\}$. For any $a \in \{1, \dots, A\}$, if $i \in \mathbb{R}_a$ then $i \notin S$. Hence, the colluding users are in the set $\bigcup_{\mathbb{T}_a \neq \emptyset} (\mathbb{T}_a \cup \mathbb{R}_a)$ of $t + k$ users. Now the general pirate decoder working as follows by running the basic pirate decoder in the above section.

Equivalent Decryption: Initialize a set $S'' = \{1, \dots, n\}$ and set a threshold T , where $t + k \ll T \ll n$. When receiving a ciphertext $Hdr = (C_0, C_1, \dots, C_A)$, the set of receivers S , update $S'' \leftarrow S'' \cap S$. If the cardinality of S'' is less than T , Output *error*. Else randomly choose $a \in \{1, \dots, A\}$ satisfying that $\mathbb{T}_a \neq \emptyset$, run the basic pirate decoder:

$$K = \mathcal{D}(S, \mathbb{T}_a \cup \mathbb{R}_a, dk(\mathbb{T}_a \cup \mathbb{R}_a, PK_a), Hdr(S, PK_a, K), PK_a).$$

where $Hdr(S, PK_a, K) = (c_0, C_a)$.

The correctness and intractability of the attack on the general BGW broadcast encryption derive directly from the equivalent key attack on the basic BGW scheme and hence are omitted.

4.3 Equivalent Key Attack on DVD Content Protection Using BGW Broadcast Encryption

When the BGW scheme is applied to a DVD content protection scenario, the dealer who is the owner of the DVD content does not need to publish the public keys of the BGW broadcast encryption schemes. We rewrite the BGW broadcast encryption scheme for this scenario. For simplicity, we just rewrite the basic BGW scheme. For the general BGW scheme, it can be done similarly.

Setup(n): Let G be a bilinear group of prime order p . The algorithm first picks a random generator $g \in G$ and a random $\alpha \in \mathbb{Z}_p$. It computes $g_i = g^{\alpha^i} \in G$ for $i = 1, 2, \dots, n, n+2, \dots, 2n$. Next, it picks a random $\gamma \in \mathbb{Z}_p$ and sets $v = g^\gamma \in G$. The (private) encryption key of the dealer is:

$$EK = (g, v \prod_{j=1}^n g_{n+1-j}) \in G^2$$

Similarly, the private key for the subscriber $i \in \{1, \dots, n\}$ is set as $d_i = (g_i, g_i^\gamma \prod_{j=1}^{n \wedge j \neq i} g_{n+1-j+i}) \in G^2$. Note that $d_i = v^{\gamma^i}$. d_i is sent to the subscriber i via a secure channel while the encryption key EK is kept confidentially to all the users.

Encrypt(PK): Let $EK = (g, h) \in G^2$. Pick a random r in $\mathbb{Z}_p$ and set $K = e(g_{n+1}, g)^r \in G$. The value $e(g_{n+1}, g)$ can be computed as $e(g_n, g_1)$. Next, set $Hdr = (g^r, h^r) \in G^2$ and output the pair (Hdr, K) . Hdr is broadcasted to all the subscribers. K is the secret session key to encrypt the DVD content.

Decrypt(S, i, d_i, Hdr, PK): Let $Hdr = (C_0, C_1)$ and $d_i = (d_{i,0}, d_{i,1})$. Then, the subscriber i can compute $K = e(d_{i,0}, C_1)/e(d_{i,1}, C_0)$. With K , the subscriber i can decrypt the encrypted DVD content.

Now we show that keeping the encryption key confidential does not prevent the pirate attack. Let malicious subscribers $\mathbb{T} = \{i_1, \dots, i_t\} \subset S \subseteq \{1, \dots, n\}$ and revoked users $\mathbb{R} = \{i_{t+1}, \dots, i_{t+k}\} \subset \{1, \dots, n\}$ collude to produce a pirate decoder, where $t+k > 1$, and $i_{t+1}, \dots, i_{t+k} \notin S$. They work as follows.

Equivalent Key Generation: For $i \in \mathbb{S} \cup \mathbb{T}$, select $s_i \in_R \mathbb{Z}_p$ such that

$$\sum_{i \in \mathbb{S} \cup \mathbb{T}} s_i = 1, \bar{d}_{i,0} = \prod_{i \in \mathbb{T} \cup \mathbb{R}} g_i^{s_i}, \bar{d}_{i,1} = \prod_{i \in \mathbb{T} \cup \mathbb{R}} d_i^{s_i}.$$

Equivalent Decryption: With equivalent decryption key $\bar{d}_i = (\bar{d}_{i,0}, \bar{d}_{i,1})$, one can compute

$$K = e(\bar{d}_{i,0}, c_1)/e(\bar{d}_{i,1}, c_0).$$

The correctness of the pirate decoder is trivial by noting that $\sum_{i \in \mathbb{S} \cup \mathbb{T}} s_i = 1$. Interestingly, keeping confidential the encryption key of the BGW schemes does not provide additional security. In fact, it makes the pirate attack more serious. In this scenario, the equivalent decryption key can be published while the dealer cannot trace the traitors of the system even if it has unlimited computational power: For $n \gg \log p$, there are exponentially many subsets $\tilde{\mathbb{T}}$ such that there exist $\tilde{s}_i \in \mathbb{Z}_p$ satisfying $\sum_{i \in \tilde{\mathbb{T}}} \tilde{s}_i = 1$, $\bar{d}_{i,0} = \prod_{i \in \tilde{\mathbb{T}}} g_i^{\tilde{s}_i}$ and $\bar{d}_{i,1} = \prod_{i \in \tilde{\mathbb{T}}} d_i^{\tilde{s}_i}$. The same problem appears in the general BGW broadcast encryption scheme when applied to DVD content protection. Hence, the BGW scheme *cannot* be used in such commercial applications.

5 Conclusion

In this paper, we presented pirate attacks on the BGW broadcast encryption schemes when they are used in Pay-TV or DVD content protection applications. Any legit decoder(s) who collude with the revoked decoders can produce exponentially many equivalent decryption keys, while the dealer cannot find any one of the inner and outsider colluding decoders who have been involved in the pirate behavior. We do not claim that we break the BGW schemes but rather showing that their broadcast encryption schemes is not suitable for satellite TV subscription services or DVD content protection applications. Their schemes can only be applicable in trusted environments such as conference key distribution.

References

1. Anderson, R. (ed.): Security Engineering: A Guide to Building Dependable Distributed Systems. John Wiley and Sons, New York (2001)
2. Touretzky, D.S.: Gallery of CSS descramblers. Webpage, Computer Science Department of Carnegie Mellon University (November 17, 2005) <http://www.cs.cmu.edu/~DeCSS/gallery>
3. Boneh, D., Franklin, M.: Identity-based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
4. Boneh, D., Gentry, C., Waters, B.: Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 258–275. Springer, Heidelberg (2005)
5. Dodis, Y., Fazio, N.: Public Key Broadcast Encryption for Stateless Receivers. In: Feigenbaum, J. (ed.) DRM 2002. LNCS, vol. 2696, pp. 61–80. Springer, Heidelberg (2003)
6. Fiat, A., Naor, M.: Broadcast Encryption. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 480–491. Springer, Heidelberg (1994)
7. Goodrich, M.-T., Sun, J.Z., Tamassia, R.: Efficient Tree-based Revocation in Groups of Low-state Devices. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 511–527. Springer, Heidelberg (2004)
8. Halevy, D., Shamir, A.: The lsd Broadcast Encryption Scheme. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 47–60. Springer, Heidelberg (2002)
9. Joux, A., Nguyen, K.: Separating Decision Diffie-Hellman from Diffie-Hellman in Cryptographic Groups. Cryptology ePrint Archive, Report 2001/003, 2001 (October 5, 2006), <http://eprint.iacr.org/>
10. Joux, A.: A One Round Protocol for Tripartite Diffie-Hellman. In: Bosma, W. (ed.) Algorithmic Number Theory. LNCS, vol. 1838, pp. 385–394. Springer, Heidelberg (2000)
11. Naor, D., Naor, M., Lotspiech, J.: Revocation and Tracing Schemes for Stateless Receivers. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 41–62. Springer, Heidelberg (2001)

A Digital Signature Mechanism and Authentication Scheme for Group Communication in Grid^{*}

Yunfa Li, Hai Jin, Deqing Zou, Jieyun Chen, and Zongfen Han

Services Computing Technology and System Lab

Cluster and Grid Computing Lab

School of Computer Science and Technology

Huazhong University of Science and Technology, Wuhan, 430074, China

hjin@mail.hust.edu.cn

Abstract. Group communication is one of the most important strategies to realize large-scale information resource sharing in grid. However, it is very difficult to ensure the security of group communication in grid. In this paper, we propose a digital signature mechanism with one time secret key. In the mechanism, the trust center only needs to issue the partial secret key one time for each group member; and each group member can generate its different secret key each time. Moreover, the size of the signature is independent of the number of group members and the group public key is constant. According to this mechanism, we present a new authentication scheme. In order to verify the efficiency of the scheme, a series of simulation experiments are presented. The results show that the scheme is efficient for group communication in grid.

1 Introduction

With the widely applying of group communication, it has been an urgent and serious problem to protect the security of communication information. In order to solve this problem, numerous research projects have been done in recent years.

At present, a lot of group signature mechanisms and group authentication schemes have been found. Although these signature mechanisms and authentication schemes can resolve some secure requirements of group communication, owing to the complexity and difficulty that group members authenticate each other in large-scale, dynamic grid, they can not be satisfied with the requirement of application. There are still a lot of questions, which need to be resolved.

In grid, group communication is one of the most important strategies to realize large-scale information resource sharing. When group communication begins, each group member in a *Virtual Organization* (VO) [1] needs to exchange some strategy information with its domain administrators (ADs). Meanwhile, the group manager in a VO needs to verify group members and judge whether they are legal group members or not. These strategies may enhance the security of group communication. However, in grid, the scale of information exchanging is very large and the members

^{*} This paper is supported by National Science Foundation of China under grant 90412010.

who do not attend group communication may also need to exchange some information. All of these will affect the security of group communication. So, in grid, it is very necessary to have some strategies of digital signature and authorization to enhance the security of group communication.

The rest of this paper is organized as follows. We discuss the related works in section 2. In section 3, we present a group signature mechanism with one time secret key. In section 4 and 5, we analyze the correctness and the security of the group signature mechanism, respectively. We describe an authentication scheme for group communication in section 6, which can be applied in grid. In section 7, a series of simulation experiments are performed and the results of experiments are analyzed. Finally, the conclusions are drawn in section 8.

2 Related Works

The digital signature and authentication are usually used to ensure the security of communication. Most of the previous authentication mechanisms may be inefficient and not suitable for group-oriented applications. In order to solve this problem, a lot of researchers have made a great progress in this area.

Chaum et al presented a concept about group signature and design a group signature scheme [2]. But, in this scheme, the complexity of computation increases with the growth of the number of members. Lee and Chang presented an efficient group signature scheme based on the *discrete logarithm problem* (DLP) [3]. Although this scheme is efficient, it is obviously linkable. Therefore, it needs to be improved. To provide unlinkability, an improved group signature scheme was proposed in [4]. Regretfully, the improved group signature scheme is still linkable [5]. In addition, Tseng and Jan proposed an ID-based group signature scheme [6]. Since Tseng-Jan ID-based scheme is universally forgeable, it also needs to be improved. In order to solve the question, Popescu presented a modification to the Tseng-Jan ID-based scheme in [7]. In additional, Camenisch and Groth proposed a novel group signature scheme [8]. In the scheme, the protocol for member to join the group only takes two rounds. Therefore, the complexity of computation does not increase with the growth of the number of members. However, for each round, the computation is complex.

In [9], Chang et al proposed a group-oriented authentication mechanism with key exchange. But, in the mechanism, it only involves one-way hash function and exclusive-or operations. Safavi-Naini et al propose an authentication system for group communication [10]. In this system, any member in the group can broadcast an authenticated message to the rest of the group so that every other member or other group can individually verify the authenticity of the message.

In large-scale, dynamic grid environment, for these group signature mechanisms, the number of secret parameters will be very large and the updating of these parameters will be very frequent. Thus, the signature and verification will also be very frequent and complex. Similarly, for these group-oriented authentication schemes, the exchange of certificates will be very frequent. Therefore, these mechanisms and schemes can not satisfy the requirement of group communication. So, it is an important research area to enhance the security of group communication in grid.

In order to ensure the security of group communication, reduce the total computation cost of signature in grid, in this paper, we propose a digital signature mechanism with one time secret key. According to the mechanism, we present an authentication scheme for secure group communication in grid.

3 Our Proposed Signature Mechanism

Our proposed signature mechanism consists of four phases. Each phase is described as follow:

(1) The partial secret key generation phase

Step 1: The trusted center randomly chooses two large primes p_1 and p_2 such that $p_1=2p_1'+1$, $p_2=2p_2'+1$ (where both p_1' and p_2' are also large primes), and computes $N=p_1*p_2$.

Step 2: Let g be a generator of a multiplicative subgroup Z_N^* with order $r=p_1'*p_2'$. The trusted center randomly chooses an integer e which satisfies $\gcd(e, r)=1$, and computes d satisfying $d*e=1 \bmod r$.

Step 3: Let $h()$ be a one-way collision-resistant hash function. The trusted center randomly chooses an integer x as the group secret key and computes the corresponding group public key $y=g^x \bmod N$.

Step 4: Let $G=\{M_1, M_2, \dots, M_n\}$ be the group of n members. The trusted center randomly chooses an integer x_{GM} as the group manager's secret key, and computes the group manager's public key $y_{GM}=g^{x_{GM}} \bmod N$.

Step 5: For the group member $M_i \in G$, the trusted center randomly chooses a large prime $ID(M_i)$ as M_i 's secret identity information, and computes M_i 's partial secret key $x_i=x*d*ID(M_i) \bmod r$, the signer M_i 's identity verification parameter $V_i=g^{ID(M_i)^{-1}} \bmod N$, and M_i 's public key $y_i=V_i^{x_{GM}} \bmod N$.

Step 6: The trusted center sends $\{x_i, V_i, ID(M_i)\}$ to the member M_i , and sends $\{x_{GM}, V_i\}$ to the group manager via a secure channel.

Step 7: The trusted center keeps $p_1, p_2, p_1', p_2', d, x$ and r secret, and publishes $e, y, N, g, h(), y_i (i=1, 2, \dots, n)$ and y_{GM} .

(2) The signature generating phase

On receiving the secret information $\{x_i, V_i, ID(M_i)\}$, each M_i can authenticate whether $\{x_i, V_i, ID(M_i)\}$ is valid or not by checking the equations:

$$y=V_i^{x_i * e} \bmod N \quad (1)$$

$$y_{GM} = y_i^{ID(M_i)} \bmod N \quad (2)$$

Assuming that M_i wants to sign a message ME , it will perform:

Step 1: M_i randomly chooses a large prime b , computes $z=b*ID(M_i)$, $w=g^b \bmod N$, and its secret key $s_i=x_i*b$.

Step 2: M_i computes $c=h(w^e \bmod N, z, w, ME)$, $s=b-s_i*c$, $A=V_i^c \bmod N$, and sends the group signature $\{c, s, z, w, A\}$ to the verifier.

(3) The verification signature phase

M_j (the verifier) can verify the validity of the group signature by using the following steps.

Step 1: M_j computes $W = g^{s^*e} y^{z^*c} \bmod N$.

Step 2: M_j checks the following congruence relations

$$c = h(W, z, w, ME) \quad (3)$$

$$W = w^e \bmod N \quad (4)$$

$$A^z = w^c \bmod N \quad (5)$$

If all the above relationship holds, the group signature $\{c, s, z, w, A\}$ is verified.

(4) The opening signature phase

In case of disputing later, the group manager can open the signature by checking the following equation:

$$w = V_i^z \bmod N \quad (6)$$

In order to convince other verifiers that M_i with the public key y_i is indeed the actual signer, the group manager first randomly chooses an integer k_{GM} , and computes $r_{GM} = V_i^{k_{GM}} \bmod N$, $s_{GM} = x_{GM} * r_{GM} + c * k_{GM}$. Then, it publishes the identification information (r_{GM}, s_{GM}) and the M_i 's public key y_i . The verifier may identify M_i with y_i for the group signature $\{c, s, z, w, A\}$ by checking the following equation:

$$w^{s_{GM}} = y_i^{z * r_{GM}} r_{GM}^{c * z} \bmod N \quad (7)$$

If the above equation holds, the member with the public y_i is identified.

4 Correctness Analysis

In this section, we will analyze the correctness of our proposed signature mechanism. The analysis processes are described as follows.

Theorem 1. The validity of the secret information $\{x_i, V_i, ID(M_i)\}$ can be verified by each group member M_i in Equation (1) and (2).

Proof: We can rewrite Equation (1) as follow:

$$\begin{aligned} V_i^{x_i * e} \bmod N &= g^{ID(M_i)^{-1} * x_i * e} \bmod N = g^{ID(M_i)^{-1} * x * d * ID(M_i) * e} \bmod N \\ &= g^{x * d * e} \bmod N = g^x \bmod N = y \end{aligned}$$

Similarly, we can rewrite Equation (2) as follow:

$$y_i^{ID(M_i)} \bmod N = V_i^{x_{GM} * ID(M_i)} \bmod N = g^{x_{GM}} \bmod N = y_{GM}$$

Theorem 2. The group signature $\{c, s, z, w, A\}$ can be verified by the receiver in Equation (3), (4) and (5).

Proof: We can rewrite Equation (4) as follow:

$$\begin{aligned} w^e \bmod N &= g^{b^*e} \bmod N = g^{(s+s_i^*c)e} \bmod N = g^{s^*e} g^{x_i^*b^*c^*e} \bmod N = g^{s^*e} \\ &g^{x^*d^*ID(M_i)^*b^*c^*e} \bmod N = g^{s^*e} g^{x^*ID(M_i)^*b^*c} \bmod N = g^{s^*e} y^{z^*c} \bmod N = W \end{aligned}$$

Similarly, we can rewrite Equation (5) as follow:

$$w^c \bmod N = g^{b^*c} \bmod N = g^{ID(M_i)^{-1}^*b^*ID(M_i)^*b^*c} \bmod N = V_i^{z^*c} \bmod N = A^z$$

Therefore, Equation (3) can be rewritten as follow:

$$h(W, z, w, ME) = h(w^e \bmod N, z, w, ME) = c$$

Theorem 3. The validity of identity about the group member M_i can be verified by the group manager in Equation (6).

Proof: We can rewrite Equation (6) as follow:

$$V_i^z \bmod N = g^{ID(M_i)^{-1}^*b^*ID(M_i)} \bmod N = g^b \bmod N = w$$

Theorem 4. The validity of identity about the group member M_i can be verified by the other group member in Equation (7).

Proof: We can rewrite Equation (7) as follow:

$$\begin{aligned} y_i^{z^*r_{GM}} r_{GM}^{c^*z} \bmod N &= V_i^{x_{GM}^*z^*r_{GM}} V_i^{c^*z^*k_{GM}} \bmod N = V_i^{z^*(x_{GM}^*r_{GM} + c^*k_{GM})} \bmod N \\ &= V_i^{z^*s_{GM}} \bmod N = w^{s_{GM}} \end{aligned}$$

5 Security Analysis

In this section, we will analyze the security of our proposed mechanism. All possible attacks against the proposed mechanism are presented as below.

Attack 1: The attacker (or group member) tries to reveal the secret parameters.

Analysis of attack 1: If the attacker wants to reveal the group secret key x form $y = g^x \bmod N$, or finds d with the known e , he will face the difficulties of DLP, or the factoring problem. On the other hand, since the trust center keeps r secret, each group member cannot reveal x , d or x^*d from $x_i = x^*d^*ID(M_i) \bmod r$. In addition, it is infeasible for the attacker to reveal (b, s_i) from $s = b^*s_i^*c$ because there are two variables in the equation $s = b^*s_i^*c$.

Attack 2: If the attacker intercepts a valid group signature $\{c, s, z, w, A\}$, he tries to identify the previous group signature.

Analysis of attack 2: In the signature mechanism, when a group member wants to sign a message, he will generate a new secret key $s_i = x_i^*b$ according to the partial

secret key x_i . Since $z=b*ID(M_i)$ is chosen to be impossible to factor, the attacker does not obtain b and $ID(M_i)$. Moreover, even if the group manager publishes the identification information (r_{GM}, s_{GM}) , the anonymity of M_i 's previous signatures is not damaged. The reason is that the information (r_{GM}, s_{GM}) is only provided for the specific group signature $\{c, s, z, w, A\}$. Therefore, the attacker cannot identify the previous group signature even if the attacker intercepts a valid group signature.

Attack 3: A group member tries to forge a valid group signature that is untraceable by the group manager.

Analysis of attack 3: In order to achieve the aim of untraceability, the group member may first choose two large primes b and b' , and computes $z=b*ID(M_i)$, $w=g^{b'} \bmod N$, $s_i=x_i*b$, $c'=h(w'^e \bmod N, z, w', ME)$, $s'=b'-s_i*c'$, $A'=V_i^{c'} \bmod N$. Then, he sends the spurious group signature $\{c', s', z, w', A'\}$ to the verifier.

The verifier computes $W'=g^{s'*e}y^{z*c'} \bmod N$, and verifies if $c'=h(W', z, w', ME)$, $W'=(w')^e \bmod N$, $(A')^z=(w')^{c'} \bmod N$ or not. However, the equation $(A')^z=(w')^{c'} \bmod N$ cannot feasibly be solved. Therefore, it is impossible for each group member to forge a valid group signature that is untraceable by the group manager.

Attack 4: A group member wants to impersonate the other group members to forge a valid group signature.

Analysis of attack 4: If M_j has the signature $\{c, s, z, w, A\}$ of a message ME generated by M_i , and wants to impersonate M_i to forge a valid group signature for the message ME' , he may compute $z'=z*ID(M_j)=b*ID(M_i)*ID(M_j)$, $w'=w^{ID(M_j)} \bmod N = g^{b*ID(M_j)} \bmod N$, $s_j=x_j*b$, $c'=h(w'^e \bmod N, z', w', ME')$, $s'=b*ID(M_j)-s_i*c'$ and $A'=V_j^{c'} \bmod N$. Thus, $\{c', s', z', w', A'\}$ is the group signature. However, M_j cannot compute $s'=b*ID(M_j)-s_i*c'$ because b (or $b*ID(M_j)$) is an unknown number in equation $z'=z*ID(M_j)$ and $w'=g^{b*ID(M_j)} \bmod N$.

Therefore, it is impossible for each group member who impersonates the other group members to forge a valid group signature.

Attack 5: The group manager, who colludes with M_i , tries to publish the M_j 's identification information (r_{GM}, s_{GM}) from the group signature $\{c, s, z, w, A\}$ generated by M_i .

Analysis of attack 5: If the group manager colludes with M_i and tries to publish the M_j 's identification information (r_{GM}, s_{GM}) from the group signature $\{c, s, z, w, A\}$ generated by M_i , he may compute as follows:

Randomly chooses an integer k_{GM} , and computes $r_{GM}=V_i^{k_{GM}} \bmod N$, $s'_{GM}=x_{GM}*r_{GM}+c*k_{GM}$, then he finds $s_{GM}=ID(M_j)^{-1}*ID(M_j)*s'_{GM}$ such that $w^{s_{GM}}=V_i^{z*s_{GM}} \bmod N$. However, because both the group manager and M_i cannot find $s_{GM}=ID(M_j)^{-1}*ID(M_j)*s'_{GM}$, it is infeasible that the group manager or M_i find the $ID(M_j)^{-1}$ of M_j . That is, the proposed scheme can resist conspiracy attack.

6 The Authentication Scheme

In order to realize large-scale resource sharing, a virtual organization can be dynamically built [1]. A VO has a VO strategy set, the server of VO can use these strategies to deal with various services. These strategies can also be applied to group communication. If we adopt these authentication strategies presented in [9-10], the group public key will change with the growing number of group members. Moreover, the storage space and the exchanging certificates that the system needs will increase with the number of group members. In order to solve these problems, we present a new authentication scheme. The scheme is mainly made up of *group member* (GM), *administrator of domain* (AD) and *virtual organization* (VO), shown in Fig.1

In the new authentication scheme, we construct a group certificate for group members. The certificates are described as follows:

$$\text{Cert}_{\text{GM}} = \{ \text{GroupID}, \text{Data}, \text{PK}_{\text{GM}}, (\text{GroupID}, \text{Data}, \text{PK}_{\text{GM}})_{\text{SK}_{\text{VO}}} \}$$

$$\text{Cert}_{\text{AD}} = \{ \text{ID}_{\text{AD}}, \text{Data}, \text{PK}_{\text{AD}}, (\text{ID}_{\text{AD}}, \text{Data}, \text{PK}_{\text{AD}})_{\text{SK}_{\text{VO}}} \}$$

$$\text{Cert}_{\text{VO}} = \{ \text{GridID}, \text{ID}_{\text{VO}}, \text{Data}, \text{PK}_{\text{VO}}, (\text{GridID}, \text{ID}_{\text{VO}}, \text{Data}, \text{PK}_{\text{VO}})_{\text{SK}_{\text{CA}}} \}$$

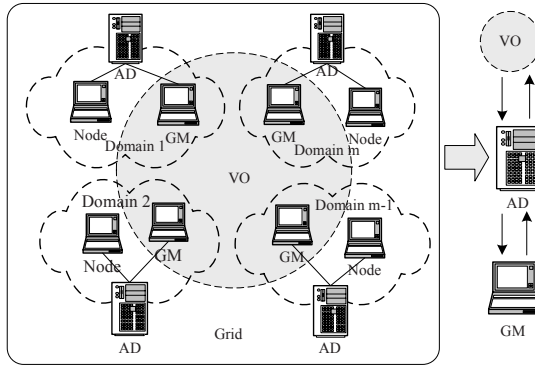


Fig. 1. The authentication scheme in grid

According to the characteristic of group communication in grid, we present an authentication scheme based on the digital signature proposed in this paper. The authentication scheme is shown as follow:

1. GM→AD: Service_Request: $\text{Cert}_{\text{GM}} \parallel (\text{E}(\text{K}_1)_{\text{PK}_{\text{AD}}})_{\text{SK}_{\text{GM}}}$
2. AD→VO: Service_Request: $\text{Cert}_{\text{AD}} \parallel (\text{E}(\text{K}_2)_{\text{PK}_{\text{VO}}})_{\text{SK}_{\text{AD}}}$
3. The server of VO→AD: Service_Response: $\text{Cert}_{\text{VO}} \parallel \text{E}_{\text{K}_2}(\text{K}_2 \parallel \text{K}_3)$
4. AD→GM: Service_Response: $\text{Cert}_{\text{AD}} \parallel \text{E}_{\text{K}_1}(\text{K}_1 \parallel \text{K}_2)$

Here, the session key of GM and AD is $\text{K}_1 \oplus \text{K}_2$, and the session key of AD and VO is $\text{K}_2 \oplus \text{K}_3$.

7 Simulation Experiments and Results Analysis

In order to validate the efficiency of this authentication scheme, we have done a series of simulation experiments. By using the simulation method, we built two kinds of group communication scenarios in HUSTGrid. In the first scenario, we use the authentication scheme presented in our paper. In the second scenario, we do not use any authentication scheme. In these two scenarios, the network and the group communication condition are the same. The communication scheme of group communication in these simulation experiments is one-to-many scheme. Moreover, the average length of each message is 1000 bytes.

7.1 Simulation Experiments

In the first scenario, let the length of the hash value is 80bits, which is exported by hash function. By these experiments, we can get a figure about the number of members of group communication, the number of source messages ME (ME_{num}) and the total time needed by the sender who sends the ME_{num} messages to other members. The results are shown in Fig.2.

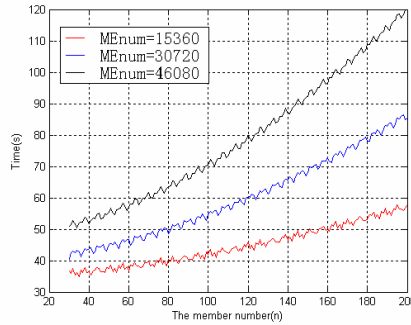


Fig. 2. The time needed by the sender, who sends the ME_{num} messages to the n members of group communication in the first scenario

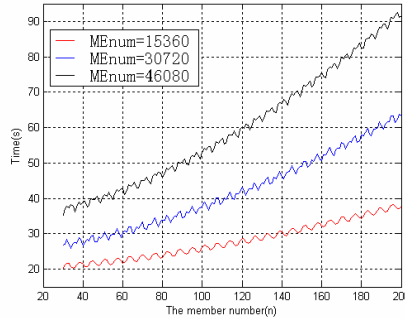


Fig. 3. The sending time needed by the sender, who sends the ME_{num} source messages to the n members of group communication in the second scenario

For the second scenario, we can get another figure about the number of members of group communication, the number of source messages ME (ME_{num}) and the sending time needed by the sender, who only sends the ME_{num} source messages to other members. The results are shown in Fig.3.

7.2 Result Analysis

From Fig.2, we find two situations: (1) when the number of messages ME is a constant, the time needed by the sender increases with the growth of the number of members n . (2) When the number of members n is a constant, the time needed by the sender increases with the growth of the number of messages ME.

Similarly, from Fig.3, we also find two situations. The two situations are the same as that of Fig.2. But, the results that the two figures are different. The main reason is that the authentication scheme presented in our paper is used in the first scenario. So, in the first scenario, the total time that is needed by the sender includes two aspects: (1) the sending time needed by the sender; (2) the time needed by digital signature and authentication. In the second scenario, the total time needed by the sender only includes the sending time. In order to explore the effect of the authentication scheme, we can analyze the results of the two scenarios as follow.

Suppose $T_1(ME, n)$ denotes the total time needed by the sender in the first scenario, which includes the sending time needed by the sender, the time needed by digital signature and authentication. $T_2(ME, n)$ denotes the sending time needed by the sender in the second scenario. Because the network and the condition of group communication in the first scenario are the same as those of group communication in the second scenario, the sending time in the first scenario nearly equals to the sending time in the second scenario for the messages ME and the number of members n .

Suppose $T_{S-A}(ME, n)$ denotes the time needed by digital signature and authentication in first scenario, then $T_{S-A}(ME, n)$ can be described as follow:

$$T_{S-A}(ME, n) \approx T_1(ME, n) - T_2(ME, n) \quad (8)$$

Let the ratio $\lambda = T_2(ME, n)/T_{S-A}(ME, n)$, then we can get $\lambda \approx T_2(ME, n)/(T_1(ME, n) - T_2(ME, n))$. So, based on the results of the first scenario and the second scenario, we can get the relationship about the ratio λ , the number of members n and the messages ME, shown in Fig.4.

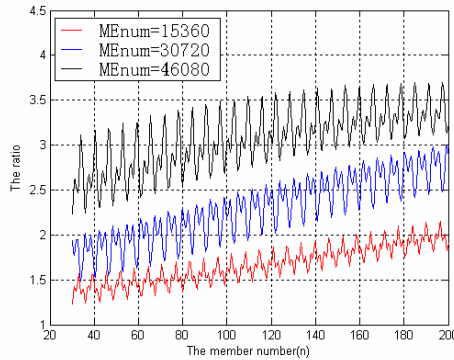


Fig. 4. The relationship about λ , the member number n and the messages ME

From Fig.4, we can find two situations: (1) the ratio λ is larger than 1; (2) for the messages ME, when the number of members n increases, though the number of the ratio λ is variational, its whole variation trend is increasing. The two situations mean that the sending time needed by the sender increases faster than the time that is needed by digital signature and authentication. So, in the first scenario, the reason about the growth of the total time needed by the sender is mainly affected by the growth of the total of messages.

Based on the result analysis, we can conclude that the authentication scheme presented in the paper is efficient.

8 Conclusion

In this paper, we propose a novel group signature mechanism with one time secret key. In the mechanism, the size of the signature is independent of the number of group members and the group public key is constant. With this novel property, we construct a new authentication scheme for secure group communication in grid. Based on the security analysis and the simulation experiments, we draw a conclusion that the group signature mechanism presented in the paper is secure and its authentication scheme is efficient for group communication in grid.

References

1. Demchenko, Y.: Virtual organizations in computer grids and identity management. Information Security Technical Report 9(1), 59–76 (2004)
2. Chaum, D., van Heyst, E.: Group Signatures. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 257–265. Springer, Heidelberg (1991)
3. Lee, W., Chang, C.: Efficient Group Signature Scheme Based on the Discrete Logarithm. IEE Proceedings Computers & Digital Techniques 145(1), 15–18 (1998)
4. Tseng, Y M., Jan, J K.: Improved Group Signature Based on Discrete Logarithm Problem. Electronics Letters 35(1), 37–38 (1999)
5. Sun, H.: Comment: Improved Group Signature Scheme Based on Discrete Logarithm Problem. Electronics Letters 35(16), 1323–1324 (1999)
6. Tseng, Y M., Jan, J K.: A Novel ID-based Group Signature. In: Proceedings of Workshop on Cryptology and Information Security at 1998 International Computer Symposium, Tainan, pp. 159–164 (1998)
7. Popescu, C.: A Modification of the Tseng-Jan Group Signature Scheme. Studia Universitatis Babes-Bolyai Informatica XLV(2), 36–40 (2000)
8. Camenisch, J., Groth, J.: Group signatures: Better efficiency and new theoretical aspects. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 120–133. Springer, Heidelberg (2005)
9. Chang, Y S., Wu, T C.: Group-oriented authentication mechanism with key exchange. Computer Communication 21(5), 485–497 (1998)
10. Safavi-Naini, R., Wang, H.X.: Broadcast authentication for group communication. Theoretical Computer Science 269(1-2), 1–21 (2001)

Cryptanalysis of Server-Aided RSA Key Generation Protocols at MADNES 2005

Fanyu Kong^{1,2}, Jia Yu³, Baodong Qin^{1,2}, and Daxing Li^{1,2}

¹ Institute of Network Security, Shandong University, 27 Shanda Nanlu Road,
Jinan 250100, China

fanyukong@sdu.edu.cn

² Key Laboratory of Cryptographic Technology and Information Security,
Ministry of Education, Jinan 250100, China

³ College of Information Engineering, Qingdao University, Qingdao 266071, China

Abstract. At MADNES 2005, Y. Chen et al. proposed two improved server-aided RSA key generation protocols, which are claimed to be secure against collusion attack. However, at ISPEC 2006, T. Cao et al. presented a collusion attack on Chen's standard server-aided RSA key generation protocol and can get the plaintext from a ciphertext. In this paper, we propose a full cryptanalysis of Chen's two server-aided RSA key generation protocols. Firstly, we give a further analysis of Chen's standard protocol and can recover the factorization of the RSA modulus N with the complexity $O(\log^3(N))$. Secondly, we propose two collusion attacks on Chen's unbalanced RSA key generation protocol. It is proved that we can decrypt any ciphertext with the complexity $O(\log^3(N))$ and find the secret prime p with the complexity $O(\log^4(N))$. Therefore, neither of Chen's two server-aided RSA key generation protocols can resist collusion attack.

1 Introduction

With the wide application of hand-held devices such as smart cards, PDAs and mobile phones, it becomes important to construct the secure communication among the remote hand-held devices. However, most of the hand-held devices have very limited computational resources for cryptographic computation and it is desirable to improve their performance with the help of untrusted servers, that is called server-aided secret computations (SASC). From CRYPTO 1988 to the date, several server-aided RSA signature protocols [1,2,3,4,5] were proposed and some corresponding passive or active attacks [6,7,8,9] were also presented.

At CT-RSA 2000, Modadugu, Boneh and Kim [10] introduced the server-aided RSA key generation protocols. Using these protocols, the limited-power hand-held devices can generate an RSA key with the help of untrusted servers. However, they can't resist collusion attack. At MADNES 2005, Chen et al. proposed two improved protocols [11] against collusion attack. However, at ISPEC 2006, T. Cao et al. [12] shows that Chen's standard RSA key generation protocol can't resist collusion attack.

In this paper, we propose a full cryptanalysis of Chen's two server-aided RSA key generation protocols. Firstly, we give a further analysis of Chen's standard server-aided RSA key generation protocol and show that we can recover the factorization of the RSA modulus N with the complexity $O(\log^3(N))$. Secondly, we propose two collusion attacks on Chen's unbalanced RSA key generation protocol. It is proved that the first method can decrypt any ciphertext with the complexity $O(\log^3(N))$ and the second method can find the secret prime p with the complexity $O(\log^4(N))$. Therefore, Chen's two server-aided RSA key generation protocols are both broken via collusion attack.

The rest of this paper is organized as follows. In Section 2, we review Chen's two server-aided RSA key generation protocols and Cao's attack on the standard one. In Section 3, we present a further cryptanalysis of Chen's standard protocol and can recover the factorization of the modulus N . In Section 4, we present two attacking algorithms and show that Chen's unbalanced server-aided RSA key generation protocol is also insecure and does not resist collusion attack. Finally, in Section 5 we conclude the paper.

2 Preliminaries

2.1 Review of RSA Cryptosystem and the Unbalanced RSA Key

The standard RSA cryptosystem [13] can be described as follows. Firstly, the user chooses two random large primes p and q and computes the RSA modulus $N = pq$. Secondly, he chooses a public exponent e that is relatively prime to $\phi(N) = (p-1)(q-1)$ and publishes N and e as his public key. The corresponding private key $d = e^{-1} \pmod{\phi(N)}$ can be obtained by the extended Euclidean algorithm and must be kept secret. The encryption and decryption of RSA cryptosystem is shown as follows.

1. A plaintext m is encrypted by computing $c = m^e \pmod{N}$.
2. The ciphertext c is decrypted by computing $m = c^d \pmod{N}$.

Shamir [14] introduced a variant RSA by using the unbalanced key, which has a modulus $N = pR$ where p is a 512-bit prime and R is a 4096-bit random number. While the public key is N and e , the private key is p and d where $d = e^{-1} \pmod{(p-1)}$. Similarly, the encryption and decryption is described as follows.

1. A plaintext m is encrypted by computing $c = m^e \pmod{N}$.
2. Differently from the standard RSA, the ciphertext c is decrypted by computing $c_p = c \pmod{p}$ and the plaintext $m = c_p^d \pmod{p}$.

2.2 Chen's Two Server-Aided RSA Key Generation Protocols

At MADNES 2005, Chen et al. proposed two improved server-aided RSA key generation protocols [11], which include the standard and unbalanced protocol. Chen's standard RSA key generation protocol is briefly described as follows.

1. Hand-held device generates two $n/2$ -bit candidates p and q and computes $N = pq$ and $\phi(N) = N - p - q + 1$, then selects a random $g \in Z_N^*$.
2. Hand-held device picks randomly a large odd integer $k \in [-N, N]$ such that $\gcd(k, \phi(N)) = 1$. It then picks integers $t_1, t_2 \in [-N^2, N^2]$ such that $t_1 + t_2 = k\phi(N) = T$, where none of t_1, t_2 and T is a multiple of either p or q . Hand-held device sends $\langle N, g, t_1 \rangle$ to server 1 and $\langle N, g, t_2 \rangle$ to server 2.
3. Server 1 computes $X_1 = g^{t_1} \pmod{N}$. Server 2 computes $X_2 = g^{t_2} \pmod{N}$. The results are sent back to hand-held device.
4. Hand-held device checks if $X_1 X_2 \equiv 1 \pmod{N}$. If so, p and q are considered to be potential primes. Then, $N = pq$ is declared as a potential RSA modulus where $\phi(N) = N - p - q + 1$. Otherwise, goto Step 1.
5. The rest steps are seen in [11].

Chen's unbalanced server-aided RSA key generation protocol is briefly described as follows.

1. Hand-held device generates a $n/2$ -bit candidate p , where $p \equiv 3 \pmod{4}$, and picks a random number R which is 8-10 times longer than the size of p . It computes $N = pR$.
2. Hand-held device picks at random a large odd integer $k \in [-p, p]$ such that $\gcd(k, \phi(p - 1)) = 1$. It then picks integers $t_1, t_2 \in [-N, N]$ randomly such that $t_1 + t_2 = k(p - 1) = T$. Hand-held device picks a random number $g \in Z_N^*$, then sends $\langle N, g, t_1 \rangle$ to Server 1 and $\langle N, g, t_2 \rangle$ to Server 2.
3. Server 1 computes $X_1 = g^{t_1} \pmod{N}$. Server 2 computes $X_2 = g^{t_2} \pmod{N}$. The results X_1 and X_2 are sent back to hand-held device.
4. Hand-held device checks if $X_1 X_2 \equiv 1 \pmod{p}$. If so, p is considered to be potential primes. Then, $N = pR$ is declared as a potential RSA modulus, where $\phi(p)$ must be $p - 1$. Otherwise, goto Step 1.
5. The rest steps are seen in [11].

2.3 Cao's Collusion Attack on Chen's Standard Protocol

In [11], Chen et al. remark that there exists an attack for finding the private exponent d in the concluding remarks. In [12], Cao et al. present the detailed collusion attack, which can decrypt any ciphertext. The attacker uses T directly, where $T = t_1 + t_2 = k\phi(N)$, and need not to guess the random integer k .

Cao's attack is effective on Chen's standard RSA key generation protocol, as claimed in the following theorem.

Theorem 1. [12] *Chen et al.'s standard RSA generation Protocol is not secure. One can get the plaintext from ciphertext via the collusion attack.*

3 Further Cryptanalysis of Chen's Standard Protocol

In this section, we show that the factorization of the RSA modulus N can be recovered in Chen's standard protocol via collusion attack. Indeed, Cao's attack

is to compute the private key d by using the $k\phi(N)$ and the public key e , which method is used in the common modulus attack [15]. Note that, in [16], Delaurentis proposed another method to recover the factorization of the RSA modulus N using the square root finding algorithm if the $k\phi(N)$ is known, which was early found in [17].

Since that the $T = k\phi(N)$ is revealed via collusion attack on Chen's standard protocol, we can directly use the method in [16,17] to factor the modulus N . The algorithm is described as follows.

Algorithm 1. [16,17] Factoring the modulus N using $T = k\phi(N)$

Input: The modulus N , $T = k\phi(N)$.

Output: The primes p and q , the factorization of the RSA modulus $N = pq$.

1. Select randomly $a \in (1, N)$, we have $\gcd(a, N) = 1$ and $a^T \equiv 1 \pmod{N}$.
Otherwise, we obtain $p|N$ or $q|N$ and succeed.
 2. Let $T = 2^s h$, where h is odd. Find the least integer $j \in [1, s]$, which satisfies $a^{2^j h} \equiv 1 \pmod{N}$ and $a^{2^{j-1} h} \not\equiv 1 \pmod{N}$.
 3. Let $b = a^{2^{j-1} h} \pmod{N}$. If $b = -1 \pmod{N}$, goto Step 1.
Otherwise, we have $p = \gcd(b + 1, N)$ or $p = \gcd(b - 1, N)$ and factor N .
-

The above Algorithm 1 does work because of the fact that finding square roots modulo N is computationally equivalent to factoring N . The leak of $T = k\phi(N)$ makes it easy to find the square roots modulo N and hence we can factor N . Therefore, Chen's server-aided standard protocol can't resist collusion attack, which does reveal the value of the $T = k\phi(N)$.

Note that in Step 3 of Algorithm 1, the probability that $b = -1 \pmod{N}$ is $1/2$ and hence we succeed with the probability 96.9% after 5 selections of a . In Algorithm 1, one modular exponentiation has the computational complexity $O(\log^3(N))$ and the Euclidean algorithm has the computational complexity $O(\log^2(N))$. Therefore, the overall complexity of Algorithm 1 is $O(\log^3(N))$.

4 Cryptanalysis of Chen's Unbalanced Server-Aided RSA Key Generation Protocol

For the case of Chen's unbalanced server-aided RSA key generation protocol, the cryptanalysis is a little more complicated. Now we propose two methods for breaking this protocol via collusion attack, which leaks the value of $T = k(p-1)$. In the first method, we can decrypt any ciphertext. In the second method, we can obtain the secret prime p . Therefore, any of two cryptanalytic methods can break Chen's unbalanced server-aided RSA key generation protocol.

4.1 Our First Method

Now we propose the first method. Note that in the previous Chen's standard protocol, the private exponent d can be directly computed since that we know

$k\phi(N)$ and all the computations are arithmetic modulo N . However, in Chen's unbalanced RSA key generation protocol, the decryption is computed modulo p , which is kept secret, and the private exponent d satisfies $ed \equiv 1 \pmod{(p-1)}$. Therefore, for Chen's unbalanced protocol, we can't directly obtain the private exponent d using the above Algorithm 1 even if the value $T = k(p-1)$ is leaked in collusion attack.

Fortunately, we find the following method, which can decrypt any ciphertext if the value $T = k(p-1)$ is known.

Algorithm 2. Decrypting any ciphertext using $T = k(p-1)$

Input: The modulus $N = pR$, $T = k(p-1)$ and any ciphertext $c = m^e \pmod{N}$.

Output: The plaintext m of the ciphertext c .

1. Select randomly $a \in (1, N)$, which satisfies that $\gcd(a, N) = 1$.
 2. Compute $b = a^{k(p-1)} \pmod{N}$.
 3. Compute $x = \gcd(b-1, N)$ via the Euclidean algorithm.
 4. Compute d_1 , which satisfies that $ed_1 \equiv 1 \pmod{k(p-1)}$.
 5. We compute $m = c^{d_1} \pmod{x}$ and m must be the plaintext.
-

Theorem 2. *Algorithm 2 can decrypt any ciphertext with the computational complexity $O(\log^3(N))$.*

Proof. (Correctness) According to Chen's unbalanced RSA key generation protocol, hand-held device picks at random an odd integer $k \in [-p, p]$ such that $\gcd(k, (p-1)) = 1$.

Let $w = \gcd(k, \phi(R)) = k/k' = \phi(y)$, where $y|R$ holds. Thus, we have

$$a^{k(p-1)} \equiv a^{k'w(p-1)} \equiv a^{k'\phi(y)\phi(p)} \equiv a^{k'\phi(py)} \equiv 1 \pmod{py}. \quad (1)$$

Then we obtain that

$$x = \gcd(b-1, N) = \gcd(a^{k(p-1)} - 1, pR) = py. \quad (2)$$

We compute d_1 , which satisfies $ed_1 \equiv 1 \pmod{k(p-1)}$. Finally, we compute $m = c^{d_1} \pmod{x}$, which is described as follows, and m must be the plaintext.

$$\begin{aligned} c^{d_1} \pmod{x} &= (m^e \pmod{N})^{d_1} \pmod{py} \\ &= m^{ed_1} \pmod{py} \\ &= m^{l[k(p-1)]+1} \pmod{py} \\ &= m. \end{aligned} \quad (3)$$

(Efficiency) In algorithm 2, one modular exponentiation has the computational complexity $O(\log^3(N))$ and the Euclidean algorithm has the computational complexity $O(\log^2(N))$. Therefore, the overall complexity of Algorithm 2 is $O(\log^3(N))$. $\square$

4.2 Our Second Method

Now we consider another attacking method for finding the secret prime p . In Algorithm 1 in Section 3, we have $N = pq$ and $N|(b-1)(b+1)$ since that $b \in (1, N)$, $b \not\equiv \pm 1 \pmod{N}$ and $b^2 \equiv 1 \pmod{N}$. Hence we have $p = \gcd(b+1, N)$ and $q = \gcd(b-1, N)$, or not we have $p = \gcd(b-1, N)$ and $q = \gcd(b+1, N)$. Thus the factorization of the modulus $N = pq$ is recovered.

However, in the unbalanced RSA key generation protocol, the modulus $N = pR$, where p is a 512-bit prime and R is a 4096-bit random number which may have multiple prime factors and have a prime factor at least 512-bit long with high probability. Therefore, in general, we can't directly obtain the secret prime p by computing $\gcd(b-1, N)$ or $\gcd(b+1, N)$ using the similar method as Algorithm 1. In [18], T. Cao suggests that he can choose many integers $g_i, 1 \leq i \leq r$ and compute the common factor N' of $\{pR, (g_1^T \pmod{N} - 1), \dots, (g_r^T \pmod{N} - 1)\}$. However, we remark that this common factor N' maybe the form of $N'|R$ or $N' = pR'$, where $R'|R$, and hence p may not be directly obtained for some cases. Therefore, the probability distribution of the common factor N' for different integer r should be computed accurately to determine the probability of finding p exactly.

Indeed, we find the provable and efficient method for computing the secret prime p , which is described as the following Algorithm 3. Let $|x|$ denote the bit length of x . Suppose that the prime p is a 512-bit integer in Algorithm 3, which can be easily applied to the primes with a different length.

Theorem 3. *Algorithm 3 can find the secret prime p with the computational complexity $O(\log^4(N))$.*

Proof. (Correctness) Since that $k \in [-p, p]$ such that $\gcd(k, (p-1)) = 1$ and $x = \gcd(a^{k(p-1)} \pmod{N} - 1, N)$, we have that $k(p-1)$ and x are both at most 1024 bits long. Hence x may have another 512-bit prime factor besides p . Note that $u = \gcd(b-1, x)$ and $v = \gcd(b+1, x)$. If only one of u and v is at least 512 bits long, it must have only one prime factor p . Otherwise, if $|v| \geq 512$, $|u| \geq 512$ and $|\gcd(u, v)| \geq 512$, we have that $\gcd(u, v)$ must be divisible by p and let $x = \gcd(u, v)$. For the case that $|v| \geq 512$, $|u| \geq 512$ and $|\gcd(u, v)| < 512$ hold, we conclude that u and v have two distinct 512-bit prime factors respectively and need to try twice for finding x , where $p|x$.

From Step 5 to 7, x must have only one 512-bit prime factor p and other prime factors are less than 512 bits long. Note that, in Step 6, $b \in (1, x)$, $u = \gcd(b-1, x)$ and $v = \gcd(b+1, x)$ hold, hence we have that $u < x$, $v < x$, $u|x$, and $v|x$. Therefore, when a loop of Step 5 to 7 is finished, the bit length $|x|$ of x must be reduced by at least 1. After at most 512 loops repeatedly, we must finally obtain the secret prime p .

(Efficiency) For every time that we compute a new x via factoring x using $u = \gcd(b-1, x)$ and $v = \gcd(b+1, x)$, we reduce at least one bit of the bit length of x . Hence the algorithm can succeed in $O(\log(p))$ loops. In each step, modular exponentiations and the Euclidean algorithm are used. One modular exponentiation has the computational complexity $O(\log^3(N))$ and the Euclidean algorithm has

Algorithm 3. Finding the prime p using $T = k(p - 1)$

Input: The modulus $N = pR$, $T = k(p - 1)$.

Output: The prime p .

1. Select randomly $a \in (1, N)$, which satisfies that $\gcd(a, N) = 1$ and compute $a^{k(p-1)} \pmod{N}$.
 2. Compute $x = \gcd(a^{k(p-1)} \pmod{N} - 1, N)$ via the Euclidean algorithm.
 Since that $k \in [-p, p]$, x must be at most 1024 bits long.
*/*From Step 3 to 4, we find the integer x , which has only one 512-bit prime p .*/*
 3. Find an integer $b \in (1, x)$, which satisfies $\gcd(b, x) = 1$, $b \not\equiv \pm 1 \pmod{x}$ and $b^2 \equiv 1 \pmod{x}$.
 4. Compute $u = \gcd(b - 1, x)$ and $v = \gcd(b + 1, x)$ via the Euclidean algorithm.
 At least one of u and v is at least 512 bits long.
 - 4.1 If $|u| \geq 512$ and $|v| < 512$, let $x = u$.
 - 4.2 If $|v| \geq 512$ and $|u| < 512$, let $x = v$.
 - 4.3 If $|v| \geq 512$ and $|u| \geq 512$ then
 - 4.3.1 If $|\gcd(u, v)| \geq 512$, let $x = \gcd(u, v)$.
 - 4.3.2 else let $x = u$ or $x = v$ respectively and execute Step5-7 twice.*/*From Step 5 to 7, we find the 512-bit prime p .*/*
 5. Find an integer $b \in (1, x)$, which satisfies $\gcd(b, x) = 1$, $b \not\equiv \pm 1 \pmod{x}$ and $b^2 \equiv 1 \pmod{x}$.
 6. Compute $u = \gcd(b - 1, x)$ and $v = \gcd(b + 1, x)$ via the Euclidean algorithm.
 At least one of u and v is at least 512 bits long.
 - 6.1 If $|u| \geq 512$ and $|v| < 512$, let $x = u$.
 - 6.2 If $|v| \geq 512$ and $|u| < 512$, let $x = v$.
 - 6.3 If $|v| \geq 512$ and $|u| \geq 512$ then let $x = \gcd(u, v)$.
 7. If $|x| > 512$, goto Step 5. Otherwise, we have $p = x$ and succeed.
-

the computational complexity $O(\log^2(N))$. For the sake that N is several times long as p , the overall complexity of Algorithm 1 is $O(\log^3(N)\log(p))$, namely $O(\log^4(N))$. $\square$

4.3 Further Attack on the Standard RSA with Known $k(p - 1)$ or $k(q - 1)$

Generally, the standard RSA with the modulus $N = pq$ may be seen as a special case of the unbalance RSA with $N = pR$, where $q = R$. Therefore, our proposed Algorithm 3 can be applied to break the standard RSA if $k(p - 1)$ or $k(q - 1)$ is known.

5 Conclusion

We have presented collusion attacks on Chen's two server-aided RSA key generation protocols, which are insecure since that we can decrypt any ciphertext and recover the factorization the modulus N .

Indeed, we remark that $k(p-1)$ in the standard and unbalanced RSA cryptosystem must be kept secret and otherwise the cryptosystem can be broken. Furthermore, similar cryptanalysis can be applied to various cryptographic protocols which have the same security flaws.

Acknowledgements

Supported by National 863 High-tech Research and Development Program of China (2003AA141120, 2004AA001260). The authors would like to thank Dr. Tianjie Cao for his interesting discussion and the anonymous referees for their valuable comments.

References

1. Matsumoto, T., Kato, K., Imai, H.: Speeding up secret computation with insecure auxiliary devices. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 497–506. Springer, Heidelberg (1990)
2. Matsumoto, T., Imai, H., Lai, C.S., Yen, S.M.: On verifiable implicit asking protocols for RSA computation. In: Zheng, Y., Seberry, J. (eds.) AUSCRYPT 1992. LNCS, vol. 718, pp. 296–307. Springer, Heidelberg (1993)
3. Kawamura, S., Shimbo, A.: Fast server-aided secret computation protocols for modular exponentiation. IEEE Journal on Selected Areas Communications 11(5), 778–784 (1993)
4. Burns, J., Mitchell, C.J.: Parameter selection for server-aided RSA computation schemes. IEEE Transactions on Computers 43(2), 163–174 (1994)
5. Beguin, P., Quisquater, J.J.: Fast Server-Aided RSA Signatures Secure Against Active Attacks. In: Coppersmith, D. (ed.) CRYPTO 1995. LNCS, vol. 963, pp. 57–69. Springer, Heidelberg (1995)
6. Anderson, R.J.: Attack on Server-Assisted Authentication Protocols. IEE Electronics Letters 28(15), 1473 (1992)
7. Pfizmann, B., Waidner, M.: Attacks on protocols for server-aided RSA computation. In: Rueppel, R.A. (ed.) EUROCRYPT 1992. LNCS, vol. 658, pp. 153–162. Springer, Heidelberg (1993)
8. Lim, C.H., Lee, P.J.: Security and Performance of Server-Aided RSA Computation Protocols. In: Coppersmith, D. (ed.) CRYPTO 1995. LNCS, vol. 963, pp. 70–83. Springer, Heidelberg (1995)
9. Nguyen, P., Stern, J.: The Beguin-Quisquater Server-Aided RSA Protocol. In: Ohta, K., Pei, D. (eds.) ASIACRYPT 1998. LNCS, vol. 1514, pp. 372–379. Springer, Heidelberg (1998)
10. Modadugu, N., Boneh, D., Kim, M.: Generating RSA Keys on a Handheld Using an Untrusted Server. In: CT-RSA 2000 (2000) Available at: <http://crypto.stanford.edu/~dabo/pubs.html>
11. Chen, Y., Safavi-Naini, R., Baek, J.: Server-Aided RSA Key Generation against Collusion Attack. In: Burmester, M., Yasinsac, A. (eds.) MADNES 2005. LNCS, vol. 4074, pp. 27–37. Springer, Heidelberg (2006)
12. Cao, T., Mao, X., Lin, D.: Security Analysis of a Server-Aided RSA Key Generation Protocol. In: Chen, K., Deng, R., Lai, X., Zhou, J. (eds.) ISPEC 2006. LNCS, vol. 3903, pp. 314–320. Springer, Heidelberg (2006)

13. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM* 21(2), 120–126 (1978)
14. Shamir, A.: RSA for Paranoids. *CryptoBytes* 1(3), 1–4 (1995)
15. Simmons, G.J.: A weak privacy protocol using the RSA crypto algorithm. *Cryptologia* 7(2), 180–182 (1983)
16. Delaurentis, J.M.: A further weakness in the common modulus protocol for the RSA crypto algorithm. *Cryptologia* 8(3), 253–259 (1984)
17. Miller, G.L.: Reimanns hypothesis and tests for primality. *J. Comput. System Sci.* 13, 300–317 (1976)
18. Cao, T., Mao, X.: Collusion Attack on a Server-Aided Unbalanced RSA Key Generation Protocol. In: *International Conference on Communication Technology* (2006) (to appear)

Service-Context Knowledge-Based Solution for Autonomic Adaptation

Marcel Cremene¹ and Michel Riveill²

¹ Technical University of Cluj-Napoca,
Cluj-Napoca, Romania
`cremene@com.utcluj.ro`

² University of Nice,
Sophia-Antipolis, France
`riveill@unice.fr`

Abstract. In most adaptive systems, the adaptation control is based on developer-made *rules* and *strategies* that are specific for each service and context. Our proposal for autonomic computing is to replace this mechanism with a machine-based reasoning. The key element in making this possible is a *service-context* model that offers a knowledge support for the adaptive platform, which can diagnose the service adequacy to the context and search for solutions. We have tested our model using a prototype that adapts a service by inserting the 'right' component at the 'right' place into the service architecture.

1 Introduction

Autonomic service adaptation is a high-interest research direction in the last years. The term '*autonomic computing*' is used in [8] by Kephart, for naming a future direction in software, where computing systems manage themselves according to high-level objectives expressed by human operators. As the author motivates, the interest for autonomic computing comes from the fact that the complexity of software systems is continuously increasing while a dynamical evolution is needed. In such complex systems, where the human-based administration becomes more and more difficult, the service developers encounter several difficulties some of which are described below.

It is impossible for the developer to completely anticipate the system evolution at the moment of its creation. The context [3] (user preferences, physical resources, physical and social environment, etc.) may change dynamically and unpredictably, especially in open environments [4] such as mobile systems. New context elements relevant for the services may appear a posteriori. If the developer tries to anticipate all possible context elements and their states, he will be overwhelmed by a combinatorial explosion. He also needs to figure out all possible ways for adapting the system. Thus, choosing the best solution is not an easy task. Some systems (telecommunications, banking, airport, military, etc.) cannot be stopped and they must be reconfigured dynamically, in a very short time.

Such systems need fast algorithms for solution search so human-based solution search may be inappropriate.

According to our approach, the computing systems include not only component-based services but also users and infrastructure (machines, networks, etc.). In such systems, the services need to be dynamically reconfigured in order to optimally: a) respond to user needs and b) use the physical resources offered by the infrastructure (memory, CPU, bandwidth, etc.). In most existing adaptive systems, the service reconfiguration is triggered by service and context-specific rules and applied using service-specific strategies. The limitation of existent systems comes from the fact that these rules and strategies are specified by a human operator (developer, administrator). This aspect comes into conflict with the autonomic computing objective that requires self-management. In our opinion, the key aspect of autonomic systems is the ability to automatically detect the problems and search for solutions.

In order to solve this problem, we propose the replacement of the developer-based reasoning with a machine-based one. The key element in making this possible is a *service-context* model that offers a knowledge support for the adaptive system which can diagnose the service adequacy to the context and search for solutions. We have tested our model using a prototype that adapts a service by inserting the right component at the right place into the service architecture.

This paper is organized as follows: the next section presents several existent adaptive systems, section three presents the proposed solution for autonomic adaptation: general architecture, service-context model, service adequacy to the context and solution search. Section four describes the prototype that we have implemented in order to test our model. The last section presents our conclusions.

2 Autonomic Adaptation Systems and Their Limitations

The autonomic adaptation domain is related to several directions such as: *unanticipated adaptation* [7]; *auto-adaptive* systems [5]; and *self-healing* systems [9]. In this section, we briefly analyze several existent adaptive systems in order to see how appropriate they are for autonomic adaptation.

According to J. Keeney [7], in a 'completely unanticipated system', the answers to questions such as: when, where, what and how to reconfigure a service, are known only at runtime. The author proposes the "Chisel" platform that supports dynamic rule modification. Let us analyze an example of rules and strategies used by this platform:

```
ON WirelessNetworkDisconnect
  NetworkConnectionService.WiredConnectionBehaviour
IF (NetworkConnectionService.WiredConnectionAvailable == TRUE &&
  WirelessNetworkDisconnect.IsTemporary == FALSE ) ||
  (UserPreferences.getPreferredComms()).compareTo("Wireless") != 0
```

This rule says that if the wireless network is disconnected we apply the strategy 'WiredConnectionBehaviour', but only if a wired connection exists and the

wireless disconnection is not just temporary. The adaptation technique is based on reflective language support and will not be detailed here. We may observe that, even if these rules may be changed at runtime (as the author claims), the strategies are still predefined (i.e. 'WiredConnectionBehaviour' strategy) and so are the events that come from the context (i.e. 'WirelessNetworkDisconnect'). Another inconvenient is that, adding new rules requires the developer's intervention because the user is not usually able to write such complex syntaxes or deal with possible conflicts between different rules.

An auto-adaptive platform is proposed by J. Floch in [5]. The idea is to use an abstract service architecture description that may be automatically projected towards several possible concrete implementations, function of the particular context. The implementation choices are based on some 'utility functions' which are defined using rules. A similar idea is proposed in [4]. The utility functions are service-specific and they must be specified by the developer. In this case, we have the same situation that is not suitable for autonomic computing: the use of service-specific rules/functions defined by the service developer.

IBM has an entire department working in this area. They have created the "IBM Autonomic Computing Toolkit". This toolkit comes with a general architecture inspired from the biological model: nerves-probes, brain-controller, and muscles-effectors. This architecture is based on the closed-loop principle and we find it very similar to the "Rainbow" architecture [6]. Unfortunately, today we do not have yet a 'completely autonomous' system, but rather some support technologies that should make it possible.

We have studied several other platforms that are not discussed here and we have observed the same thing: the adaptation control part is not suitable for autonomic adaptation because it is specific to a particular service/context and predefined. The general architectures (i.e. closed loop based) and the adaptation techniques may be reused because they do not influence essentially the adaptation autonomy.

3 Proposed Solution for Autonomic Service Adaptation

3.1 General Architecture

Figure 1 depicts the proposed general architecture that is based on the classical closed-loop (same as the biological model) control principle. In this architecture, we separate three main parts: a) the *adapted part* is the service that is executed on top of a reflexive platform allowing dynamic reconfiguration, b) the *observation part* periodically monitors the context/service state and dynamically builds a *service-context model*; c) the control part uses the service-context model in order to check the service adequacy to the context (*S-C Adequacy Verifier*) and to look for solutions if necessary (*Solution Search Engine*).

The key element of this architecture, compared to existent architectures, is the service-context model that replaces the service-specific rules and strategies. This model, which is the heart of our proposal, is explained in detail in the next section.

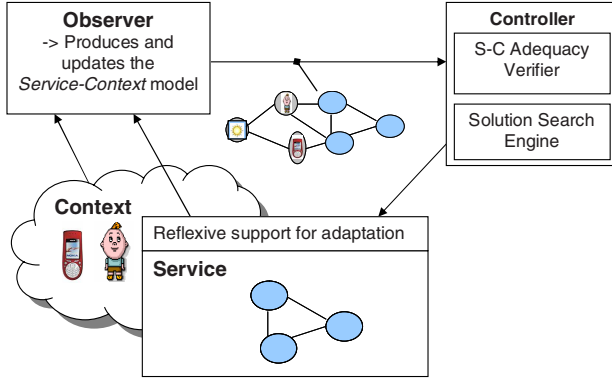


Fig. 1. General architecture for autonomic service adaptation

3.2 Service-Context Model and Profiles

In order to facilitate the model explanation, we propose to apply it for a concrete scenario. This scenario is based on a forum service. The forum architecture is composed by a client, itself composed by a message editor *TreeViewUI* and a message composer *EditorUI*, and a server: *Forum Server*.

The service meta-model describes the service architecture as a graph having as nodes the components and as arcs the interconnections, similar to an ADL (Architecture Description Language) description. We observed that a "pipe-and-filter" perspective significantly simplifies our model without limiting it. A filter has one input and one output, a sink has only one input, and a source has only one output. A complex component with several inputs and several outputs will be represented (at the service model level) as decomposed in several distinct filters.

The context is represented today using different model types: list of attributes, object oriented models, ontology based models[10], contextual graph [2]. In order to have a common model, we propose the use of the same model for the context and the service. This means, the context is also seen as an architecture composed by (interconnected) *contextual components*: user, terminal, network, environment, etc. There are two types of interrelations between software components and contextual components: a) *Information exchange*, for instance, the user exchanges information with the service through the HMI (Human Machine Interface) and b) *Resources utilization*, for instance, the service uses a certain amount of the terminal memory and a certain bit rate of a network.

A problem is that actual service and context models do not reveal the service-context interrelations. In order to solve this issue, we introduce the *profile* concept. The profile is a meta-data associated to a component (software or contextual), provided by the component developer. The role of the profile is to describe how software and contextual components interact with each other. These interactions enable us to check the service-context adequacy. For instance, if a user interacts with a service, it would be normal to use the same language and

information type (visual, voice). If a component uses the terminal memory, there must be enough free memory in order to have adequacy.

Each complex component may be decomposed in several filters. The profile describes each filter by the following elements: A) *Type*. The type (may be software or contextual); B) *Component attributes*. The component attributes are related to the whole component. Examples of such attributes are: memory, CPU, APIs, OS, and they are related usually to the physical and logical resources. Each attribute has a name and a definition domain (values); and C) *Flow attributes*. The flow attributes are related to the input and output information flows that enter or exit the component ports. These attributes characterize the information content. Some examples are: data type, language, compression, delay, bit rate. Each attribute has a name and a definition domain. For each attribute we define a *transfer function* $\mathbf{H}$, as for the electrical filters. This function indicates the relationship between the output value and the input value for a certain attribute. The $\mathbf{H}$ function may have parameters that are associated with the component parameters. If the $\mathbf{H}$ relationship cannot be known a priori, it will be marked as unknown.

In order to manipulate the service-context model, we implement it as a graph. The graph for the forum example is depicted in figure 2. The graph nodes correspond to software (white) and contextual (gray) components. A node is an object that includes all the profile attributes: component attributes, flow attributes and transfer functions. The graph arcs are all oriented and correspond either to information flows (normal arrows) or to resource utilization (dotted arrows).

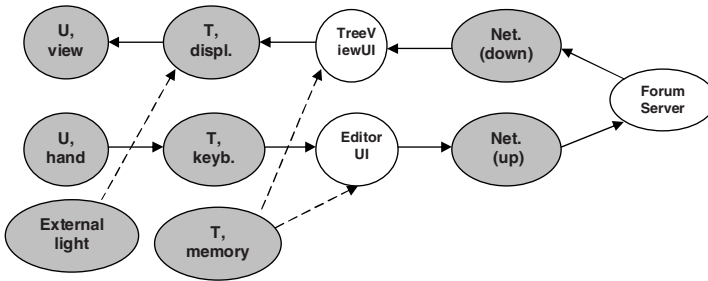


Fig. 2. Service-context model represented as a graph

Several contextual components are involved: *User* that interacts with the forum service through its HMI, *Terminal* that includes two components: the display as output device and the keyboard as input device. The terminal devices are interposed between the user and the software components. External light may reduce the screen contrast. The *Network* connects the terminal with the internet access point and finally with the server machine. Several networks may be concatenated. Uplink and downlink are seen as separated filters.

3.3 Service-Context Adequacy

The proposed service-context model allows us to check the service adequacy to the context. The first thing is to define what 'adequacy' means. In order to do that, we consider two general, service-independent rules. The first one concerns the information exchange and the second one the resource utilization. These rules and their application are described bellow.

Rule 1: A service S is adequate to its context C if the information exchanged between S and C is compatible. The information flow compatibility is checked for each flow attribute (specified in the component profiles). The compatibility validation rule is: *for each output port interconnected with an input port, the output attribute value must be included in the input interval.*

The information flow circulates directly through the interconnections (but may suffer modification through the filters). For each attribute, the chain effect is given by the mathematical composition of the transfer functions for each filter in the chain (functions $\mathbf{H}$ specified in the component profile). If a filter profile does not specify an attribute, we suppose that the filter does not affect that attribute and its output value is equal to its input value. An unspecified function $\mathbf{H}$ is equivalent to the identity function, and this allows us to compose the functions.

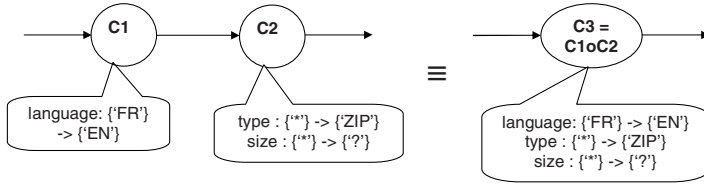


Fig. 3. Component chain composition

As it is depicted in figure 3, a chain made of two components $C1$ and $C2$ is equivalent to a component $C3 = C1 \circ C2$ (composition). The $C3$ profile may be calculated automatically and contains all the attributes of the $C1$ and $C2$ profiles. The symbol '*' means any value and the symbol '?' means unknown value. The unknown values will be detected using specialized components, the observers, (i.e. language detector). The chain flow attributes are given by the reunion of all flow attributes of each filter. If two filters affect the same attribute (fact indicated by their profiles), we use the standard mathematical function composition in order to determine the chain global transfer function. The information composition is analog to signal composition for electrical filter chains.

Rule 2: A service is adequate to its context if all the resources required by the service are offered by its context. A service is adequate to its context (physical infrastructure: terminals, machines, networks) if it does not need more resources than those available. The nature of the resources is usually additive. In some cases, the resource utilization does not fit into a simple additive relation. For instance, two visual components may be displayed on the same display at the

same time or consecutively. The required screen surface is different in the two situations. In order to enable the developer to express different situations, we offer the possibility of using different composition and validation operators, not only the addition and the inferiority operators. The attribute definitions and the operators must be shared by all component developers in order to be able to integrate components produced by different developers.

3.4 Solution Search and Application

To find a solution is: a) to choose the reconfiguration type: replacement, insertion, removal, migration and parametrization of components and b) to decide what component/interconnection to reconfigure. We will not present here advanced solution search algorithms, we will only show that the proposed model allows us to do it.

The proposed model allows us to automatically create different service-context configurations and test the service adequacy to the context. If the inadequacy is caused by information flow incompatibility, one general solution is to insert an additional filter that acts like an adapter. A translator is an adapter between two components (software or contextual) "speaking" different languages. The necessary component is searched for by its profile: the input/output function of the searched component must solve the inadequacy problem.

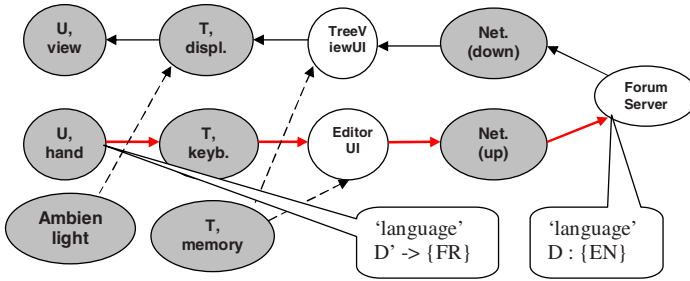


Fig. 4. Solution search for a flow related inadequacy

The right component is searched by its profile and the insertion point is searched by verifying the syntactical interface compatibility. As we see in figure 4, the graph model gives us the possibility to follow the information flows. In the forum case, a new language translator filter may be inserted after the *EditorUI* output or before the *ForumServer* input. The same idea may be used for the service output: a second translation component may be inserted after the *ForumServer* output. As we can see, the model enables us to make distinction between the user's ability to write or to read respectively (input and output) in English.

In principle, a new filter may be inserted at any point of a filter chain if two conditions are fulfilled: a) the IDL (Interface Definition Language) is compatible and b) it does not create a new inadequacy because of its own profile. An existent

filter may be replaced with another one if the same conditions are fulfilled. If several solutions exist, an additional cost function may be defined (based on component size, response time) or the user may be asked to select one.

4 Prototype

The prototype architecture is described in figure 5.

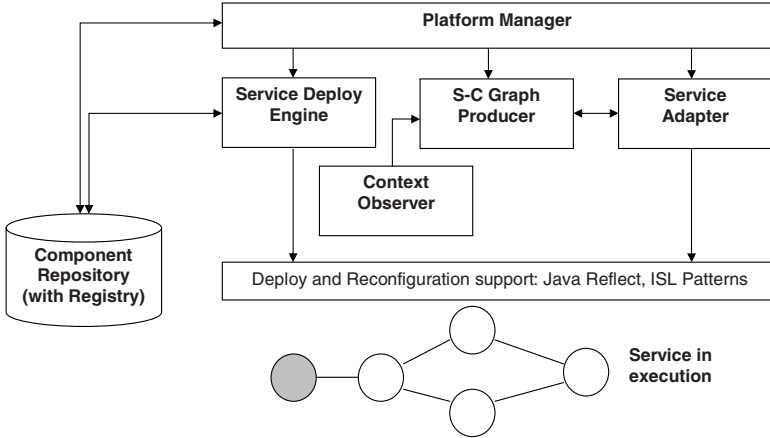


Fig. 5. Prototype Architecture

The *Component Repository* contains the available components, in our case we have placed there atomic components such as the three forum components (viewer, editor, server), a translation component, a language detector component and also the forum itself that is a composite component. Components are implemented in Java, we use CCM (Corba Component Model). Each atomic component has an IDL description and a profile described in XML. The forum is a composite component and its architecture is described using an XML based ADL. There is no need to specify profiles for composite components because their profiles are determined by the inner components' profiles (according to the graph model and the composition rules). The *Service Deploy Engine* uses Java reflection API in order to deploy the service component instances and binds the components together. When a service is deployed, the *S-C Graph Producer* initializes the service and context graph. In this case, the context is implicit: the user is always connected to the service through the HMI and each service component always uses the host resources. This example takes into account only one contextual component: the user and only one information flow attribute: the language. The *Context Observer* contains dedicated components also called 'observers' that offer the information needed in order to update the service-context graph. In this case, a graph analysis reveals the fact that we have a user and

the 'language' attribute for this user is unknown. The solution is to search an observer component capable to detect the value of this attribute. We look in the *Component Repository* and we find a language detector component. Now, the question is how to collect the necessary information in order to detect the user language? If we analyze the service-context graph (see fig. 4), we see that the informational flow passes from the user to the forum server. The language detector interface specification (IDL) does not allow us to insert it at the HMI level but we can do it at the *EditorUI* output. So, the language detector is inserted there. The value resulted at the language detector output will be used in order to update the service-context graph. The *Service Adapter* re-checks the service adequacy to the context each time the service-context graph is updated. This prototype uses only one reconfiguration type, for the moment: component insertion. Thus, to find the solution means to find the right point where to insert the component (see section 3.4). Runtime reconfiguration is applied using interaction patterns described using the ISL (Interaction Specification Language) language [1]. ISL allows us to specify a general pattern that is then used to insert new components by interception. The interception means that a message/method call between two existent components may be intercepted and redirected towards a third component. We use this mechanism when we want to insert the translation component into the forum service architecture. The *Platform Manager* implements the platform flow and assures the coherent communication between the other parts of the adaptive platform.

The platform adapts the forum service (described before) to a context based on the users's language attribute and proposes two possibilities to the user: either use a translator or leave the service unchanged. The language is detected at each message. The user may write messages in English, French and German and the service is reconfigured for each message, dynamically. Assuming that the user choses the translator, his messages are translated.

5 Conclusions and Perspectives

In this paper we have proposed a solution for autonomic, dynamic service adaptation. While most existing adaptation systems use service-specific, developer-specified adaptation rules and strategies, in our proposal we replace this by a service-context model. This model, which is dynamically updated, represents a knowledge base for the system that may reason about the service adequacy to the context and search for solutions. In our proposal, the rules are high-level defined and service-independent. In order to reveal the service to context interactions, we use *profiles* that describe how a component modifies the information and what resources it requires. One of the most important elements of a profile is the transfer function between input and output ports. The profiles may be composed in a component chain as we compose ordinary mathematical functions. As a strategy, we have tested the component insertion but the model does not restrict the possibility of using any strategy. The strategy search itself is a separate problem and remains as future development. Implementing a simple prototype we have

shown that a service can be adapted without using service-specific rules and developer-specified strategies, thus we consider our model suitable for autonomic adaptation. In future, we intend to focus on the solution search issues: strategy selection, algorithm improvement, AI techniques, feedback based learning, mechanisms allowing us to compare alternative solutions and select the best one.

References

1. Blay-Fornarino, M., Charfi, A., Emsellem, D., Pinna-Dery, A.-M., Riveill, M.: Software interactions. *Journal of Object Technology* 3(10), 161–180 (2004)
2. Brezillon, P.: Context-based modeling of operators practices by contextual graphs. In: *Proceedings of the 14th Mini Euro Conference, Human Centered Processes*, pp. 129–137 (May 2003)
3. Dey, A.K.: Understanding and Using Context. *Personal and Ubiquitous Computing Journal* 5(1), 4–7 (2001)
4. Dubus, J., Merle, P.: Vers l'auto-adaptabilité des architectures logicielles dans les environnements ouverts distribués. In: *Proceedings of the 1ere Conference Franco-phone sur les Architectures Logicielles, CAL'06, Nantes, France, Hermes Sciences*, pp. 3–29 (September 2006)
5. Floch, J., Hallsteinsen, S., Stav, E., Eliassen, F., Lund, K., Gjorven, E.: Using Architecture Models for Runtime Adaptability. *Software, IEEE* 23(2), 62–70 (2006)
6. Garlan, D., Cheng, S.W., Huang, A.-C., Schmerl, B.R., Steenkiste, P.: Rainbow: Architecture-based self-adaptation with reusable infrastructure. *Computer, IEEE* 37(10), 46–54 (2004)
7. Keeney, J.: Completely Unanticipated Dynamic Adaptation of Software. PhD Thesis, University of Dublin, Trinity College, Distributed Systems Group (October 2004)
8. Kephart, J.O.: Research challenges of autonomic computing. In: Inverardi, P., Jazayeri, M. (eds.) *ICSE 2005. LNCS*, vol. 4309, pp. 15–21. Springer, Heidelberg (2006)
9. Lemos, R., Fiadeiro, J.L.: An architectural support for self-adaptive software for treating faults. In: *Proceedings of the first Workshop on Self-healing systems*, pp. 39–42. ACM Press, New York (2002)
10. Wang, X.H., Gu, T., Zhang, D.Q., Pung, H.K.: Ontology-Based Context Modeling and Reasoning using OWL. In: *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*, pp. 18–22 (2004)

Middleware Based Context Management for the Component-Based Pervasive Computing

Di Zheng, Jun Wang, Yan Jia, Wei-Hong Han, and Peng Zou

School of Computer Science,
National University of Defence Technology,
Changsha, Hunan, China 410073
dizheng@nudt.edu.cn

Abstract. Ubiquitous computing allows application developers to build a large and complex distributed system that can transform physical spaces into computationally active and intelligent environments. Ubiquitous applications need a middleware that can detect and act upon any context changes created by the result of any interactions between users, applications, and surrounding computing environment for applications without users' interventions. The context-awareness has become the one of core technologies for application services in ubiquitous computing environment and been considered as the indispensable function for ubiquitous computing applications. The need for high quality context management is evident to the component-based middleware for it forms the basis of the component adaptation and the component deployment in the pervasive computing. Therefore, we suggest a holistic approach where context management is an integral part of a more comprehensive adaptation enabling middleware, thus enabling the development and support of context-aware, component-based applications.

1 Introduction

With the technical evolution of wireless networks, mobile and sensor technology, the vision of pervasive computing is becoming a reality. The paradigm for pervasive computing aims at enabling people to contact anyone at anytime and anywhere in a convenient way. And it also brings the new challenges to traditional applications [1, 2]. In this environment, the applications should become context aware for the reason that the resources (e.g. memory, battery, CPU) of the mobile devices may be limited and the application execution context (e.g. user location, device screen size) is variable[3]. Therefore, the applications need adapt their behaviors basing on corresponding context information. The context-awareness has become the one of core technologies for application services in ubiquitous computing environment and been considered as the indispensable function for ubiquitous computing applications.

At the same time, middleware is a widely used term to denote generic infrastructure services above operating system and protocol stack. The role of the middleware is to ease the task of designing, programming and managing distributed applications by providing a simple, consistent and integrated distributed programming

environment. As the name "middleware" suggests, it uses the hierarchical model, a layer of services which sits between applications and operating systems and it can allow the clients to invoke the operations on distributed objects without concern for object location, programming language, OS platform, communication protocols and interconnects, and hardware. Therefore, ubiquitous applications need a middleware that can detect and act upon any context changes created by the result of any interactions between users, applications, and surrounding computing environment for applications without users' interventions.

In order to provide context-awareness services, the middleware platform supporting ubiquitous computing should be able to recognize contextual changes so that applications use contexts for evaluating new environments and finding an appropriate action by the result of evaluation for these changes. Furthermore, many of the current applications are component-based and the management of the components in the pervasive computing environment will be more difficult than ever. However, the existing component based middleware (e.g., .NET[4], Enterprise JavaBeans [5], and the CORBA Component Model [6]) do not take charge of the context information. The OMG Specification of the Deployment and Configuration [7] and the deployment tool of COACH [8] only support the context that presents the deployment target environment and do not consider the context in general case. Therefore, we put forward a middleware based context management infrastructure for component-based pervasive computing in this paper and discussed according mechanisms.

2 Architecture of the Middleware Based Context Management for Component-Based Pervasive Computing

2.1 Component Based Middleware StarCCM

In terms of middleware, lots of emphasis has been given to enterprise (or server-side) component technologies, such as Enterprise Java Beans or the CORBA Component Model. As depicted in figure 1, in previous work we have developed a component based middleware StarCCM which conform to the CORBA Component Model. The components execute inside a container, which provides implicit support for distribution in terms of support for transactions, security, persistence and resource management. This offers an important separation of concerns in the development of business applications; i.e. the application programmer can focus on the development and potential re-use of components to provide the necessary business logic, and a more "distribution-aware" developer can provide a container with the necessary non-functional properties. Containers also provide additional functionality including life-cycle management and component discovery.

The OMG Specification of the Deployment and Configuration presents a data model for the description of a deployment plan which contains information about artifacts that are part of the deployment, how to create component instances from artifacts, where to instantiate them, and information about connections between them. This specification also presents a data model for the description of the domain into which applications can be deployed as a set of inter-connected nodes with bridges

routing between inter-connects. However, these data models are still insufficient for the context of the mobile devices and do not support a description of the rules achieving the adaptation of the deployment.

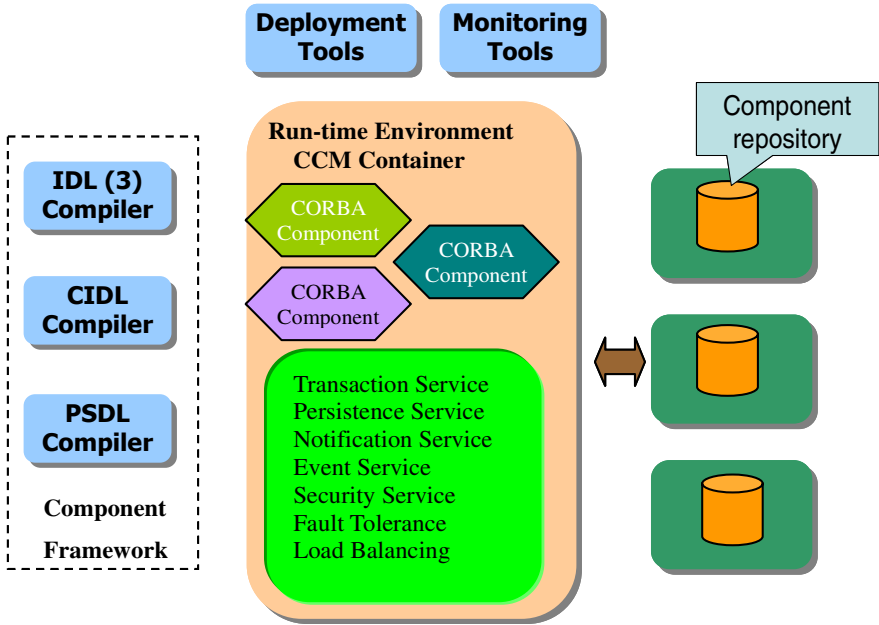


Fig. 1. The Architecture of the StarCCM

2.2 Architecture of the Middleware Based Context Management

The overall middleware architecture is shown in the figure 2. The core provides the fundamental platform-independent services for the management of applications, components and component instances. The core relies on the basic mechanisms for instantiation, deployment and communication provided by the distributed computing environment.

StarCCM Core Component Management provides platform-independent services for the management of the component based applications, components and component instances as depicted in the figure 1. It also provides uniform platform-independent access to the execution platform resources. Furthermore, the middleware offers the other three core services:

- The Context manager which monitors the user and the execution context for detection of relevant changes.
- The Adaptation Manager which reasons about the impact of the changes and decides about appropriate adaptations based on architectural description of component properties.
- The Configurator which reconfigures the application variant to put the decided adaptations into effect.

Context Manager is responsible for sensing and capturing context information and changes, providing access to context information (pull) and notifying context changes (push) to the Adaptation Manager. The Context Manager is also responsible for storing user needs and preferences on application services. The Context Manager should provide flexible context sensing. We recommend the Context Manager to be developed as a Component Framework where new context sensor components can be plugged in. The Context Manager may provide advanced reasoning operations on context. For example, it may aggregate complex context elements from elementary context elements or derive user needs from context. The Context Manager may also keep track of context change history.

Adaptation Manager is responsible for reasoning on the impact of context changes on the application(s), and for planning and selecting the application variant or the device configuration that best fits the current context. As part of reasoning, the Adaptation Manager needs to assess the utility of these variants in the current context. The Adaptation Manager produces dynamically a model of the application variant that best fits the context. We use the term “configuration template” to denote a model of an application variant where all variation points have been resolved.

Configurator is responsible for coordinating the initial instantiation of an application and the reconfiguration of an application or a device. When reconfiguring an application, the Configurator proceeds according to the configuration template for the variant selected by the Adaptation Manager. Thus, the Configurator carries out the adaptations decided by the Adaptation Manager by applying the configuration template. The Adaptation Manager and the Configurator are tightly coupled as they operate on a common information element: the configuration template.

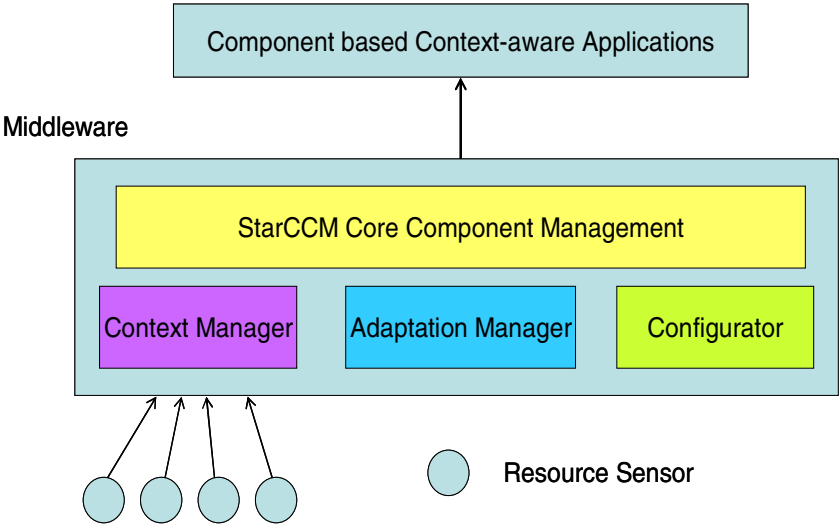


Fig. 2. Architecture of the Context-Aware Middleware

3 Context Management

3.1 Context Management Requirements

In order to adapt to different applications, the context manager must meet the following requirements:

Define context representations that the middleware can use. This implies that the context model maintained by the context manager must make no assumptions on what is context in specific applications (e.g. network context might be extremely important in one application, but less important in another). Instead the context model should define a structure which allows developers to express their application-dependent context, while at the same time remain applicable to other components of the middleware.

Support context sensing. The context manager should provide support for the definition of context sensors [9, 13, 14] enabling the sensing of information (e.g. user activity, user environment, mobile device resources, availability of various networks, network service quality and cost, and system infrastructure.)

Support context reasoning: In order to reduce context noise, the context manager should allow the creation and deployment of context reasoning mechanisms. Context reasoning can be used to derive higher level context information (e.g. a meeting is taking place) from lower level context information (such as GPS coordinates, or number of persons in the room). It should be easy for other middleware components to specify what particular context information they are interested in, and what changes in context are relevant to them. Other middleware components should not be aware of the internal process in the context manager for obtaining and managing this information. As pointed out by Austaller et al [14], in order to be useful, context management services need to implement strategies for reducing context noise.

These requirements are not exhaustive. Depending on the nature and the goals of the target application, additional features might also be required, to enable, for example, inter-operability between devices.

3.2 Context Types

One of the most critical issues that is important to be initially defined in a Context-aware application implementation is the type of the used contextual information. In most of the cases, the context types can classify the application's infrastructure by enabling or disabling underlying technologies (e.g. wireless communication that utilizes WiFi and Bluetooth technologies). Even though, it is important to identify all the possibilities due to the fact that we need to make the Middleware be able to support any type of contextual information. The analysis presented in [12] provides the following list of contextual information types that is quite interesting:

Physical context: The light and noise levels, traffic conditions, and temperature are some of the physical context information. This information describes environmental factors that can be evaluated by using specialized hardware mechanisms (e.g. noise and light sensors, etc). This contextual information is collected by using specialized Physical sensors.

Computing context: This contextual information is related to computable or measurable information that is retrieved by calculating or checking routines. Such information is the network connectivity, the communication costs and bandwidth, but also the nearby resources (like printers, displays, etc). This contextual information is constructed by using Virtual and Logical sensing mechanisms. Virtual sensors source context data from software and Logical sensors make use of a couple of information sources.

User context: This information refers to the user's profile by focusing on the user needs, preferences, interests, expertise, workload, tasks etc. The user's context is closely related to the "personalization" issue that is enabling the ability of adapting products and services either to large user groups, smaller interest groups, or the individual user. This task usually depends on the competitor's behavior, the internal resources, the market, and the customer preferences, and intends to establish a better relationship between the customer/user and the system.

3.3 Context Sensoring

The contextual information that is previously mentioned can be retrieved by using a specialized set of mechanisms that we can refer as sensors, depending on the context type. In the following paragraphs we try to categorize sensors based on the complexity of the given contextual information. This taxonomy provided in [12] that classifies context sensing based on low and high level context elements:

- **Sensing Low-level Contexts:** In this contextual information, a set of sensors is responsible to measure environmental parameters. The calculation of such parameters is quite simple and usually the result is absolutely correct.
 - a) **Time:** Most applications use their local clock to check schedule tasks or current activities. For example the Active Badge and Cyber guide use time in order to adapt the application behavior.
 - b) **Sensing the Location:** Most of the applications that we analyzed in the previous section use location information to adjust their behavior (e.g. the Call Forwarding application uses location information to find the nearest phone to the user). Such applications are using outdoor sensing mechanisms (e.g. GPS systems) and indoor sensing mechanisms (using transmitters and receivers with radio signals).
 - c) **Nearby objects:** This is feasible if the system records the position of the other objects. For example the Cyber guide application stores the information of attractions in a local database. When the user is in a specific location, the nearest attractions are detected and presented to the mobile device.
 - d) **Network bandwidth:** Such systems use kernel functions that measure the bandwidth and notify applications.
 - e) **Light level, Sound level, Temperature and Pressure:** are measured by electronic sensors.
- **Sensing High-level Contexts:** In this type of contextual information the complexity in the evaluation is higher. These sensing mechanisms are combining the Physical context with Computing and User's context in order to construct "intelligent" sensing mechanisms. Such mechanism can be used to support for example:

- a) Machine vision, based on camera technology and image processing.
- b) Artificial Intelligence techniques to recognize complex context by combining several simple low-level sensors.

Context sensing is just a supporting mechanism that provides information to the adaptation process. Adaptation is the major tasks that can be efficiently supported by using low-level sensors. Even if we are implementing low-level sensors we have avoided using location-awareness in our scenarios. In order to support such type of information, it was required to use GPS systems or radio transmitters and receivers. Therefore, we use sensors that can be easily implemented by using kernel functions (time, bandwidth, available networks, etc) that eliminate costs and efforts.

3.4 Context Modeling

The variety of contextual information can support a broader set of adaptation rules based on different combinations. But, in order to handle the contextual information that is provided by the sensing mechanism, it should be stored in data structures thanks to a given schema or ontology. There are different ways with which such information can be expressed and modeled. Modeling of location information is quite complex. There are some strict context models that are used in most of the applications that are dealing with location information. We have mentioned in the previous subsection that we are not using location information in our project because it is resource consuming task. Thus we are using a simple model that despite its simplicity, it can be easily modified to model location. In most of the cases, these structures fall into the following categories:

- Key-value pairs: The contextual information is stored in a key-value pair that is using the key to refer to the environmental variable and the value of the variable holding the actual context data. In most of the cases such context-aware applications are using pattern-matching queries to notify their adaptation mechanisms.
- Tagged encoding: The contexts are modeled as tags and corresponding fields.
- Object-oriented model: The contextual information is embedded in the states of the object, and the object provides methods to access and modify the states.
- Logic-based model: Context data are expressed as facts in rule-based systems.

We primarily use the key-value pairs that can be easily embedded into modeling tools. This model enables integration with the storing mechanism (use contextual historical data in reconfiguration decision-making), and the adaptation mechanisms (utility functions that can use the key-value pairs for evaluating statements).

The context information is encompassed in Context Elements. The context elements can be composite (i.e. elements within elements). Context elements extend the Entity of Interest component. Our middleware conceives a software system and its context to be a set of interacting components. Entities of interest (for the middleware) are then either software entities or context entities.

Elements have Values. The value is the information available in the element. Every value instance encapsulates the actual data. An element can have several values. For example, the WLAN element above can have two values corresponding to bandwidth (e.g. 1Mbit) and cost. In their simplest form, a context element consists of an

identifier, and a value entry. The context elements must be uniquely identifiable, so that the adaptation manager can map application variants need for context information, to context elements administered by the context manager. Values are also associated with metadata that functions as Quality of Context parameters. Such metadata is important when performing reasoning on context information.

3.5 Architecture of the Context Manager

As we have discussed above, with the development of the pervasive computing, dealing with the direct access to context information during the development of a context-aware service or application is expensive, error-prone, and the applications will be complex and non-portable. So we must try to separate the context-aware infrastructure from the context-aware applications. The context-aware infrastructure deals with the activities like communicating with context sources, collecting context data, storing and managing context data, and finally transforming context data into higher level context and refine them according to the applications needs. All the actions are transparent to the end users and the applications supported may be more scalable. The middleware provides abstractions for the fusion of the sensor information to obtain high-level context information whereas the context-aware applications are responsible for adaptation and reaction to context changes. Furthermore, the middleware allows designers to build context-aware applications and to interact with context-aware services by providing a meta-model for describing context and adaptation policies and the middleware communicates with the underlying execution environment to collect context information, processes them to identify relevant changes, and propagates those changes up to the context-aware components or adapt the applications to context changes.

As depicted in figure 3, the context manager provides two important interfaces to context clients, the context listener and the context access. The clients can either request

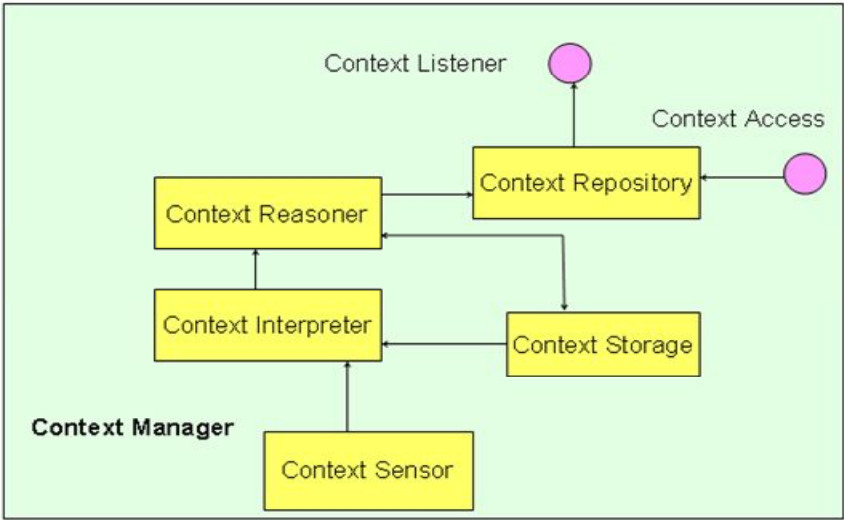


Fig. 3. Architecture of the Context Manager

to be notified of certain events using the context listener interface (push), or they can use the context access interface to explicitly query context information (pull).

The context repository is the main entry point for clients to the context manager. The primary tasks of the context repository are to maintain a context model, register and notify listeners, give access to context elements, and keep registry of available components (sensors, reasoners and storage). In order to get access to a specific context element, a context client (such as the adaptation manager) either registers as an observer to that element, or directly accesses it via the context access interface.

The context sensors are components which provide context information to the context repository (a type of context source). Sensors can be wrappers around specialized hardware drivers, or legacy code used for monitoring context, such as battery, memory, and network information. This component is in charge of discovering, managing sensors and collecting raw data from them through agents. It shields the heterogeneity of all kinds of sensors and provides universal interfaces and transmits the raw context data. At the same time, it does some preprocess work, for example, it will filter the data by certain rules, wipe off redundancies and encapsulate them in a unified format. This component has to activate and deactivate agents according to the context to which applications or services are sensitive.

The context interpreter abstracts raw or low level context information into richer or higher level information according to interpretation rules described by using the context meta-model provided by the middleware. Furthermore, this component can fuse kinds of basic information into more comprehensive elements; for instance, it can merge temperature, luminance, humidity and other context within the room into a ROOM context.

The context reasoners can produce one or more context elements using other context elements as input. This component is used to filter context information to determine relevant ones, and notify the subscribed component of these context changes. It is an important task for the context manager to reduce context noise, by filtering out unnecessary context information, which is not relevant for adapting applications. The adaptation manager should only be notified when a significant change occurs in context. Consequently, the context reasoners need to implement filtering mechanisms. These mechanisms can vary from very simple, rule-based logic, to more advanced techniques. The reasoners are “plug and play” in order to make it possible to target reasoners according to different needs and domains. For example, the applications can provide the context manager with Quality of Context (QoC) requirements, such as precision and refresh rate.

The context storage keeps the track of historical context information which is often required in order to determine trends in context data (for example trends in user behaviour, network stability, etc). The need for storage mechanisms was shown in Dey [9], where context widgets stored all context information they sensed.

4 Future Works

SoA has been suggested for use in context aware computing (see for example the Service-Oriented Context-Aware Middleware, SOCAM, by Gu et al [15]). A promising application of these technologies could for example involve the formation

of ad-hoc networks between context manager services, enabling sharing of relevant information. SoA in isolation does not accomplish this, as it needs better ways of describing services behavior and the Quality of Service (QoS). Additionally, further work is required to guarantee the semantics of the data used by the service. Although Ontologies and SoA have their limitations, they both appear to be promising approaches for context management in the area of adaptation enabling middleware solutions. Their usefulness is particularly illustrated when we consider collaborative context management, among distributed peers, forming ad-hoc networks.

5 Conclusions

Ubiquitous computing allows application developers to build a large and complex distributed system that can transform physical spaces into computationally active and intelligent environments. Ubiquitous applications need a middleware that can detect and act upon any context changes created by the result of any interactions between users, applications, and surrounding computing environment for applications without users' interventions. The context-awareness has become the one of core technologies for application services in ubiquitous computing environment and been considered as the indispensable function for ubiquitous computing applications. The need for high quality context management is evident to the component-based middleware for it forms the basis of the component adaptation and the component deployment in the pervasive computing. Therefore, we suggest a holistic approach where context management is an integral part of a more comprehensive adaptation enabling middleware, thus enabling the development and support of context-aware, component-based applications. We have proved that such an approach is feasible, by developing and evaluating several different applications. Experimenting with the approach, we have identified ontologies and service oriented architectures as relevant approaches in relation to adaptation enabling middleware.

Acknowledgements

This work was funded by the National Grand Fundamental Research 973 Program of China under Grant No.2005cb321804 and the National Natural Science Foundation of China under Grant No.60603063.

References

1. Weiser, M.: Some Computer Science Problems in Ubiquitous Computing. *Communications of the ACM* 75–84 (1993)
2. Roy, W., Trevor, P.: System Challenges for Ubiquitous & Pervasive Computing. In: *Proceedings of the 27th International Conference on Software Engineering*, pp. 9–14 (2005)
3. Dey, A.K.: Understanding and Using Context. *Personal and Ubiquitous Computing* 5, 4–7 (2001)
4. Microsoft Corporation: An Introduction to Microsoft.NET, White Paper (2001)
5. Sun Microsystems: Enterprise JavaBeans Specification 2.0 (2002)

6. OMG CORBA Components Version 3.0. An Adopted Specification of the Object Management Group (2002)
7. OMG. Specification for Deployment and Configuration of Component Based Distributed Applications (2003)
8. IST. COACH WP2: Specification of the Deployment and Configuration, D2.4 (2003)
9. Dey, A.: Providing Architectural Support for Building Context-Aware Applications. Ph.D. Thesis Dissertation, College of Computing, Georgia Tech (2000)
10. Dey, A., Abowd, G.D.: Towards a Better Understanding of Context and Context Awareness. Technical Report, GITGVU-99-22, Georgia Institute of Technology (1999)
11. Schmidt, A.: Ubiquitous Computing- Computing in Context. Ph.D. Thesis, Lancaster University, UK (2002)
12. Chen, G., Kotz, K.: A Survey of Context-aware Mobile Computing Research. Technical report TR2000-381. Department of Computer Science, Dartmouth College, Dartmouth (2000)
13. Huebscher, M.C., McCann, J.A.: An Adaptive Middleware Framework for Context-aware Applications. *Personal and Ubiquitous Computing* 10, 12–20 (2006)
14. Austaller, G., Kangasharju, J.: Using Web Services to Build Context-Aware Applications in Ubiquitous Computing. In: *The 4th International Conference of Web Engineering*. Munich, Germany, pp. 483–487 (2004)
15. Gu, T., Pung, H.K., Zhang, D.Q.: A Service Oriented Middleware for Building Context-aware Services. *Journal of Network and Computer Applications* 28, 1–18 (2005)

Building Autonomic and Secure Service Oriented Architectures with MAWeS

Valentina Casola¹, Emilio Pasquale Mancini², Nicola Mazzocca¹,
Massimiliano Rak³, and Umberto Villano²

¹ Dipartimento di Informatica e Sistemistica,
Università degli studi di Napoli Federico II
`{casolav,n.mazzocca}@unina.it`

² RCOST and Dip. di Ingegneria,
Università del Sannio

`{epmancini, villano}@unisannio.it`

³ Dipartimento di Ingegneria dell'Informazione,
Seconda Università di Napoli
`massimiliano.rak@unina2.it`

Abstract. Service-oriented architectures (SOA) and, in particular, Web Services designs are currently widely used for the development of open, large-scale interoperable systems. In those systems performance, security and trustability are challenging open issues. As regards performance, in Web Services designs, abstraction layers completely hide the underlying system to all users, and classical techniques for system optimization (such as *ad-hoc* tuning, performance engineered software development, ...) are not applicable. As far as security is concerned, the need of trust mechanisms in open environments is a well-known problem, widely discussed in the literature, but few techniques for security evaluation are available. In this paper we propose an integrated solution to optimize performance and to guarantee security of Web Services architectures. The proposed solution is based on a framework, MAWeS, which adopts simulation in order to predict system performances and adopts policies for security description and evaluation.

1 Introduction

Service-oriented architectures (SOA) and, in particular, Web Services designs are currently widely used for the development of open, large-scale interoperable systems. As a matter of fact, many enterprises rely on them to interconnect critical services and to provide added quality to their customers [1,2,3,4]. However, reliability, availability, performance and security of these architectures are completely open issues. As regards performance, in Web Services designs abstraction layers completely hide the underlying system to all users, and classical techniques for system optimization (such as *ad-hoc* tuning, performance engineered software development, ...) are not applicable. As far as security is concerned, the need of trust mechanisms in open environments is well-known problem, widely discussed

in the literature. Different approaches to define and assess trust have been proposed [5,6,7,8]. In particular, the concept of trust includes different aspects such as access control, reputation, security and service level agreements [9,6,10,11].

The object of this paper will be to study an integrated solution to performance and security of Web Services architectures, which is based on the use of an autonomic framework exploited to auto-configure and to auto-tune the system, guaranteeing optimal performance and fulfillment of given security levels.

Autonomic computing [12,13,14,15], whose name derives from the autonomic nervous system, aims to bring automated self-management capabilities into computing systems. Autonomic capabilities are usually classified in four different categories:

Self-configuring. The system can dynamically adapt to changing environments;

Self-healing. The system can discover, diagnose and react to disruptions;

Self-optimizing. The system can monitor and tune resources automatically;

Self-protecting. The system can anticipate, detect, identify and protect itself against threats.

Most of the times, the optimizations carried out by a “self-optimizing” system are related to one or several performance metrics (i.e., response time, throughput, quality of service, ...). However, optimizations are not necessarily linked only to system performance, as the object of optimization could also be to obtain a given security level, or to reconfigure for performance, but in such a way that the desired security level is guaranteed. For example, given a web application that relies on several component web services, replicated and variously distributed on multiple servers, an optimization step may be to find out the best-performing configuration of component services, in the current or predicted load conditions, such that the composed service guarantees an acceptable security level.

In previous papers, we have introduced MAWeS (MetaPL/HeSSE Autonomic Web Services)[16], a framework whose aim is to support the development of self-optimizing autonomic systems for Web service architectures. It adopts a simulation-based methodology, which allows to predict system performance in different status and load conditions. The predicted results are used for a feedforward control of the system, which self-tunes *before* the new conditions and the subsequent performance losses are actually observed.

In this paper we describe an extension of the MAWeS framework and its use for the development of autonomic SOA applications that optimize themselves in a proactive way, both for performance and security. In addition to the existing tools, adopted by MAWeS to simulate multiple system configurations and to choose the optimal one, a new security evaluation component has been developed that provides information on the security level of the various system configurations. This component is based on a policy-based approach, and relies on the methodology previously published in [17]. The combination of security and performance predicted figures for multiple possible system configurations are exploited by the MAWeS optimization engine. This performs the feedforward control of the system taking into account the system state, the future load

conditions and the possible behaviors of the service to be carried out. So the system autonomically self-tunes before the new load conditions are actually observed, settling in a configuration that guarantees optimal performance without decreasing the system security level.

The remainder of the paper is structured as follows. Sections 2 and 3 describe the security and performance evaluation issues in Web Services Architectures and introduce the HeSSE/MetaPL prediction methodology and the security evaluation methodology. In Section 4 we describe the MAWeS framework with the new security extension, explaining how it can be used to choose the best configuration in terms of both performance and security requirements. Then we present a detailed description of the framework components, of the integration with the security evaluation component, of their behavior and of their usage. Finally, some conclusions are drawn and the directions of our future work are outlined.

2 Security and Trustability in Web Services

The need of trust mechanisms in open environments is a common problem, widely discussed in the literature. Different approaches to define and assess trust have been proposed [5,6,7,8]. In particular, the concept of trust includes different aspects such as access control, reputation, security and service level agreements [6,9,10,11].

In this paper we are interested in trustability concepts in open and distributed systems and, in particular, in Service Oriented Architectures, often implemented by means of Web Services technologies. IBM and Microsoft have recently proposed a reference architecture specification and a “roadmap for developing a set of Web Service Security specifications” [18]. The proposed architecture stems from the OASIS WS-security specification, which aims at describing enhancements to SOAP messaging and providing quality of protection through message integrity and message confidentiality. The reference architecture arranges available specifications on security and trust in layers.

The WS-Security framework [19] defines the specifications for the construction of Trusted Domains (TDs). A TD is “an administered security space in which the source and the target of a request (i.e., requester and provider agents) can determine and agree whether particular sets of credentials from a source satisfy the relevant security policies of the target”. Inside a TD, when a requester R requests a service to a provider P, WS-Security provides the information needed to establish if both R and P have the needed credentials and respect their policies.

The WS specifications need to be implemented to build a secure system and grant secure and trusted communications. The resulting trust communication model is illustrated in [19]; note that each *Server Provider* (Web Service) publishes its own security policy and implements access control mechanisms, and the *Service Requester* needs to hold the specified credentials. Security credentials are enveloped in security tokens that are managed by a Trusted Third Party (the Security Token Service). We explicitly note that the policy contains not only security provisions but also non-functional requirements, as quality of service or

Service Level Agreements. It is manually evaluated by the requesters when they need to choose a service for cooperation and to extend their trust to it.

In a service oriented scenario, a Requester exploits component services in order to complete its task. In general, the component services are variously distributed and, possibly, offered by different providers with a different *security level*, i.e., the amount of security being provided (as defined in [8]). State of the art WS Specifications allow to find a service that offers requested functionalities but currently no automatic tool is available to evaluate if the chosen service is able to guarantee minimal security requirements.

It is a common approach to describe security requirements through a policy, but current solutions make the user responsible for policy comparison and for deciding whether to trust or not a provider on the basis of its exposed policy. So, the general trustability problem can be related to the problem of comparing the exposed policies and classifying the related security levels.

Anyway, due to the complexity of a real-world policy, the process of policy comparison may be very complex and error prone. Furthermore, the need for human intervention in a web services transaction is an undesirable condition as far as the interoperability requirements of third generation Web Services is concerned. To this aim, we have developed a security evaluation methodology to formalize policies, and to evaluate the security level associated to the corresponding service, as shown in next sections.

3 Self Optimization and Trusting

In a typical real-world scenario, a web application exploits component services in order to complete its task. Component services could be atomic or composed of other services, and they could be variously distributed and, possibly, offered by different providers with a different *security level* and with different load conditions.

Figure 1 tries to summarize the problem that we address; it shows two main actors, a service requester (R) (the web application or the composed service) looking for a specific component service, and a set of service providers (Pi) offering such service. The requester searches for a service that fulfills a set of desired security requirements expressed by means of a policy, the *Requested Policy (RP)*, and optimizes the overall response time. On the other hand, each service provider offers the service with different security levels, exposing them through published policies, the *Offered Policies (OPs)*. On the other side, performance parameters are not included in the policy but need to be evaluated or predicted with external tools.

The main questions to be answered are:

1. Does a service exist with an associated OPi that respects the security requirements needed by R?
2. If multiple configurations are available for such a service, which is the best in terms of performance and security?

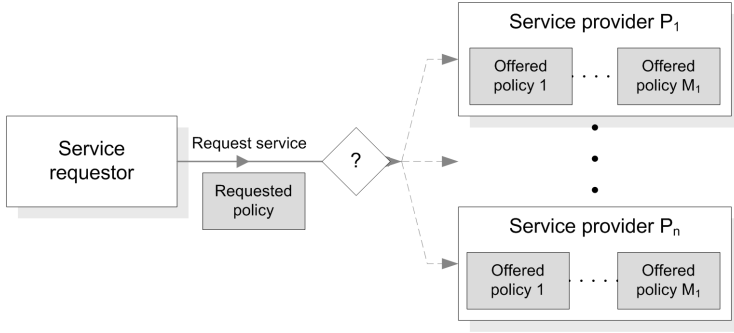


Fig. 1. The problem of trusted security requests

As regards the first point, we proposed an evaluation methodology for characterizing a policy and defining a metric to evaluate the security level associated to a service [17]. As for the second point, we implemented MAWeS, a framework for the choice of the best configuration of web services to optimize the overall performance parameters. In this paper, we will present an enhancement of MAWeS able to choose the best configuration that also guarantees that the minimal security requirements are fulfilled. In the following subsections we will briefly summarize the adopted security and performance evaluation methodologies.

3.1 The Security Evaluation Methodology

The methodology core is the Reference Evaluation Model (REM), defined as the following triplet:

$REM = \langle Formalization; Technique; ReferenceLevels \rangle$

where:

Formalization represents the formal representation of the policy. A formalized policy may be expressed by a tag language (an XML schema or a DTD) or by a mathematical representation (such as a matrix of given dimension). We will adopt the standard WS-Policy language to describe the security features of each service.

Technique represents the evaluation technique that can be applied to compare policies; it strictly depends on the policy formal representation. We will adopt a distance criterium to numerically evaluate and compare policies.

Reference Levels are different policies, which represent different security levels.

Applying the REM, we are able to obtain the so-called Global Security Level (GSL), which is a value associated to the overall security provided by a system implementing the policy.

The detailed description of the methodology is out of the scope of this paper, but we will implement a security evaluation component to ascertain if a service provider is able to guarantee a required security level. The interest reader is referred to [17] for details.

3.2 The Performance Evaluation Methodology

In previous papers [16,20] we presented a framework, MAWeS, able to build self-optimizing Web Services-based applications. The focus in the original framework was on time-related metrics for optimization (application response time, system throughput, ...), while the security problem was not dealt with.

The MAWeS Framework is based on two existing technologies: the MetaPL language and the HeSSE simulation environment [16]. MetaPL is an XML-based metalanguage for parallel programs description, which, similarly to other prototype-based languages, can be used when applications are in the design phase or when they are not completely available. It is language-independent, and can be adopted to support different programming paradigms; it is structured in layers, with a core that can be extended through *Language Extensions*, implemented as XML DTDs. These extensions introduce new constructs into the language. Starting from a MetaPL program description, a set of extensible filters makes it possible to produce different program *views*.

Filters can be used to generate views for program comprehension or documentation purposes, but also to produce traces of program activity annotated with timing information. These traces are amenable to direct simulation in HeSSE, thus making it possible to obtain fairly reliable predictions of program performance, even at the very first steps of code development.

HeSSE is a simulation tool that allows to simulate the performance behavior of a wide range of distributed systems for a given application, under different computing and network load conditions. It makes it possible to describe Distributed Heterogeneous Systems by interconnecting simple components, which reproduce the performance behavior of a section of the complete system (for instance a CPU, a network ...).

The MAWeS framework relies on MetaPL descriptions, describing the software structure, and on HeSSE configuration files, describing the hardware/software execution environment, to run HeSSE simulations and to obtain performance data. Through the execution of multiple simulations, corresponding to different parameterized configurations, MAWeS can choose the parameter set that optimizes the software execution according to one or several criteria (e.g., shortest response time).

The MAWeS framework makes it possible to optimize Web Services applications at two different, non exclusive, levels:

Application Level. The autonomic system affects user applications, modifying their resource use and the order of the actions they perform. Examples of use of MAWeS for application-level optimizations are presented in [16].

Service Level. The autonomic system affects the service provider, i.e., the optimization actions have impact on the tuning of service parameters and on the workload management policies of the set of offered services. Examples are presented in [16,20].

4 Security MAWeS Architecture

In the following, we will describe the new architecture of the framework, dealing with both self-optimization and trusting features in an integrated way.

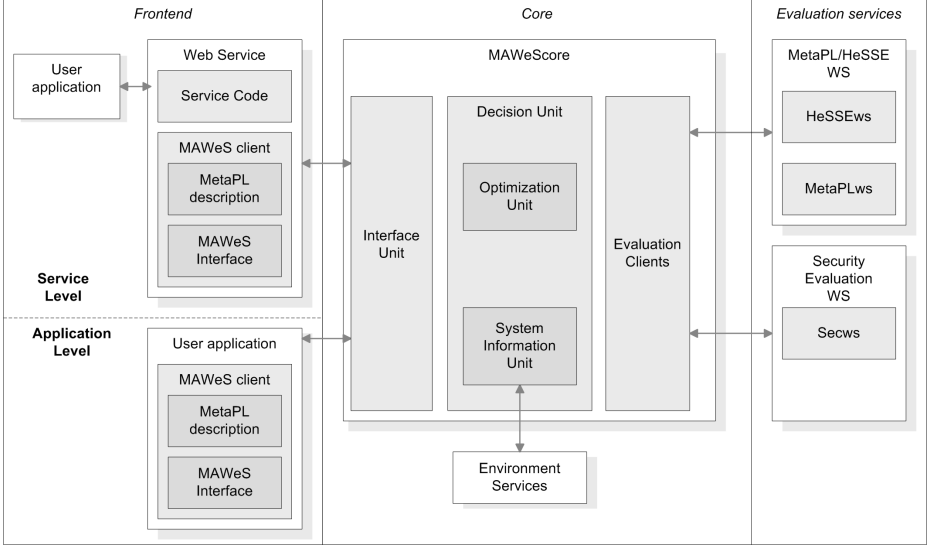


Fig. 2. The MAWeS Framework Basic Architecture

As shown in Figure 2, the Framework underlying architecture is structured in three layers:

Frontend: Made up of the software modules used by the requester (i.e. web applications or other services) to access the MAWeS services;

Core: Composed of the software that makes optimization decisions;

Evaluation Services: The services that perform the system evaluation in order to make decisions.

The layers will be described in detail in the following subsections.

4.1 MAWeS Frontend

The MAWeS FrontEnd has not been changed, as compared to the one in the previous versions of the framework. Hence, for brevity's sake, here we will give only some essential information about it.

Self-optimization services may be offered, as previously pointed out, at application or service level. In the first case (application level optimization) the web application searches for a set of composing services which, granting the security level requested, minimizes its response time. This means that it adapts itself

predicting the performance of the composing services. The MAWeS Frontend, in this case, is a standard client which can be integrated into the web application, and sends to the MAWeS core all the application-related information (performance metrics, behaviour, security requirements, ...) needed to perform the simulations. Details about this kind of optimization can be found in [16].

In Service level optimizations, the MAWeS core aims at minimizing the response time of a self-optimizing service. In this case, the MAWeS frontend is integrated in the service, which invokes the MAWeS core in order to optimize its execution. Invocations can take place in two modalities: for each service (*Service Call Approach*) invocation, or in correspondence with particular events (*Reactive Approach*). Details about this kind of optimization can be found in [20].

In both cases, the MAWeS Frontend produces a MetaPL application description embedded with information useful for self-optimization (i.e., the optimization target, the application parameters that can possibly be modified, the target configurations, ...) and sends it to the MAWeS core.

4.2 MAWeS Core

The MAWeS Core exploits environment services (i.e., the services offered by the environment to monitor and possibly to configure itself) and the Evaluation Web Services to find out optimal execution conditions (in terms of both performance and security).

The MAWeS Core is subdivided in three logical components: Interface Unit, which receives the MetaPL document and retrieves the optimization information, the Decision Unit, which controls the execution of the evaluation processes in order to make decisions, and the Evaluation Clients, which invoke the Evaluation Services.

The MAWeS core can work on-line, i.e., when the application/service requests the optimization, and off-line, i.e., independently of the execution of the applications or services. Furthermore, MAWeS is invoked off-line, in order to publish the available configurations or new available services; it collects the information needed to perform the optimizations.

Interface Unit. The task of the Interface Unit (IU) is to find the parameters that may affect the application performance and to pass this information to the Decision Unit. In order to do so, it extracts from the MetaPL application description all information that may possibly be used to optimize the application. The MetaPL description explicitly declares the optimization target by means of the **Target** element, whose attribute **kind** specifies if the target is to maximize a **performance** metric (default), or to guarantee a desired minimal security level (**GSLmin**). A single MetaPL description may contain more than one **Target**.

Furthermore, the service requester (web application or composed service) needs to indicate explicitly the type of security credentials that it intends to adopt. This information will be also used to choose the most suitable configuration. To include such credential information, MetaPL has been enriched with

a new element, **credential**, whose attribute **kind** specifies the kind of credential. At the state of the art, the working prototype supports login/password credentials (**pwd** value), digital **certificate**, or no credential (**nothing**).

Evaluation Clients. Evaluation Clients are a set of clients that let the MAWeS core to interoperate with the Evaluation Services, which perform performance predictions and policy evaluations. They are the MH Client and the SEC Client.

The M/H (MetaPL/HeSSE) Client is a software unit that implements a client for the MetaPL/HeSSE web services. Its implementation is threaded, in that each client instance runs as a new thread. This makes it possible for the MAWeS core to start in parallel as many M/H clients as necessary. Each client invokes the services needed to perform the simulation and analysis for each different configuration, and gets the corresponding results. The Decision Unit collects all the results, compares them and makes its decisions.

The SEC Client is a software unit that implements a client for the Security Evaluation web service; it is usually invoked off-line to evaluate the GSL of each published service that will be stored in the SIU.

Decision Unit. The Decision Unit (DU) contains all the framework autonomic intelligence. It applies the optimization rules defined by the framework administrator to optimize the target applications by means of the feedforward approach described in the introduction.

The DU is made out of two different units: the System Information Unit (SIU) and the Optimization Unit (OU). The former maintains information about the system status, i.e., the list of valid configurations with their GSLs. The latter requests to the MetaPL/Hesse client to simulate all admissible configurations and compare them. When MAWeS manages a new request, the SIU extracts the list of configurations whose GSL is greater than GSL_{min} , while the OU invokes the MetaPL/HeSSE clients in order to obtain performance predictions.

In particular, the **Optimization Unit** analyzes all possible configurations after their evaluation, and chooses the one with maximum performance among the ones that meet the minimal security requirements.

The OU requests to the SIU all available configurations that are accessible with user provided credentials, and whose GSL is greater than the requested GSL_{min} . It asks to the MetaPL/Hesse WS (through the MH client) the evaluation of all admissible configurations and, after that, it chooses the best configuration. It should be noted that in the current version of the framework we assume that all service policies are published and evaluated off-line by the System Information Unit. This uses a Security Evaluation Service to perform the GSL of each available service, as explained later in this section.

The **System Information Unit** holds all system and status information, as follows:

- available simulator configurations (which correspond to the different real system configurations);

- security policies related to different configurations and the corresponding security level (GSL);
- associations between policies and configurations.

Furthermore, the SIU includes a **Monitoring and Discovery Unit** that updates the temporal parameters for the simulator. Moreover, it invokes the Security Evaluation Client to evaluate the GSL of each policy to be maintained.

4.3 Evaluation Components

Evaluation components include a set of Web Services that make it possible to obtain performance predictions and security evaluation of the submitted configuration. Simulation and performance prediction services that implement the HeSSE simulator and apply the MetaPL filters are described in detail in [16,20]. The security Evaluation service implements the novel security evaluation methodology proposed in [17] and previously summarized, that allows to evaluate the Security Level provided by a service or by a system from its published policy. It receives the policy, described according to WS-Policy language, applies the formalization process (as described in [21]), applies the evaluation technique, and returns the GSL.

4.4 The MAWeS Execution Flow

In conclusion, MAWeS performs on-line the optimization on request from the application or services, but collects the information about configuration off-line.

Figure 3 illustrates how MAWeS works “on-line”: (1) the MAWeS Frontend submits the request to the core, attaching the MetaPL document containing the GSLmin, the types of credential and the application description. (2) The interface unit extracts from MetaPL the GSLmin, the credentials, the performance optimization target, the application parameters and the application description and forwards them to the DU. (3) The DU optimization unit gets the parameters from IU and requests the available configurations to the SIU. The SIU extracts from the local database the list of valid configurations whose GSL, evaluated off-line, is greater than GSLmin. (4) The SIU returns the list of valid configurations. (5) The optimization Unit starts the cycle of simulations, submitting the configuration and the MetaPL application description to the MH clients, which generate the traces and simulate them on the target configuration (5a and 5b). (6) The MH client returns the set of performance measurements for valid configurations. (7) The optimization unit chooses the best configuration and returns it to the IU. (8) The final results are sent to the Frontend.

The grey box describes the “off-line” publication of configurations and policies, together with the policy evaluation process. When the new couple configuration/policy is published, the SIU forwards the policy to the SEC client, which performs the GSL evaluation and returns the value. The SIU stores the GSLs ordered by growing GSL values.

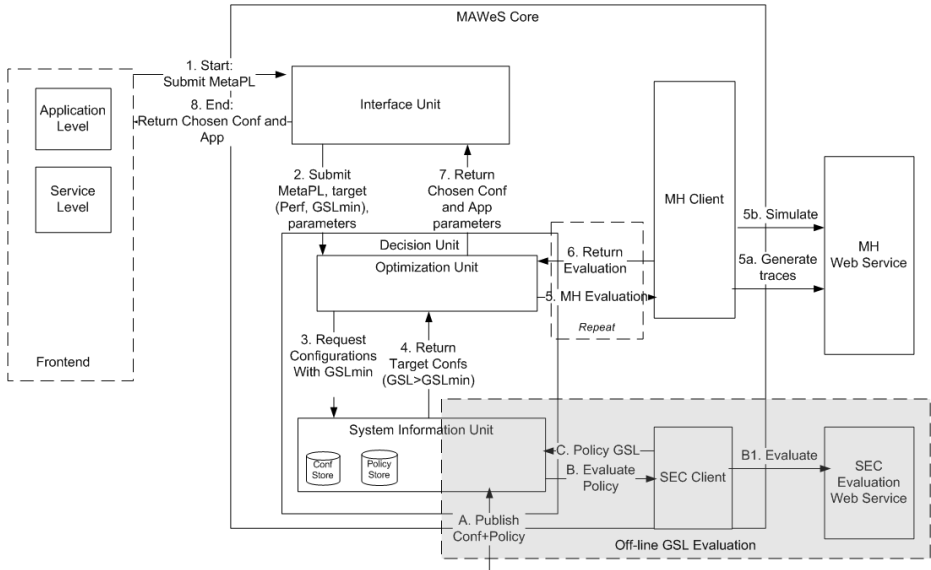


Fig. 3. The MAWeS execution Flow

5 Conclusions

In this paper we have proposed an innovative approach to the development of self-optimizing autonomic systems for Web Services architectures with trust requirements, based on the adoption of simulation for performance prediction and feedforward control and a security evaluation methodology to guarantee the security level of the adopted services. The approach relies on the autonomic framework MAWeS, now enriched with security features. We have shown the new framework architecture and how it works.

The proposed framework opens a new way for the development of autonomic and trusted systems, which, in a way transparent to users and easily manageable by developers, let services and applications to self-optimize, maintaining a given security level.

References

1. Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., Orchard, D.: Web Services Architecture. W3C Web Services Architecture Working Group (2004) <http://www.w3.org/TR/2003/WD-ws-arch-20030808>
2. Balasubramanian, V., Bashian, A.: Document management and web technologies: Alice marries the Mad Hatter. In: Commun. ACM., vol. 41(7), pp. 107–115. ACM Press, New York (1998)
3. Chandrasekaran, S., Silver, G., Miller, J.A., Cardoso, J., Sheth, A.P.: Web service technologies and their synergy with simulation. In: Proc. of Winter Sim. Conf., San Diego, California, USA, vol. 1, pp. 606–615. ACM, New York (2002)

4. Chandrasekaran, S., Miller, J.A., Silver, G., Arpinar, I., Sheth, A.P.: Performance analysis and simulation of composite web services. In: *Electronic Markets, USA*, Routledge, vol. 13(2), pp. 120–132 (2003)
5. Bishop, M.: *Computer Security*. In: *Art and Science*, Addison-Wesley, London (2003)
6. Bicarregui, J., Dimitrakos, T., Matthews, B.: Towards security and trust management policies on the web (2000)
7. Beth, T., Borcherding, M., Klein, B.: Valuation of trust in open networks. In: Gollmann, D. (ed.) *Computer Security - ESORICS 94. LNCS*, vol. 875, pp. 3–18. Springer, Heidelberg (1994)
8. Department of Defense: Trusted computer system evaluation criteria, Department of Defense Standard 5200.28-STD (Orange Book) (1985)
9. Xiong, L., Liu, L.: Building trust in decentralized peer-to-peer electronic communities. In: *Proceedings of ICECR-5, Montreal, Canada* (2002)
10. Dini, O., Moh, M., Clemm, A.: Web services: Self-adaptable trust mechanisms. In: *Proc. of Advanced Industrial Conference on Telecommunication/Service Assurance with Partial and Intermittent Resource Conference /E-Learning on Telecommunication Workshop*, IEEE Press, New York (2005)
11. Chung, J., Zhang, J., Zhang, L.: WS-Trustworthy: A framework for web services centered trustworthy computing. In: *Proc. of IEEE Int. Conf. on Services Computing (SCC 04)*, Washington, DC, USA, pp. 186–193. IEEE Computer Society, Los Alamitos (2004)
12. Birman, K.P., van Renesse, R., Vogels, W.: Adding high availability and autonomic behavior to web services. In: *Proc. of 26th Int. Conf. on Software Engineering (ICSE 2004)*, pp. 17–26. IEEE Computer Society, Los Alamitos (2004)
13. IBM Corp.: An architectural blueprint for autonomic computing, USA (2004)
14. Kephart, J.O., Chess, D.M.: The vision of autonomic computing. In: *Computer*, vol. 36(1), pp. 41–50. IEEE Computer Society Press, Los Alamitos (2003)
15. Zhang, Y., Liu, A., Qu, W.: Software architecture design of an autonomic system. In: *Proc. of 5th Australasian Workshop on Soft. and System Arch.* pp. 5–11 (2004)
16. Mancini, E.P., Rak, M., Villano, U.: Predictive autonomicity of web services in MAWeS framework. *Journal of Comp. Science* 2, 513–520 (2006)
17. Casola, V., Mazzeo, A., Mazzocca, N., Vittorini, V.: A security metric for public key infrastructures. *Journal of Computer Security* 15(2) (2007)
18. Whitepaper: Security in a web services world: A proposed architecture and roadmap (2002)
19. Atkinson, B., et al.: Ws-security specification, web services security ver. 1.0 (2002)
20. Mancini, E.P., Rak, M., Villano, U.: Autonomic service oriented architectures with mawes. To be published in *Journal of Autonomic and Trusted Computing*, American Scientific Publishers (2007)
21. Casola, V., Coppolino, L., Rak, M.: An architectural model for trusted domains in web services. *Journal of Information Assurance and Security* 1(2) (2006)

Biology as Inspiration Towards a Novel Service Life-Cycle

David Linner, Heiko Pfeffer, Ilja Radusch, and Stephan Steglich

Technische Universität Berlin,
Sekt. FR 5-14, Franklinstrasse 28/29, 10587 Berlin, Germany
{david.linner, heiko.pfeffer, ilja.radusch,
stephan.steglich}@tu-berlin.de

Abstract. A crucial challenge for future computing environments is the development and management complexity caused by an increase in mobility and heterogeneity of networked devices. Autonomous, service-oriented computing environments promise to significantly reduce the management overhead and the need for human intervention. Making services autonomous, requires a shift in the design of service life-cycles, away from static, partially human-controlled mechanisms towards self-control principles. In this work, a biologically inspired service life-cycle is proposed. This life-cycle utilizes evolutionary principles for the adaptation and evaluation of services in highly dynamic computing environments. Additionally, approaches for the autonomous service creation are integrated to the bio-inspired service life-cycle in order to rapidly address users' needs on demand, but also to come up with completely new services.

1 Introduction

Driven by W3C's Web Service Architecture (WSA) [1] service-oriented computing has become a widely accepted concept for the design of applications in open, distributed systems. But apart from WSA, the success of service-oriented computing is strongly related to the straightforward nature of the service pattern. Each interaction of service consumer and service provider is basically composed of a two-way message exchange, started with a service request and terminated with a service response. The involved parties are loosely coupled, i.e., dependence exists only for the moment of service provision. These characteristics make service-oriented computing also a suitable approach for the realization of distributed applications in highly dynamic computing environments.

Within the scope of this work we assume to have a decentralized and mobile network infrastructure, which is characterized by the temporally formed islands of devices dependent on the movement patterns of the user. On top of this networking infrastructure we are establishing a user-centric service environment that allows to seamlessly integrating services of different devices in the surrounding of the user to fulfill the user's needs. The reference for this provision of integrated services is solely the description of the user goal. Accordingly we introduce the concept of an

automatic service life-cycle as core paradigm of the envisioned service environment. However, to cope with the dynamicity of the underlying network infrastructure and the volatility of the computational entities this life-cycle is strongly interweaved with biologically inspired principles. The resulting bio-inspired service life-cycle addresses two main weaknesses of traditional service life cycles in terms of autonomy. First, the service adaptability is upgraded to service evolution, i.e. services are not only able to adapt to current situations according to predefined transition table entries, but to evolve with respect to their environment and assembled knowledge. Second, service autonomy principles are extended from run-time to design-time phase, paving the way for a holistic autonomous service life cycle enabled through biologically inspired concepts. To complement the bio-inspired service life-cycle, also principles for the decentralized control of life-cycle activities are discussed.

The next chapter gives an overview on the computing environment we address in our investigations. The third chapter explains the notion of service emergence, as one of the central motivations for the bio-inspired service life-cycle. Moreover, this chapter describes service principles and outlines different approaches for rapid service composition. The fourth chapter introduces our concept of a bio-inspired service life-cycle, the single phases and the constituting activities. Finally, chapter six concludes this work and outlines some prospectively striven activities.

2 Computing in the Pervasive Space

Our investigations on autonomous services and corresponding solutions for coherent service delivery platforms are driven by the trend towards pervasive service environments. In those environments features of several local devices and global services provided though local gateways are made available in order to best address the users' needs in each situation. Physically there are various, most different devices with a high degree of mobility. Consequently, devices are following abstracted as mobile nodes moving through the environment and establishing connections to other nodes. A set of nodes connected in a way that there is a path between every two nodes is referred to as an island. Hence, we assume the service environment to be composed of a set of islands which again consist of a set of nodes. Since nodes are supposed to be mobile, they can enter current islands and leave them at every point in time. However, nodes are not necessarily mobile. They can also represent infrastructural or embedded components like network gateways or sensors.

The collaborative utilization of node services in this network topology implies control and management procedures that abstain from central entities. Moreover, the autonomous nature of each node entails further constraints with regard to the reliability of its services. As these services are provided by different authorities there is an eventuality for malfunction or even malicious behavior. The interior node functionality can neither be monitored nor controlled, but just the node behavior in conjunction with other nodes. Consequently, computational concepts that aim at the corporate provision of node services to users need to be designed in accordance to these characteristics.

3 Autonomous Emergence of Services

From the functional point of view the users' goals are even more relevant to service provision than the physical network characteristics. These goals define the required behavior and characteristics of services. Hence, they are the requirements basis that serves as trigger for the emergence of services and starting point for the service life-cycle. The classical service life-cycle is composed of two life phases, *design-time* and *run-time*. While the design-time basically comprises activities as modeling, implementation, and testing, activities handled at run-time are provision, execution, monitoring, and reconfiguration of services. Aiming at autonomous emergence of service means to automate the design-time activities and shorten the response time of the service environment to a goal-oriented service request by the user.

In this scope we distinguish between two kinds of goals, explicitly expressed and implicitly given ones. The first kind of goal is based on a typical service consumption pattern, where the service environment is supposed to respond on demand to particular service request expressed by the user. However, based on the observation of the long-term behavior of the user and the interpretation of user data, needs and preferences can be derived. This information may serve as implicit goal and help to come up with a useful, although not explicitly demanded service.

Before differentiating the idea of automating the service design-time activities more in detail, the following section introduces the principles our concepts are based on.

3.1 Service Model and Principles

Since our service platforms address the user, we are mainly regarding application level services, based on message- and data-stream- based interactions. In contrast to, e.g., component models, the service model just defines an interface for consumption and configuration of a particular functionality. This interface view allows us to abstract from the actual implementation of services, but also entails new characteristics with regard to the control of computed tasks. Service consumer and provider may physically be situated on different entities and run on behalf of different authorities. Hence, they are only loosely coupled and the service consumer needs to be aware of the fact that a service provider may behave different than promised, e.g. with regard to timing or delivered results.

In order to match the user goals, i.e. find an appropriate service that corresponds to the described requirements, we go back to the principle of service composition. Service composition is the central element for the creation of complex functionality though arranging several other services within a particular execution order. For simplicity, we distinguish between two basic structures in the following. An atomic service that cannot be broken into constituent parts is referred to as Service Cell. A composition of Service Cells is called Service Individual. Service Individuals may also be composed of other Service Individuals, but completely decomposing a Service Individual again result in a set of Service Cells. Service Individuals and Service Cells are supposed to have the same appearance, at least with regard to the interface. For that reason, either a Service Cell or a Service Individual is meant when talking about services in general.

The composition of Service Individuals is supposed to be described by a set of Services and at least two directed graphs constituting the interrelation of the services, as illustrated in Fig. 1. The first graph describes the flow of control data among the Services and therefore implicitly the execution sequence. Hence, it determines an initiation cascade along with an appropriate parameterization of the services. The second graph describes the flow of application data among the Services. With regard to our service model we distinguish between these two graphs since we do not expect each Service in a composition to participate in the creation or processing of application data.

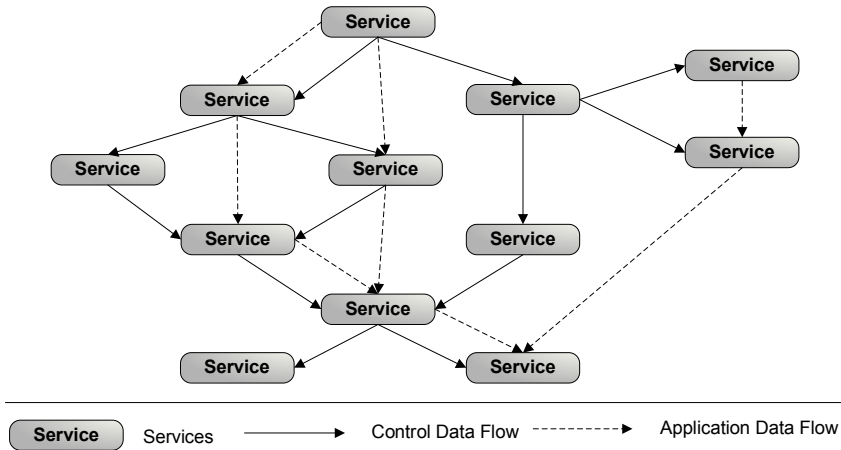


Fig. 1. Exemplary visualization of a service graph

3.2 Role of Service Semantics

When talking about autonomous service emergence on behalf of the user we mainly address the rapid (optimally on-demand) creation of Service Individuals through composition. Service Cells are assumed to be still covered by the conventional service life-cycle. As indicated above the intention of autonomous service emergence is to automate service design-time activities with respect to the decentralized nature of the computing environment. For the beginning we see three approaches of addressing this challenge, which can be differentiated according to the way knowledge about the semantics of services available for composition is applied. We classify today's automatic or semi-automatic composition approaches according to three different principles for interviewing knowledge about service semantics with the process of rapid service creation by composition of other services.

3.2.1 Manual Service Composition

In the first approach the user manually arranges order, conditions, and flow of control and media data between the single constituting services. Hence, the user directly utilizes own knowledge about the available services and their semantics to create an appropriate composition likely to address a particular goal. This kind of composition

often comes along with suitable graphical tools. However, the applicability of user-generated service approaches decreases with the service complexity. While simple problems can be addressed with simple service compositions, finding service compositions for the solution of complex problems may soon exceed users' skills.

3.2.2 Automated Service Composition

In the second approach the service developer is supposed to deliver atomic services that have passed the conventional life-cycle together with a formal description of the service semantics. In this approach the service itself is described with meta information, which comprises descriptions of interface semantics and functional semantics. This idea is for instance followed in semantic Web Service technologies as WSMO [2], SWSO [3] and, OWL-S [4]. The semantic meta information can be utilized by planning algorithms to find a reasonable order for the execution of services and a suitable flow for control and media data. The user just needs to define the goals to be achieved. Thus, when requiring a service the user determines the problem but not the solution as in the manual creation. Although automatic service composition based on semantic service descriptions is a theoretically promising and in practice partially proofed concept, there is a trade-off between the required service development efforts and the actual applicability in large-scale open systems. Today's approaches for semantic service descriptions are usually static and focused on a particular application domain. Hence, they are rarely suited for dynamic systems as the pervasive service environment we envision. Moreover, the complexity of semantic descriptions exceeds especially for simple service the complexity of the service implementation.

3.2.3 Automated and Self-reflective Service Composition

The third approach is an advancement of the second approach. Instead of static service annotation describing the service semantics, the service environment is supposed to learn how certain services can be composed and progressively annotates the constituting service accordingly. The only initially available information about each service is a light-weight description that allows formally correct compositions without telling anything about the semantics. However, the atomic services utilized for the composition additionally need to be able to operate in a trial-mode, in order to not change the physical state of any entities in the learning phase. While positive examples for self-learning systems such as heuristic error-feedback- [5] or Multilayer Perceptron-based [6] systems are currently only known from other domains, respective features perfectly fit into the notion of self-emerging services. Thus, in the following we are concentrating on this approach for service composition.

4 Bio-inspired Service Life-Cycle

The bio-inspired service life-cycle represents an attempt towards the application of autonomous service emergence in the presence of a highly dynamic computing environment. Thus, on the one hand the bio-inspired service life-cycle aims at the automatic creation of services. On the other hand evolutionary concepts are utilized to

support seamless and continuous adaptation of services to permanently changing environment conditions (e.g. network topology, available services) and user needs.

The bio-inspired service life-cycle as depicted in Fig. 2 comprises four phases, *Initiation*, *Integration*, *Evolution*, and *Retirement*. The Initiation phase deals with the preprocessing of user goal-descriptions and their coordination with respect to the current context of the system in order to create appropriate Service Individuals that meet the users' needs. The Integration phase comprises activities aiming at provision, execution and evaluation of service, while the Evolution phase concerns the improvement and adaptation of services. The Retirement phase addresses the handling of Services that are supposed to be removed from the service environment when not being appropriate any longer.

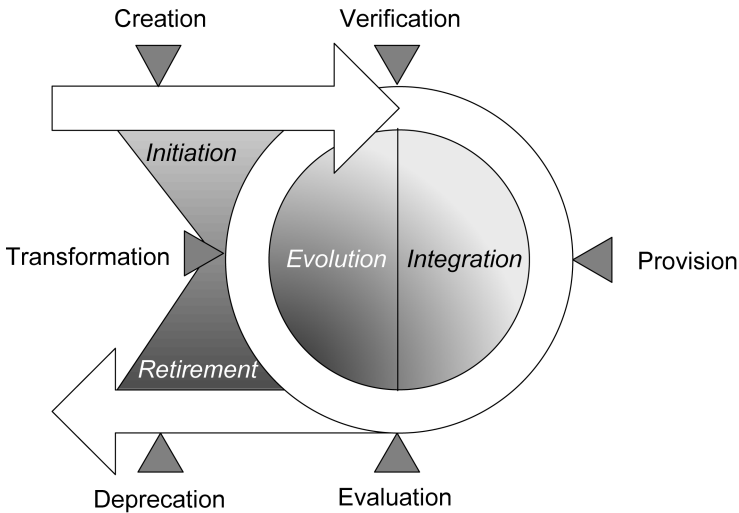


Fig. 2. Bio-inspired Service Life-Cycle

4.1 Initiation

Usually, there is a big difference between the formalization of user's needs and formal descriptions of functions provided by computing devices. When formulating the goal for service execution the user utilizes concepts describing the problem from the human point of view. For instance, a user who requests support for getting from one point in the city to another does not want to reflect about the steps required by the computing environment to support this demand with an appropriate service. In contrast, the Service Cells provided by the nodes in the service environment are supposed to be very special and their description may be far apart from the user's notion. For instance, such Service Cells could provide the user's current location, render street maps as images or display images on screens. However, as this simple example reveals there is a suitable composition of services that corresponds to the description of the user goals. The purpose of the initiation phase is to automatically find this mapping between the user's goal, given a service request, and a

corresponding Service Individual and its graph representation. The service creation, as central activity in the initiation phase, is supposed to also incorporate information about the current context of all involved entities (i.e. current situation, user preferences, etc.). A final verification of the formal correctness for each Service Individuals helps to prevent malfunctions and unpredictable system behavior.

4.2 Integration

The Integration phase of bio-inspired service life-cycle aims at the organization of services in the computing environment. Integration is perhaps the most classical part of the bio-inspired service life-cycle while addressing the deployment, provision and execution of Service Individuals in the service environment. Since each Service Individual can be regarded as solutions for a particular problem, the deployment of the underlying composition may either help to solve an equal user request or serve as source of knowledge about possible service combinations. The latter case could at least speed up further service creation processes later on. Service provision reflects the active participation of service providing entities in the service environment. The purpose of service execution is to control the executions of all Service Cells constituting a Service Individual. In the decentralized environment that is focused on, the service execution involves collaborative control processes. This will be explained as showcase more in detail later on.

4.3 Evolution

In order to allow for seamless adaptation of the service environment according to modifications of user goals or changes in the configuration of the underlying physical system, the bio-inspired service life-cycle includes an evolution phase.

Therein, each Service Individual is transformed into numerous variations with different compositions and configurations. For the transformation of Service Individuals it is envisioned to apply methods taken from genetic programming, e.g., mutation and cross-over [7][8] and evolutionary computing [9][10]. The transformation is supposed to be an ongoing process in order to find the Service Individuals that best address functional and non-functional requirements of the user request while taking the characteristics of the physical environment into account. This procedure follows the concept of survival of the fittest. However, the environment of Service Individuals is also defined by artificial characteristics like user goals, which need to be considered for the selection of Service Individuals. Consequently, the evolution phase of the bio-inspired service life-cycle explicitly addresses the evaluation and selection of Service Individuals.

The intention of evaluation is to estimate or test performance and adequacy of Service Individuals in the destination environment, at least in a trial operation mode. However, the evaluation of Service Individuals in a certain computing environment is not only about the composition plan, but also addresses instances of this plan, i.e. sets of actual Service Cells that substitute the nodes in the composition graph. Consequently, it is expected that for each Service Individual there exist multiple, partially intersecting sets of Service Cell instances as binding. For the evaluation each Service Individual is regarded together with its possible bindings in the destination

environment. Hence, Service Individuals should not only be functionally correct, but effective with regard to various physical criteria (e.g. calculation time, battery power, storage consumption). For the evaluation of the correspondence between Service Individual and user goals, feedback mechanisms for users are included. That way, the system is enabled to learn user preferences with regard to the current context. Notably, since the given criteria could compete among and influence each other, the evaluation of the Service Individual fitness has to be considered as a multi-criterion optimization problem.

The evaluation rates the fitness of the available Service Cells which are considered to constitute Service Individuals. In biological evolution, fitter individuals of a species stand up to others by being able to breed faster; they thereby pass on more of their genetic material than other individuals. The faster breeding of fitter Service Cells is envisioned to be realized by picking fitter Service Cells with higher priorities. This process is common for consideration of the adaptation level of individuals in nearly all evolutionary algorithms [11].

4.4 Retirement

While Service Individuals that perform well in the current context will serve as parent generation for the next evolution step, negatively evaluated services are retired from the service environment in a progressive process. However, the composition plan contained by each individual describes how a particular problem can be solved in a certain context. Although a composition may not perform well at the moment the Service Individual is evaluated, the represented information can become useful again. For that reason the removal of service is, indeed, an inevitable step to save resources, but supposed to be implemented smooth and adapted to the frequency of changes in the context of the service environment.

5 Conclusion and Outlook

Service-oriented system in highly dynamic computing environments can significantly benefit from the interweavement of autonomous management concepts. Service life-cycles based on the notion of autonomy are a first, but very promising step in this direction. The bio-inspired service life-cycle introduced in this work marks the beginning of our investigations in the field of autonomous services. Although early probes of the concepts and ideas we have presented worked fine in practice, there is still the need evaluate all concepts, especially in conjunction with each other.

Nonetheless, we are also refining concepts and developing new ones to close the bio-inspired service life-cycle. Part of our ongoing work is for instance a state-based approach for Initial Service Creation, wherein the user goal description represents origin and target state, while a combination of services is searched that may serve as an appropriate state transition between these states. Another point of further investigation is the deployment of services. We try to find the best dissemination strategies of Service Individuals, as these are the main carriers for information about problem solutions and central element of our service evolution.

References

- [1] Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., Orchard, D.: Web Services Architecture (2004) Available: <http://www.w3.org/TR/ws-arch/>
- [2] Roman, D., Keller, U., Lausen, H., Bruijn, J.d., Lara, R., Stollberg, M., Polleres, A., Feier, C., Bussler, C., Fensel, D.: The Web Service Modelling Ontology, pp. 77–106. IOS Press, Amsterdam (2005)
- [3] Battle, S., Bernstein, A., Boley, H., Grosz, B., Gruninger, M., Hull, R., Kifer, M., Martin, D., McIlraith, S., McGuinness, D., Su, J., Tabet, S.: Semantic Web Services Ontology, Available (2005) <http://www.daml.org/services/swsf/1.0/swso/>
- [4] The OWL Services Coalition: OWL-S: Semantic Markup for Web Services (2004) Available: <http://www.daml.org/services/owl-s/1.1/>
- [5] Lo, J.-C., Yang, C.-H.: A heuristic error-feedback learning algorithm for fuzzy modelling, *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 686–691 (1999)
- [6] Reifman, J., Feldman, E.: Multilayer perceptron for nonlinear programming. *Computers and Operations Research archive*, 1237–1250 (2002)
- [7] Koza, J.R.: Survey of genetic algorithms and genetic programming. In: *Proceedings of the Wescon 95 - Conference Record: Microelectronics, Communications Technology, Producing Quality Products, Mobile and Portable Power, Emerging Technologies*, 7–9 (1995)
- [8] Piszcz, A., Soule, T.: A survey of mutation techniques in genetic programming. In: *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pp. 951–952 (2006)
- [9] Yamamoto, L., Tschudin, C.: Genetic Evolution of Protocol Implementations and Configurations. In: *Proc. IFIP/IEEE International Workshop on Self-Managed Systems & Services* (2005)
- [10] Miorandi, D., Yamamoto, L., Dini, P.: Service Evolution in Bio-Inspired Communication Systems. In: *Proceedings of the International Conference on Self-Organization and Autonomous Systems in Computing and Communications* (2006)
- [11] Kicinger, R., Arciszewski, T., Jong, K.: Evolutionary computation and structural design: A survey of the state-of-the-art. *Computers & Structures* 1943–1978 (2005)

Design of Service-Based Systems with Adaptive Tradeoff Between Security and Service Delay

Stephen S. Yau, Min Yan, and Dazhi Huang

Department of Computer Science and Engineering
School of Computing and Informatics
Arizona State University
Tempe, AZ 85287-8809, USA
{yau,min.yan,dazhi.huang}@asu.edu

Abstract. Service-based Systems (SBS) have the advantage of composing distributed systems from various services provided by multiple providers transparently. In addition to functional correctness, multiple non-functional QoS requirements should also be satisfied in such systems. Among these QoS requirements, security protection and real-time performance are the two major concerns. However, neither application users, nor service providers, have adequate control over such QoS of SBS. In this paper, an approach to the design of SBS with the capability of tradeoff between security and service delay of composite services running across various service hosts is presented in order to satisfy both security and real-time performance requirements simultaneously.

1 Introduction

Service-Oriented Architecture (SOA) enables rapid integration of loosely-coupled and reusable services for on-demand applications [1]. A service is a self-contained software entity with network-addressable interfaces that offer well-defined capabilities using standard protocols like SOAP. In SOA, services which do not invoke other services are considered as *atomic services*. *Composite services* are services composed of atomic services provided by various service providers following a specific workflow. Systems based on SOA, called *service-based systems (SBS)*, comprise of various services offered by distributed service providers over networks [2], [3], [4]. For most mission-critical applications in SBS, the services should be provided with not only functional correctness, but also with nonfunctional properties, such as timeliness, security, availability, throughput, and fault-tolerance, which are referred to as QoS attributes [5]. Development of SBS to satisfy the requirements of multiple QoS imposes great challenges because the satisfaction of one QoS requirement often requires certain sacrifices for the others. Therefore, proper tradeoffs among the requirements of QoS attributes are often required. However, current software engineering techniques cannot effectively support such tradeoffs.

Among multiple QoS attributes, we will focus on the design of SBS with tradeoff between security and real-time performance requirements, because they are two major concerns of many mission-critical applications in various areas, such as healthcare,

e-commerce, military and homeland security. In this paper, we will present an approach to the design of SBS with the capability of tradeoff between security and service delay under various situations. Our approach includes providing an ontology for developers to specify functionalities and QoS properties of atomic services with service interfaces, as well as a methodology for the user to specify security and real-time performance requirements. In order to satisfy security and real-time performance requirements simultaneously, our approach employs a control-based mechanism to perform the tradeoff between security and service delay of SBS when situations change.

2 Current State of Art

For the real-time aspect, a heuristic algorithm for searching the optimal service composition for SBS was presented in [6]. A generic algorithm to find optimal or near optimal migration decisions was introduced in [7] to facilitate the execution of real-time web services. In [5], [8], performance control architectures and algorithms were presented for software systems and web servers based on feedback control theory in order to optimize throughput.

For the security aspect, an approach to provide development and runtime support for situation-aware security (SAS) in autonomic computing systems was presented in [9]. A UML profile describing QoS in SOA, such as security and fault-tolerance, was presented [10]. The security vector of a computing task was presented in [11] to characterize the security requirements in n-dimensional space, where each dimension specifies a Boolean security requirement, which is either satisfied by the computing task or not satisfied.

Some research has been done to consider both security and real-time performance. For web-based information systems, a method to improve the performance gain by adaptively adjusting the security level of the system was presented [12]. The concept of security vector is extended to a vector of discrete security levels [13], instead of Boolean values, to represent a security requirement of particular protection strength. By dynamically controlling the security levels of computing tasks, several dynamic security-aware scheduling algorithms were presented to improve the systems' overall performance in terms of security and throughput for distributed computing systems. Some heuristic approaches have been presented to solve the problem of QoS-aware service composition to satisfy users' QoS requirements [14]. However, these approaches have the difficulty to be directly applicable to SBS, in which the situations of users, service providers and system environments are dynamically changing.

3 Our Overall Approach

In order to design SBS which can dynamically satisfy both the security and service delay requirements at runtime, the following challenges need to be addressed:

C1) How to generate formal specification of the security and service delay?

C2) How to model user's requirements of the security and service delay of a service?

C3) How to evaluate the security protection status for atomic and composite services at runtime?

C4) How to dynamically perform the tradeoff between security and service delay for a composite service involving multiple service providers such that both requirements are met?

Our approach, which addresses these challenges, consists of the following six steps:

Given an SBS, with the information of which services are provided by which service providers, and the application requirements of the user,

S1) Service providers use the ontology in Sec. 4 to specify the service interfaces.

S2) The user specifies the expected security protection and service delays of the composite services under various situations in the form of *SPARQL query* [15] to be described in Sec. 4.

S3) Based on the user's expected QoS of composite service in **S2)**, a QoS-aware service composition process [14] selects the atomic services with appropriate service interfaces specified in **S1)** to generate composite service.

S4) The developer uses *User Expectation Function (UEF)* (see Sec.5) for evaluating services' security protection status at runtime when system situation changes.

S5) The developer uses the situation awareness (SAW) agent approach in [3] to synthesizing QoS monitoring modules in order to collect context data, including system states, events generated by services and other extraneous events in the system environment. Context data is reported to the tradeoff controller as *situation* [16].

S6) Our tradeoff controller developed in Sec. 6 dynamically evaluates the security protection status using *UEF* in **S4)**, and analyzes the tradeoff between security and service delays with feedback of system situations from QoS monitoring in **S5)**. If both security and service delay requirements are satisfied in current situation, the controller will either adjust the service's security vector [13] to optimize security protection or achieve better real-time performance. Otherwise, the user's security and service delay requirements need to be relaxed, or the service composition needs to be reconfigured to improve its real-time performance, or both.

In our approach, the service interface in our ontology (**S1**) is associated with the security vector and service delay of the atomic service under a specific situation, which tackles Challenge **C1**). The SPARQL query (**S2**) tackles Challenge **C2**) by specifying user's QoS requirements for services' security and service delays. *UEF* (**S4**) is derived from SPARQL query to evaluate services' security protection status during runtime, which tackles Challenge **C3**). The tradeoff controller (**S6**) tackles Challenge **C4**) by adjusting security vector to meet user's security and service delay requirements under all situations.

4 Specifying Service Interface and User's Expectation

In this section, we will present the ontology for specifying service interfaces in **S1**) and the SPARQL query [15] for specifying user's security and real-time requirements in **S2**).

4.1 Specification of Service Interfaces for Security and Service Delay

To compose a service with QoS constraints [14], an atomic service should have clear service QoS capability specification and related context and situation information for the service to be discovered and used in service composition. To achieve this, we use the extended *OWL-S with situation ontology (SAW-OWL-S)* [18] to model QoS properties of a service under various situations, as shown in Figure 1. Security levels in SBS are normalized as discrete levels in the range of [0, 1] to represent various security protection capabilities using the security level models [13], [17]. Tasks with high security level usually have higher protection strength, lower throughput, consume more resources and cause longer delay than tasks with low security level. Each atomic service in an SBS follows the SBS's definition of security level. The security vector is set to certain levels according to current situation, as described in Sec. 6. A service's execution is considered as a process in *SAW-OWL-S*. A process with different security vectors generates different service delays. An atomic service's functionality, together with its specific security vector and designated service delay, is called a *service interface*. Service's delay for a specific security vector can be obtained using distributed real-time benchmark suite techniques [19]. Based on service interface specifications, appropriate atomic services can be chosen for service composition with QoS constraints [14]. Note that atomic services that have dependent relations in a workflow require their service interfaces to have appropriate security correlation. For example, if two services need to communicate with each other at some point in a workflow, their security level of encryption/decryption algorithm should be the same. Otherwise, they will not be able to understand each other's message.

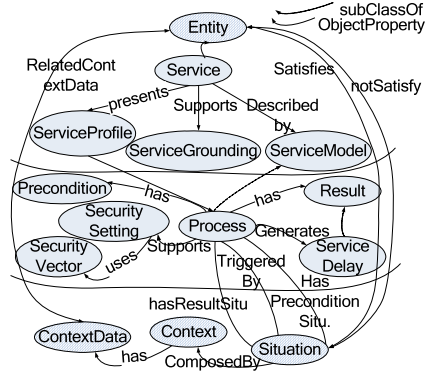


Fig. 1. SAW-OWL-S for security and service delay specification (only related and important classes and relations are shown)

According to current situation, as described in Sec. 6. A service's execution is considered as a process in *SAW-OWL-S*. A process with different security vectors generates different service delays. An atomic service's functionality, together with its specific security vector and designated service delay, is called a *service interface*. Service's delay for a specific security vector can be obtained using distributed real-time benchmark suite techniques [19]. Based on service interface specifications, appropriate atomic services can be chosen for service composition with QoS constraints [14]. Note that atomic services that have dependent relations in a workflow require their service interfaces to have appropriate security correlation. For example, if two services need to communicate with each other at some point in a workflow, their security level of encryption/decryption algorithm should be the same. Otherwise, they will not be able to understand each other's message.

4.2 Specification of User's Expectations for Composite Service

Using SPARQL Query Language for OWL-S [15], a user can specify the requirements for the security protection and service delay of a composite service under situation *situ* as follows:

```
PREFIX situ: <CS.owl#aS-3-3>
PREFIX pref: <P (P1, P2, ...Pn)>
PREFIX ns: <example domain>
SELECT ?service_name
WHERE { ?x ns:S ?S. ?y ns:delay ?delay.
?z ns:tradeoff ?tradeoff. ?k ns:security_rating ?security_rating.
FILTER (?S1 >= B1 && ?S2 >= B2 &&... && ?Sn >= Bn
```

$\&\&?delay < d \ \&\& ?tradeoff = flag \ \&\&?security\_rating > r)$
 $?x \ ns: service\_name \ ? service\_name. \}$

In the above requirement specification in SPARQL query, *situ* is a *situation* [3], [4], which is a set of contexts in SBS over a period of time that affects future system behavior. $\mathbf{P}$ is the vector of security preference, given by the user for relative priority of different security mechanisms in vector S_i . $P = (P_1, P_2, \dots, P_n)$, $0 \leq P_i \leq 1$, $\sum P_i = 1$. For example, if $P_1 = 0.4$, $P_2 = 0.2$, this means that the user thinks confidentiality aspect (S_{i1}) of service is more important than integrity aspect (S_{i2}). Symbol “?” is used in SPARQL language for querying about the variables (such as *service_name*, security vector S , etc.) of the services that meet the requirements in *FILTER* statement. $\mathbf{B}$ is the vector of baseline security levels, given by the user in the form of $B = (B_1, B_2, \dots, B_n)$. $\mathbf{r}$ is the lowest acceptable security rating of any security mechanism. In Secs. 5 and 6, we will discuss the details of security rating and tradeoff algorithm. $\mathbf{B}$, $\mathbf{P}$ and $\mathbf{r}$ are generated by security requirement engineering [10]. d is the longest tolerable service delay set by the user. d can be obtained from real-time requirement engineering process [20]. During QoS-aware service composition process [14], constraint solver in SBS can map d of composite service A to every atomic services using existing off-line constraint deduction process [4] with the input of timing constraints, workflow specification, system status and system control service information. *flag* is the tradeoff Boolean indicator. If the user wants a composite service to have shortest service delay possible and only baseline security protection, user will use 0 for *flag*. If the user wants optimal protection, and only requires service delay not bigger than d , the *flag* is set to 1.

5 Runtime Security Protection Evaluation Criteria

In *S4*) of our approach, the evaluation of service’s security protection status at runtime is needed. A *service_i*’s real-time performance can be measured by service delay d_i . A *service_i*’s security protection status can be obtained in a similar way as the security measurement presented in [13], [21]:

$$G_{Si}(s_i, situ, t) = s_i \times P = s_{i1} * P_1 + s_{i2} * P_2 + \dots + s_{in} * P_n, \quad (1)$$

The measurement of security protection strength $G_{Si}(s_i, situ, t)$ is based on the level of each security mechanism in use and vector s_i of security preference P . However, only this measurement cannot reflect a service’s security protection status in the dynamic environment. Intuitively, a service functioning in a hostile environment has more risk for security breaches than a service with the same security vector running in a safe environment. Safety of the environment can be reflected by the security-related system events monitored, such as failure login attempts, the number of encrypted packets using the same key, the number of illegitimate DHCP server packets filtered out by firewall and TLS/SSL handshake failure rate. Hence, we introduce the concept of *security rating* SR_i of *service_i*:

$$SR_i = (sr_{i1}, sr_{i2}, \dots, sr_{in}), \quad (2)$$

where $0 \leq sr_{ij} \leq 1$, sr_{ij} represents the monitored status of the j^{th} aspect of the service's security mechanisms. Initially, all sr_{ij} in SR_i is set to 1 because no security event has been monitored. As the service keeps running, SR_i can be updated according to organization-specific *security event processing rules* made by security domain experts. Security event processing rules can be categorized into the following two classes:

- **Rate-based rules**, in which the security rating is a function of the amount of security events occurred. For example, a security domain expert of an organization can make a rule that sr_1 will reduce 2% whenever 10^2 packets are encrypted with the same key. This rule depicts the wear out of trust towards an encryption key being used constantly.
- **Exception-based rules**, in which security rating is only related to whether a specific security event occurs in the system environment. For example, if the security log is modified by an unknown user, this indicates that an attacker may have an illegitimate privilege for system resource access. Therefore, the security domain expert can set the rule that the security rating of authorization mechanism will drop to 0 under this condition.

In our approach, AS³ logic is used to specify such rules because AS³ logic provides modalities for declarative specifications of SAW [3].

Similar to (1), we have the measurement of security rating $G_{SR_i}(SR_i, situ, t)$:

$$G_{SR_i}(SR_i, situ, t) = SR_i \times P = sr_{i1} * P_1 + sr_{i2} * P_2 + \dots + sr_{in} * P_n \quad (3)$$

Now we define the *User Expectation Function*, $UEF(s_i, situ, t)$, as a runtime evaluation criteria of a *service_i*'s security protection at time t under situation *situ* as follows:

$$UEF(s_i, SR_i, situ, t) = G_{Sf}(s_i, situ, t) + G_{SR_i}(SR_i, situ, t) \quad (4)$$

UEF can be interpreted as a service's security protection status which is affected by both the security mechanisms used by the service and the service's execution environment.

In service composition of a composite service, an atomic service is invoked based on its position in the workflow and whether its specified precondition situations are satisfied. Therefore, we define the *readiness for service* under *situ*: $r_i(situ)$ as 1 if the preconditions for the atomic service *service_i* have been satisfied under situation *situ*, and the *service_i* is ready for execution or is currently being executed; otherwise, $r_i(situ)$ is defined as 0.

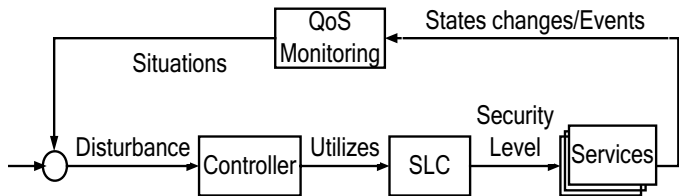


Fig. 2. Our control model for the security and service delay tradeoff in a composite service

The overall security protection $UEF(A, situ, t)$ of a composite service A can be measured by the sum of contribution from $UEF(serv_i, situ, t)$ of all the ready atomic services as follows:

$$UEF(A, situ, t) = \sum r_i(situ) * UEF(serv_i, SR_i, situ, t) \quad (5)$$

6 Control-Based Tradeoff

In this section, we will present a tradeoff controller for tradeoff between security and service delay for SBS in **S6**). The model of the tradeoff controller is shown in Figure 2. Generally, there are many system-specific control services provided by the underlying SBS for managing system settings. One such service, called *Security Level Controller (SLC)* [13], is used by tradeoff controller to adjust security levels of the atomic services of a composite service. The execution process of the composite service produces security events and results in system state change, which are captured by QoS Monitoring module and reported to tradeoff controller as feedback of situations. With the input of situation $situ$, the current security vector s_i and previous security vector s_i' , the estimated $service_i$'s delay on host j can be generated by SLC similar to *earliest start time* in [13]:

$$estd_i(serv_i, s_i', s_i, situ) = r_j + e_i(serv_i, s_i, situ) + h(s_i', s_i) \quad (6)$$

where r_j represents the remaining execution time of a previous service on host j , $e_i(serv_i, s_i, situ)$ represents the estimated execution time of $service_i$ obtained from the service interface, and $h(s_i', s_i)$ represents the time overhead of reconfiguring $service_i$'s security from s_i' to s_i . SLC is for security level reconfiguration and delay estimation for a single atomic service. The overall security and service delay tradeoff of a composite service is achieved by the tradeoff controller using algorithm shown in Figure 3.

The tradeoff controller analyzes if every atomic service of a composite service can satisfy the security requirement B while still satisfy service delay requirement in line 4. If B cannot be satisfied by any atomic service, the composite service will be rejected in line 23, and both the user and service providers will be notified. If user's tradeoff focus for a composite service is on minimizing service delay ($flag=0$), we will use B for all its atomic services in line 5; If user's tradeoff focus for a composite service is on improving security protection ($flag=1$), then the tradeoff controller uses SLC to adjust security levels of atomic services in lines 6-21. SLC will start from the security mechanism whose security rating has been decreased most, and stop until the mechanism whose security rating been increased most. For security mechanisms whose security ratings have changed the same amount of value, SLC will start from the mechanism with the highest security preference value P_m to the lowest value P_n . If the result of adjusting the security level of an atomic service $service_i$ by SLC can increase the composite service's UEF and does not lead to a delay violating requirement d_i (in line 19), the current security level will be accepted for $service_i$. Otherwise, $service_i$ will go back to its previous security level (in line 20).

Let us consider an example service of enhanced netmeeting eNM , which is composed of the atomic services VoIP ($Serv_{VoIP}$), FTP ($Serv_{FTP}$), and video-on-demand ($Serv_{VOD}$). We only consider three aspects of security mechanisms: confidentiality (s_1), integrity (s_2) and authorization (s_3). Suppose for "video conference" situation, the user

sets $P = (0.2, 0.3, 0.5)$ and 0.65 as lowest acceptable security rating r , and 1 for $flag$ value. For confidentiality S_j , $Serv_{VoIP}$ supports three levels: level 0.22: DES, level 0.44: 3-DES, and level 1: AES. The levels are obtained by dividing the key lengths of DES (56bits) and 3DES (112bits) by the key length of the strongest AES (256bits) encryption. Assume at time $t-I$, the security vector of VoIP is $s_{VoIP} = (0.22, 1, 1)$, and the security rating is $sr_{VoIP} = (0.7, 1, 1)$. The voice compression standard for VoIP suggests the maximum acceptable delay for VoIP in wired communication to be 150 ms [22].

```

1. for each host server  $j$  do
2.   Sort component atomic services of the composite service A on  $j$  according to
   service priority in workflow of A from high to low
3.   for each  $service_i$  in the sorted service group of the host  $j$  do
4.     if  $estd_j(serv_i, s_i', B, situ) \leq d_b$  then
5.       if  $flag == 0$ , then  $s_i = B$ , except security level correlated, continue;
6.       else if  $flag == 1$ , and ,
7.         if there exists an  $sr_{ik} < r$ , then  $s_i = B$ , except security level correlated;
8.         Use (5) to calculate  $UEF(A, situ, t)$  and store the result in  $U$ ;
9.         for each  $sr_{ik}$  of  $service_i$  do
10.           $\Delta sr_{ik} = sr_{ik}(t) - sr_{ik}(t-I)$ ;
11.        end for
12.        Sort  $s_{ik}$  of  $service_i$ 's security level  $s_i$  according to  $\Delta sr_{ik} * P_k$ , so that
         $\Delta sr_{i1} * P_1 < \Delta sr_{i2} * P_2 < \dots < \Delta sr_{in} * P_n$ ; if  $\Delta sr_{im} * P_m = \Delta sr_{in} * P_n$ , sort  $s_{im}$  and  $s_{in}$  accord-
        ing to  $P$ , so that  $P_m > P_n$ ;
13.        for each  $s_{ik}$ ,  $1 \leq k \leq n$ , do
14.          if  $s_{ik}$  has not been correlated, then
15.            while  $s_{ik} \leq \max \{S_{ik}\}$  do
16.              Increase security level of  $s_{ik}$  with  $SLC$  to next level;
17.              Correlate security level with other services with dependency in
              workflow;
18.              Use (5) to calculate  $UEF(A, situ, t)$  with current  $s_i$  and  $SR_i$  of
               $service_i$ ;
19.              if  $UEF(A, situ, t) \geq U$  AND  $estd_i(serv_i, s_i', s_i, situ) \leq d_i$ , then
               $U = UEF(A, situ, t)$ ;
20.              else decrease  $s_{ik}$ , break;
21.            end while
22.          end for
23. else reject composite service A, notify both users and service providers of A;
24. end for

```

Fig. 3. The tradeoff algorithm for the controller of a composite service

Assume that the security domain expert gives a rate-based security event processing rule that “security rating of confidentiality will reduce 2% whenever 10^2 packets are encrypted”, which can be specified in AS³ logic as follows:

SERVI $Encrypt\_pack(int(pack\#), rbr, saw\_rbrAgent) \rightarrow serv(int(pack\#), rbr, saw\_rbrAgent)$
ASI $serv(int(pack\#), rbr, saw\_rbrAgent) \quad pack\# \geq 10^2 \rightarrow diam(k[Decrease, 0.02], monitor\_until(-1, success), saw\_packageAgent))$

In the above rate-based rule, *saw_rbrAgent* is a synthesized SAW agent responsible for services related to rate-based rules (rbr). *saw_packageAgent* is a synthesized SAW agent monitoring the situations related to packages on networks.

At time t , the QoS Monitoring detects that the number of packets encrypted is 500 packets per second. Hence, sr_i decreases to 0.6. According to lines 9-11 of our tradeoff algorithm in Figure 3, we can calculate the change of security rating $\Delta sr_{VoIP1} = 0.6 - 0.7 = -0.1$. Assume that the security levels and ratings of S_{FTP} and S_{VOD} unchanged at time t and there is no security dependency. For S_{VoIP} , after the sorting process of line 13, we have $\Delta G_{SRVoIP} = \Delta sr_{VoIP1} * P_I = -0.1 * 0.2 = -0.02$. Hence, as in lines 13-20, we increase the security level of confidentiality S_{VoIP1} . According to the experimental results of voice over IPSec [23], the VoIP service with different encryption algorithms for confidentiality purpose have the delays shown in Table 1.

Table 1. Tradeoff options for service S_{VoIP}

S_{VoIP1}	ΔG_{SLVoIP}	delay	$\Delta_{UEF}(eNM, s, t)$
0.44	$(0.44 - 0.22) * 0.2 = 0.044$	110 ms	$0.044 - 0.02 = 0.024$
1	$(1 - 0.2) * 0.2 = 0.16$	152 ms	$0.16 - 0.02 = 0.14$

Although the security level 1 of S_{VoIP1} can increase $UEF(eNM, s, t)$ further, the delay at level 1 becomes 152 ms, which is unbearable according to line 20. Based on our tradeoff algorithm in Figure 3, S_{VoIP} will adapt its confidentiality security level to $s_{VoIP1} = 0.44$ (3DES), which will improve $UEF(eNM, s, t)$ and guarantee that eNM finishes less than 150ms as well.

7 Conclusion and Future Work

In this paper, we have presented a control-based approach to the design of SBS with the dynamic balance between service's security and performance due to the change of the situations of SBS. We have extended *SAW-OWL-S* to specify service interface. *SPARQL query* is used to depict user's expectations for security, service delay and tradeoff preference. *User Expectation Function* has been derived to measure security protection at runtime. Future work includes simulation of our approach to evaluate its effectiveness, and relaxation of QoS requirements and composite service reconfiguration after service requests are rejected.

Acknowledgment

This research was supported by National Science Foundation under grant number CNS-0524736 and DoD/ONR under MURI Program, contract number N00014-04-1-0723. The authors would like to thank Zhaoji Chen, Junwei Liu, Yin Yin, and Luping Zhu for many helpful discussions.

References

1. Jones, S.: Toward an Acceptable Definition of Service. *IEEE Software* 22(3), 87–93 (2005)
2. Yau, S.S., et al.: Situation-Awareness for Adaptable Service Coordination in Service-based Systems. In: *Proc. 29th Annual Int'l. Computer Software and Application Conf.* pp. 107–112 (2005)
3. Yau, S.S., et al.: Automated Agent Synthesis for Situation Awareness in Service-based Systems. In: *Proc. 30th Annual Int'l. Computer Software and App. Conf.* pp. 503–510 (2006)
4. Yau, S.S., et al.: A Software Cybernetic Approach to Deploying and Scheduling Workflow Applications in Service-based Systems. In: *Proc. 11th Int'l. Workshop on Future Trends of Distributed Computing Systems*, pp. 149–156 (2007)
5. Abdelzaher, T.F., et al.: Feedback Performance Control in Software Services. *IEEE Control Systems Magazine* 23(3), 74–90 (2003)
6. Tsai, W.T., et al.: RTSOA: Real-Time Service-Oriented Architecture. In: *Proc. 2nd IEEE Int'l. Workshop on Service-Oriented System Engineering*, pp. 49–56 (2006)
7. Hao, W., et al.: An Infrastructure for Web Services Migration for Real-Time Applications. In: *Proc. 2nd IEEE Int'l. Workshop on Service-Oriented System Engineering*, pp. 41–48 (2006)
8. Lu, C., et al.: Feedback Control Architecture and Design Methodology for Service Delay Guarantees in Web Servers. *IEEE Trans. on Parallel and Distributed Systems* 17(9), 1014–1027 (2006)
9. Yau, S.S., Yao, Y., Yan, M.: Development and Runtime Support for Situation-Aware Security in Autonomic Computing. In: *Proc. 3rd Int'l. Conf. on Autonomic and Trusted Computing*, pp. 173–182 (2006)
10. Wada, H., Suzuki, J., Oba, K.: A Service-Oriented Design Framework for Secure Network Applications. In: *Proc. 30th Annual Int'l. Computer Software and App. Conf.* pp. 359–368 (2006)
11. Spyropoulou, E., Levin, T., Irvine, C.: Calculating Costs for Quality of Security Service. In: *Proc. 16th Annual Conf. Computer Security Applications*, pp. 334–343 (2000)
12. Son, S.H., Zimmerman, R., Hansson, J.: An Adaptable Security Manager for Real-Time Transactions. In: *Proc. 12th Euromicro Conf. on Real-Time Systems*, pp. 63–70 (2000)
13. Xie, T., et al.: Real-Time Scheduling with Quality of Security Constraints. *Int'l. Jour. High Performance Computing and Networking* (2006)
14. Berbner, R., et al.: Heuristics for QoS-aware Web Service Composition. In: *Proc. Int'l Conf. on Web Services*, pp. 72–82 (2006)
15. SPARQL Query Language for RDF. W3C Working Draft 26 (2007)
<http://www.w3.org/TR/rdf-sparql-query/>
16. Yau, S.S., Liu, J.: Functionality-based Service Matchmaking for Service-Oriented Architecture. In: *Proc. of 8th Int'l. Symp. on Autonomous Decentralized Systems*, pp. 147–152 (2007)
17. Kang, K., Son, S.: Systematic Security and Timeliness Tradeoffs in Real-Time Embedded Systems. In: *Proc. 12th IEEE Int'l. Conf. on Embedded and Real-Time Computing Systems and Applications*, pp. 183–189 (2006)
18. Yau, S.S., Liu, J.: Incorporating Situation Awareness in Service Specifications. In: *Proc. 9th IEEE Int'l. Symp. on Object and Component-oriented Real-time Distributed Computing*, pp. 287–294 (2006)

19. Cavanaugh, C.D.: Toward a Simulation Benchmark for Distributed Mission-Critical Real-time Systems. In: Proc. Networking, Sensing and Control, pp. 1037–1042 (2005)
20. Goldsack, S.J., Finkelstein, A.C.W.: Requirements Engineering for Real-time Systems. *Jour. Software Engineering* 6(3), 101–115 (1991)
21. Wang, C., Wulf, W.A.: A Framework for Security Measurement. In: Proc. National Information Systems Security Conf. pp. 522–533 (1997)
22. Barbieri, R., Bruschi, D., Rosti, E.: Voice over IPsec: Analysis and Solutions. In: Proc. 18th Annual Computer Security Applications Conference, pp. 261–270 (2002)
23. Nascimento, A., Passito, A., Mota, E.: Can I Add a Secure VoIP Call. In: Proc. 2006 Int'l. Symp. On a World of Wireless, Mobile and Multimedia (2006)

Provably Secure Identity-Based Threshold Unsignryption Scheme^{*}

Bo Yang¹, Yong Yu², Fagen Li³, and Ying Sun¹

¹ College of information,
South China Agricultural University, Guangzhou, 510642, P.R. China
`{byang,sunying}@scau.edu.cn`

² National Key Lab. of ISN,
Xidian University, Xi'an, 710071, P.R. China
`yuyong@mail.xidian.edu.cn`

³ School of Computer Science and Engineering,
University of Electronic Science and Technology of China, Chengdu, 610054, China
`fagenli@21cn.com`

Abstract. Signcryption is a cryptographic primitive that performs signature and encryption simultaneously. In this paper, we propose an identity based threshold unsignryption scheme, which is the organic combination of the signcryption scheme, the (t, n) threshold scheme and zero knowledge proof for the equality of two discrete logarithms based on the bilinear map. In this scheme, a signcrypted message can be decrypted only when at least t members join an unsignryption protocol. We also prove its security in a formal model under recently studied computational assumptions and in the random oracle model. Specifically, we prove its semantic security under the hardness of q -Bilinear Diffie-Hellman Inversion problem and its unforgeability under the q -Strong Diffie-Hellman assumption.

1 Introduction

Identity based (ID-based) cryptosystem was introduced by Shamir [1] in 1984. Its main idea is that public keys can be derived from arbitrary strings while private keys can be generated by the trusted Private Key Generator (PKG). This removes the need for senders to look up the receiver's public key before sending out an encrypted message. ID-based cryptography is supposed to provide a more convenient alternative to conventional public key infrastructure.

Signcryption, first proposed by Zheng [2], is a cryptographic primitive that performs signature and encryption simultaneously, at a lower computational costs and communication overheads than the signature-then-encryption approach. Followed by the first constructions given in [2], a number of new schemes and improvements have been proposed [3, 4, 5, 6, 7, 8, 9, 10, 11, 12]. In [7], Malone-Lee proposed the first ID-based signcryption scheme. Libert and Quisquater [8] pointed

^{*} This work was supported by the National Natural Science Foundation of China under Grants No. 60372046 and 60573043.

out that Malone-Lee's scheme [7] is not semantically secure and proposed three provably secure ID-based signcryption schemes. However, the properties of public verifiability and forward security are mutually exclusive in their schemes. To overcome this weakness, Chow *et al.* [9] designed an ID-based signcryption scheme that provides both public verifiability and forward security. In [11], Chen and Malone-Lee improved Boyen's scheme in efficiency. In [12], Barreto *et al.* constructed the most efficient ID-based signcryption scheme to date.

All of the above schemes consist of only a single receiver. In many cases such as in a sealed-bid auction scheme [15], however, we need to prohibit a single receiver from recovering a signcrypted message in order to prevent a single point of failure or abuse. In 2001, Koo *et al.* [16] proposed a new signcryption in which at least t receivers must participate in an unsignryption process. However, their scheme is based on discrete logarithm problem, not ID-based. Recently, Li *et al.* [17] proposed an ID-based threshold unsignryption scheme from pairings.

Our Contributions: In this paper, we propose an efficient ID-based threshold unsignryption scheme, which is the organic combination of the signcryption scheme, the (t, n) threshold scheme and zero knowledge proof for the equality of two discrete logarithms based on the bilinear map. In our scheme, a signcrypted message can be decrypted only when at least t members join an unsignryption protocol. We also prove its security in a formal model under recently studied computational assumptions and in the random oracle model. Specifically, we prove its semantic security under the hardness of q -Bilinear Diffie-Hellman Inversion problem and its unforgeability under the q -Strong Diffie-Hellman assumption.

Roadmap: The rest of this paper is organized as follows. Section 2 presents the basic concepts of bilinear map groups and the hard problems underlying our proposed scheme. Section 3 gives the syntax and security notions of ID-based threshold unsignryption schemes. We describe ID-based threshold unsignryption scheme and prove its security in section 4. We draw our conclusion in section 5.

2 Preliminaries

2.1 Bilinear Pairings and Related Computational Problems

Let $\mathbb{G}_1, \mathbb{G}_2$ be cyclic additive groups generated by P_1, P_2 respectively, whose orders are a prime q . Let $\mathbb{G}_T$ be a cyclic multiplicative group with the same order q . We assume there is an isomorphism $\psi : \mathbb{G}_2 \rightarrow \mathbb{G}_1$ such that $\psi(P_2) = P_1$. Let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ be a bilinear mapping with the following properties:

1. *Bilinearity:* $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$ for all $P \in \mathbb{G}_1, Q \in \mathbb{G}_2, a, b \in \mathbb{Z}_q$.
2. *Non-degeneracy:* There exists $P \in \mathbb{G}_1, Q \in \mathbb{G}_2$ such that $\hat{e}(P, Q) \neq 1_{\mathbb{G}_T}$.
3. *Computability:* There exists an efficient algorithm to compute $\hat{e}(P, Q)$ for all $P \in \mathbb{G}_1, Q \in \mathbb{G}_2$.

The computational assumptions for the security of our schemes were formalized by Boneh and Boyen [14] and are reviewed in the following. Let us consider bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ and generators $P \in \mathbb{G}_1$ and $Q \in \mathbb{G}_2$.

1. **q -Strong Diffie-Hellman problem(q -SDHP).** Given a $(q+2)$ -tuple $(P, Q, \alpha Q, \alpha^2 Q, \dots, \alpha^q Q)$, find a pair $(c, \frac{1}{c+\alpha}P)$ with $c \in \mathbb{Z}_p^*$.
2. **q -Bilinear Diffie-Hellman Inversion problem(q -BDHIP).** Given a $(q+2)$ -tuple $(P, Q, \alpha Q, \alpha^2 Q, \dots, \alpha^q Q)$, compute $e(P, Q)^{\frac{1}{\alpha}} \in \mathbb{G}_T$.

2.2 Baek and Zheng's Zero Knowledge Proof for the Equality of Two Discrete Logarithms Based on the Bilinear Map

We omit this section for space limitation and refer the readers to [18] for details.

3 Formal Model of ID-Based Threshold Unsigncryption

We omit this section for page limitation.

4 The Proposed Scheme and Security Results

4.1 The Proposed Scheme

In this section, we propose an ID-based threshold unsigncryption scheme. The proposed scheme involves four roles: the PKG, the sender Alice, a legitimate user U that wants to unsigncrypt the ciphertext and the message receiver group $B = \{B_1, B_2, \dots, B_n\}$. It consists of the following eight algorithms.

Setup: Given a security parameter k , the PKG chooses bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ of prime order $p > 2^k$ and generators $Q \in \mathbb{G}_2$, $P = \psi(Q) \in \mathbb{G}_1$, $g = e(P, Q) \in \mathbb{G}_T$. It then chooses a master key $s \in \mathbb{Z}_p^*$, a system-wide public key $Q_{pub} = sQ \in \mathbb{G}_2$ and Hash functions $H_1 : \{0, 1\}^* \rightarrow \mathbb{Z}_p^*$, $H_2 : \{0, 1\}^* \times \mathbb{G}_T \rightarrow \mathbb{Z}_p^*$, $H_3 : \mathbb{G}_T \rightarrow \{0, 1\}^n$ and $H_4 : \mathbb{G}_T \times \mathbb{G}_T \times \mathbb{G}_T \rightarrow \mathbb{Z}_p^*$. The public parameters are $\text{params} = \{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P, Q, g, Q_{pub}, e, \psi, H_1, H_2, H_3, H_4\}$.

Keygen: For an identity ID , the private key is $D_{ID} = \frac{1}{H_1(ID)+s}Q \in \mathbb{G}_2$.

Keydis: Suppose that we have chosen a threshold value t and n satisfying $1 \leq t \leq n < q$. The PKG picks $R_1, R_2, \dots, R_{t-1}$ at random from G_2^* and constructs a function $F(x) = D_{ID_B} + \sum_{j=1}^{t-1} x^j R_j$. Then, the PKG computes the private key $D_i = F(i)$ and the verification key $y_i = e(P, D_i)$ for the receiver $B_i (1 \leq i \leq n)$. Subsequently, the PKG secretly sends the private key D_i and the verification key y_i to B_i . B_i then keeps D_i secret while making y_i public.

Signcrypt: Suppose Alice whose identity is ID_A wants to signcrypt a message $m \in \{0, 1\}^*$ to the receiver group B , she computes the ciphertext $\sigma = (c, S, T)$ as follows:

1. Pick randomly $x \in \mathbb{Z}_p^*$, compute $r = g^x$ and $c = m \oplus H_3(r)$.
2. Set $h = H_2(m, r)$.
3. Compute $S = (x + h)\psi(D_{ID_A})$.
4. Compute $T = x(H_1(ID_B)P + \psi(Q_{pub}))$.

Sharegen: A legitimate user U sends σ to each member of group B and requests unsigncryption shares. Each $B_i (1 \leq i \leq n)$ computes $\tilde{r}_i = e(T, D_i)$, $\tilde{u}_i = e(T, T_i)$, $u_i = e(P, T_i)$, $v_i = H_4(\tilde{r}_i, \tilde{u}_i, u_i)$ and $W_i = T_i + v_i D_i$ for random $T_i \in \mathbb{G}_2$ and sends $\sigma_i = (i, \tilde{r}_i, \tilde{u}_i, u_i, v_i, W_i)$ to the user U . Otherwise, B_i returns *Invalid Ciphertext*.

Sharever: U firstly computes $v'_i = H_4(\tilde{r}_i, \tilde{u}_i, u_i)$ and checks whether $v'_i = v_i$, $e(T, W_i)/\tilde{r}_i^{v'_i} = \tilde{u}_i$ and $e(P, W_i)/y_i^{v'_i} = u_i$. If these tests hold, the σ_i from B_i is a valid unsigncryption share. Otherwise, U returns *Invalid Share*.

Sharecom: When U collects valid unsigncryption shares from at least t members in the group B , U computes $r' = \prod_{j=1}^t \tilde{r}_j^{N_j}$, where $N_j = \prod_{i=1, i \neq j}^t \frac{-ID_i}{ID_j - ID_i}$ and recovers the message $m = c \oplus H_3(r')$.

Sigver: U computes $h' = H_2(m, r')$ and accepts the message (signature) if $r' = e(S, H_1(ID_A)Q + Q_{pub})g^{-h'}$.

4.2 Security Results

The correctness of the scheme can be verified easily. The following theorems claim the security of the scheme in the random oracle model under the same irreflexivity assumption as Boyen's scheme [10]: the signcryption algorithm is assumed to always take distinct identities as inputs. In other words, a principal never encrypts a message bearing his signature using his own identity.

Theorem 1. *In the random oracle model, assume that an IND-IDTUSC-CCA $\mathcal{A}$ has an advantage ε against the proposed scheme when running in time t , asking q_{h_i} queries to random oracles $H_i (i = 1, 2, 3, 4)$, q_{se} signcryption queries and q_{ds} decryption share queries. Then there is an algorithm $\mathcal{B}$ to solve the q -BDHIP for $q = q_{h_1}$ with probability*

$$\epsilon' > \frac{\varepsilon}{q_{h_1}(2q_{h_2} + q_{h_3})} \left(1 - q_{se} \frac{q_{h_2} + q_{h_2}}{2^k}\right) \left(1 - \frac{q_{ds}q_{h_4}}{2^k}\right)$$

within a time $t' < t + O(q_{se} + q_{ds})t_p + O(q_{h_1}^2)t_{mult} + O(q_{ds}q_{h_2})t_{exp}$ where t_{exp} and t_{mult} are respectively the costs of an exponentiation in $\mathbb{G}_T$ and a multiplication in $\mathbb{G}_2$ whereas t_p denotes the computation time of a pairing computation.

Proof. Algorithm $\mathcal{B}$ takes as input $(P, Q, \alpha Q, \alpha^2 Q, \dots, \alpha^q Q)$ and aims to extract $e(P, Q)^{\frac{1}{\alpha}}$. In a preparation phase, $\mathcal{B}$ builds a generator $G \in \mathbb{G}_1$ such that it knows $q - 1$ pairs $(\omega_i, \frac{1}{\omega_i + \alpha} G)$ for $\omega_1, \dots, \omega_{q-1} \in \mathbb{Z}_p^*$. To do so,

1. It picks $\omega_1, \dots, \omega_{q-1} \in \mathbb{Z}_p^*$ and expands $f(z) = \prod_{i=1}^{q-1} (z + \omega_i)$ to obtain

$$c_0, \dots, c_{q-1} \in \mathbb{Z}_p^* \text{ so that } f(z) = \sum_{i=0}^{q-1} c_i z^i.$$

2. It sets generators $H = \sum_{i=0}^{q-1} c_i (\alpha^i Q) = f(\alpha)Q \in \mathbb{G}_2$ and $G = \psi(H) = f(\alpha)P \in$

$\mathbb{G}_1$. The public key $H_{pub} \in \mathbb{G}_2$ is fixed to $H_{pub} = \sum_{i=1}^q c_{i-1} (\alpha^i Q)$ so that $H_{pub} = \alpha H$ although $\mathcal{B}$ does not know α .

3. For $1 \leq i \leq q-1$, $\mathcal{B}$ expands $f_i(z) = f(z)/(z + \omega_i) = \sum_{i=0}^{q-2} d_i z_i$ and
- $$\sum_{i=0}^{q-2} d_i \psi(\alpha^i Q) = f(\alpha)P = \frac{f(\alpha)}{\alpha + \omega_i} P = \frac{1}{\alpha + \omega_i} G.$$

The pair $(\omega_i, \frac{1}{\alpha + \omega_i} G)$ are computed using the left member of above equation. Then $\mathcal{B}$ chooses randomly $i \in \{1, \dots, q\} \setminus \{l\}$ and $\omega_1, \dots, \omega_{l-1}, \omega_{l+1}, \dots, \omega_q \in \mathbb{Z}_p^*$. For $i \in \{1, \dots, q\} \setminus \{l\}$, it computes $I_i = I_l - \omega_i$. Using the technique described above, it sets up generators $G_2 \in \mathbb{G}_2$, $G_1 = \psi(G_2) \in \mathbb{G}_1$ and another element $U = \alpha G_2$ such that it knows $q-1$ pairs $(\omega_i, H_i = \frac{1}{\omega_i + \alpha})$ for $i \in \{1, \dots, q\} \setminus \{l\}$. The system-wide public key Q_{pub} is set to be $Q_{pub} = -U - I_l G_2 = (-\alpha - I_l) G_2$ so that its (unknown) private key is implicitly set to $x = -\alpha - I_l \in \mathbb{Z}_p^*$. For all $i \in \{1, \dots, q\} \setminus \{l\}$, we have $(I_i, -H_i) = (I_i, (1/I_i + x) G_2)$.

$\mathcal{B}$ then initialize a counter $v=1$ and starts $\mathcal{A}$ on input (G_1, G_2, Q_{pub}) . Throughout the game, we assume that H_1 -queries are distinct and any query involving an identity ID comes after a H_1 -query on ID . We also assume that ciphertext returned from a signcryption query will not be used by $\mathcal{A}$ in an decryption share query. Now we explain how the queries are treated by $\mathcal{B}$. During the game, $\mathcal{A}$ will consult $\mathcal{B}$ for answers to the random oracles H_1, H_2, H_3 and H_4 . Roughly speaking, these answers are randomly generated, but to maintain the consistency and to avoid collision, $\mathcal{B}$ maintains four lists L_1, L_2, L_3, L_4 respectively to store the answers used.

H_1 -queries: Let ID_t be the input of the t^{th} query on H_1 , $\mathcal{B}$ answers I_t and increments t . Then $\mathcal{B}$ stores (ID_t, I_t) in L_1 .

H_2 -queries: On a $H_2(m, r)$ query, $\mathcal{B}$ returns the defined value if it exists in L_2 and a random $h_2 \in \mathbb{Z}_p^*$ otherwise. To anticipate possible decryption share queries, $\mathcal{B}$ additionally simulates random oracles H_3 on its own to obtain $h_3 = H_3(r)$ and stores the information $(m, r, h_2, c = m \oplus H_3, \eta = r \cdot e(G_1, G_2)^{h_2})$ in L_2 .

H_3 -queries: For a query $H_3(r)$, $\mathcal{B}$ returns the previously assigned value if it exists in L_3 and a random $h_3 \in \{0, 1\}^n$ otherwise. In the latter case, the input r and the response h_3 are stored in the list L_3 .

H_4 -queries: For a query $H_4(\tilde{y}_e, \tilde{u}_e, u_e)$, $\mathcal{B}$ returns the previously assigned value if it exists in list L_4 . Otherwise he chooses randomly $v \in \mathbb{Z}_q^*$, gives it as an answer to the query and puts the tuple $(\tilde{y}_e, \tilde{u}_e, u_e, v)$ into L_4 .

Keygen queries: For a query $\text{Keygen}(ID_t)$, if $t = l$, then $\mathcal{B}$ fails. Otherwise, it knows that $H_1(ID_t) = I_t$ and returns $-H_t = \frac{1}{I_t + x} G_2$.

Signcryption queries: For a signcryption queries on a plaintext m and identities $(ID_S, ID_R) = (ID_u, ID_v)$ for $u, v \in \{1, \dots, q_{h_1}\}$. If $u \neq l$, $\mathcal{B}$ knows the sender's private key $D_{ID_u} = -H_u$ and can answer the query by running Signcryption algorithm. We thus assume that $u = l$ and hence $v \neq l$ by the irreflexivity assumption. Therefore, $\mathcal{B}$ knows the receiver's private key $D_{ID_v} = -H_v$. In order to answer $\mathcal{A}$'s query, $\mathcal{B}$ randomly chooses $t, h \in \mathbb{Z}_p^*$ and computes $S = t\psi(D_{ID_v}) = -t\psi(H_v)$, $T = t\psi(Q_{ID_l}) - h\psi(Q_{ID_v})$, where $Q_{ID_v} = I_v G_2 + Q_{pub}$ in order to obtain the desired equality $r = e(T, D_{ID_v}) =$

$e(S, Q_{ID_l})e(G_1, G_2)^{-h} = e(\psi(D_{ID_v}), Q_{ID_l})^t e(G_1, G_2)^{-h}$ before patching the hash value $H_2(m, r)$ to h . $\mathcal{B}$ fails if H_2 is already defined but this only happens with probability $(q_{h_2} + q_{se})/2^k$. The ciphertext $\sigma = (m \oplus H_3(r), S, T)$ is returned to $\mathcal{A}$.

Decryption share queries to the uncorrupted members: Suppose that the t -th member has not been corrupted by $\mathcal{A}$. When $\mathcal{A}$ observes a ciphertext $\sigma = (c, S, T)$ for identities $(ID_S, ID_R) = (ID_u, ID_v)$ for $u, v \in \{1, \dots, q_{h_1}\}$, he may want to ask $\mathcal{B}$ for the decryption share of σ . If $v \neq l$, $\mathcal{B}$ knows the receiver's private key $D_{ID_v} = -H_v$ and can normally run *Keydis* and *Sharegen* algorithms to answer $\mathcal{A}$'s queries. So we assume $v = l$, and therefore $u \neq l$ by the irreflexivity assumption. Consequently, $\mathcal{B}$ has the sender's private key D_{ID_u} and also knows that, for all valid ciphertexts, $\log_{D_{ID_u}}(\psi^{-1}(S) - hD_{ID_u}) = \log_{\psi(Q_{ID_v})}(T)$, where $h = H_2(m, r)$ is the hash value obtained in the signcryption algorithm and $Q_{ID_v} = I_v G_2 + Q_{pub}$. Therefore, we have the relation $e(T, D_{ID_u}) = e(\psi(Q_{ID_v}), \psi^{-1}(S) - hD_{ID_u})$ which yields $e(T, D_{ID_u}) = e(\psi(Q_{ID_v}), \psi^{-1}(S))e(\psi(Q_{ID_v}), D_{ID_u})^{-h} = e(S, Q_{ID_v})e(\psi(Q_{ID_v}), D_{ID_u})^{-h}$. This query is thus handled by computing $\eta = e(S, Q_{ID_u})$ where $Q_{ID_u} = I_u G_2 + Q_{pub}$, and searching through list L_2 for entries of the form $(m_i, r_i, h_{2,i}, c, \eta)$. If none is found, σ is rejected. Otherwise, $\mathcal{B}$ first runs *Keydis* algorithm for r_i to obtain private/verification key pairs $\{D_l, y_l\}$, where $1 \leq l \leq n$ and computes $\tilde{r}_t = e(T, D_t)$. Next, he chooses W_t and v_t uniformly at random from $\mathbb{G}_2$ and Z_q^* respectively, and computes $\tilde{u}_t = e(T, W_t)/\tilde{r}_t^{v_t}$ and $u_t = e(P, W_t)/y_t^{v_t}$. Then, $\mathcal{B}$ sets $v_t = H_4(\tilde{r}_t, \tilde{u}_t, u_t)$. Finally, he check if L_4 contains a tuple $(\tilde{r}_t, \tilde{u}_t, u_t, v'_t)$ with $v'_t \neq v_t$. In this case, $\mathcal{B}$ repeats the process with another random pair (W_t, v_t) until finding a tuple $(\tilde{r}_t, \tilde{u}_t, u_t, v_t)$ whose first three elements do not figure in a tuple of the list L_4 . Otherwise, $\mathcal{B}$ returns the simulated value $\sigma_t = (t, \tilde{r}_t, \tilde{u}_t, u_t, v_t, W_t)$ as a unsigncryption share corresponding to σ and saves $(\tilde{r}_t, \tilde{u}_t, u_t, v_t)$ to L_4 . The above simulated decryption share generation perfectly simulates the real one except the collision in the simulation of H_4 occurs. Note that this happens with probability $q_{h_4}/2^k$. Adding up all the decryption queries up to q_{ds} , so $\mathcal{B}$ fails in the simulation with probability at most $q_{ds}q_{h_4}/2^k$.

At the challenge phase, $\mathcal{A}$ outputs two messages (m_0, m_1) and identities (ID_S, ID_R) for which she never obtained ID_R 's private key. If $ID_R \neq ID_l$, $\mathcal{B}$ aborts. Otherwise, it randomly chooses $\theta \in \mathbb{Z}_p^*$, $c \in \{0, 1\}^n$ and $S \in \mathbb{G}_1$ to return the challenge ciphertext $\sigma^* = (c, S, T)$ where $T = -\theta G_1$. If we define $\rho = \theta/\alpha$ and since $x = -\alpha - I_l$, we can check that $T = -\theta G_1 = -\alpha \rho G_1 = (I_l + x)\rho G_1 = \rho I_l G_1 + \rho \psi(Q_{pub})$. $\mathcal{A}$ cannot recognize that σ^* is not a proper ciphertext unless she queries H_2 or H_3 on $e(G_1, G_2)^\rho$. $\mathcal{A}$ then performs a second series of queries which are treated in the same way as the first one and finally she outputs a bit b' .

$\mathcal{B}$ ignores $\mathcal{A}$'s output and takes a random entry (m, r, h_2, c, η) from L_2 or $(r, \cdot)$ from L_3 . As L_3 contains no more than $q_{h_2} + q_{h_3}$ records, therefore, with probability at least $1/(2q_{h_2} + q_{h_3})$, the chosen entry will contain the right element

$r = e(G_1, G_2)^\rho = e(P, Q)^{f(\alpha)^2\theta/\alpha}$, where $f(z) = \sum_{i=0}^{q-1} c_i z^i$ is the polynomial for which $G_2 = f(\alpha)Q$. The q -BDHIP solution can be extracted by noting that, if $\eta^* = e(P, Q)^{1/\alpha}$, then

$$e(G_1, G_2)^{1/\alpha} = \eta^{*(c_0^2)} e\left(\sum_{i=0}^{q-2} c_{i+1}(\alpha^i P), c_0 Q\right) e\left(G_1, \sum_{j=0}^{q-2} c_{j+1}(\alpha^j) Q\right)$$

We now have to access $\mathcal{B}$'s probability of success. Note that it only fails in providing a consistent simulation because one of the following independent events:

E_1 : $\mathcal{A}$ does not choose to be challenged on ID_l .

E_2 : a key extraction query is made on ID_l .

E_3 : $\mathcal{B}$ aborts in a *signcryption query* because of a collision on H_2 .

E_4 : $\mathcal{B}$ aborts in a *decryption share query* because of a collision on H_4 .

We have $Pr[\neg E_1] = 1/q_{h_1}$ and we know that $\neg E_1$ implies $\neg E_2$. We have also observed that $Pr[E_3] \leq q_{se}(q_{se} + q_{h_2})/2^k$ and $Pr[E_4] \leq q_{ds}q_{h_4}/2^k$. Therefore, we find that

$$Pr[\neg E_1 \wedge \neg E_3 \wedge \neg E_4] \geq \frac{1}{q_{h_1}} \left(1 - q_{se} \frac{q_{se} + q_{h_2}}{2^k}\right) \left(1 - \frac{q_{ds}q_{h_4}}{2^k}\right)$$

Note that $\mathcal{B}$ selects the correct element from L_2 or L_3 with probability $1/(2q_{h_2} + q_{h_3})$. Therefore, $\mathcal{B}$'s probability of success is as follows.

$$Adv(\mathcal{B}) = \epsilon' > \frac{\epsilon}{q_{h_1}(2q_{h_2} + q_{h_3})} \left(1 - q_{se} \frac{q_{se} + q_{h_2}}{2^k}\right) \left(1 - \frac{q_{ds}q_{h_4}}{2^k}\right)$$

The running time is dominated by $O(q_{h_1}^2)$ multiplications in the preparation phase, $O(q_{se} + q_{ds})$ pairings and $O(q_{ds}q_{h_2})$ exponentiations in $\mathbb{G}_T$ in the simulation of *signcryption* and *decryption share* oracles.

Theorem 2. *In the random oracle model, if there exists an ESUF-IBSC-CMA attacker $\mathcal{A}$ that makes q_{h_i} queries to H_i ($i = 1, 2, 3, 4$), q_{se} signcryption queries and q_{ds} decryption share queries. Assume that, within a time t , $\mathcal{A}$ produces a forgery with probability $\epsilon \geq 10(q_{se} + 1)(q_{se} + q_{h_2})/2^k$. Then, there exists an algorithm $\mathcal{B}$ that is able to solve the q -SDHP for $q = q_{h_1}$ in expected time*

$$t' \leq 120686q_{h_1}q_{h_2}(t + O((q_{se} + q_{ds})t_p) + q_{ds}q_{h_2}t_{exp}) / (\epsilon(1 - 1/2^k)(1 - q/2^k)) + O(q^2 t_{mult})$$

where t_{mult} , t_{exp} and t_p denote the same quantities as in theorem 1.

Proof. (sketch). It shows that a forger in the ESUF-IBSC-CMA game implies a forger in a chosen-message and *given* identity attack. Using the forking lemma [20], the latter is in turn shown to imply an algorithm to solve q -Strong Diffie-Hellman problem. More precisely, queries to the *signcryption* and *decryption share* oracles are answered as in the proof in theorem 1 and, at the outset of the game, the simulator chooses public parameters in such a way that it can extract private keys associated to any identity but the one which is given as a challenge to the adversary. In this way, it is able to extract plain message-signature pairs from ciphertexts produced by the forger. We refer the readers to [12] for more details about how to extract plain message-signature pairs.

5 Conclusion

We have successfully integrated the design ideas of the ID-based signcryption scheme, the (t, n) threshold scheme and zero knowledge proof for the equality of two discrete logarithms based on the bilinear map, and have proposed an ID-based threshold unsignryption scheme. In this scheme, a signcrypted message can be decrypted only when at least t members join an unsignryption protocol. We have also proven its security in a formal model under recently studied computational assumptions and in the random oracle model. Further work is on the way to construct efficient and provably-secure threshold unsignryption schemes *without random oracles*.

References

1. Shamir, A.: Identity-based cryptosystem and signature scheme. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 120–126. Springer, Heidelberg (1985)
2. Zheng, Y.: Digital signcryption or how to achieve $\text{cost}(\text{signature} \ \& \ \text{encryption}) \ll \text{cost}(\text{signature}) + \text{cost}(\text{encryption})$. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 165–179. Springer, Heidelberg (1997)
3. Petersen, H., Michels, M.: Cryptanalysis and improvement of signcryption schemes. IEE proceedings-Computers and Digital Techniques 145(2), 149–151 (1998)
4. Bao, F., Deng, R.H.: A signcryption scheme with signature directly verifiable by public key. In: Imai, H., Zheng, Y. (eds.) PKC 1998. LNCS, vol. 1431, pp. 55–59. Springer, Heidelberg (1998)
5. Zheng, Y., Imai, H.: How to construct efficient signcryption schemes on elliptic curves. Information Processing Letters 68(5), 227–233 (1998)
6. Malone-Lee, J., Mao, W.: Two birds one stone: signcryption using RSA. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 211–226. Springer, Heidelberg (2003)
7. Malone-Lee, J.: Identity based signcryption. Cryptology ePrint Archive. Report 2002/098 (2002)
8. Libert, B., Quisquater, J.J.: A new identity based signcryption scheme from pairings. In: 2003 IEEE information theory workshop. Paris, France, pp. 155–158 (2003)
9. Chow, S.S.M., Yiu, S.M., Hui, L.C.K., Chow, K.P.: Efficient forward and provably secure ID-based signcryption scheme with public verifiability and public ciphertext authenticity. In: Lim, J.-I., Lee, D.-H. (eds.) ICISC 2003. LNCS, vol. 2971, pp. 352–369. Springer, Heidelberg (2004)
10. Boyen, X.: Multipurpose identity based signcryption: a swiss army knife for identity based cryptography. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 383–399. Springer, Heidelberg (2003)
11. Chen, L., Malone-Lee, J.: Improved identity-based signcryption. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 362–379. Springer, Heidelberg (2005)
12. Barreto, P.S.L.M., Libert, B., McCullagh, N., Quisquater, J.J.: Efficient and provably-secure identity based signatures and signcryption from bilinear maps. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 515–532. Springer, Heidelberg (2005)
13. Zheng, Y.: Signcryption and its applications in efficient public key solutions. In: Okamoto, E. (ed.) ISW 1997. LNCS, vol. 1396, pp. 291–312. Springer, Heidelberg (1998)

14. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
15. Kudo, M.: Secure electronic sealed-bid auction protocol with public key cryptography. IEICE Trans. Fundamentals E81-A(1), 20–26 (1998)
16. Koo, J.H., Kim, H.J., Jeong, I.R.: Jointly unsignryption signcryption schemes. In: Proceedings of WISA 2001, pp. 397–407 (2001)
17. Li, F., Gao, J., Hu, Y.: ID-based threshold unsignryption scheme from pairings. In: Feng, D., Lin, D., Yung, M. (eds.) CISC 2005. LNCS, vol. 3822, pp. 242–253. Springer, Heidelberg (2005)
18. Baek, J., Zheng, Y.: Identity-based threshold decryption. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 262–276. Springer, Heidelberg (2004)
19. Malone-Lee, J.: Identity based signcryption. Cryptology ePrint Archive, Report 2002/098 (2002). Available at: <http://eprint.iacr.org/2002/098>
20. Pointcheval, D., Stern, J.: Security arguments for digital signature and blind signature. Journal of Cryptology 13(3), 361–396 (2000)

Final Fantasy – Securing On-Line Gaming with Trusted Computing

Shane Balfe¹ and Anish Mohammed²

¹ Royal Holloway, University of London
Egham, Surrey, TW20 8XF, U.K.
`s.balfe@rhul.ac.uk`

² Capgemini UK PLC, Floor 1-5
76-78 Wardour Street, London, W1F 0UU, U.K.
`anish.mohammed@capgemini.co.uk`

Abstract. On-line gaming has seen something of a popular explosion in recent years, and is rapidly becoming the predominant focus of many gaming platforms. Unfortunately, honesty is not a virtue favoured by all players in these networks. This paper proposes a Trusted Computing based security framework for gaming consoles that will be resilient to platform modification based cheating mechanisms. In addition to this, we propose a Trusted Computing based auction mechanism that can be used for auctioning in-game items.

1 Introduction

The history of computer games can be traced back to 1952, when Douglas developed a graphical version of the game noughts and crosses (tic-tac-toe) in a University of Cambridge laboratory [1]. Since then computer games have evolved from a programmer-driven leisure activity into a multi-billion pound global phenomenon. However, with this has come increased disquiet (on the part of console manufacturers) over security. This is particularly true with reference to platform ‘moding’ which allow game consoles to run illegal (pirated) games and homebrew software by circumventing a console’s security system. Modding can be split into two categories, “soft-mods” and “mod-chips”. Soft-mods represent a class of attacks that bypass a console’s security through software only methods, typically through buffer overflows. By contrast, mod-chips achieve the same results by attaching a small chip to a console’s main circuit board. The use of platform modding potentially translates into lost revenue for manufacturers, who typically sell their consoles at discounted rates, which they later try to recoup through software sales.

In order to combat the threat posed by modding, many console manufacturers, in providing on-line services, attempt to detect compromised systems during network authentication. One such service is Sony’s Dynamic Network Authentication System (DNAS) [2], which uses a set of codes in a protected area of a game’s DVD, together with serial numbers from the console’s EEPROM, to authenticate a game. However, given the static nature of such protection mechanisms, tools have been developed that circumvent these checks by reporting known good

values in order to authenticate a modified platform. Traditionally, the console and game manufacturers' primary concern has been the prevention/detection of software piracy. However, as the on-line gaming market continues to grow, an additional concern has arisen, namely to stop players from cheating.

The concept of cheating, as we will see in Section 2, covers a multitude of undesirable behaviour. It addresses what we intuitively envisage cheating to be, namely one player gaining an unfair advantage over another player, but it is also beginning to incorporate an aspect of financial loss. This is not surprising when one considers the virtual economies that are being spawned by on-line games. There have been moves by console/game manufactures to capitalise on this trend, such as Sony's Station Exchange for the "EverQuest II" game which allows players to buy and sell the right of use for an in-game item. Indeed, the value of such items should not be underestimated; for example, in MindArk's "Project Entropia", a virtual space resort recently sold for £57,000 [3].

This paper proposes the use of Trusted Computing platforms as hosts for game consoles. We examine the infrastructural requirements necessary for TPM-enabled consoles and propose a Trusted Computing based payment scheme for digital goods. We show how the on-line capabilities of many consoles can be modelled using the functionality provided by the Trusted Computing Group (TCG) Trusted Network Connect (TNC) specifications [4]. Through a process of assessment, isolation and remediation we can either deny 'cheaters' access to the network or filter cheaters into their own private networks, preserving the quality of the experience for honest players. Cheating in our context chiefly relates to a class of vulnerabilities that can be broadly called "system design inadequacies" [5]. For the remainder of this paper, stopping cheating will refer to preventing a player from surreptitiously altering game state through the modification of client-side software. Finally, we highlight how Trusted Computing in conjunction with virtualisation can offer emulation of older consoles in order to allow games to be played on incompatible hardware. Indeed, this particular feature may become a unique selling point, as seen with the inclusion of a "virtual console" in Nintendo's Wii, allowing players access to some of Nintendo's back catalogue.

The applicability of our approach is directly dependent on the availability of the following components in a Trusted Gaming Platform (TGP), a Trusted Platform Module (TPM), processor/chipset extensions such as Intel's LaGrande [6] or AMD's AMD-V [7], and Operating System support such as Microsoft's Next Generation Secure Computing Base (NGSCB) Nexus Kernel [8] or the European Multilateral Secure Computing Base (EMSCB) with Perseus¹.

The remainder of this paper is laid out as follows. In Section 2 we examine a number of security requirements for computer games, as well as looking at a number of anti-cheating technologies. In Section 3 we provide a brief overview of Trusted Computing. In Section 4, we examine the infrastructure requirements for a Trusted Gaming Platform, and describe how such a platform would join a gaming network. In Section 5, we describe an English auction for the distribution of in-game items. We conclude in Section 6.

¹ www.perseus-os.org

2 A Thing Worth Having Is a Thing Worth Cheating for

A number of requirements need to be satisfied when providing an on-line gaming service; these are inter-related, and are as follows.

The prevention of piracy: The computer games industry is a multi-billion pound a year business, with a revenue model based on software sales. Consequently, it is often argued, illegal copying of games causes loss of revenue to the industry, thus providing less incentive for developers to pursue new titles [1]. Traditionally, piracy protection employed non-standard formatting techniques, such as using game cartridges as a mechanism for code distribution. However, new content distribution models, such as Valve Software's Steam or Microsoft's Xbox Live Marketplace, provide software only downloads, potentially making piracy prevention more difficult.

Preventing Cheating: The inclusion of network connectivity in many of the newer consoles has added a new dimension to a player's gaming experience. Unfortunately, it has also created a new problem: on-line cheating. Cheating in networked games comes in many forms but for this paper our primary concern is the unauthorised modification of game logic. These exploits are typically manifested in either direct alteration of game files, or through the surreptitious running of a program in parallel to an executing game that modifies a game's output before sending it to a server. Examples of such cheats include wallhacking (changing the properties of walls within a game, allowing players to see through them), maphacking (being able to see more of a map than a player should), nocliping (providing players with ethereal like qualities, allowing them to float through walls and ceilings), and aimbots (assisting a player in aiming at a target).

In combatting these exploits, a number of schemes to detect memory-resident cheating applications have been proposed, such as PunkBuster² (client-side cheat detection) and Valve Anti-Cheat (proprietary server-side cheat detection). Once cheating has been detected, a player will be either temporarily or permanently removed from the game. Both these approaches involve examining visible processes within a customer's platform and comparing them to a database of banned applications. Unfortunately, there are two problems with these approaches. Firstly, some cheat-enabling applications have been known to cloak their presence within a system, thus rendering them invisible to the examining application [9]. Secondly, the reporting of executing processes to third-party servers may be seen as a violation of user privacy. For a full treatment of cheating in on-line games we refer interested readers to [5].

3 Trusted Computing

This section highlights a number of important specifications that are germane to our discussion, namely the Trusted Platform Module (TPM) [10], Processor

² <http://www.punkbuster.com/>

Support [6], Operating System (OS) support [8,11] and the Trusted Network Connect (TNC) specifications [4]. Trusted Computing, as discussed here, relates directly to the type of system standardised by the Trusted Computing Group (TCG): a system which behaves in an expected manner for a particular purpose. For readers unfamiliar with Trusted Computing concepts, introductory material can be found in [12] and [13].

3.1 TPM Specifications

The TPM is the core component of TCG's definition of a trusted system. The TPM comes in the form of a microcontroller with Cryptographic Co-Processor (CCP) capabilities that resides on a platform's motherboard. TPM capabilities include: tick counters, monotonic counters, RSA key generation, SHA-1 and HMAC computation, as well as random number generation. Additionally, the TPM provides secure areas in which a platform can operate on sensitive data. The TPM is assumed capable of making (and storing) intrinsically reliable integrity measurements pertaining to a platform's current state. These measurements represent a snap-shot of the current configuration of a platform and are recorded internally to the TPM in special hardware registers called Platform Configuration Registers (PCRs). The TPM also has the ability to faithfully recount a platform's current operating state to third parties. The mechanism through which this is achieved is known as 'remote attestation', and involves signing a record of a platform's current state (as recorded in the PCRs) using the private part of a special non-migratable key called an Attestation Identity Key (AIK). In order for an AIK to be verified by an external party, it is necessary for the platform to obtain a credential for the public component of an AIK from a trusted third party called a Privacy CA.

3.2 Secure Boot with OS and Processor Support

Both Operating System and processor support are integral components of Trusted Computing. The process of booting a Trusted Computing platform begins with a TPM-supported secure initialisation facility which measures the operational state of an OS as a platform transitions from a pre-boot into a post-boot state. This process begins with a platform's Core Root of Trust for Measurement (CRTM). The CRTM is an immutable portion of the host platform's initialisation code, that executes upon a host platform reset. This code exists either as the BIOS or the BIOS boot block, and comprises the executable component of the Root of Trust for Measurement (RTM). This code subsequently measures the BIOS and stores a representation of its code in one (or more) of the TPM's PCRs. The BIOS in turn measures and stores the OS loader, which finally measures and stores the static OS, forming a transitive trust chain called the Static CRTM (S-CRTM). From this S-CRTM, a platform can launch a Dynamic CRTM (D-CRTM). The concept of a D-CRTM, as defined by Intel in the La Grande system architecture, refers to a protected isolation domain running on-top of a Measured Virtual Machine Monitor (MVMM). This isolated partition runs in parallel to the standard OS

partition without requiring a system reboot, and can be used to run arbitrary code free from observation from other partitions within a single platform.

Through remote attestation a verifier can compare the a platform's post-boot state (as recorded in a TPM's PCRs) against some agreed-upon "good" value. Provided they match then the OS can be seen to be functioning correctly. This correctly functioning OS can then provide a stable baseline from which the future execution of programs can be measured. As well as providing access to TPM functionality, a Trusted Computing aware OS will be capable of launching Virtual Machines (VM) in which applications can run. In this respect, processor and chipset extensions will provide the hardware support for the creation of these VMs, and act as a basis for enforcing application level sandboxing within main memory. In this regard, Microsoft's NGSCB (Next Generation Secure Computing Base) [8,11] forms an illustrative example of OS support, whilst Intel's Lagrande initiative provides an example of processor and chipset extension support [6].

3.3 TNC Specification

TNC [4] offers a way of verifying an endpoint's integrity to ensure that it complies with a particular predefined policy. A particular example of this would be ensuring that a certain software state exists on a platform prior to being granted network access, requiring that certain firmware or software patch updates be installed. This is achieved using a three-phase approach of assess, isolate and remediate, which we now briefly discuss.

The assess phase primarily involves an Access Requestor (AR) wishing to gain access to a restricted network. In this phase an Integrity Measurement Verifier (IMV) on a Policy Decision Point (PDP) examines the integrity metrics coming from an Integrity Measurement Collector (IMC) on the AR's platform and compares them to its network access policies. From this process of reconciliation the PDP informs a Policy Enforcement Point (PEP) of its decision regarding an AR's access request. The PEP is then responsible for enforcing the PDP's decision. As an extension to the assessment phase, in the event that the AR has been authenticated but failed the IMV's integrity-verification procedure, a process of isolation may be instigated whereby the PDP passes instructions to the PEP which are then passed to the AR directing it to an isolation network. The final phase, remediation, is where the AR on the isolation network obtains the requisite integrity-related updates that will allow it to satisfy the PDP's access policy.

4 Trusted Network Gaming

This section describes the steps necessary for a computer game to gain access to a secure gaming network. In order to establish that the requesting platform is free from modifications, particularly soft-mods, we need to establish that a gaming system has securely booted into a known good state.

4.1 Launching the Console — Secure Boot

When powering up a Trusted Gaming Platform, a TGS's CRTM in combination with its TPM (and its connections to a platform's motherboard) forms a platform's Trusted Building Block (TBB), and is responsible for loading a platform's static OS, as shown in Figure 4.2. To enable a secure boot facility, the PCRs which reflect the S-CRTM need to be compared against known good values for a correctly functioning S-CRTM. These good values will typically be measured during functional testing when the gaming platform is believed to be in a stable state, and will be inserted into non-volatile storage on a TPM. A console may then, depending on the policy in force, either allow a console's OS to load or suspend the loading of the console's OS if the measurements stored in the TPM fail to match the runtime measurements of the S-CRTM. From this S-CRTM a TGS can launch an MVMM, which will be capable of offering an isolation layer in which computer games may run, which in turn will be attestable to game servers. Much like the case for the C-RTM, in the event of an MVMM's runtime state diverging from what a game server deems to be an acceptable state, an MVMM may still be allowed to load. However, when joining the TGS network it would be placed in a 'cheaters' network until it could prove it no longer diverged from an acceptable state. In addition to compartment isolation, the MVMM may offer an emulation layer which would expose the expected console interfaces to executing games. Adopting this approach would potentially avoid expensive porting of back catalogues to new system architectures.

4.2 Enrolling with the Game Server

On the first occasion that a game tries to access the gaming network, the TGS will need to enrol with the game server. For an architecture based on Trusted Computing, this will mean enrolling with a game-provider specific Privacy CA (see Section 3.1) in order to obtain a certificate from a gaming server. This certificate will later be used to validate signed platform metrics demonstrating both possession, and correct usage of, a valid game.

This process involves the TGS generating a game key pair per game (S_G and P_G , for the private/public portions respectively) and having the public part incorporated into a game certificate, issued by the game-provider's Privacy CA. Generating a new game key within a platform involves the generation of an AIK. This is achieved by using the TPM Make Identity command [14, pp.147]. In addition to this, the game executable is loaded into memory and a representative hash of the game's state is recorded to one or more PCR registers in the TPM. The P_G public key, a signature of the game state reflected in the TPM's PCRs (as well as the S-CRTM and D-CRTM) and various platform credentials (that describe the binding of the TPM to the TGP) are sent to the game-provider specific Privacy CA. After authenticating a particular game by comparing the signature of game PCRs to a known good value and satisfying itself that the request is coming from a genuine platform, the game server's Privacy CA will issue a game certificate (AIK credential) to the game platform.

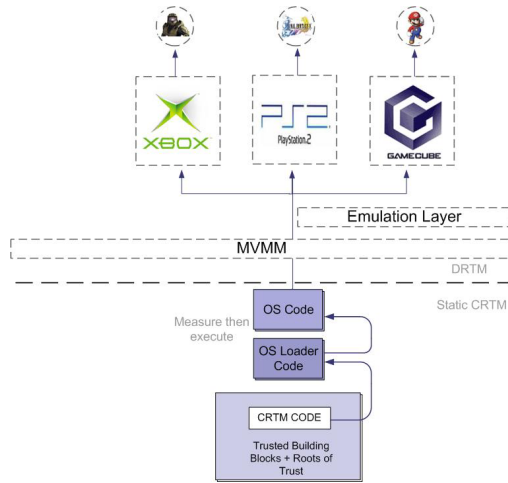


Fig. 1. An Architecture for Trusted Gaming

The process of certificate acquisition is achieved as follows. A `Tspi_TPM_CollateIdentityRequest` command [15, pp.111] is issued by a platform prior to the generation of a gaming (AIK) key pair; this command gathers all the information necessary for a gaming server (Privacy CA) to examine the requestor's platform. This information includes various credentials that vouch for the trustworthiness of both the TPM and the platform. Provided the evidence presented by a gamer's platform is validated by the Privacy CA, the Privacy CA will send the gaming certificate to the requesting platform. After receiving this certificate, the gaming platform runs the `TPM_ActivateIdentity` command [14, pp.151]. This allows a private key component to be subsequently used to generate signatures over platform integrity metrics, as reflected in PCRs. Additionally, the presence of this certificate indicates to a third party that a game has been activated within the system. A game server Privacy CA may also wish to modify the game certificate using certain X.509 v3 extensions. For example, it would be possible to add key and policy information to the certificate, such as setting a private key usage period under which the AIK signing key will operate. Through this, the server could enforce different payment models for on-line access to a game server's network.

4.3 Joining the Network

The introduction of Trusted Computing into the on-line gaming world allows game service providers to classify gaming platforms into two distinct categories: those which cheat (using soft-modding techniques) and those which do not. In this environment, a game's access to the network is controlled by its ability to demonstrate that it is free from soft-mods. Through a process of assessment and isolation, a game service provider can effectively filter platforms in which

cheating is detected into a special network. Instead of blacklisting platforms in which cheating has been detected, cheaters would end up in a ‘cheating network’ in which they would play against each other. In this scenario, cheaters gain no unfair advantage over other players, as every player in their network will themselves be cheating. This may encourage cheating players to become honest and disable platform modifications.

Assessment Phase. The assessment phase deals primarily with determining if a particular TGP (AR) should gain access to its game provider’s network. This is achieved through a process of checking console end-point integrity for compliance with predefined integrity policies. In this phase, the IMV on a game server (PDP) checks integrity metrics coming from the requesting platforms’s IMC against its network access policies. Here the TGP’s IMC may be a part of the game disk provided and signed by the content producer or a downloaded component from the game server’s network, and would monitor executing processes on a customer’s platform. The PDP informs the game server’s PEP (here the PDP and PEP could be the same game server or distinct architectural components) of its decision regarding an AR’s access request after comparing the TGP’s supplied IMC metrics against its security policy. In this setting the AR would need to authenticate themselves to the PEP using some form of authentication protocol, for example Radius with EAP. Using this protocol a game would communicate authentication information (a signature of a PDP supplied challenge using the certified private key S_G from the enrolment phase) in conjunction with its IMC-collated integrity metrics.

Isolation Phase. In the event that the AR has been authenticated but failed the game server’s IMV integrity-verification procedure (possibly as a result of the intrusion of some undesirable third party application, as evidenced in the IMC reported metrics), a process of isolation may be instigated whereby the PDP passes instructions to the PEP which are then passed to the AR directing it to an isolation network. The player can then be instructed in the removal of any detected modding.

Remediation Phase. The remediation phase represents a successful completion of PEP instructions by the AR’s platform, where the AR on the isolation network obtains the requisite integrity-related updates that will allow it to satisfy the PDP’s access policies. After this the player may gain access to the regular gaming network by rerunning the assessment phase.

5 A Trusted Computing English Auction

In economic theory, auctions are used when the value of an item varies sufficiently that it is difficult to establish a set price point. This is precisely the case with in-game items. There is an idiosyncratic element to gaming auctions, in which a bidder may take special advantage of the asset being auctioned, particularly if it complements other in-game items held by that player. Such auctions

are becoming extremely popular as evidenced by in-game auctions in World of Warcraft and Sony’s station-exchange for Everquest II. For the sale of items economic theory suggests a number of different auction strategies, such as: English, Dutch, first-price sealed bid, Vickery and double auctions [16]. Each of these auctions can be modelled in a Trusted Computing setting, but under the assumption of revenue equivalence theory and for brevity of exposition we concentrate on English auctions [17]. English auctions, also known as ascending auctions, are auctions in which an auctioneer starts with the lowest acceptable price and solicits successively higher bids until there is no one willing to bid more. For an overview of electronic auctions we refer readers to [16,18]. Our auction can be seen to take place in a number of phases: initialising an auction, bidding and notification of acceptance, and transfer of ownership. We assume our auction operates within a game using “virtual” currency. The transfer of virtual currency for real world remuneration is outside of the scope of this paper. Our Auction takes place as follows.

5.1 Initialising an Auction

Our auction scheme requires that players be registered with a game provider’s Auction Server (AS). This registration must occur after (or in tandem with) the console’s enrollment with the game server, as described in Section 4.2. During registration the console, on behalf of the gamer, creates a new auction key pair (S_A , P_A). The public component P_A , is then signed using the game server’s private key S_G . This process involves the console using the TPM_CreateWrapKey command to generate the auction key pair. The private key, S_A , is assigned to be non-migratable, and its usage is set to be contingent on a specific game state being present in a platform’s PCRs. The public auction key (P_A) is then input into the TPM_CertifyKey command, which uses S_G to create a digitally signed statement that “ P_A is held in a TPM-shielded location, bound to a particular game and will never be revealed outside of the TPM.” Certification of this P_A key is obtained through Subject Key Attestation Evidence (SKAE) CA [19] which, in this instance, will be the same entity as the game-provider’s Privacy CA.

An AS, like the player’s platform, should be TPM-enabled in order to provide assurance to bidding players that the server will behave fairly. Additionally, the AS should have its own private/public key pair, (S_{AS} , P_{AS}), used for signing auction broadcasts, and another key pair (S'_{AS} , P'_{AS}), used for maintaining secrecy of bids. We assume that, in bidding, the bidder has obtained an authenticated copy of the AS’s public key, P_{AS} , which may come embedded in the game disc. Our auction begins with the AS broadcasting the description of the item to be auctioned and the time at which auction closes. Here I-ID is the item being sold, || means concatenation, T is the time the auction is due to end, B-ID is a broadcast identifier (which allows the AS to distinguish multiple auctions), and $S_x(Y)$ is a signature generated by a TPM using key x over data Y .

AS $\rightarrow$ Network: I-ID || T || $S_{S_{AS}}(B-ID || I-ID || T)$

5.2 Bidding

The bidder submits a bid to the Auction Server. The bid contains the signature of the bidder created using S_A over: the identifier of the item, the public key certificate of the bidder (P_A), K , a structure specifying the properties of the bidder's private auction key expressed as a `TPM_CERTIFY_INFO` structure [20, pp.95], and B , the value of bid, all encrypted with the public key of the Auction Server, P_{AS} . Here $E_x(Y)$ means encrypt using the TPM data item Y using key x .

$$\text{TGP} \rightarrow \text{AS}: E_{P_{AS}}\{P_A \parallel K \parallel S_{S_A} (\text{I-ID} \parallel B)\}$$

The AS decrypts the received blob and checks to see if the bid is greater than the current highest bid for Item-ID. If so, the AS verifies the bidder's signature by examining the link between P_G and P_A from the `TPM_CERTIFY_INFO` structure (since S_G is used to sign P_G). If the AS successfully verifies the signature, it publicly distributes the new highest bid. This can be achieved by either broadcasting the result or making it available via some publicly readable bulletin board, as is done in eBay.

5.3 Notification of Acceptance and Transfer of Ownership

The AS waits for the designated time at which point the auction is declared closed and the highest bidder is notified of success. At this point, the successful bidder makes an appropriate payment to the AS. The payment could be completed by a variety of means, independent of the operation of the auction scheme. Once payment has been received, the AS transfers the item to the winner as follows:

$$\text{AS} \rightarrow \text{TGP}: E_{P_A}(\text{item})$$

Provided the game platform is in a predetermined state, the console's TPM decrypts the item and loads it into the player's game inventory. This scheme is designed on the assumption that the TPM is tamper-resistant. Both the game platform's keys (S_G and S_A) are non-migratable and should never exist in-the-clear outside of a console's TPM chip. The P_G certificate provides evidence as to the existence of an AIK within a TPM, and hence existence of identity associated with a gaming machine. Ordinarily, a single key could be used for both platform attestation and bidding in an auction; however, the TPM specifications dictate usage constraints for AIK signature keys. These keys are only used to sign platform metrics and certify non-migratable keys, thus requiring the generation of an additional key pair.

6 Conclusions and Future Work

In this paper we examined the infrastructural requirements for Trusted Gaming platforms. The architecture discussed here could be used as the underlying

security framework for any on-line gaming service, and is not just limited to custom gaming platforms. Any sufficiently well equipped TPM-enabled Personal Computer could replicate a console system though emulation. The idea of a Trusted Platform acting as host for game consoles is not an unrealistic one, as evidenced by the thriving console emulation community. Extensive emulation packages for the majority of console systems currently exist. Playstation 2 games can be played using PCSX2, GameCube games can be played using Dolphin, and preliminary results for Xbox games have been demonstrated using xeon and Cxbx. We note however, that newer systems, such as Xbox 360 and the PS3, are less amenable to such emulation techniques because of their highly customised hardware.

Future work will examine implementation strategies for various types of auction, and examine how bidder anonymity can be achieved using Trusted Computing's Direct Anonymous Attestation (DAA) protocols. We will also look at methods for enabling Trusted Computing enhanced payments for the transfer of value between virtual currencies and real world payments.

Acknowledgments. We would like to thank Chris Mitchell, Stéphane Lo Presti and the anonymous reviewers for their comments and suggestions during the preparation of this paper.

References

1. Hoglund, G., McGraw, G.: *Exploiting Online Games: How to Break Multi-user Computer Games*. Addison-Wesley, London (2007)
2. Sony Computer Entertainment Inc: DNAS (Dynamic Network Authentication System) (2003) <http://www.us.playstation.com/DNAS>
3. BBC News: Virtual property market booming (2005) <http://news.bbc.co.uk/1/hi/technology/4421496.stm>
4. Trusted Computing Group: TCG Trusted Network Connect TNC Architecture for Interoperability. 1.1 revision, 6 edn. (2006)
5. Yan, J., Randell, B.: A systematic classification of cheating in online games. In: *NetGames '05: Proceedings of 4th ACM SIGCOMM workshop on Network and system support for games*, pp. 1–9. ACM Press, New York (2005)
6. Intel Corporation: *LaGrande Technology Architectural Overview* (2003)
7. Strongin, G.: Trusted computing using AMD Pacifica and Presidio secure virtual machine technology. Information Security Technical Report 10, 120–132 (2005)
8. Abadi, M., Wobber, T.: A logical account of NGSCB. In: de Frutos-Escrig, D., Núñez, M. (eds.) *FORTE 2004*. LNCS, vol. 3235, pp. 1–12. Springer, Heidelberg (2004)
9. Lemos, R.: World of warcraft hackers using Sony BMG rootkit (2005) <http://www.securityfocus.com/news/10232>
10. Trusted Computing Group: *TPM Main: Part 1 Design Principles*. 1.2 revision, 93 edn. (2006)
11. Peinado, M., Chen, Y., England, P., Manferdelli, J.: NGSCB: A trusted open system. In: Bauknecht, K., Tjoa, A.M., Quirchmayr, G. (eds.) *E-Commerce and Web Technologies*. LNCS, vol. 2738, pp. 86–97. Springer, Heidelberg (2003)

12. Mitchell, C. (ed.): Trusted Computing. IEE Press, New York (2005)
13. Pearson, S. (ed.): Trusted Computing Platforms: TCGA Technology in Context. Prentice Hall PTR, Englewood Cliffs (2002)
14. Trusted Computing Group: TPM Main: Part 3 Commands. 1.2 revision, 93 edn. (2006)
15. Group, T.C.: TCG Software Stack Specification Version 1.2 Level 1 (2006)
16. Klemperer, P.: Auctions: Theory and Practice (The Toulouse Lectures in Economics). Princeton University Press, Princeton (2004)
17. Ivanova-Stenzel, R., Salmon, T.C.: Revenue equivalence revisited. In: Discussion Papers 175, SFB/TR 15 Governance and the Efficiency of Economic Systems. Free University of Berlin, Humboldt University of Berlin, University of Bonn, University (2006)
18. Omote, K.: A Study on Electronic Auctions. PhD thesis, Japan Advanced Institute of Science and Technology (2002)
19. Trusted Computing Group — TCG Infrastructure Workgroup: Subject Key Attestation Evidence Extension. V1.0 revision, 7 th edn. (2005)
20. Group, T.C.: TPM Main: Part 2 Structures of the TPM. 1.2 revision, 93 rd edn. (2006)

An Efficient and Secure Rights Sharing Method for DRM System Against Replay Attack*

Donghyun Choi¹, Yunho Lee¹, Hogab Kang², Seungjoo Kim^{1, **},
and Dongho Won¹

¹ Information Security Group, School of Information and Communication
Engineering, Sungkyunkwan University,
Suwon-si, Gyeonggi-do, 440-746, Korea
{dhchoi, younori, skim, dhwon}@security.re.kr

² DRM inside,
#403, Doosanweve BD, 98, Garakbon-dong, Songpa-gu, Seoul, 135-805, Korea
hgkang@drminside.com

Abstract. In the past years there has been an increasing interest in developing DRM (Digital Rights Management) systems. The purpose of DRM is to protect the copyrights of content providers and to enable only designated users to access digital contents. From the consumers' point of view, they have a tendency to go against complex and confusing limitations. Consumers want to enjoy contents without hassle and with as few limitations as possible. The concept of Authorized Domain (AD) was presented to remove such problems. However, the previous work on authorized domain has two problems. The first is that it requires a rather expensive revocation mechanism. The second is that the devices still can play contents which are previously obtained even though they are currently out of the authorized domain. On the contrary, our scheme prevents the content from being played by devices which are out of the domain for better security. Furthermore our scheme does not need to maintain a revocation list and can prevent replay attack.

1 Introduction

In the past years there has been an increasing interest in developing DRM systems[1,2,3]. Under the influence of development of computer technology, we could manufacture high quality digital content. In addition, increasing internet usage combined with expansion of communication technology has strengthened interrelation between computers. The development of technology generated more demands toward multimedia data such as digital music, digital movies and digital books. However, this has also caused illegal reproduction of original digital content because such content can be easily linked to the internet. This is threatening the digital content market. In that point, DRM helps to settle this

* This work was supported by the University IT Research Center Project funded by the Korea Ministry of Information and Communication.

** Corresponding author.

problem. That's why DRM is essential in the digital market field to protect copyrights[4,5,13].

From the consumers' point of view, they tend to go against complex and confusing limitations. They want to enjoy content without hassle and with as few limitations as possible. Moreover, consumers' rights of use of the content obtained legally were frequently harmed by arbitrary limitations. This violates consumers' legal rights of "fair use." Consumers want to build a network of their devices and have easy access to any type of content at home. Especially, if a client purchases digital content with a paid guarantee, he/she wants to play the content freely on any of his multiple devices such as PC, PDA, and MP3 players. The solution of this problem is to organize domain for DRM where legally acquired digital content can be freely played by any device in that domain.

The concept of Authorized Domain was presented to remove such problem[6,7]. The consumer organizes the devices in a group, which is named an (authorized) domain, and the access rights to the content are given to the group instead of each device. However, AD systems have a rather expensive revocation mechanism. Furthermore, the device can still play its content which are previously obtained even though it is currently out of the authorized domain.

In this paper, we present rights sharing method for DRM system against replay attack. The key idea of our scheme is to encipher the license two times using commutative encryption. Furthermore, our scheme uses time-stamp and digital signature to prevent replay attack. Our device not only can play content freely on any other devices in a domain but it also blocks playing content in the other domains. This method guarantees consumers' right and content providers' right. In addition, when new devices join the domain and existing devices withdraw the domain, the proposed scheme does not need additional process. Our scheme does not need to maintain a revocation list.

The rest of the paper is organized as follows: in Section 2, we review related works, and describes about problems of other DRM system. In Section 3, we describe our rights sharing method, and compare with other DRM systems. In Section 4, we describes about security analysis. Finally, in Section 5 we present our conclusions.

2 Related Works

2.1 DRM Technology Outline

DRM is an umbrella term that refers to set of several technologies used by publishers or copyright owners to control access, usage of digital data, hardware and restrictions associated with a specific instance of a digital work or device[14].

Digital content is utilized through information network system and digital media. DRM technology must protect digital content from unlawful reproduction and usage perfectly. When digital content go out of places where are designed to be, DRM system must offer traitor tracing so that can cope with issues related to illegal distributions.

2.2 Existing DRM Systems

Microsoft DRM. Microsoft's Windows Media Right Management (WMRM) allows protection of audio and video. The content can be played on a Windows PC or portable device.

The following is explanation about service that WMRM offers. The content service can be established using content hosting, probably in combination with content packaging when content needs to be protected on the fly. The license service functionality is part of the license clearinghouse. The tracking service can be implemented using part of the license clearinghouse functionality. A payment service is not supported, but existing payment solutions can be integrated when a DRM system is developed. An import service can be implemented using content packaging. No payment service and access service are supported in WMRM, but existing payment or access control solutions can be integrated. No information has been found about support for an identification service. The general process of WMRM is described bellows (see Fig.1)[5,15].

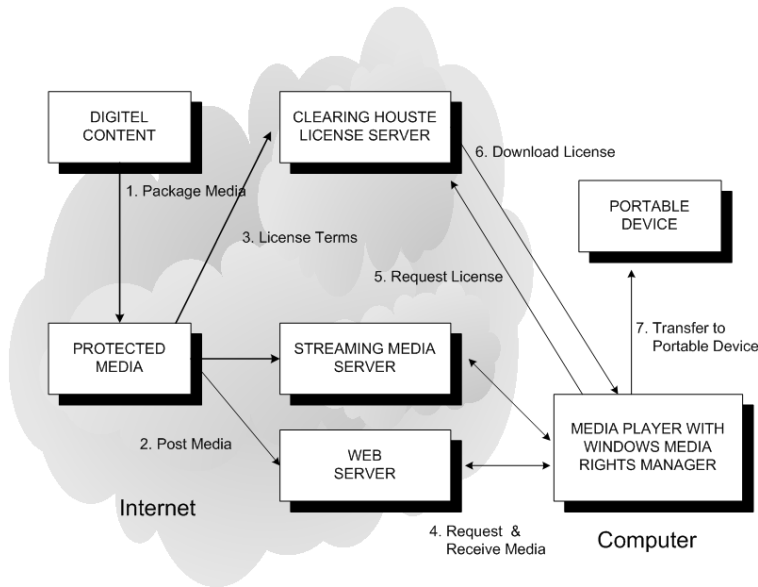


Fig. 1. WMRM work flow

1. Packaging: Packaging is a process of encryption which converts a media file to protected content. Usually, DRM system uses a symmetry encryption. The encrypted media file has a metadata in header file. The metadata includes a URL which is License Server address. A header file is not encrypted by DRM system.

2. Distribution: Distribution plays a role to distribute protected media file to consumers. Protected content is distributed according to business model through web server, streaming server, CD or DVD.
3. Establishing a License Server: License Server sends a license to user who purchases content legally. License transaction starts with authentication process. License includes a content encryption key and content uses rules.
4. License acquisition: The right user requests and obtains license from the license server. Microsoft's DRM (WORM) have four transmission methods (non-silent, silent, non predelivered and pre-delivered). When user acquire license, the silent method does not require any action to users but the remaining three methods require several actions to user.
5. Playing a media file: WORM guarantees that protected content should be used according to use rules in license. License can not be transmitted from one place to different places, it is strictly limited to the window media player

Light Weight DRM. The usual DRM schemes are strong in the sense that they enforce the usage rules at the consumer side very strictly. To overcome this problem, they propose Light Weight DRM (LWDRM), which allows consumers to do everything they want with the content they bought, except for large scale distribution. This form of use is called fair-use[5].

LWDRM uses two file formats: the Local Media File (LMF) and the Signed Media File (SMF) format. An LMF file is bound to a single local device by a hardware-driven key but can be converted into the SMF-format, which can be played on whatever device that supports LWDRM. LWDRM is not compulsive as to content reproduction. However, LWDRM uses digital signature when reproducing content, so content's owner can track a copied content[5,13].

2.3 Domain DRM System

The Authorized Domain based on DRM technology enables all devices in the same domain to use and share the DRM content freely. A domain can be a home network, personalized network, or any networks which have several rendering devices such as PC, PDA, Video Player, DVD Player, etc. The concept of AD has originated to protect content in the home environment. Former mechanisms of protecting content hardly addressed issues such as consumers' convenience and "fair use." It could support only limited number of business models. The idea was to allow content to be shared among devices owned by the same household without restrictions. Afterward, the Digital Video Broadcasting (DVB) standardization body later called this the AD concept[10].

The xCP. The xCP[11], proposed by IBM, is DRM technology based on the AD. This employs the broadcast encryption for secure content distribution. This is the only DRM architecture solely based on symmetric key cryptographic algorithm which has an advantage from an economical point of view. Due to the use of broadcast encryption, xCP needs expensive costs to revoke the members in a domain.

The SmartRight. The SmartRight system[12] suggested by Thompson Electronic relies on smart cards modules in CE devices. Once a device joined a domain, it shares the same symmetric domain key which is used to encrypt content. In this approach, it is required that the existing devices' domain key should be revoked and reestablished.

A DRM Security Architecture for Home Networks. [9] describes a security architecture which allows a home network to consist of consumer electronic devices. Their idea enables devices to establish dynamic groups called as Authorized Domains where legally acquired copyrighted content can be transferred seamlessly from one device to another. The key to their design is hybrid compliance checking and group establishment protocol. This is based on pre-distributed symmetric keys, with minimal reliance on public key cryptographic operations. Their architecture allows a key to be revoked and updated efficiently. However, their architecture can be misused by users because creating a revocation list is made by the users. Moreover, AD system can not receive new content once its device escapes from its original domain but the concerning issue is that playing content out of the domain of AD system is possible.

3 Proposed Right Sharing Method

In this section, we describe the license sharing scheme for domain DRM system. Home network is a good example of a domain. So, we will explain the scheme under home network environment. However, our scheme is not restricted to home network environment.

3.1 Notation

We now introduce some basic notations used throughout the paper.

1. $CE_K(\cdot)/CE_K^{-1}(\cdot)$: Commutative encryption/decryption with key K
2. $E_{PUB\_A}(\cdot)$: RSA encryption with A's public key
3. $D_{PRI\_A}(\cdot)$: RSA decryption with A's private key
4. $DS_A(\cdot)$: Digital signature with A's private key
5. $H(\cdot)$: A hash function
6. K_C : Content encryption key
7. DC : DRM client
8. DM : Domain manager

3.2 Domain Creation

Step 1: Domain manager creation. Creating a new domain requires one domain manager. When creating a new domain, the domain manager generates a domain id.

Step 2: Device registration. When a device joins the domain, it needs to be registered to the domain manager. The device sends its public key and device id to the domain manager. After exchanging their certificates, the domain manager and device can authenticate each other.

3.3 Domain Registration

After completing domain creation, domain manager needs to be registered to license server. The registration phase consists of two steps: domain authentication and key distribution.

Step 1: Domain authentication. Domain manager and license server exchange their certificates and authenticate each other. After completing authentication, domain manager transmits domain information (domain id, device list and devices' public keys) to the license server.

Step 2: Key distribution. License server assigns each secret key to the domain manager and clients in the domain (see Fig.2).

$$LicenseServer \xrightarrow{E_{PUB\_DM}(K_{DM})} DM \quad (1)$$

$$LicenseServer \xrightarrow{E_{PUB\_DC_i}(K_{DC_i})} DC_i \quad (2)$$

3.4 Content Usage

When a device plays content, the device must pass through the following steps.

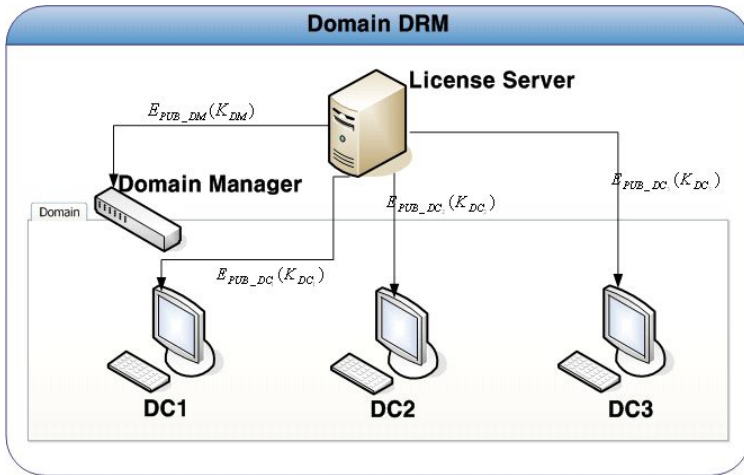


Fig. 2. Key distribution by license server

Step 1: License issuing. The license server received content encryption keys from the packaging server beforehand. If DC_2 buys content, license server transmits content encryption key K_C by encryption it twice using a commutative encryption. An encryption system $CE_K(\cdot)$ is called commutative if $CE_{K_1}(CE_{K_2}(M)) = CE_{K_2}(CE_{K_1}(M))$ holds.

$$LicenseServer : DL_2 = CE_{K_{DM}}(CE_{K_{DC_2}}(K_C)) \quad (3)$$

License server sends DC_2 to the buyer of contents.

$$LicenseServer \xrightarrow{DL_2} DC_2 \quad (4)$$

Step 2: License decryption by domain manager. After receiving DL_2 from license server, DC_2 sends it to the domain manager. Domain manager decrypts the DL_2 using the K_{DM} and then concatenates time-stamp T_n , the current time information, yielding CL_2 (see Fig.3). Domain manager applies a hash function to the CL_2 , and then digitally sign the resulting hash value (see equation (7)).

$$DC_2 \xrightarrow{DL_2} DM \quad (5)$$

$$DM : CL_2 = CE_{K_{DM}}^{-1}(CE_{K_{DM}}(CE_{K_{DC_2}}(K_C))) || T_n \quad (6)$$

$$DM : S_2 = DS_{DM}(H(CL_2)) \quad (7)$$

The domain manager sends CL_2 and S_2 to DC_2 . If the calculated hash value match the result of the decrypted signature and the time difference of T_n and DC_2 is within threshold value, DC_2 recognizes validation of received information.

DC_2 remove T_n from CL_2 and then decrypts TL_2 this using its secret key K_{DC_2} . After decrypting TL_2 , DC_2 gets the content encryption key K_C and can play the content.

$$DM \xrightarrow{DL_2, S_2} DC_2 \quad (8)$$

$$DC_2 : TL_2 = CE_{K_{DC_2}}(K_C) || T_n - T_n \quad (9)$$

$$DC - 2 : K_C = CE_{K_{DC_2}}^{-1}(CE_{K_{DC_2}}(K_C)) \quad (10)$$

3.5 Sharing License

To share a license in the same domain, DC_2 decrypts the license by K_{DC_2} , and then sends the decrypted value SL to DC_3 .

$$DC_2 : SL = CE_{K_{DC_2}}^{-1}(CE_{K_{DM}}(CE_{K_{DC_2}}(K_C))) \quad (11)$$

$$DC_2 \xrightarrow{SL} DC_3 \quad (12)$$

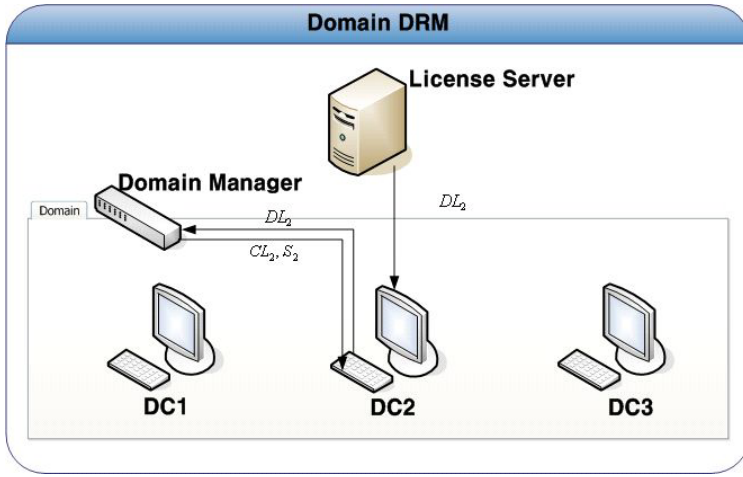


Fig. 3. Content usage

Upon receiving SL from DC_2 , DC_3 encrypts it with its own secret key K_{DC_3} and then sends it to the domain manager.

$$DC_3 : DL_3 = CE_{K_{DC_3}}(CE_{K_{DM}}(K_C)) \quad (13)$$

$$DC_3 \xrightarrow{DL_3} DM \quad (14)$$

Domain manager decrypts the DL_3 using the K_{DM} and then concatenates time-stamp T_n , the current time information, yielding CL_3 . Domain manager applies a hash function to the CL_3 , and then digitally sign the resulting hash value (see equation (16)).

$$DM : CL_3 = CE_{K_{DM}}^{-1}(CE_{K_{DM}}(CE_{K_{DC_3}}(K_C))) || T_n \quad (15)$$

$$DM : S_3 = DS_{DM}(H(CL_3)) \quad (16)$$

The domain manager sends CL_3 and S_3 to DC_3 . If the calculated hash value matches the result of the decrypted signature and time difference of T_n and DC_3 is within threshold value, DC_3 recognizes validation of received information.

DC_3 remove T_n from CL_3 and then decrypts TL_3 using its secret key K_{DC_3} obtained from initial registration process. After decrypting TL_3 , DC_3 acquires K_C and can play the content.

$$DM \xrightarrow{DL_3, S_3} DC_3 \quad (17)$$

$$DC_3 : TL_3 = CE_{K_{DC_3}}(K_C) || T_n - T_n \quad (18)$$

$$DC_3 : K_C = CE_{K_{DC_3}}^{-1}(CE_{K_{DC_3}}(K_C)) \quad (19)$$

In the case of DC_1 , it could share the license using the same process, and can acquire the packaged content from other devices using super distribution.

3.6 Comparison with Other DRM Systems

As you can see in table 1, while providing content sharing facility in a domain as other systems, the propose system preserves better security by prevention of playing content outside domain. Furthermore, our system does not need a revocation list, thus it does not require any resource about the revocation and can prevent replay attack.

Table 1. Functionality comparison result between the proposed system and the previous systems

	xCP	DRM security architecture for home networks	Proposed scheme
Sharing a content in a domain	○	○	○
Maintaining of revocation list	○	○	×
Content protection outside domain	×	×	○

4 Security Analysis

4.1 Content Encryption Key

DRM modules store a content encryption key in memory which is protected from user and then if rendering is ended, the modules erase the content encryption key immediately. Thus, users can not access to the content encryption key.

4.2 Black Box

DRM modules can be regarded as a black box which does not leak secret information to anyone. The steps of content rendering and license sharing are processed in black box. Therefore, users can not inflict any modification or obtain any information.

4.3 Device Join and Withdraw

When new devices join the domain, the proposed scheme does not need additional process because if the device receives packaged contents and *SL* from other device, it can render those contents. When existing devices withdraw the domain, the proposed scheme does not need additional process (domain key revocation mechanism or withdraw process) too. Because the device outside the domain can not obtains decrypted license from *DM*. Therefore, the proposed scheme can be much more simple process than the existing schemes.

4.4 Security Against Relay Attack

Our method uses time-stamp and digital signature. Thus, our method is secure against replay attack. Even if an attacker acquires license information that is

transmitted from domain manager to DC, replay attack is impossible because DRM modules check validation of the time-stamp and digital signature.

5 Conclusions

From the consumers' point of view, DRM systems have complex and confusing limitations. So, consumers' rights of using legally obtained contents are frequently harmed by arbitrary limitations. The concept of the AD was presented to remove such problem. But AD systems have expensive revocation mechanism and the device can still play its contents which were previously obtained even though it is currently out of the authorized domain.

Our scheme not only can play the contents freely on any other devices in a domain but also can prevent devices from playing content outside the authorized domain. Furthermore our scheme does not require a revocation mechanism and can prevent replay attack.

References

1. Ripley, M., Traw, C.B.S., Balogh, S., Reed, M.: Content Protection in the Digital Home. Intel Technology journal 49–56 (2002)
2. Eskicioglu, A.M., Delp, E.J.: An overview of multimedia content protection in consumer electronic devices. Signal Processing: Image Communication 681–699 (2001)
3. Eskicioglu, A.M., Town, J., Delp, E.J.: Security of Digital Entertainment Content from Creation to Consumption. Signal Processing: Image Communication 237–262 (2003)
4. Liu, Q., Safavi-Naini, R., Sheppard, N.P.: Digital rights management for content distribution. In: proceedings of the Australasian information security workshop conference on AISW frontiers, pp. 49–58 (2003)
5. Michiels, S., Verslype, K., Joosen, W., Decker, B.: Towards a Software Architecture for DRM. In: Proceedings of the Fifth ACM Workshop on Digital Rights Management, pp. 65–74 (2005)
6. van den Heuvel, S., Jonker, W., Kamperman, F., Lenoir, P.: Secure Content Management in Authorized Domains. In: Proc. IBC, pp. 467–474 (2002)
7. Sovio, S., Asokan, N., Nyberg, K.: Defining Authorization Domains Using Cirtual Devices. In: SAINT Workshops, pp. 331–336 (2003)
8. Tuecke, S., Welch, V., Engert, D., Pearlman, L., Thompson, M.: Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile. RFC 3820 (2004)
9. Popescu, Bogdan, C., Kamperman, Frank, L.A.J., Crispo, Bruno, Tanenbaum, Andrew, S.: A DRM security architecture for home networks. In: Proceedings of the 4th ACM workshop on Digital rights management, pp. 1–10 (2004)
10. DVB: Call for proposals for content protection & copy management technologies. DVB CPT REV 1.2 (2001)
11. IBM Research Division Almadern Research Center: eXtensible Content Protection (2003)

12. THOMSON: Smartright technical white paper (2003) Available:
http://www.smartright.org/images/SMR/content/SmartRight_tech_whitepaper_jan28.pdf
13. Fraunhofer Institute, Light Weight DRM (LWDRM), <http://www.lwdrm.com>
14. Wikipedia, <http://en.wikipedia.org>
15. Microsoft, <http://www.Microsoft.com>

Establishing Trust Between Mail Servers to Improve Spam Filtering

Jimmy McGibney and Dmitri Botvich

Telecommunications Software & Systems Group, Waterford Institute of Technology,
Waterford, Ireland
{jmcgibney, dbotvich}@tssg.org

Abstract. This paper proposes a new way to improve spam filtering based on the establishment and maintenance of trust between mail domains. An architecture is presented where each mail domain has an associated trust manager that dynamically records trust measures pertaining to other domains. Trust by one mail domain in another is influenced by direct experience as well as recommendations issued by collaborators. Each trust manager interacts with local spam filtering and peer trust managers to continuously update trust. These trust measures are used to tune filter sensitivity. A simulation set-up is described with multiple nodes that send and receive mail, some of which is spam. Rogue mail servers that produce spam are also introduced. Results of these simulations demonstrate the potential of trust based spam filtering, and are assessed in terms of improvements in rates of false positives and false negatives.

1 Introduction

Unsolicited bulk e-mail, or spam, is probably the greatest single nuisance for users of the Internet. Despite significant anti-spam efforts and the development of powerful spam filtering technologies, the incidence of spam remains stubbornly high. Estimates of the incidence of spam as a proportion of total email traffic vary widely, with the highest estimates close to 90% and the lowest still above 50%.

The main anti-spam techniques in practical use are based on message content, domain name system (DNS) blocklists and collaborative filtering databases. *SpamAssassin* [1], for example, processes each incoming mail and assigns it a score based on a combination of values attributed to possible spam indicators. The higher the score, the more likely it is that the mail is spam. A score threshold is then used to filter mail. DNS blocklisting is a complementary approach where mail is simply filtered based on where it comes from.

Countering spam is difficult though. Spammers are quite resourceful and adapt to filtering advances. Anti-spam techniques also evolve and improve to meet these new challenges, but there is usually some delay in propagating updates. With content filters, it is difficult to avoid having false positives and false negatives. A false positive occurs when a genuine mail message scores above the threshold and is flagged as spam. A false negative occurs when spam scores below the threshold and is accepted. There are also difficulties with blocklists, such as when well-intentioned mail servers

are attacked and exploited, or when they fall somewhere in between – for example, where servers are well managed but client machines are not patched frequently and are at risk of hijack by spammer rootkits.

This paper proposes a new approach to establishing and maintaining trust between mail domains and its application to improve spam filtering. In this approach, mail domains dynamically record trust scores for other mail domains; trust of one in another is influenced by direct experience of the domain (i.e. based on mail received from its server) as well as recommendations issued by collaborating domains. As well as modelling trust interactions between mail domains, we explore how mail filtering can use these trust values with existing mail filtering techniques. We also consider the case of rogue mail domains that issue false recommendations. We also report on experimental simulations that measure the effectiveness of our approach by examining to what extent we have achieved a reduction in false positives and false negatives.

Note that we focus on mail transfer agents (MTAs) rather than user clients in this paper. We interchangeably use the terms *mail server*, *mail domain* and, simply, *node* in place of MTA throughout this paper.

The remainder of this paper is organised as follows. The next section summarises related work and establishes the novelty of the work reported upon in this paper. In section 3, we summarise the principles of trust management that relate to this work, identify requirements for a trust-based anti-spam system and describe the architecture of our system. Section 4 discusses simulations that attempt to assess the system's effectiveness and section 5 concludes the paper.

2 Related Work

Several innovative new anti-spam techniques have been proposed, though without widespread practical application as yet. An example is the use of micro-payments for sending mail. The idea is that cost is negligible for normal users but punitive for bulk mail [2]; in one variation, the cost is more significant but is refundable if not spam [3]. Another idea is to require the sending client to solve a computational challenge for each message, greatly slowing bulk mail generation. Additional tricks, such as obfuscation of published email addresses and the use of *human interactive proofs* [4] in email account creation systems, try to frustrate spammers by making it harder to automate their activities.

The remainder of this section discusses other work that specifically uses collaborative techniques to fight spam.

Some widely-implemented spam filters use *centralised* trust information. For example, *SpamAssassin* [1] has a facility to make use of, in addition to other measures, collaborative filtering databases where trust information is shared and used to help to detect spam. Our approach differs from this, in that trust information is in our case generally managed independently by each mail domain and shared as desired. If required though, a centralised database may be modelled as a node in our network, and each mail domain may assign a trust value to it if it wishes.

Kong et al. [5] present a collaborative anti-spam technique that is different from ours in that the system focuses on end user email addresses. When a user flags a mail

as spam, this information is made available to other users' spam filters, which is useful as the same spam messages are usually sent to a large number of users. Golbeck and Hendler [6] present a technique, inspired by social networks, that allows end users to share ratings information. This technique requires direct user interaction. Neustaedter et al. [7] also use social networks, but to assist more general email management, not limited to spam filtering. Han et al. [8] also focus on users that generate spam, but more specifically *blog comment spam*.

Foukia et al. [9] consider how to incentivise mail servers to restrict output of spam. Their approach is agent based – each participating mail server has an associated Federated Security Context Agent that contributes to, and draws on, an aggregated community view. They also use quotas to control the volume of mail output by a server in an attempt to prevent temporary traffic bursts that are typical of spammer activity.

Though the idea of auto-tuning spam filters is not entirely new, our method of auto-tuning is different from other approaches, for example from Androutsopoulos et al. [10] who use game theory to model interaction between spammers and email users.

3 Trust-Based Approach to Spam Protection

3.1 Modelling Trust

A well-known definition of trust is “*a particular level of the subjective probability with which an agent will perform a particular action*” (Gambetta, [11]). Trust is primarily a social concept and, by this definition, is personalised by the subject.

Almost any transaction between entities requires the establishment of trust between them. The decentralised nature of many Internet services means that a model of trust is necessary for effective operations. The scope for hierarchical top-down solutions is limited due to the lack of centralised control, the desire for privacy or anonymity, and the increased use of services in a one-off ad hoc fashion. We can identify some specific requirements for a trust-based anti-spam system:

- *Compatibility with existing infrastructure.* The existing email system should not require changing.
- *The meaning of trust.* Trust is defined as being between two nodes, in this case mail servers – i.e. node i has a certain level of trust in node j . Each node has access to a measure of level of trust in each other node. Each node should be capable of managing its own trust level independently.
- *Trust updates based on experience.* It is possible for trust to be built up by experience. Although there may initially be little or no trust between node i and node j , it must be possible to establish trust based on interactions between them.
- *Trust updates based on recommendations.* Node i 's level of trust in node k may be influenced by node j 's level of trust in node k (communicated by node j to node i).
- *Robustness against attack, including collaboration between spammers.* Spammers tend to adapt to new anti-spam systems. For a new trust-based approach to be effective, it should be difficult for spammers to corrupt it. This could include the possibility of spammers actively participating in the trust system, possibly in concert, issuing false recommendations.

- *Stability*. The system should be stable – i.e. consistent treatment of mail, few oscillations in state, and rapid convergence to new state on changes. There should be a way to allow a poorly behaved domain to regain trust following recovery from a hijack or change in its administration policy.

3.2 Trust Overlay Architecture

We propose the overlay of a distributed trust management infrastructure on the Simple Mail Transfer Protocol (SMTP) mail infrastructure, with a view to using trust information at each domain to assist with spam filtering. Note that we do not propose any changes or extensions to SMTP or other mail protocols – rather, mail domains exchange mail as normal, but this is overlaid with a *trust management layer*. For each mail domain, there is a logical *trust manager* operating at this layer.

This trust management overlay works as follows. Each mail domain with associated spam detection, operating at the mail transport layer, records mail statistics including incidences of spam and reports this direct experience to its associated trust manager. The trust manager uses this data to form measures of trust about the sources of mail reported upon. This local trust information is then shared with other peer trust managers in the form of recommendations. Each trust manager independently decides how to handle these recommendations, using a trust transitivity algorithm to assist in differentiating between recommendations from well-intentioned nodes and those from those that are unreliable or deliberately false. This trust information maintained by the trust manager is then fed back to the mail domain at the mail transport layer to allow it to re-tune its spam filters (typically by raising or lowering thresholds based on trust) to be more effective.

The mechanics of these interactions requires a trust management overlay protocol. The mechanics of such a trust management overlay protocol, and its associated architecture, have already been described by these authors [12]. There are three major types of communications, as follows:

1. *Experience report: Mail host $\rightarrow$ Trust manager*. The mail host has an associated spam filter. All mail is processed by this spam filter and flagged as spam or else accepted. This experience information is made available to the trust manager.
2. *Trust recommendation: Trust manager $\leftrightarrow$ Trust manager*. Nodes' trust managers collaborate to share trust information with one another. Trust information from a third party node may be based on its experience of mail received from the node in question and/or reputation information that it has gleaned from other nodes. This relates to trust transitivity.
3. *Policy update: Trust manager $\rightarrow$ Mail host*. The third part of this collaboration architecture is responsible for closing the loop. Direct experience is recorded by nodes and shared between them. The result of this experience and collaboration is then used to inform the mail host to allow it to operate more effectively.

Initialisation. Note that choosing an initial value of trust to assign to such new arrivals is a non-trivial task. The first option is to initialise the trust level at zero so that nodes have no trust initially and must earn it, thus removing the threat of the so-called *Sybil attack* [13]. The second option is to assume some *default trust* exists and

initialise the trust level at some value greater than zero for previously unknown nodes. The great benefit of Internet mail is in being able to advertise a mail address and receive mail from anyone, even if this person is previously unknown, so the second option is perhaps the better, though we will use simulations and experience to best determine this. In our experiments, we choose a modest initial value of 0.2.

Distribution of Trust Information. It is important, for reasons of scalability and reliability of reputation information, to consider how often and to whom trust advertisements are provided.

- *When to issue trust advertisements?* If the trust level in a node is significantly increased or (more likely, due to sudden appearance of spam) decreased, then this should be advertised. Trust advertisements may also be issued periodically.
- *To whom?* Trust advertisements are restricted to nodes that are defined as within the neighbourhood of the issuer. A node may define its neighbourhood for trust advertisements as it wishes. The set of neighbours could contain, for example, most frequent contacts, most trusted contacts, most reliable recommenders, or nodes located nearby.

3.3 Using Trust Scores to Filter Email

Assume that a spam filter applies some test to incoming email. In each case, a decision is made whether to accept the mail. A negative result (in the spam test) means that the mail is accepted and a positive result means that the mail is marked as spam.

Most spam filters combine a variety of measures into a suspicion score and compare this with a *pre-defined* threshold. This threshold is a fixed value, which may be tuned manually. In our system, we attempt to improve spam filtering by allowing the threshold to vary. The threshold level depends on the trustworthiness of the node that sent the message. So, we use the sender's trust score (as perceived by the receiver) to define the threshold – i.e. the more trusted a node is, the higher we set the threshold for marking a received mail message as spam. There are several ways to cause this threshold to vary. In our initial experiments, the threshold for mail received from a domain is simply a linear function of the trust score of that domain (as the mean of the trust score range is 0.5 and the default threshold for *SpamAssassin* is 5, we set the threshold to be simply ten times the trust score).

The motivation for this is that mail received from nodes that are known to be well managed and are unlikely to produce spam has an increased likelihood of passing through the filter, reducing the incidence of false positives. There is also scope for reduced processing load if less stringent spam filtering is required in these situations.

In practice, the dynamics of trust applied to spam filtering allows an organisational mail domain to process email for spam in a way that depends on its trust in the sending node. In some cases, this trust level will be somewhere in the middle, between trusted and untrusted. Table 1 illustrates the possible effects of maintaining trust scores for a variety of mail domains, and a desired implicit classification.

Table 1. Possible effects of recording trust scores for a variety of mail domains

Trust score range	Category of mail domain	Spam filter threshold (Trust score * 10)	Effect
1.0	Internal host	10	All mail accepted
0.8 – 1.0	Business partner	8 – 10	Most mail accepted
0.7 – 0.8	University	7 – 8	Mail checked for spam but mostly ok
0.5 – 0.7	Popular ISP	5 – 7	Mail checked for spam
0.3 – 0.5	Public webmail service	3 – 5	Mail checked thoroughly for spam
0.1 – 0.3	Lazy configuration	1 – 3	Spam quite likely
0 – 0.1	Open mail relay; known spam source	< 1	Most mail flagged as spam

4 Simulations and Results

This section reports on initial simulations that aim to evaluate whether this new proposed approach is likely to be worthwhile.

In summary, the objectives of these experiments were to see if the rates of false positives and/or false negatives could be reduced, as well as to observe the overall stability of the spam filtering system as trust levels converge to steady-state levels.

4.1 Generation of Test Emails

Email traffic is inhomogeneous in practice. Some mail domains are much more active than others. Mail domains tend to communicate mostly with selected mail domains for business, social or geographical reasons. Spammers produce email in vast quantities compared to regular email users.

In our simulations, each node has a *neighbourhood* defined. This is a set of nodes that are somehow close to the node in question, with the expectation of above average frequency of communication.

Our generated email has the following statistical properties. For some experiments, every node is a neighbour of every other node. For other experiments, we use a sparse neighbourhood definition. This second, more sparse, neighbourhood of each node is built randomly as follows:

1. Initially, the neighbourhood of each node is the empty set.
2. Choose two nodes at random. Add one to the neighbourhood of the other.
3. Repeat (2) until all nodes are connected via neighbour relations (if the set of nodes is modelled as a graph with neighbourhood relationships as edges, repeat until the graph is connected).

From each node, 50% of emails are sent to a randomly chosen neighbour. The other 50% are sent to any randomly chosen node.

Each email contains a value S' that models aggregated indicators of spam, in the style of *SpamAssassin*. This value is used by the receiving node to test for spam. S' has a Gaussian (normal) probability density function with mean μ and standard

deviation σ . Mail that is actually spam tends to have a relatively high value of μ . Normal mail tends to have a lower value of μ . The standard deviation determines the tendency for the filtering system to be prone to false positives and false negatives.

For the purpose of our experiments, whether or not an email is actually spam is indicated by a binary value communicated separately to the receiver. This is not used in spam detection, but is used afterwards in evaluating how well the spam filter worked.

4.2 Trust Establishment and Maintenance in the Experiments

We denote the *local trust* that node i has in node j as $T_{i,j}$, $0 \leq T_{i,j} \leq 1$. Consistent with Gambetta's definition [11], this is the probability assigned by node i that an email from j can be trusted to be spam free. Initially, $T_{i,j} = x$, where x is the default trust.

Updating Local Trust Based on Direct Experience. On receipt of an email from node j , node i applies a test based on its contents and a threshold (which is function of $T_{i,j}$) to determine whether it is spam. Set the binary value S to 1 if spam is found and 0 otherwise. In our simulations, we use an exponential averaging technique to update the local trust that node i has in node j as follows:

$$T_{i,j} = \alpha S + (1 - \alpha)T_{i,j}, \quad (1)$$

where parameter α that can be viewed as the *rate of adoption of trust*, $0 \leq \alpha \leq 1$. Note that having α set to 0 means that the trust value is unaffected by the experience. Having α set to 1 means that local trust is always defined by the latest experience and no memory is retained. The higher the value of α , the greater the influence of recent mails from a node on the trust value recorded for that node. Lower values of α encourage stability of the system. If a succession of experiences of mail from node j return the same spam determination, S , then the trust value $T_{i,j}$ converges towards S (towards 0 for a sequence of spam or towards 1 for a sequence of normal messages).

Updating Local Trust Based on Recommendation by Another Node. Node i may receive a message from a third party node, k , indicating a level of trust in node j . This can be modelled as node i adopting some of node k 's trust level in node j . As well as introducing a new parameter β indicating the level of influence of recommender trust on local trust, we also use $T_{i,k}$, how much domain i trusts domain k . In our simulations, we use an exponential averaging technique to update trust as follows:

$$T_{i,j} = \beta T_{i,k} T_{k,j} + (1 - \beta T_{i,k}) T_{i,j}, \quad (2)$$

where β is a parameter indicating the level of influence that recommender trust has on local trust, $0 \leq \beta \leq 1$. Note that, the larger the value of $T_{i,k}$, (i.e. the more i trusts k), the greater the influence of k 's trust in j on the newly updated value of $T_{i,j}$. Note that, if $T_{i,k} = 0$, (i.e. i has no trust in k), this causes $T_{i,j}$ to be unchanged.

4.3 Initial Results and Analysis

In this subsection, we present initial results of our experiments. Firstly, we examine how each node draws on both direct experience and recommendations from collaborating

nodes to reach a stable trust evaluation of another node. Secondly, we observe the effects of feedback of trust values for more effective spam filtering.

Convergence of Trust. How trust converges to stable values in our system depends on a number of factors, such as the size of the network, the means of updating trust, the extent of spam, the average size of each node's neighbourhood, and the starting point (how trust scores are initialised). The means of updating trust relates to whether we use direct experience only or how a mix experience and reputation inputs are combined. In our system, we use exponential averaging with parameters α and β respectively, so the choice of these values influences convergence.

Results shown here vary just some of these parameters. Each of the figures below is based on a network of fifty nodes. The default trust is set to 0.2 for all nodes.

Fig. 1 compares the option of using direct experience only to update trust with the combination of direct experience and frequent recommendations from neighbours. In this example, there are fifty good nodes in the network and there is no spam, meaning that all trust values should eventually converge to 100%. Setting parameter β to zero removes any effect of recommendations. Note that, in the direct experience only case ($\alpha = 0.1, \beta = 0$), trust level converges less smoothly – each jump occurs when a (non-spam) mail is received from the node in question. Recommendations allow trust values to be updated on receipt of mail by *another* node.

Fig. 2 examines the effect of the sizes of parameters α and β on convergence. Not surprisingly, the higher their values, the faster is convergence. It is important not to set these values too high though to avoid the risk of trust values oscillating widely.

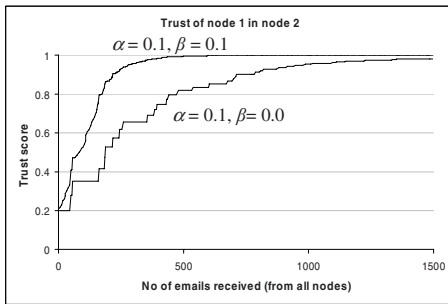


Fig. 1. Effect on convergence of using recommendations to update trust score

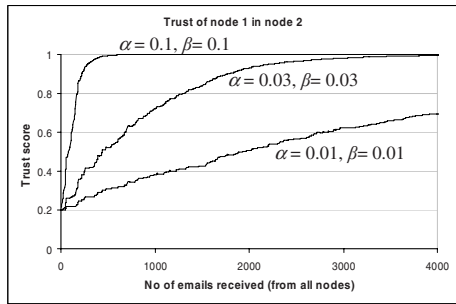


Fig. 2. Effect on convergence of varying parameters α and β

These illustrations just consider trust between good nodes. What happens when some nodes occasionally produce spam (perhaps due to poor configuration, or due to relatively open access policies)? Experiments with a small number of nodes producing a varying level of spam have shown that the trust recorded by a good node for each of these unreliable nodes converges to a different value, depending on the level of spam. This is encouraging as we can envisage trust filters that work by subjecting mail to a degree of scrutiny that is appropriate to the trust level in the source node.

Influence of Trust on Effectiveness of Filtering. The main objective of the work described in this paper is to see if we can get an improvement in the effectiveness of spam filtering by applying trust scores. The figures below show how this has been achieved in an illustrative case. In this experiment, a network of fifty nodes is chosen; also there is a single spammer who is responsible for 50% of all email generated in the system. Trust convergence for normal nodes is moderately fast, with parameters α and β both set to 0.03. Furthermore, the neighbourhood of each node consists on average of one-seventh of all nodes.

For this experiment, we choose relatively flat (but distinct) probability density functions for both spam and non-spam email. For spam mail, the aggregate spam indicator has a Gaussian (normal) distribution with a mean of 8.0 and a standard deviation of 4.0. For non-spam, the mean is 1.0 and the standard deviation is 2.0. This attempts to model the fact that parameters of non-spam mail tend to deviate less than those of spam (and hence fewer false positives than false negatives).

As already mentioned, most spam filters combine a variety of measures into a suspicion score and compare this score with a pre-defined threshold. For our experiments, a fixed threshold of 5.0 is chosen (*SpamAssassin* default) and used as a benchmark. As can be seen in Fig. 3 and Fig. 4 below, a significant reduction in both false positives and false negatives can be achieved with auto-tuning of the threshold (based on trust values). Auto-tuning is of course most effective in a steady-state situation when trust values have stabilised. A range of other predefined threshold values were also tried, but with no better results than the value of 5.0 shown. Choosing a higher predefined threshold causes an increase in false negatives and choosing a lower predefined threshold causes an increase in false positives.

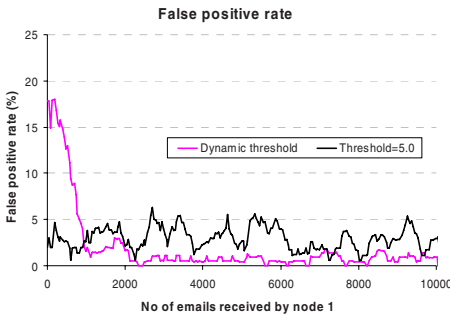


Fig. 3. Comparison of dynamic vs. fixed threshold: impact on rate of false positives

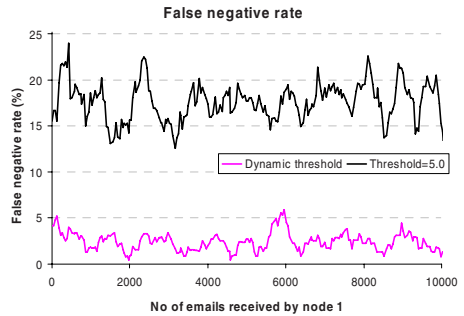


Fig. 4. Comparison of dynamic vs. fixed threshold: impact on rate of false negatives

5 Conclusions and Future Work

We have described a collaboration system for mail domains and how this can be used to make spam filtering more efficient and more effective. The collaboration system is lightweight, and relies on the decentralised maintenance of simple trust scores by individual mail domains. Stability and speed of convergence are influenced by a

number of tuneable parameters, and our initial simulations have investigated various parameter settings for email traffic with certain statistical characteristics.

Our simulations have also shown (again for systems and email traffic with specific statistical characteristics) that using trust measures to dynamically refine spam filter thresholds can improve effectiveness by reducing false positives and false negatives.

Further work is required to assess the applicability of this approach to situations with various topological and email traffic patterns. It is proposed to use real email data sets such as [14] to more realistically model incidence of spam and examine performance issues with our system. There is also significant scope for refinement of the system's dynamics, including how recommendations and new experiences are interpreted and used to update trust scores. Further experiments are required to explore the effects of various ways to define node neighbourhoods. It should also be possible to improve filter throughput as mails from trusted domains may require less processing.

It would also be very interesting to examine how spammers might try to get around this system, either by looking for weaknesses in the system's dynamics, or by collaborating with each other.

References

1. Schwartz, A.: SpamAssassin. O'Reilly (2004)
2. Goodman, J., Rounthwaite, R.: Stopping outgoing spam. In: Proc. ACM Conference on E-Commerce, New York (2004)
3. Abadi, M., Birrell, A., Burrows, M., Dabek, F., Wobber, T.: Bankable postage for network services. In: Saraswat, V.A. (ed.) ASIAN 2003. LNCS, vol. 2896, pp. 72–90. Springer, Heidelberg (2003)
4. Naor, M.: Verification of a human in the loop or identification via the Turing test. Unpublished manuscript (1996) <http://w.wisdom.weizmann.ac.il/naor>
5. Kong, J., Rezaei, B., Sarshar, N., Roychowdhury, V., Oscar Boykin, P.: Collaborative spam filtering using e-mail networks. IEEE Computer 39(8), 67–73 (2006)
6. Golbeck, J., Hendler, J.: Reputation network analysis for email filtering. In: Proc. Conf. on Email and Anti-Spam (2004)
7. Neustaedter, C., Bernheim Brush, A., Smith, M., Fisher, D.: The social network and relationship finder: social sorting for email triage. In: Proc. Conf. on Email and Anti-Spam (2005)
8. Han, S., Ahn, Y., Moon, S., Jeong, H.: Collaborative blog spam filtering using adaptive percolation search. In: Proc. International World Wide Web Conference. Edinburgh (2006)
9. Foukia, N., Zhou, L., Neuman, C.: Multilateral decisions for collaborative defense against unsolicited bulk e-mail. In: Stølen, K., Winsborough, W.H., Martinelli, F., Massacci, F. (eds.) iTrust 2006. LNCS, vol. 3986, pp. 77–92. Springer, Heidelberg (2006)
10. Androutsopoulos, I., Magirou, E., Vassilakis, O.: A game theoretic model of spam e-mailing. In: Proc. Conf. on Email and Anti-Spam, Stanford (2005)
11. Gambetta, D.: Can we trust trust? In: Gambetta, D. (ed.) Trust: making and breaking cooperative relations, pp. 213–237. Blackwell, Oxford (1988)
12. McGibney, J., Botvich, D.: A trust overlay architecture and protocol for enhanced protection against spam. In: Proc. Conf. on Availability, Reliability & Security, Vienna, pp. 749–756 (2007)
13. Douceur, J.: The Sybil attack. In: Proc. International Workshop on P2P Systems (2002)
14. Klimt, B., Yang, Y.: The Enron corpus: a new dataset for email classification research. In: Proc. European Conf. on Machine Learning, pp. 217–226 (2004)

An Architecture for Self-healing Autonomous Object Groups

Hein Meling

Department of Electrical Engineering and Computer Science,
University of Stavanger, N-4036 Stavanger, Norway
`hein.meling@uis.no`

Abstract. Jgroup/ARM is a middleware for developing and operating dependable distributed Java applications. Jgroup integrates the distributed object model of Java RMI with the *object group* paradigm, enabling construction of replicated servers that offer dependable services to clients. ARM aims to improve the dependability characteristics of systems through fault treatment, focusing on operational aspects where the gain in terms of improved dependability is likely to be the greatest. ARM offers two core mechanisms: recovery from *node*, *object* and *network failures* and distribution of replicas. ARM identifies failures and reconfigures the system according to its dependability requirements.

This paper proposes an enhancement of the ARM framework in which replica placement is performed in a *distributed* manner, eliminating the need for a centralized manager with global information about all object groups. Instead each autonomous object group handles their own replica placement based on information from nodes. Assuming that multiple objects groups are deployed in the system, this constitutes a *distributed replica placement* scheme. This scheme enables the implementation of self-healing object groups that can perform fault treatment on themselves. Advantages of the approach: (a) no need to maintain global information about all object groups which is costly and limits scalability, (b) reduced infrastructure complexity, and (c) less communication overhead.

1 Introduction

Networked computer systems are prevalent in most aspects of modern society, and we have become dependent on such computer systems to perform many critical tasks. Moreover, making such systems *dependable* is an important goal. However, dependability issues are often neglected when developing systems due to the complexities of the techniques involved. A common technique used to improve the dependability characteristics of systems is to *replicate* critical system components whereby the functions they perform are repeated by multiple replicas. Replicas are often distributed geographically and connected through a network as a means to render the failure of one replica independent of the others. However, the network is also a potential source of failures, as nodes can become temporarily disconnected from each other, introducing an array of new

problems. The majority of previous projects [1,2,3,4,5] have focused on the provision of middleware libraries aimed at simplifying the development of dependable distributed systems, whereas the pivotal deployment and operational aspects of such systems have received very little attention.

This paper presents an architecture for *Distributed Autonomous Replication Management* (DARM), aimed at improving the dependability of systems through a self-managed fault treatment mechanism that is adaptive to network dynamics and changing requirements. Consequently, the architecture improves the deployment and operational aspect of systems, where the gain in terms of improved dependability is likely to be the greatest, and also reduces the human interactions needed. The architecture builds on our experience [6,7] with developing a prototype that extends Jgroup [2] with fault treatment capabilities. The new architecture relies on a distributed approach for replica distribution (placement), thereby eliminating the need for a centralized management infrastructure used in our previous work [6,7]. Distributed replica placement enables deployed applications (implemented as object groups) to implement autonomic features such as self-healing by performing fault treatment on themselves. Fault treatment represents a non-functional aspect and is easily implemented as a separate protocol module to separate it from application concerns.

Jgroup [2] is a group communication service that integrates the Java RMI distributed object model with object groups. It supports *partition-awareness*: replicas placed in disjoint network partitions are informed about the current state of the system, and may take appropriate actions to ensure the availability of the provided service in spite of the partitioning. By supporting partitioned operation, Jgroup trades consistency for availability, whereas other systems takes a *primary partition* approach [8], ensuring consistency by allowing only a single partition to make progress. A *state merging service* is provided to simplify the re-establishment of a consistent global state when partitions merge.

DARM offers automated mechanisms for performing management activities such as distributing replicas among sites and nodes, and recovering from replica failures, thus reducing the need for human interactions. These mechanisms are essential to operate a system with strict dependability requirements, and are largely missing from existing group communication systems [3,4,2]. DARM achieves its goal through three core paradigms: *policy-based management* [9], where application-specific distribution and fault treatment policies are used to enforce dependability requirements; *self-healing* [10], where failure scenarios are discovered and handled through recovery actions with the objective to minimize the period of reduced failure resilience; *self-configuration* [10], where objects are relocated/removed to adapt to uncontrolled changes such as failure/merge scenarios, or controlled changes such as scheduled maintenance (e.g. OS upgrades), as well as software upgrade management [11]. DARM follows a non-intrusive system design, where the operation of deployed services is decoupled from DARM during normal operation. Once a service is installed, it becomes an “autonomous” entity, monitored by DARM until explicitly removed. This design principle enables support for a large number of object groups. The

Jgroup/DARM framework shares many of its goals with other fault tolerance frameworks, notably Delta-4 [12], AQuA [13], FT CORBA [14], and our previous implementation called ARM [6]. The novel features of Jgroup/DARM when compared to other frameworks include: autonomous management facility based on policies, distributed replica distribution and fault treatment, support for partition awareness, and interactions based solely on RMI.

Organization: Section 2 presents the system model and Section 3 gives an overview of Jgroup/DARM. In Section 4 the DARM framework is described. Section 5 compares DARM with related work and Section 6 concludes.

2 System Model and Assumptions

The context of this work is a distributed system comprising a collection of nodes connected through a network and hosting a set of client and server *objects*. The set of nodes, N , that may host application services and infrastructure services, in the form of server objects (or *replicas*), is called the *target environment*. The set N is comprised of one or more subsets, N_i , representing the nodes in site i . Sites are assumed to represent different geographic locations in the network, while nodes within a site are in the same local area network. A node may host several different replica types, but it may not host two replicas of the same type.

The system is *asynchronous* in the sense that neither the computational speed of objects nor communication delays are assumed to be bounded. Furthermore, the system is unreliable and failures may cause objects to *crash*, whereby they simply stop functioning. Once failures are repaired, they may return to being *operational* after an appropriate *recovery* action. Byzantine failures are not considered. Communication channels may omit to deliver messages; a communication substrate handles message retransmission, also using alternative routes [2]. Long-lasting *partitionings* may also occur, in which certain communication failure scenarios may disrupt communication between multiple sets of objects forming *partitions*. Objects in the same partition can communicate among themselves, but cannot communicate with objects in other partitions. When communication between partitions is re-established, we say that they *merge*.

Developing dependable applications to be deployed in these systems is a complex and error-prone task due to the uncertainty resulting from asynchrony and failures. The desire to render services partition-aware to increase their availability adds significantly to this difficulty. Jgroup/DARM is designed to simplify the development and operation of partition-aware, dependable applications by abstracting complex system events such as failures, recoveries, partitions, merges and asynchrony into simpler, high-level abstractions with well-defined semantics.

3 Jgroup/DARM Overview

Jgroup [2] supports dependable application development by means of replication, based on the *object group* paradigm [8], where a set of server objects form a group to coordinate their activities and appear to clients as a single server.

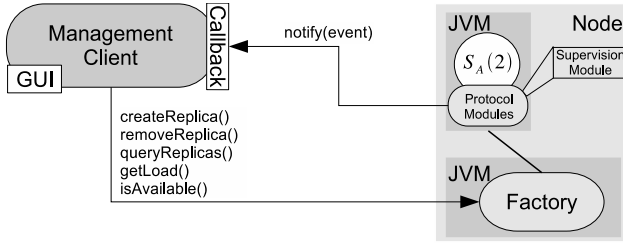


Fig. 1. Overview of DARM components

Jgroup provides a *partition-aware group membership service* (PGMS), a *group method invocation service* (GMIS) and a *state merging service* (SMS). The PGMS provides replicas with a consistent view of the group's current membership, enabling coordination of their actions. Reliable communication between clients and groups is handled by the GMIS and takes the form of *group method invocations* (GMIs) [2], that result in methods being executed by replicas forming the group. To clients, GMIs are indistinguishable from ordinary RMI: clients interact with the object group through a *client-side group proxy* that acts as a representative object for the group, hiding its composition. The proxy maintains information about the group composition, and handles invocations on behalf of clients by establishing communication with replicas and returning the result to the invoking client. On the server side, the GMIS enforces reliable communication among replicas. The SMS facilitate re-establishing a consistent shared state when partitions merge by handling state diffusion among partitions. Jgroup also includes a *dependable registry* (DR) allowing clients to locate object groups.

The ARM framework presented in [7,6] supports seamless deployment and operation of dependable services. Within the target environment, issues related to service deployment, replica distribution and recovery from failures are automatically managed by ARM, following the rules of user-specified distribution and fault treatment policies. Maintaining a fixed redundancy level is a typical requirement specified in the fault treatment policy.

In this paper, DARM is proposed in which fault treatment and replica distribution is performed in a distributed manner, rather than relying on a centralized (but replicated) replication manager (RM) component to handle these important mechanisms. The RM implemented in ARM [6] maintains global information about all object groups, which is costly as complex protocols are needed to maintain consistency across RM replicas, and each object group must report view changes to the RM replicas. This imposes an additional delay before fault treatment is activated, but more importantly it also limits the scalability (no. of groups) that can be supported by ARM. The proposed algorithm for distributed replica placement enables the implementation of a distributed fault treatment mechanism. However, it also introduces additional challenges with respect to appropriate load balancing of replicas on the nodes in the target environment.

Fig. 1 illustrates the core components and interfaces supported by the DARM framework: a supervision module associated with each application replica (S_A),

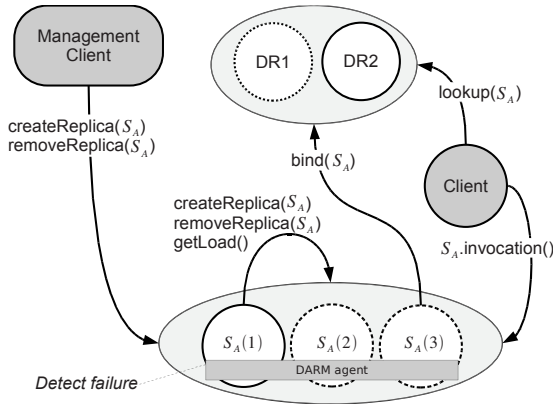


Fig. 2. The Jgroup/DARM architecture

an object factory deployed at each node in the target environment, and a management client used to interact with object factories to install/remove replicas.

The *supervision module* is the DARM agent co-located with each replica which is responsible for collecting and analyzing failure information obtained from view change events generated by the PGMS, and reconfigure the system on-demand according to the configured policies. It is also responsible for decentralized removal of excessive replicas. The *object factories* enable the management client to install/remove replicas, as well as to respond to queries about replicas on the local node, and its current load. The *management client* provide administrators with an interface through which to install and remove applications in the system and to specify and update the distribution and fault treatment policies to be used. It can also be used to obtain monitoring information about running services. Overall, the interactions among these components enable the DARM agent to make proper recovery decisions, and allocate replicas to suitable nodes in the target environment.

Next, a brief description of a minimal Jgroup/DARM deployment is given, as shown in Fig. 2. Only two different groups are shown. The DR service represent the naming service infrastructure component and is required in all Jgroup/DARM deployments. In addition, each application service must contain the DARM agent, the supervision module, as discussed above. The figure also illustrates a service labeled S_A that is implemented as a simple object group managed through Jgroup/DARM. Finally, two clients are shown: one client interacts with the S_A object group, while the other is the management client used to create and remove object groups by interacting with object factories. Object factories are not shown, but are present at each node in the target environment. The main communication patterns are shown as graph edges. For example, the DARM agent associated with an object group detect failures by monitoring the current membership of the group, and activate fault treatment actions as needed

to recover from various failure scenarios. When joining the system, replicas must bind themselves to the same name (e.g. S_A) in the dependable registry, to be looked up later by clients. After obtaining references to object groups, clients may perform remote invocations on them. The object group reference hides the group composition from the client.

4 The DARM Framework

This section describes the main elements of the DARM architecture and provide an informal discussion of the algorithms related to failure analysis and recovery. Algorithms are provided in [15]. The DARM architecture borrows parts of its infrastructure from ARM [6], and where appropriate the differences between the two are explained.

4.1 The Management Client

The management client enables a system administrator to install or remove services on demand. The initial deployment of replicas is handled by the management client using the distribution policy discussed below. The management client may also perform runtime updates of the configuration of a service. In the Jgroup/ARM implementation [6], updates are restricted to changing the redundancy level attributes. Additionally, the management client may subscribe to events associated with one or more object groups. These events are passed to the management client through the `Callback` interface, permitting appropriate feedback to the system administrator. Currently, two management client implementations exist, one providing a graphical front-end to ease human interaction, and one that supports defining scripts to perform automated installations. The latter was used to perform experimental evaluations using the original centralized ARM implementation as reported in [16,7].

4.2 Replication Management Policies

Policy-based management [9] is aimed at enabling administrators to specify how a system should autonomically react to changes in the environment — with no human intervention. These specifications are called *policies*, and are typically defined through high-level declarative directives describing how to manage various system conditions. Policy-based management architectures are often organized using two key abstractions [17]: a manager component and a set of managed resources controlled by the manager. Typically the manager is a centralized entity and is called the *policy decision point* (PDP), and the managed resources are called *policy enforcement point* (PEP). In the DARM architecture, the decision and enforcement points can easily be co-located on the managed resources enabling implementation of decentralized policies.

In DARM two separate policy types are defined to support the autonomy properties: (1) the *distribution policy* and (2) the *fault treatment policy*, both of

which are specific to each deployed service. Alternative policies can be added to the system. The policies used here is just the minimum set.

The purpose of a distribution policy is to describe how service replicas should be allocated onto the set of available sites and nodes. Two types of input are needed to compute the replica allocations of a service: (1) the target environment, and (2) the number of replicas to be allocated. The latter is obtained at runtime from the fault treatment policy. The distribution policy in DARM is similar to the one used in ARM [7]: `DisperseOnSites` will avoid co-locating two replicas of the same service on the same node, while at the same time trying to disperse the replicas evenly on the available sites. In addition, the least loaded nodes in each site is selected. The same node may host multiple distinct service types. The primary objective of this policy is to ensure available replicas in all *likely* network partition that may arise. Secondly, it will load balance the replica placements evenly over each site. A distribution policy algorithm is given in [15].

Each service is associated with a fault treatment policy, whose primary purpose is to describe how the redundancy level of the service should be maintained. Two inputs are needed: (1) the target environment, and (2) the initial (R_{init}) and minimal (R_{min}) redundancy level of the service. The current fault treatment policy called `KeepMinimalInPartition` has the objective to maintain service availability in all partitions, i.e. to maintain R_{min} in each partition that may arise (see [15] for details). Alternative policies can easily be defined, e.g. to maintain R_{min} in a primary partition only.

Policy specifications are part of a sophisticated configuration mechanism, based on XML, enabling administrators to specify (1) the target environment, (2) deployment-specific parameters, and (3) service-specific descriptors.

4.3 The Object Factory

The purpose of object factories is to facilitate installation and removal of service replicas on demand. To accomplish this, each node in the target environment must run a JVM hosting an object factory, as shown in Fig. 1. In addition, the object factory is also able to respond to queries about which replicas are hosted on the node. The availability status of a node (factory) can also be checked by invoking the `isAvailable()` method on the factory. This method is used by the distribution policy to determine if a node is available before selecting it to host a replica, whereas the `getLoad()` method obtains load information about the node.

The factory maintains a table of local replicas; this state need not be preserved between node failures since all replicas would have crashed as well. Thus, the factory can simply be restarted after a node repair and support new replicas. Furthermore, object factories are not replicated and thus do not depend on any Jgroup or DARM services. Replicas run in separate JVMs, to avoid that a misbehaving replica causes the failure of other replicas within a common JVM.

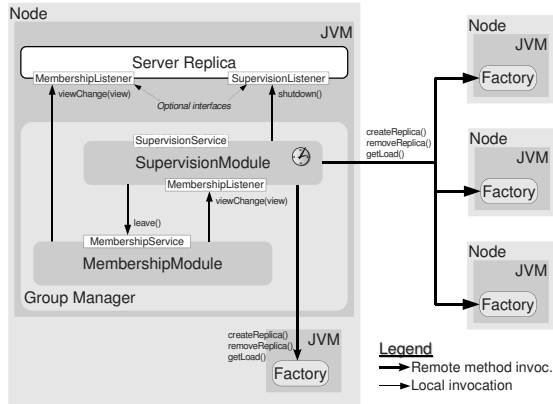


Fig. 3. The Distributed ARM architecture

4.4 Monitoring and Controlling Services

Keeping track of service replicas is essential to enable discovery of failures and to rectify any deviation from the dependability requirements. The purpose of DARM is (1) to distribute service replicas in the target environment, to (best) meet the operational policies for all services (see Section 4.2); (2) to collect and analyze information about failures, and (3) to recover from them.

Fig. 3 shows the Distributed ARM architecture. The architecture follows an *event-driven* design in that events are reported to the supervision protocol module rather than having to continuously probe individual components. Hence, the supervision module exploits synergies with existing Jgroup modules, the membership module in particular. Applications that wish to support fault treatment must include the supervision module in its protocol composition.

The supervision module operates on group-level events, also called *view change* events received from the membership module. A group leader (associated with each application service) is responsible for detecting failures and activating fault treatment actions (see Section 4.6). In this way, the failure detection costs incurred by the PGMS are shared with other modules that need membership information. The group leader is elected implicitly by the total ordering of the group members, hence there is no additional cost of leader election. If the group leader fails, a new group leader is implicitly elected by the total ordering of group members in the new view installed by the group. Note that membership events cannot discriminate between crash failure and network partition failures.

Unfortunately, group-level events are not sufficient to cover *group failure* scenarios in which all remaining replicas fail before fault treatment can be activated. This can occur if multiple nodes fail in rapid succession, or if the network partitions, e.g., leaving only one replica in a partition whom fails shortly thereafter. A solution to this could be to have the various groups monitor each other using a *lease renew* mechanism similar to the approach taken in the centralized ARM [6] architecture, where the centralized manager tracks all groups.

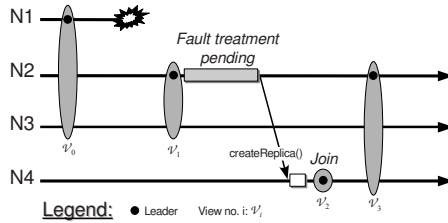


Fig. 4. An example crash failure-recovery sequence where $R_{\min} := 3$

Both tracking mechanisms can be managed by supervision modules. View changes are received by the supervision module of all replicas in the group, but only the group leader activates the fault treatment action, e.g. to replace a failed replica or remove an excessive replica, as discussed in Section 4.6 and 4.5.

An example of a common failure-recovery sequence is shown in Fig. 4, in which node $N1$ fails, followed by a recovery action causing the supervision module to install a replacement replica at node $N4$. In the centralized ARM implementation [6], the recovery action was performed by a centralized RM, which would have a complete view of all installed applications within the target environment. Recomputing the replica allocations in a distributed manner offers a considerable challenge.

4.5 The Remove Policy

The supervision module may optionally be configured with a remove policy to account for any excessive replicas that may be installed. The reason for the presence of excessive replicas is that during a partitioning, a fault treatment action may have installed additional replicas in one or more partitions to restore a minimal redundancy level. Once partitions merge, these replicas are in excess and no longer needed to satisfy the fault treatment policy.

Let $\mathcal{V}$ denote a view and $|\mathcal{V}|$ its size. If $|\mathcal{V}|$ exceeds the initial redundancy level R_{init} for a duration longer than a configurable time threshold (remove policy delay), the supervision module requires one excessive replica to leave the group. If more than one replica needs to be removed, each remove is separated by the remove policy delay. The choice of which replicas should leave is made deterministically based on the view composition, enabling decentralized removal. This mechanism is shown in Fig. 5, where the dashed timelines indicate the duration of the network partition. After merging, the supervision module detects one excessive replica, and elects $N4$ to leave the group.

4.6 Failure Recovery

The supervision module handles failure recovery for its associated application, as follows: (i) determine the need for recovery, (ii) determine the nature of the failures, and (iii) the actual recovery action. The first is accomplished through a

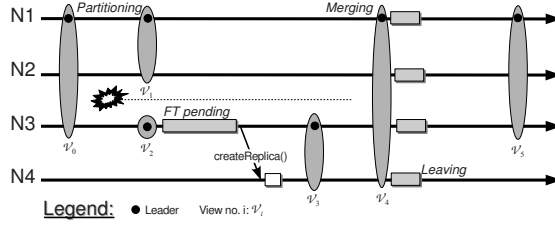


Fig. 5. A sample network partition failure-recovery scenario where $R_{\text{init}} := 3$ and $R_{\text{min}} := 2$. The partition separates nodes $\{N1, N2\}$ from $\{N3, N4\}$.

reactive mechanism based on service-specific timers, while the last two use the abstractions of the fault treatment and distribution policies, respectively.

The supervision module receive events and maintains the state necessary to determine the need for recovery, according to the fault treatment policy of the associated service. Each instance of the supervision module maintains a *Service Monitor* (SM) timer for its associated application service. The purpose of the SM timer is to delay the activation of a fault treatment action until the current membership has stabilized. Moreover, the recovery algorithm is invoked if the SM timer expires. To prevent activating unnecessary recovery actions, the SM timer must either be rescheduled or canceled before it expires. The SM status is updated by means of *ViewChange* events associated with the service: If the received view $\mathcal{V}$ is such that $|\mathcal{V}| \geq R_{\text{min}}$, the SM timer is canceled, otherwise the SM is rescheduled, pending additional view changes. Upon expiration of the SM timer and detecting that the service needs recovery, the recovery algorithm is executed with the purpose of determining the nature of the current failure scenario. Recovery is performed through two primitive abstractions: *restart* and *relocation*. Restart is used when the node's factory remains available, while relocation is used if the node is considered unavailable. The actual installation of replacement replicas is done using the distribution policy.

5 Related Work

Fault treatment techniques similar to those provided by DARM were first introduced in the Delta-4 project [12]. Delta-4 was developed in the context of a fail-silent network adapter and does not support network partition failures. Due to its need for specific hardware and OS environments, Delta-4 has not been widely adopted. None of the most prominent Java-based fault tolerance frameworks [4,1] offers mechanisms similar to those of DARM, to deploy and manage dependable applications with only minimal human interaction. These management operations are left to the application developer. However, the FT CORBA standard [14] specify certain mechanisms such as a generic factory, a centralized RM and a fault monitoring architecture, that can be used to implement a centralized management facilities similar to ARM [7,6]. DARM as presented in this paper enable distributed fault treatment. Furthermore, the standard makes

explicit assumptions that the system is not partitionable, a unique feature of Jgroup/DARM. Eternal [5] is probably the most complete implementation of the FT CORBA standard, and uses a centralized RM. It supports distributing replicas across the system, however, the exact workings of their replica placement approach has not been documented. DOORS [18] is a framework that provides a partial FT CORBA implementation, focusing on passive replication. It uses a centralized RM to handle replica placement and migration in response to failures. The RM component is not replicated, and instead performs periodic checkpointing of its state tables, limiting its usefulness since it cannot handle recovery of other applications when the RM is unavailable. Also the MEAD [19] framework implements parts of the FT CORBA standard, and supports recovery from node and process failures. However, recovery from a node failure requires manual intervention to either reboot or replace the node, since there is no support for relocating the replicas to other nodes as in DARM. AQuA [13] is also based on CORBA and was developed independently of the FT CORBA standard. AQuA is special in its support for recovery from value faults, while DARM is special in supporting recovery from partition failures. AQuA adopts a closed group model, in which the group leader must join the dependability manager group in order to perform notification of membership changes (e.g. due to failures). Although failures are rare events, the cost of dynamic joins and leaves (run of the view agreement protocol), can impact the performance of the system if a large number of groups are being managed by the centralized dependability manager. The ARM [20,7,6] framework uses a centralized RM to handle distribution of replicas (replica placement), as well as fault treatment of both network partition failures and crash failures. The ARM framework uses the open group model, enabling object groups to report failure events to the centralized manager without becoming a member of the RM group.

DARM essentially supports the same features as ARM, but instead uses a distributed algorithm to perform replica placement according to a distribution policy. This enables each group to handle their own allocation of replicas to the sites and nodes in the target environment. Thereby, eliminating the need for a centralized RM that maintains global information about all object groups in the system, which is required in all frameworks discussed above. Furthermore, none of the other frameworks that support recovery focus on tolerating network partitions. Nor do they explicitly make use of policy-based management, which allows DARM to perform recovery actions based on predefined and configurable policies enabling self-healing and self-configuration properties, ultimately providing autonomous fault treatment.

6 Conclusions and Future Work

This paper has presented an architecture for distributed autonomous replication management based on our previous experiences with building a centralized ARM architecture [6]. The new architecture enables seamless self-healing of dependable applications through a distributed fault treatment policy implemented in the

protocol modules associated with applications. There are still a few open issues in our system; e.g. how to cope with multiple applications recovering simultaneous, which may result in several new replicas being allocated to the same *least loaded* node, causing the node to become highly overloaded. This is an artifact of our distributed approach. Once the implementation has been completed, we intend to perform elaborate experimental evaluations similar to our previous work on failure recovery measurements [7,16,6]. That is, the injection of multiple nearly-coincident node and network failures to test the failure-recovery success rate of our system and to iron out any design and implementation flaws.

Acknowledgments. The author wish to thank Alberto Montresor and Bjarne Helvik for valuable comments on this work.

References

1. Amir, Y., Danilov, C., Stanton, J.: A Low Latency, Loss Tolerant Architecture and Protocol for Wide Area Group Communication. In: Proc. Int. Conf. on Dependable Systems and Networks, New York (2000)
2. Montresor, A.: System Support for Programming Object-Oriented Dependable Applications in Partitionable Systems. PhD thesis, Dept. of Computer Science, University of Bologna (2000)
3. Felber, P., Guerraoui, R., Schiper, A.: The Implementation of a CORBA Object Group Service. Theory and Practice of Object Systems 4, 93–105 (1998)
4. Ban, B.: JavaGroups – Group Communication Patterns in Java. Technical report, Dept. of Computer Science, Cornell University (1998)
5. Narasimhan, P., et al.: Eternal - a Component-Based Framework for Transparent Fault-Tolerant CORBA. Softw. Pract. Exper. 32, 771–788 (2002)
6. Meling, H.: Adaptive Middleware Support and Autonomous Fault Treatment: Architectural Design, Prototyping and Experimental Evaluation. PhD thesis, Norwegian University of Science and Technology, Dept. of Telematics (2006)
7. Meling, H., Montresor, A., Helvik, B.E., Babaoğlu, Ö.: Jgroup/ARM: A Distributed Object Group Platform with Autonomous Replication Management. Technical Report No. 11, University of Stavanger (2006)
8. Chockler, G.V., Keidar, I., Vitenberg, R.: Group Communication Specifications: A Comprehensive Study. ACM Computing Surveys 33, 1–43 (2001)
9. Sloman, M.: Policy driven management for distributed systems. Journal of Network and Systems Management 2 (1994)
10. Murch, R.: Autonomic Computing. On Demand Series. IBM Press (2004)
11. Solarski, M., Meling, H.: Towards Upgrading Actively Replicated Servers on-the-fly. In: Proc. Workshop on Dependable On-line Upgrading of Distributed Systems in conjunction with COMPSAC 2002, Oxford, England (2002)
12. Powell, D.: Distributed Fault Tolerance: Lessons from Delta-4. IEEE Micro 36–47 (1994)
13. Ren, Y., et al.: AQuA: An Adaptive Architecture that Provides Dependable Distributed Objects. IEEE Trans. Comput. 52, 31–50 (2003)
14. Object Management Group: Fault Tolerant CORBA Specification. OMG Document ptc/00-04-04 (2000)
15. Meling, H.: An Architecture for Self-healing Autonomous Object Groups. Technical Report No. 21, University of Stavanger (2007)

16. Helvik, B.E., Meling, H., Montresor, A.: An Approach to Experimentally Obtain Service Dependability Characteristics of the Jgroup/ARM System. In: Proc. Fifth European Dependable Computing Conference (2005)
17. Agrawal, D., Lee, K.W., Lobo, J.: Policy-Based Management of Networked Computing Systems. *IEEE Commun. Mag.* 43, 69–75 (2005)
18. Natarajan, B., et al.: DOORS: Towards High-performance Fault Tolerant CORBA. In: Proc. 2nd Int. Symp. Distributed Objects and Applications (2000)
19. Reverte, C.F., Narasimhan, P.: Decentralized Resource Management and Fault-Tolerance for Distributed CORBA Applications. In: Proc. 9th Int. Workshop on Object-Oriented Real-Time Dependable Systems (2003)
20. Meling, H., Helvik, B.E.: ARM: Autonomous Replication Management in Jgroup. In: Proc. 4th European Research Seminar on Advances in Distributed Systems, Bertinoro, Italy (2001)

A Generic and Modular System Architecture for Trustworthy, Autonomous Applications

G. Brancovici and C. Müller-Schloer

University of Hannover, Institute of Systems Engineering,
System and Computer Architecture
Appelstr. 4, 30167 Hannover, Germany
{George.Brancovici, cms}@sra.uni-hannover.de

Abstract. We propose a generic architecture to facilitate the systematic design of autonomous, adaptive and safe applications. We specify generic modules including a trustworthiness enforcement layer dedicated to ensure the system's functional stability as seen by the human owner. Instead of building a monolithic system, we encourage modularization based on the cognitive function of the components. A key premise is that domain knowledge is explicitly specified as a parameter of each application, with the side effect of enabling seamless integration with other remote autonomous or infrastructure applications. The design choices we have made are exemplified on a demonstrative travel management application.

1 Introduction

We describe our efforts to improve a generic architecture that support the systemic and modular design of autonomous and adaptive applications, capable of seamless organic integration through interaction with other compatible provider and consumer applications, while enjoying extensive confidence from the humans. This reflects how trustworthy the autonomous system is.

Trustworthiness requires that the system's behavior has no immediate or future negative effects on the user's or system's health and does not harm their interests. In other words, the system functions as directly or indirectly expected by the user. It can be achieved partially through careful verification and validation. When the complexity of the adaptive system and of its environment increases, it may become practically impossible to use this approach in a theoretically sound manner. Sometimes trustworthiness enforcement needs to be done at runtime.

The main test bed for our ideas is a full-fledged application that can provide a reactive, active, adaptive and trustworthy interface to travel management services. Alternative use cases for our architecture including an intelligent distributed calendaring application and other types of intelligent, context-aware and collaborating assistants are being investigated.

An overview of the architecture is given in the second chapter, with the Travel Manager (TMGR) as an example application. Background theoretical aspects that led to this proposal are discussed in the third chapter. The main generic components of

the corresponding Intelligent Adaptive Systems Framework (IASF) are presented in more detail in the fourth chapter. The status of the implementation and next steps are discussed throughout the fifth chapter. We end our proposal with a conclusion.

2 The Architecture

2.1 Motivation

The proliferation of complex adaptive systems capable to think and act autonomously from humans is already happening. These applications that can handle different types of tasks from process control to user assistance are usually organically interconnected and have a sense of their environment. Examples of such complex systems are flight control systems on airplanes, robotic control systems used in space exploration just as well as personal assistants such as travel managers, medical assistants or collaborative personal information management applications.

The actions these systems do can have potentially far-reaching or dangerous (side-) effects, either immediately or in time. Sometimes they can produce immediate unacceptable (e.g. travel) costs for the user or cause delayed problems (e.g. missed appointments and loss of associated profits) or can even be life-threatening for the user (e.g. wrongly administered medication) or for the system itself and humans (e.g. space exploration and flight control, respectively). Other times these actions can be obviously suboptimal or simply unacceptable for the user.

The quality of the autonomy of such applications depends mostly on the ability to adapt their behavior to situations that may appear and do this trustworthily. Humans could supervise everything; however this would drastically decrease the quality of the autonomy and may be sometimes impossible to do, in real-time or at all.

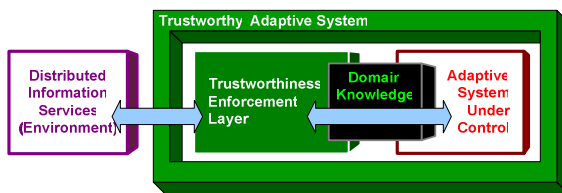


Fig. 1. An overview of the concept

We propose a systematic way to avoid this problem (Fig. 1). The adaptive system under control is placed within a shell that guarantees trustworthiness for one or more such systems. The interaction with the external world is done through a Trustworthiness Enforcement Layer that observes and controls the behavior. The common ground is established by imposing that at least a part of the domain knowledge be explicitly described and shared, in order to allow monitoring the system's behavior. This serves as a basis for the automatic trustworthiness enforcement mechanism.

The proposed architecture enables the development of intelligent applications with mixed capabilities, depending on the desired level of intelligence and on the power of

the targeted device. Modules with similar function can be swapped or alternatively used, while modules capable to fulfill distinct types of cognitive activities can be combined.

The modularity, interchangeability and reusability of modules are possible due to the interest we take in providing standardized interfaces between them and especially in the explicit representation of knowledge throughout the architecture. Knowledge is nothing but a parameter, defined in a symbolic, human-readable way.

2.2 The Travel Manager (TMGR)

The Travel Manager's role is to assist the user while planning and ordering complex trips. This requires planning the necessary steps and executing them, which includes both searching and booking travel related items like train rides or hotels. The user does not have to supply the exact details of the trip, but just some already known parameters. The complete plans are computed actively by the TMGR. Once the user makes a choice, the TMGR takes care of all details just like an assistant would do. TMGR attempts to learn what strategies the user would favor.

Our application is similar in complexity with other autonomous applications, including those capable to control spacecrafts in space exploration. Both kinds of systems are capable to deal with high-level goals autonomously, with the exception that the TMGR receives them directly while the others are transmitted remotely. Although the TMGR does not seem to be as potentially dangerous as a spacecraft control system, this is only a subjective difference relative to the actual observer.

2.3 An Overview of the Architecture

In this section we present the general architecture (Fig. 2) as seen by the designer of the TMGR. Our proposal is extended to a modular architecture for trustworthy adaptive autonomous systems that specifies both the Trustworthiness Enforcement Layer itself and a corresponding flexible structure for the adaptive system under control. The modular aspect of the architecture relies partly on the model of the human cognition. This consists of several basic types of cognitive processes, which can be used individually or combined. The associated Intelligent Adaptive Systems Framework (IASF) collects a series of components common to intelligent applications and simplifies the development of new ones.

The TMGR interacts with the external world through a series of sensors and of known web services capable of processing travel-related information. These web services can be provided on central servers of the travel service providers or by other peer applications, e.g. belonging to fellow travelers. An IASF-level (*System*) *Context Manager* offers a dynamic unified image of the device's context, by caching sensor information. All the information gained by the intelligent system is ultimately stored in the local knowledge base within the *Planner*. In addition to being information repositories, the web services also constitute the environment in which the intelligent TMGR executes its actions.

We will discuss the levels of autonomy (reactive, active, adaptive and trustworthy) that applications using our architecture (Fig. 2) can exhibit towards the user. In our example, these levels are built on each other, respectively.

At the lowest intelligence level, the (*reactive*) TMGR is capable to process atomic tasks that are directly commanded by the user through a *TMGR GUI*. They are transformed into messages by the *Direct Request TMGR* and sent to remote web services, selected using the *Peer Service Router*. This collects information about known remote entities and their capabilities. In this configuration, the *Behavior Guardian* simply forwards messages. When the user issues a search request (e.g. for a simple flight), the *Direct Request TMGR* will receive the answers that directly match the request. They will be displayed within the GUI and the user can act on the respective resource (e.g. order the flight).

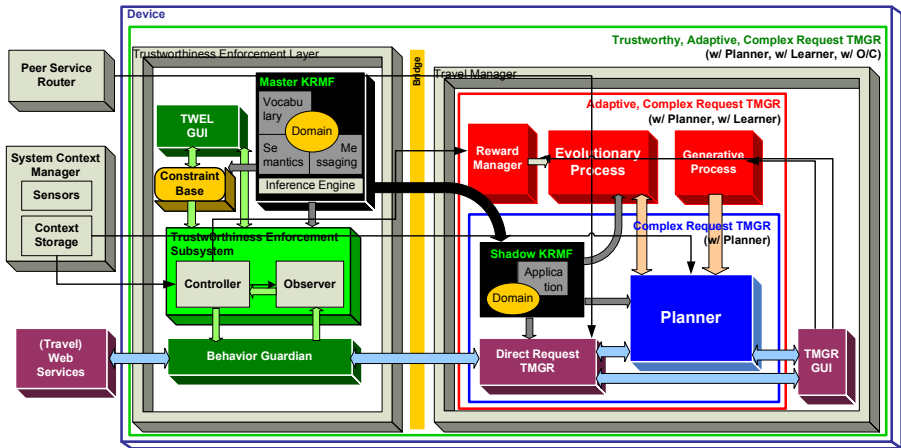


Fig. 2. The general architecture exemplified on the Travel Manager

The user has full control over the system, since the TMGR possesses little intelligence. However, the burden of planning each step of a trip lies on the user's shoulders. Whenever no appropriate travel solution can be found for a leg of the trip planned by the user, he has to repeat the entire planning process. An application that can search for alternative solutions automatically or that even allows the user to specify tasks in an abstract manner and does the planning itself is preferred.

At the next level of intelligence, the (*active*) TMGR can process and plan abstract tasks using a set of explicit and hard-coded rules for task decomposition. The request issued by the user is now sent to a domain-configurable *Planner*. This uses an iterative process to find a plan, during which it communicates with the remote web services using the functionality provided by the *Direct Request TMGR*. The system inherently searches for travel solutions to be automatically bound to each leg of the planned trip. As soon as complete plans have been found, the user can accept one of them (e.g. order the associated travel solutions) or not.

This system works deterministically. Since the decomposition methods are explicitly hard-coded by an expert, the plans the system delivers should be generally acceptable and presumably optimal, at least for the situations considered during the design phase. There are two disciplines at which this system will fail at times. First of

all, there might be unforeseen situations in which the system will be unable to find solutions. Secondly, even when solutions will be found, the TMGR might systematically favor some that are not considered optimal by the user. Since the system relies on hard-coded rules, its behavior will not improve.

The TMGR as described above relies on hard-coded but explicitly written rules in order to find plans. The merit of this design is that the system can be made to exhibit a different behavior, simply by modifying the set of rules. We can use external parameters like feedback from the user to decide whether a change made to the set of rules was beneficial or not. This way we can direct the update algorithm to help it converge towards a set of rules that deliver optimal plans for the user and its context. Finding plans according to very specific rules is also faster. Similarly, rules can be generated to yield plans in previously uncovered situations.

At this level of intelligence, the (*adaptive*) TMGR works as before, primarily. The rules' definitions comply with a evolutionary population specification. This allows us to define a set of valid operations capable to reshape these rules, including unary and binary transformations. They are used within a generalized *Evolutionary Process*. The transformations are similar to those used by traditional genetic algorithms (unary and binary transformations can be e.g. mutation and crossover, respectively). The *Reward Manager* evaluates the feedback from all sources and uses it to drive the *Evolutionary Process* towards building an optimal set of rules. The *Generative Process* supervises the user's activity and provides functionality to generate rules based on observations as well as to manage facts important to the system.

Provided that the system was initially supplied with proper rules and that the user has trained it sufficiently, it is now able deliver optimal plans to its user. Even in situations where it would have not been able to find plans during a single attempt, the system can try repeatedly to find a solution. The downside of this approach is that the behavior of the system is not bounded anymore by a well defined set of rules. Actually, the set of rules can change dynamically, so the behavior of the system cannot be easily estimated over time. The system's behavior has practically become nondeterministic, which can lead to potentially hazardous behavior in certain circumstances. This problem will be handled in the next sections.

3 An Analysis of the Architecture

3.1 Reasoning Patterns

Different types of reasoning ([6]) have been mentioned implicitly throughout the description of the architecture of the Travel Manager and are summarized in (Fig. 3).

The Planner is the key component of our intelligent system. It is implemented using a planning engine, which is a specialized type of inference engine. We use the JSHOP2 planner ([9]) in our reference implementation. Other planners or even generic inference engines can be used instead or in parallel.

Responsible for the adaptive character of the system are the Evolutionary and the Generative Processes, which rely on techniques similar to the inductive reasoners.

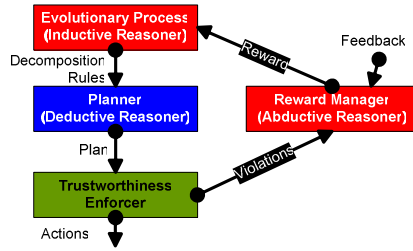


Fig. 3. Types of reasoning within the architecture

The Evolutionary Process we are working on induces structural changes to the rules, by affecting their buildup out of basic operations. Other inductive reasoners could use softer types of learning, such as adaptation of the parameters of basic operations or of the quality parameters used for conflict resolution. The Generative Process monitors solutions proposed by the user and uses heuristics to generate new rules.

The Reward Manager relies on abductive reasoning techniques to decide which the causes of constraint violations (see the Trustworthiness Enforcement Layer) were and to distribute the feedback accordingly.

3.2 Domain Knowledge and Its Modeling

One of our initial decisions was to explicitly encode the domain knowledge (examples will follow). The consequent centralization of knowledge inside the system greatly simplifies the effort to update or replace it or parts of it.

As shown in (Fig. 4), the knowledge about information representation is divided into several sections. The most general concepts are included in a basic *vocabulary*; more specific terms are placed in an *application*-specific set, while knowledge about operations on concepts is kept in the *messages* section (due to our communication-centric approach). The representational knowledge mentioned above is immutable and is kept as an ontology within the Knowledge Representation and Management Framework (KRMF) discussed later.

More expressive (and learnable) *inference rules* based on the concepts and operations from KRMF are needed to extend the reasoning capabilities. In the Travel Manager, these rules are called decomposition rules and are kept within the Planner, but they are modifiable from outside. The Evolutionary Process relies on the set of configurable *evolutionary operations and transformations* that it uses to extend the set of inference rules within the Planner. The immutable set of operations and transformations used by the Evolutionary Process are accompanied by a specification in a *evolutionary population specification*, defined on top of the concepts and operations in KRMF. Last but not least, the Trustworthiness Enforcement Layer contains a set of *trustworthiness constraints* which outline the functional envelope ([7] and [8]) the system must be enclosed into.

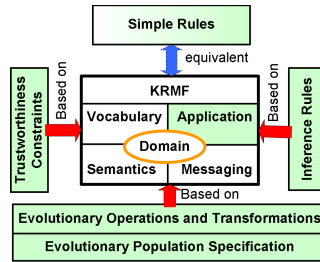


Fig. 4. An overview of the domain knowledge

3.3 Trustworthiness Enforcement

We control the information flow that characterizes the functionality of the intelligent system and build a functional envelope ([7], [8]). The trustworthiness constraints can be described and enacted in a cross-domain manner and regardless of whether the adaptive systems itself was organized in accordance with the recommendations of our architecture.

4 The Generic Modules

The domain knowledge is shared to some extent by all the components of the architecture. For example, the Evolutionary Process updates the inference rules used within the Planner. Many components need access to factual knowledge that is stored within the Planner and the System Context Manager.

Some modules are application-independent and rely on a common subset of the domain knowledge. Such modules, like the System Context Manager, the Knowledge Representation and Management Framework and the components of the Trustworthiness Enforcement Layer, are implemented within the IASF framework. They are described in the following sections.

4.1 The Knowledge Representation and Management Framework

The Semantic Web initiative showed that a standardized way to access to information on the web is needed, as a foundation for developing intelligence. When abstracting from the TMGR as an application processing heterogeneous information towards a more generic architecture, the relevance of this foundation is obvious.

Most intelligent applications are built around inference engines. A vital part of an inference engine is the knowledge base, which is structured according to an ontology. However, the information from the ontology can be used for additional purposes, than to describe the factual content of the knowledge base.

We propose the Knowledge Representation and Management Framework as a solution to standardize access to the ontology and the additional information. An example of how KRMF's functionality can be used from other components of the intelligent system is given in (Fig. 5) in the context of the TMGR.

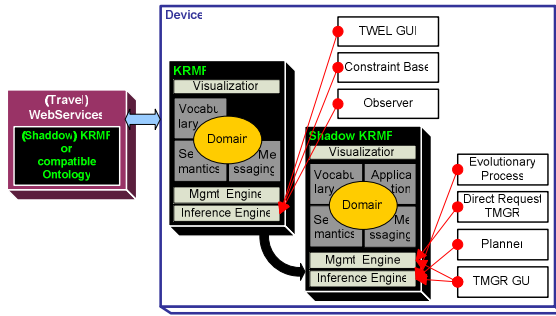


Fig. 5. Integrating KRMF in an intelligent system like the TMGR

The ontology within KRMF is vital to an intelligent system like the TMGR. Not all components need the same level of access to the ontology. This principle is illustrated in (Fig. 6). Modules that need limited ontological knowledge are connected to the “Master KRMF” and “know” about general concepts, as well as about properties and features of the other (marked) concepts. They can also rely on the simple automatic inference. Components belonging to a certain application such as the TMGR are using the “Shadow KRMF” and have full access, which means that they fully “know” the specific concepts.

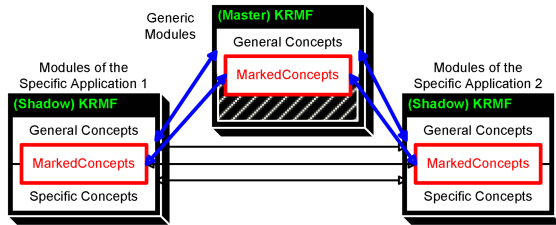


Fig. 6. Differentiated access to knowledge and knowledge representation

This design allows KRMF to be loaded as an IASF-level instance that provides services to the rest of the generic components. At the same time, each application receives a (shadow) copy of KRMF and can load additional definitions of concepts that belong to its domain (e.g. travel). The layered and extensible design of KRMF and of the ontology at its core allows applications with different view scopes about the world to coexist with generic modules.

The ontology definition is given in UML ([2], [3]) using a series of special constructs and ontology design guidelines inspired by but not limited to ([1], [4]) that are outside the scope of this document. It is used by the KRMF reference implementation which is written in Java.

4.2 The Trustworthiness Enforcement Layer

We have designed and implemented a Trustworthiness Enforcement Layer (TWEL) capable to guard and ensure the safety of applications like the TMGR. TWEL can

control multiple applications at runtime, regardless of their reasoning engine or adaptation algorithm, in a centralized and simultaneous way. This component also provides feedback when violations of the constraints contracted between the user and the TWEL occur.

The TWEL acts as an intermediary layer between the intelligent system and the external world (Fig. 2). The layer is separated from the productive system by a bridge that defines a precise interface. The information that passes through it consists mainly of the succession of messages exchanged between the TMGR and the external world. They are encoded in SOAP and are processed by a Behavior Guardian that dispatches them to an Observer. This analyzes all outgoing and incoming messages, extracts relevant information and updates the synthetic image the TWEL has about the adaptive system under control.

This image is checked by a Controller against a set of constraints. When constraints are violated, the controller decides how to handle the liable messages and informs the intelligent application about the situation. The power these constraints possess relies on the fact that they are defined similarly to OCL constraints (Object Constraint Language) ([5]) on concepts in the KRMF but also benefit from the additional semantic specifications and the automatic inference. The TWEL is an IAFS-level component but retains most of the power that an application-specific guarding mechanism would provide.

5 The Status of the Implementation and Future Work

The modules' implementations have reached different maturity levels. KRMF, as the pivotal component, has been fully implemented. In order to enable the interactive planning needed within our architecture, the JSHOP2 Planner has been adapted and extended. Advanced work has been done to implement the Complex Request TMGR. This joins the Planner that holds an updated world definition with other vital components, such as the TMGR GUI and the Direct Request TMGR. Work is underway to implement the modules of the Trustworthiness Enforcement Layer.

6 Conclusions

We have proposed a generic architecture for trustworthy adaptive autonomous systems and we have exemplified our design choices on a demonstrative Travel Manager application. We have emphasized how certain reasoning patterns used in human cognition can be mapped onto components of our architecture and the function knowledge modeling has within such a system, especially to support the enforcement of trustworthy functional envelope.

Much of the value of our architecture lies within the generic design of its modules. Most of the knowledge about travel management is stored explicitly and discretely within each component, so it can be seen as a design-time parameter of the system, while the rest is learned. This greatly simplifies the reconfiguration required to reuse the system for planning within other domains than travel management.

Throughout the system we have attempted to keep balance between conceptual clarity and rigorous, well-motivated optimizations. Computing power is saved in comparison with classical approaches, e.g. through a more rigorous knowledge definition, which limits the scope and amount of inference that is needed. The available power is used to implement additional functionality, including the trustworthiness enforcement.

References

1. Guizzardi, G., Wagner, G., Guarino, N., van Sinderen, M.: An Ontologically Well-Founded Profile for UML Conceptual Models. In: Persson, A., Stirna, J. (eds.) CAiSE 2004. LNCS, vol. 3084, Springer, Heidelberg (2004)
2. Cranefield, S., Purvis, M.: UML as an ontology modelling language. In: Proceedings of the Workshop on Intelligent Information Integration. In: 16th International Joint Conference on Artificial Intelligence (IJCAI) (1999)
3. Cranefield, S.: UML and the Semantic Web. In: Proceedings of the International Semantic Web Working Symposium (SWWS) (2001)
4. Degen, W., Heller, B., Herre, H., Smith, B.: GOL: A General Ontological Language. In: Proceedings of the International Conference on Formal Ontology in Information Systems (FOIS) (2001)
5. Gorman, J.: UML for Java Developers, Model Constraints & the Object Constraint Language, <http://www.parlezuml.com>
6. d'Avila Garcez, A., Russo, A., Nuseibeh, B., Kramer, J.: Combining Abductive Reasoning and Inductive Learning to Evolve Requirements Specifications. In: IEEE Proceedings - Software (2003)
7. Mili, A., Jiang, G., Cukic, B., Liu, Y., Ben Ayed, R.: Towards the Verification and Validation of Online Learning Systems: General Framework and Applications. In: Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS) (2004)
8. Watkins, A., Berndt, D., Aebischer, B., Fisher, J., Johnson, L.: Breeding Software Test Cases for Complex Systems. In: Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS) (2004)
9. Ilghami, O.: Documentation for JSHOP2. In: Technical Report, CS-TR-4694, Department of Computer Science, University of Maryland (2006)

Cooperative Component Testing Architecture in Collaborating Network Environment

Gaeil An¹ and Joon S. Park²

¹ Electronics and Telecommunications Research Institute (ETRI),
161 Gajeong-Dong, Yuseong-Gu, Daejeon, 305-350, Korea
fogone@etri.re.kr

² The Laboratory for Applied Information Security Technology (LAIST), School of
Information Studies, Syracuse University, Syracuse, NY 13244-4100, USA
jspark@syr.edu

Abstract. In a large distributed enterprise multiple organizations may be involved in a collaborative effort to provide software components that they developed and maintain based on their own policies. When a local system downloads a component from a remote system into such an environment, the downloaded component should be checked to find if it contains internal failures or malicious codes before it is executed in the local system. Although the software was tested by the original developer in its local environment, we cannot simply assume that it will work correctly and safely in other organizations' computing environments. Furthermore, there is a possibility that some malicious codes are added to the original component by a mistake or intentionally. To address this problem, we propose a cooperative component-testing architecture that consists of three testing schemes, a provider node testing, a multiple-aspect testing, and a cooperative testing. The proposed architecture is able to effectively and efficiently detect malicious codes in a component. The provider node testing can increase the possibility of choosing the cleanest (least infected) component among components that exist on multiple remote systems. The multiple-aspect testing can improve the ability to detect a fault or malicious contents. And the cooperative testing scheme provides fast detection speed by integrating detection schemes effectively. Finally, we simulate our proposed ideas and provide a performance evaluation.

1 Introduction

In a collaborating network environment such as GRID [1] and P2P [2], an application may span more than one organization. Organizations can dynamically download components from other organizations. Single administration of components across the boundaries of organizations is not possible, so an autonomous administration should be employed.

When a component is exported from a remote system to a local system, we must check to see if the remote component has been altered in an unauthorized manner before the component is used in the local system. Although the software was tested by the original developer in its local environment, we cannot simply assume that it will

work correctly and safely in other organizations' computing environments [3,4,5,6]. Furthermore, there is a possibility that some malicious codes have been added to the original component by a mistake or intentionally. So, we need to check if the downloaded components contain malicious codes such as a Trojan Horse, Virus, Worm, Denial of Service (DoS) attack, backdoor, etc. [7,8,9]. To detect such malicious code in a component, the existing technologies such as pattern-matching-based attack detection, checksum-based attack detection, etc. can be used [10,11,12]. This paper focuses on how to enhance the performance of existing schemes in a collaborating network environment in terms of detection accuracy and detection speed.

In this paper, we propose a cooperative component-testing architecture which consists of three testing schemes, a provider node testing, a multiple-aspect testing, and a cooperative testing. The proposed architecture is able to effectively and efficiently detect malicious codes in a component. The provider node testing is used to increase the possibility of choosing the cleanest (least infected) component from the components that exist on multiple remote systems. The multiple-aspect testing is used to improve the ability to detect a fault or malicious contents. The cooperative testing scheme provides fast detection speed by integrating detection schemes effectively.

The rest of this paper is structured as follows. Section 2 introduces existing technologies for detecting failures or malicious codes in a downloaded component. The proposed schemes and their performance analyses are described in Sections 3 and 4, respectively. Section 5 summarizes this paper and presents future research direction.

2 Technologies for Component Testing

When a user downloads a component from a remote system, the component may include failures or malicious codes. Examples of malicious codes include Virus, Worm, Trojan horse, etc. A virus is a small program written without the permission or knowledge of the user to alter the way a computer operates. A worm is a computer program that can infect other computer programs, spreads via security vulnerabilities, and does not require any action by users. A Trojan horse contains malicious code that when triggered causes loss, or even theft, of data by using a back door for the attacker.

Many technologies have been proposed to detect such malicious codes in a component. There are three kinds of detection schemes [7,10,11,12]: signature-based, anomaly detection, and integrity-based. The signature-based detection scheme detects attacks based on a known attack pattern. There may be at least two ways in a signature-based detection scheme, as follow:

- (1) Simple pattern matching -- This way is based on a set of rules (strings) that describes characteristics of well-known malicious codes. If a downloaded component matches a defined string, it is regarded as a malicious component.
- (2) Smart pattern matching -- A smart attacker can make a mutator to make detection difficult by inserting junk instructions, such as do-nothing instructions, into the source files. To detect such mutators, this way skips instructions like NOP (No Operation) in a download component and does not store such instructions in the attack signature rule.

An integrity-based detection scheme detects malicious codes by checking changes to components based on their integrity. There may be three ways in integrity-based detection scheme, as follow:

- (1) Timestamp test -- This is based on the time interval between requesting a component and receiving the built component from the remote machine. If the time interval for downloading a component is greater than an acceptance threshold, the component is suspected changed during the communication process.
- (2) Checksum test -- This uses a checksum database created on the protected system to check whether a downloaded component has been changed.
- (3) Digital signature test -- This is based on a digital signature attached to the downloaded component. If the digital signature of a component is not verified, then the component is considered to have been changed.

An anomaly detection scheme collects information that describes the normal or abnormal state (i.e., behavior) of a component to protect, and then detects malicious codes based on the behavior, without regard to actual attack scenarios. There may be two ways in anomaly detection scheme, as follow:

- (1) Abnormal behavior test way -- This, first of all, builds an attack state model that describes a state for known attack behavior (e.g., trying to get access to the password file). The downloaded component is executed on a virtual machine for a security test before it is included in the main program of host computer. If a behavior that occurs in the middle of the execution of the downloaded component is in accord with the attack state model, then the downloaded program is considered to include malicious codes.
- (2) Normal behavior test way -- This uses a set of behaviors that describe the characteristics (i.e., expected behavior when executed) of normal components. If a downloaded component forms a poor match with the defined behaviors, then it is considered to include malicious codes.

As a technology for detecting failure codes in a downloaded component, there are several schemes. One of those approaches employs black-box testing of the components. In this technique, behavioral specification [13] is provided for the component to be tested in the target system. This technique treats the target component as a black box and can be used to determine how the component behaves anomalously. The main disadvantage of this technique is that the specifications should cover all the details of the visible behavior of the components, which is impractical in many situations.

Another approach employs a source-code analysis, which depends on the availability of the source codes of the components. Software testability analysis [14] employs a white-box testing technique that determines the locations in the component where a failure is likely to occur. Another approach is software component dependability assessment [15], a modification or testability analysis which thoroughly tests each component. These techniques are possible only when the source code of the components is available.

N-Version Programming (NVP) is a well known concept in fault tolerant systems. The NVP was proposed by [16,17] for providing fault tolerance in software. N-Version techniques are used to ensure the reliability of a system by having multiple and different, yet functionally equivalent, implementations of critical software to ensure reliability of software systems.

3 Cooperative Component-Testing Architecture

In this section, we propose a cooperative component-testing architecture, which consists of three test schemes: a provider node testing, a multiple-aspect testing, and a cooperative testing.

3.1 Provider Node Testing Scheme

In this paper, we propose a provider node testing scheme that is able to increase the possibility of choosing the cleanest (least infected) component among components that exist on multiple remote systems. Provider node-testing is a way to filter potentially malicious contents even before the component consumer node proceeds to test for such contents. The consumer node that employs the provider node testing downloads from remote component provider nodes multiple copies of each component required by the consumer node and checks for coherence in their contents by comparing certain parameters. Thus, for every component N , we have several downloads, say $N1$, $N2$, $N3$ and $N4$. Each of these components is checked for parameters such as component size, last modified time, the number of specific characters contained in each component, etc., and a comparison is made between each of them to check for coherence or matches in the values for each of the mentioned parameters. Now, for instance, if component $N4$ does not match the properties of the other downloaded component copies $N1$, $N2$ and $N3$, since the latter forms a majority we can discard $N4$ by assuming that it may be a component whose contents may have been modified over the network during its transit.

Even if this technology is very simple, it is able to improve detection accuracy by making a consumer node avoid downloading a malicious component, especially a component with unknown malicious codes. It is extremely important to note that although the valid components are accepted in this case, they are far from being allowed to execute in the consumer node, because this technique does nothing to establish the validity of the provider node.

3.2 Multiple-Aspect Testing Scheme

In this paper, we propose a multiple-aspect testing approach to achieve more of such accuracy in detection of failures or malicious codes in a component downloaded from a remote provider node. The multiple-aspect testing consists of the two levels of testing that a consumer node would need to conduct, i.e., N -type and N -way testing as shown in Fig. 1.

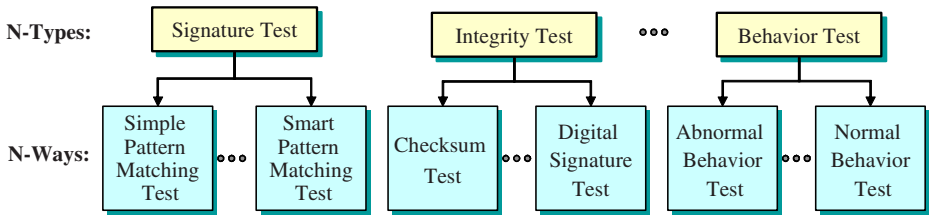


Fig. 1. Multiple-aspect testing

N-type refers to the type of tests that are used to detect failures or malicious codes in a downloaded component. N-type testing can be defined by a set of types $T = \{t_1, t_2, t_3 \dots t_n\}$ where each of the elements of T can be one of a signature-based test type, an integrity-based test type, a behavior-based test type, and so on. For each type of test, there can be several test ways that have different but conceptually equivalent implementations. For example, an integrity type testing may have a checksum test way, a digital signature test way, etc. Thus, we have N-way testing for each type under consideration. This implies that for every type we have a set of types $W = \{w_1, w_2, w_3 \dots w_n\}$ where each of the elements in W is an implementation way that belongs to that type. By using N-way testing, we have a greater chance of detecting a fault or malicious contents in downloaded components.

N-way testing is different from the N-version technique which are used to ensure the reliability of a system by having multiple and different yet functionally equivalent implementations of the critical software to ensure reliability of the system. N-way testing, on the other hand, is a way of increasing the possibility of detecting faults or malicious codes in a component by having as many ways as possible to conduct the test on the target component.

The above test mechanisms are structured in a way that allows them to provide more types and ways of testing, depending on the intensity of the threats and the criticality of the system to be protected. Thus, for a given number of types T and ways W , the total number of test modules plugged into the consumer node would be $T * W$. Thus if the criticality of the system is high, then the number of tests to be performed on the downloaded components would have to be higher to ensure higher reliability and would thus be a factor proportional to $T * W$.

3.3 Cooperative Testing Scheme

Even if the multiple-aspect testing scheme proposed in this paper has a merit that can significantly improve ability to detect a fault or malicious contents, it may have a performance overhead problem (i.e., poor detection speed) because it performs several tests to test one component.

In this paper, we propose a cooperative testing scheme to address the detection speed problem. In the cooperative testing scheme, the test ways cooperate with each other to provide faster detection capability. Before explaining the cooperative scheme, however, we introduce a Trigger Decision Gate (TDG) developed to express relations among test ways.

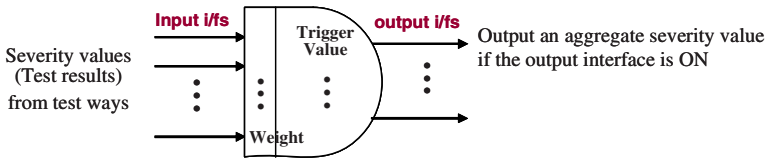


Fig. 2. Notation of Trigger Decision Gate (TDG)

Fig. 2 shows the notation of TDG. TDG collects test results (i.e., severity values) through input interfaces from test ways, and then outputs an aggregate severity value to the corresponding output interface. In Fig. 2, weight means a priority value assigned to the corresponding input interface. Trigger value in TDG indicates a minimum value for turning the corresponding output interface on. The output interface is triggered as follows:

$$\text{OutputInterface}_k = \begin{cases} \text{ON, if } \left(\text{TriggerValue}_k \leq \sum_{i=1}^{\text{input i/f \#}} \text{SeverityValue}_i < \text{TriggerValue}_{k+1} \right) \\ \text{where, } \text{TriggerValue}_k < \text{TriggerValue}_{k+1} \\ \text{Off, otherwise} \end{cases}$$

A cooperative test scheme is a kind of heuristic-based technique because we design a cooperative architecture through the analysis of the characteristics (i.e., merit and demerit) of each test type and way. For example, the signature test type generally has a merit that is able to detect malicious codes fast, but a demerit that cannot detect unknown malicious codes. On the other hand, the behavior test type can detect unknown malicious codes, but is very slow in the detection speed. The integrity test type is able to detect malicious codes done by an illegal attacker more accurately than any other test way, but has difficulty in detecting not only malicious codes (e.g., back-door) intentionally written by a legitimate but malicious user, but also failure codes accidentally written by a legitimate user.

Fig. 3 shows a cooperative test architecture based on existing schemes (i.e., pattern matching-based detection scheme, integrity-based detection scheme, and anomaly

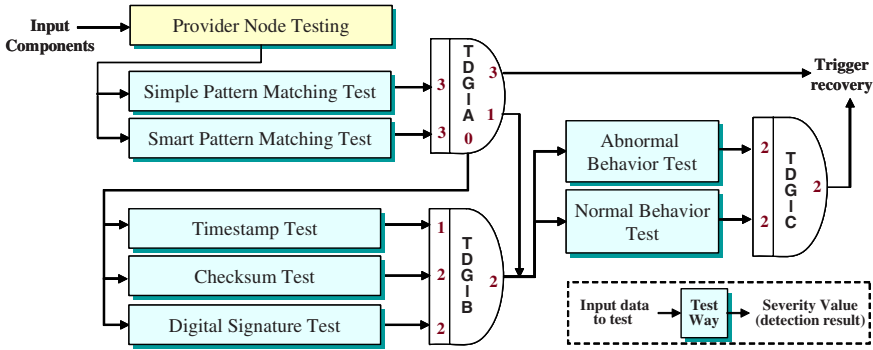


Fig. 3. Cooperative Test Architecture

detection scheme) and the two schemes (i.e., provider node-testing scheme and multiple-aspect testing scheme) proposed in this paper, which is able to enhance test performance in terms of detection accuracy and detection speed. The cooperative architecture operates as follows. When a consumer node receives several components, it first performs provider node testing to choose the cleanest component among those components. And then, the consumer tests the component by using the simple pattern matching test way and smart pattern matching test. If either of the two tests detects malicious code (i.e., if aggregate severity value is greater than or equal to 3, in TGD-A), then a recovery mechanism is triggered to recover the infected component. In this paper, we do not address the recovery mechanism. Else if either of the two test ways find something suspected as malicious code (i.e., if aggregate severity value is greater than or equal to 1, in TGD-A) in the component, then it triggers performing an abnormal behavior test way and a normal behavior test. Otherwise, it triggers the executing of three kinds of test ways: timestamp test, checksum test, and digital signature test. If the aggregate severity value outputted from the three tests is greater or equal to 2, then TDG-B performs the abnormal behavior test and the normal behavior test. Finally, if the aggregate severity value outputted from those two test ways is greater or equal to 2, then it triggers the recovery mechanism.

4 Performance Evaluation

In this section, we analyze and evaluate the performance of the provider node-testing scheme, the multiple-aspect testing, and the cooperative testing in terms of detection accuracy and speed. For this, we have implemented a component-sharing environment by using Network Simulator (NS) [18].

4.1 Simulation Environment

Fig. 4 shows a simulated network architecture for component testing. The architecture consists of 5 parts: component storage, Component Provider (CP) nodes, and a Component Consumer (CC) node, component test modules, and Integrity information DB.

There are four kinds of components in the component storage, a normal component (C1), a component with a well-known Trojan horse (C2), a component with a mutative Trojan horse (C3), and a component with an unknown Trojan horse (C4). Note that all the components are the same in that they provide identical functions, but are different from each other in that C1 is a normal component, while C2, C3, and C4 each are infected with a different type of Trojan horse. CP nodes (i.e., CP1, CP2, CP3, and CP4) each randomly select one among the four types of components and send it to the CC node whenever it receives a request from the CC node. If the CC node downloads components from CP nodes, it calls the component test modules to test them. The component test modules mean component test schemes introduced in this paper. In this experiment, we have implemented the existing three kinds of test ways, Pattern-Matching-based (PM) test, CheckSum-based (CS) test, and Abnormal Behavior-based (AB) test, and the three schemes proposed in this paper, provider node testing, multiple-aspect testing, and cooperative testing. The component test modules are able to detect the C2 or C3 type of malicious components. But, any of them can not detect the C4 type of components. The integrity information DB is used by the CS test way to get integrity information for a download component.

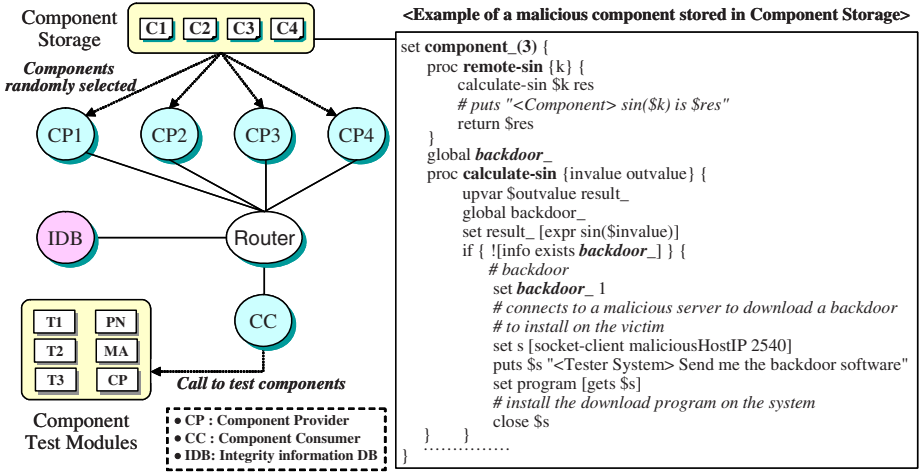


Fig. 4. Simulated network architecture for component testing

Fig. 4 also shows an example of a malicious component with a mutative Trojan horse (i.e., C3 type), which is stored in the component storage. The component is written in the Tcl language. The code of the component provides a sin function (i.e., *remote-sin* in Fig. 4), but includes a backdoor (i.e., in *calculate-sin*). Whenever the *remote-sin* function is called, it calls the *calculate-sin* function. Once the *calculate-sin* function is called, it calculates the sin value for the input value, and then checks what the value of a variable, *backdoor_*, is. If the value of *backdoor_* is not 1, then the *calculate-sin* function connects to a malicious server to download a backdoor program and install it on its system.

We have introduced the existing three kinds of test ways, PM, CS, and AB test way. The PM test uses attack signature to detect malicious components. So, in the example of Fig. 4, if the PM test has “*backdoor_*” as an attack signature, it will succeed in detecting the backdoor because the component defines *backdoor_* as a variable for the backdoor. On the other hand, the AB test detects malicious code by monitoring abnormal behavior during the execution of the component. So, if the AB test regards what the component connects to an external system as abnormal behavior, it will detect the backdoor. Finally, the CS test detects malicious code by comparing the checksum calculated from the component with the checksum downloaded from IDB, irrespective of the contents of the component. If the value of both checksums is different from each other, the CS test regards the component as a malicious component. In this experiment, the PM test does not have “*backdoor_*” as an attack signature, but the AB test defines what a component connects to an external system as abnormal behavior. So, the AB test is able to detect the backdoor in the component shown in Fig. 4.

4.2 Analysis of Simulation Results

In this simulation, there are three kinds of users: attacker, malicious user, and normal user. An attacker is one who makes and distributes a malicious component illegally

without permission. On the other hand, a malicious user means one who is a legitimate user but inserts malicious codes (e.g., a Trojan horse or a backdoor) into a normal component by accident or intentionally.

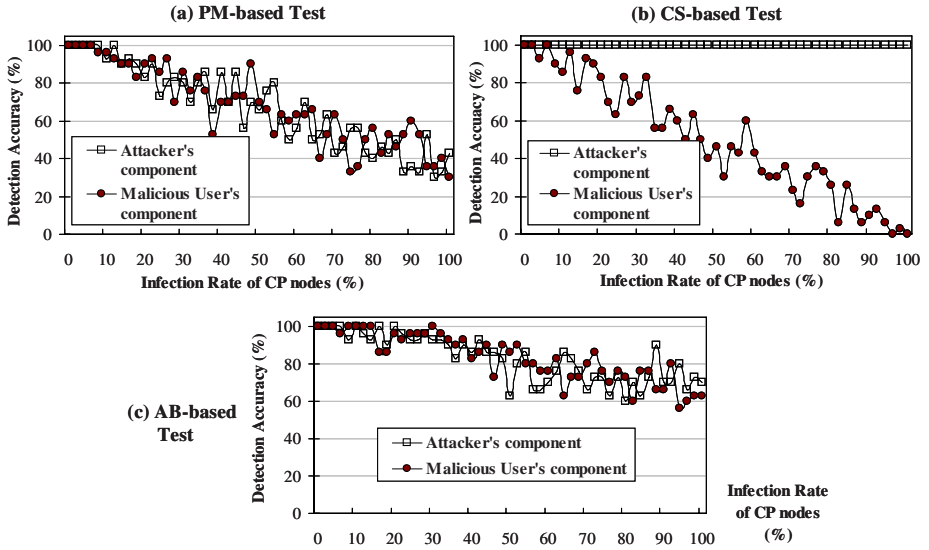


Fig. 5. Performance of Trojan horse detection in the existing component testing schemes: In this graph, an attacker is one who makes and distributes a component illegally without permission. On the other hand, a malicious user means one who is a legitimate user but makes a malicious component. The X axis in this graph, the infection rate of the CP (Component Provider) nodes indicates the rate of CP nodes infected by attackers and malicious users.

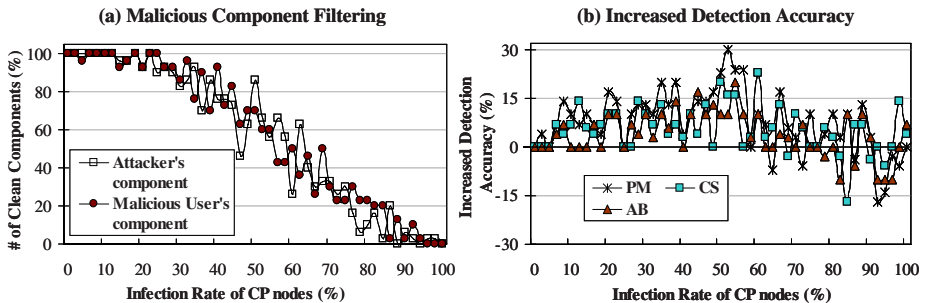


Fig. 6. Performance of Trojan horse detection in Provider Node-testing scheme

Fig. 5 shows the performance of Trojan horse detection in the existing component testing schemes. Fig. 5-(a), (b), and (c) are the performances of PM (Pattern-Matching) way, CS (Check-Sum), and AB (Abnormal Behavior), respectively. The PM way and AB way decrease in detection accuracy in proportion to the infection rate of the Component Provider (CP) nodes because the probability to download the

C4 type of components is proportional to the infection rate of the CP nodes, as shown in Fig. 5. On the other hand, the CS way has a great advantage in detecting malicious components made by attackers, but is very poor at detecting malicious components made by malicious users. This is because attackers have no right to generate checksum for components that they create illegally, whereas malicious users are legitimate users who are allowed to generate it. The performance of each test scheme shown in Fig. 5 has meaning only in itself and no concern with that of the other schemes. So, Fig. 5 does not mean that the AB way is better than the PM way in detection ability.

Fig. 6 shows the performance of Trojan horse detection in the provider node-testing scheme proposed in this paper. The provider node-testing scheme is used to increase the possibility of choosing the cleanest (least infected) component among components that exist on CP nodes by filtering malicious components as shown in Fig. 6-(a). Fig. 6-(b) shows the performance of Trojan horse detection when the existing schemes employ the provider node testing scheme. The provider node-testing scheme improves detection accuracy for all existing schemes, as shown in Fig. 6-(b). The provider node-testing scheme is very effective until the infection rate of the CP nodes is less than 60%, as shown in Fig. 6-(b).

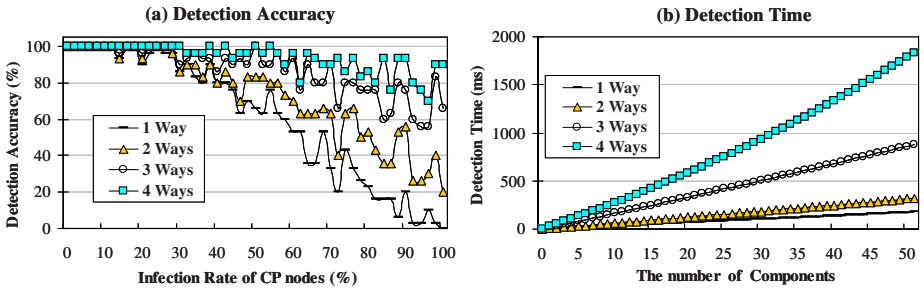


Fig. 7. Performance of Trojan horse detection in Multiple-Aspect testing scheme: In this experiment, 1 way indicates provider node (PN) testing scheme, 2 ways PN+PM, 3 ways PN+PM+CS, and 4 ways PN+PM+CS+AB

Fig. 7 shows the performance of Trojan horse detection in a multiple-aspect testing scheme. As shown in Fig. 7-(a), the multiple-aspect testing gains significantly in accuracy of testing over one-way techniques used for detecting malicious content. The detection accuracy in multiple-aspect testing is directly proportional to the number of test ways. This shows that the multiple-aspect testing scheme proposed in this paper provides a dramatically high precision of malicious code detection in a downloaded component. However, the multiple-aspect testing scheme has a performance overhead problem, because it uses several test ways to test one component. As shown in Fig. 7-(b), the detection time for the multiple-aspect testing is not good.

We have proposed the cooperative testing scheme to address the detection speed problem of the multiple-aspect testing scheme. The cooperative testing scheme can provide faster detection capability by making the test ways cooperate with each other without much impact on its system. Fig. 8 shows that the cooperative testing scheme can detect attacks as accurately as 4-ways, as shown in Fig. 8-(a), while its attack detection time is even less than that of the 4 ways, as shown in Fig. 8-(b).

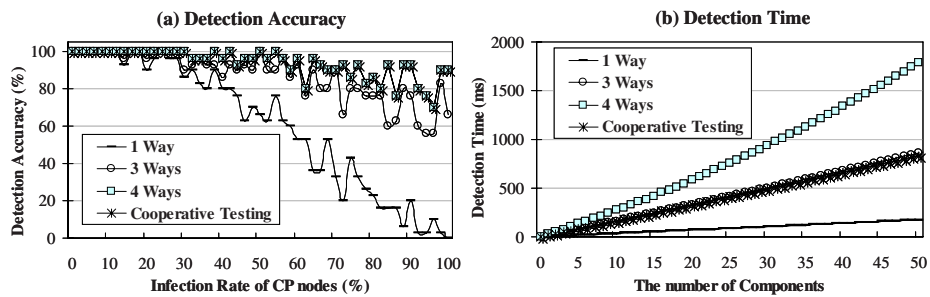


Fig. 8. Performance of Trojan horse detection in Cooperative testing scheme

Table 1. Comparison between schemes proposed in this paper

Performance Schemes	Provider Node (PN) Test	Multiple-aspect Test	Cooperative Test
Attack Detection Time	-	Increase (Bad)	Decrease (Good)
Attack Detection Precision	Increase (Good)	Increase (Good)	Increase (Good)

Table 1 shows a comparison based on the simulation results between schemes proposed in this paper.

5 Conclusion

When a component is exported from a remote system to a local system, we must check to find if the remote component has been altered in an unauthorized manner, especially if it contains malicious codes, before the component performs in the local system. This paper focuses on how to enhance the performance of existing schemes in a collaborating network environment in terms of detection accuracy and detection speed.

In this paper, we have proposed a cooperative component-testing architecture that consists of three testing schemes: provider node testing, multiple-aspect testing, and cooperative testing. The proposed architecture is able to effectively and efficiently detect malicious codes in a component. We also implemented a prototype collaboration network environment to evaluate the performances of those schemes proposed in this paper in terms of detection accuracy and detection speed. The simulation results show that provider node testing can increase the possibility of choosing the least infected component among components that exist on multiple remote systems, that the multiple-aspect testing can improve ability to detect a fault or malicious contents, and that the cooperative testing scheme can provide fast detection.

Currently, we have been implementing our schemes on real systems in Microsoft's .Net environment. Our future work will experiment our ideas using real data on real systems.

References

1. Shum, S.B., De Roure, D., Eisenstadt, M., Shadbolt, N., Tate, A.: CoAKTinG: Collaborative Advanced Knowledge Technologies in the Grid. In: Proc. of the IEEE International Symposium on High Performance Distributed Computing (HPDC) (2002)
2. Risson, J., Moors, T.: Survey of Research towards Robust Peer-to-Peer Networks: Search Methods. Internet Research Task Force (IRTF) draft-irtf-p2prg-survey-search-00.txt (2006)
3. Chen, M., Kiciman, E., Brewer, E., Fox, A.: Pinpoint: Problem Determination in Large, Dynamic Internet Services. In: Proc. of the IEEE International Conference on Dependable Systems and Networks (DSN) (2002)
4. Park, J.S., Suresh, A.T., An, G., Giordano, J.: A framework of multiple-aspect component-testing for trusted collaboration in mission-critical systems. In: Proc. of the IEEE Workshop on Trusted Collaboration (TrustCol) (2006)
5. Park, J.S., Chandramohan, P., Suresh, A.T., Giordano, J.: Component survivability for mission-critical distributed systems. *Journal of Automatic and Trusted Computing (JoATC)* (in press)
6. Park, J.S., Giordano, J.: Software component survivability in information warfare. In: *Encyclopedia of Information Warfare and Cyber Terrorism*, IDEA Group Publishing (in press)
7. Szo, P.: *The Art of Computer Virus Research and Defense*. Addison-Wesley Publishing, London (2005)
8. Kienzie, D.M., Elder, M.C.: Recent Worms: A Survey and Trends. In: Proc. of the ACM Workshop on Rapid Malcode (WORM) (2003)
9. Milenkovic, M., Milenkovic, A., Jovanov, E.: Using Instruction Block Signatures to Counter Code Injection Attacks. *Computer Architecture News* 33(1), 108–117 (2005)
10. Almgren, M., Barse, E.L., Jonsson, E.: Consolidation and Evaluation of IDS Taxonomies. In: Proc. of the Nordic Workshop on Secure IT-systems (NordSec), pp. 57–70 (2003)
11. Axelsson, S.: *Intrusion Detection Systems: A Survey and Taxonomy*. Technical Report 99-15, Depart. of Computer Engineering, Chalmers University (2000)
12. Hansman, S., Hunt, R.: A taxonomy of network and computer attacks. *Int. Journal of Computers and Security* 24(1), 31–43 (2005)
13. Abadi, M., Lamport, L.: Composing specifications. *ACM Transactions on Programming Languages and Systems* 15(1), 73–132 (1993)
14. Voas, J. M., Miller, K. W., Payne, J.: PISCES: A tool for predicting software testability. Technical Report, NASA (1992)
15. Voas, J.M., Payne, J.: Dependability certification of software components. *Journal of Systems and Software* 52(2-3), 165–172 (2000)
16. Chen, L., Avizienis, A.: N-version programming: a fault-tolerance approach to reliability of software operation. In: *Digest of the 8th International Conference on Dependable Systems and Networks (FTCS)*, pp. 3–9 (1978)
17. Cai, X., Lyu, M.R., Vouk, M.A.: An experimental evaluation on reliability features of n-version programming. In: Proc. of the 16th IEEE International Symposium on Software Reliability Engineering, pp. 161–170 (2005)
18. UCB/LBNL/VINT: Network simulator (ns) Notes and Documentation.
<http://www.isi.edu/nsnam/ns>

An Approach to a Trustworthy System Architecture Using Virtualization

Frederic Stumpf*, Michael Benz**, Martin Hermanowski, and Claudia Eckert

Department of Computer Science
Darmstadt University of Technology
Darmstadt, Germany

{stumpf,benz,hermanowski,eckert}@sec.informatik.tu-darmstadt.de

Abstract. We present a system architecture for trusted transactions in highly sensitive environments. This architecture takes advantage of techniques provided by the Trusted Computing Group (TCG) to attest the system state of the communication partners, to guarantee that the system is free of malware and that its software has not been tampered with. To achieve meaningful attestation, virtualization is used to establish several different execution environments. The attestation process is limited to a fragment of the software running on the platform, more specifically, to the part requesting access to sensitive data. The Trusted Platform Module (TPM) is virtualized, in order to make it accessible for an execution environment with a higher trust level.

1 Introduction

The complexity of emerging computer systems is rising rapidly, leading to an increase in system vulnerabilities. These vulnerabilities can be exploited to inject malicious code, which then influences other software components running in parallel. Additionally, a system free of viruses, worms and other malware is an important prerequisite for handling sensitive and personal information. Applications whose security is critical, e.g., online banking applications, are highly fragile, due to the vast number of possible attacks. Subversive programs could spy on users actions, passwords, credit card information, bids in an auction or other sensitive data by eavesdropping and replaying the recorded data in later sessions.

The introduction of Trusted Computing techniques offers a sound approach to deal with these shortcomings. The Trusted Platform Module (TPM) specified by the Trusted Computing Group (TCG) provides important functions for establishing IT security. Besides having a secure storage mechanism for storing cryptographic keys and other confidential data, the TPM also offers the possibility to attest the configuration of the local platform to a remote entity. This

* The author is supported by the German Research Foundation (DFG) under grant EC 163/4-1, project *TrustCaps*.

** The author is supported by the German Research Foundation (DFG) under grant EC 163/5-1, project *QuaP2P*.

process is called *Remote Attestation* (RA). To attest the platform configuration to a remote entity, the trusted platform module measures every code fragment before execution and stores the resulting values in a protected and shielded location. However, the process of deciding whether a platform is trustworthy or not can become quite complex, especially when several software components which have to be measured are involved. Since all processes running on the machine can mutually influence each another [1], they all need to be trustworthy. We argue that this complexity can be reduced by establishing multiple isolated execution environments with different trust levels. These separate environments are created by several different virtual machines (VM) running on a single system. This gives us the ability to keep certain processes separate, which in turn leads to a stronger degree of isolation. Multiple trusted VMs are used to execute highly sensitive code, such as transaction clients. Like Terra [2], we use different types of VMs: A *trusted VM* that is responsible for running highly sensitive code, an *open VM* for running arbitrary software components and a *management VM*, which is responsible for spawning new VMs. We extend this approach by utilizing mechanisms provided by the TCG to establish a trusted computing base.

2 Background

We will now give some background information that is important to understand our approach.

2.1 Trusted Computing

The core of the TCG mechanisms [3] is the Trusted Platform Module (TPM), which is basically a smart card soldered to the mainboard of a PC. The TPM serves as the *root of trust*. Tampering with the TPM is generally difficult, since it is implemented in hardware and uses non-migratable keys for certain cryptographic functions. The TPM can create and store cryptographic keys, both symmetric and asymmetric. These keys are marked either migratable or non-migratable, which is determined the moment the key is generated. In contrast to non-migratable keys, migratable keys can be transferred to another TPM.

In the context of this paper, the Platform Configuration Registers (PCRs) are of particular interest. These registers are initialized on startup and then used to store the software integrity values. Before execution, a software component is measured by the TPM and the corresponding hash-value is written to a specific PCR by combining the current result with the previous value of the PCR. The following cryptographic function is used to calculate the values for the specific registers:

$$\text{Extend}(\text{PCR}_N, \text{value}) = \text{SHA1}(\text{PCR}_N || \text{value}) \quad (1)$$

SHA1 refers to the cryptographic hash function used by the TPM and $||$ denotes a concatenation. The trust anchor for a so-called trust-chain is the *Core Root of Trust Measurement* (CRTM), which resides in the BIOS and is the first process

executed when a platform is powered up. The CRTM measures itself and the BIOS, and hands over control to the next software component in the trust-chain. For every measured component, an event is created and stored in the stored measurement log (SML). The PCR values, combined with the SML, can then be used to attest the platform's state to a remote party. In order to guarantee the authenticity of these values, they are signed with a non-migratable TPM signing key, namely the Attestation Identity Key (AIK). The AIK is generated on a TPM and certified using either a Privacy-CA or the Direct Anonymous Attestation [4]. With the obtained certificate, a remote platform can compare the signed values with reference values, to check whether the platform is in a trustworthy state or not.

2.2 Virtualization

Virtualization allows the execution of several different VMs, with different operating systems, on a single host entity by the introduction of a hypervisor called virtual machine monitor (VMM) [5]. This hypervisor is responsible for providing an abstract interface to the hardware and for partitioning the underlying hardware resources. The underlying resources are available to the VMs through the VMM, which maintains full control of the resources given to the VM. The main contribution of this technology is the provision of strong isolation between different VMs, established by the VMM. Additionally, the hardware abstraction presented to the VM can differ from the one physically available.

3 Related Work

Terra [2] also introduces an approach to create a virtual machine-based platform, which allows the attestation of VMs. It utilizes VMWare's ESX Server to establish different types of VMs (*Closed Box* and *Open Box*) and to report the state of a closed box machine to a remote entity. The difference to our approach is that Terra uses neither a hardware-based trust anchor (the TPM in our approach), nor allows attestation without the direct involvement of the VMM. Using a virtualized TPM, in contrast allows, on the one hand, direct attestation and the reporting of fine-grained platform configurations and, on the other hand, the full functionalities of a trusted platform module, e.g., sealing. Terra also suffers from using a large trusted computing base, which is a potential security threat, and is therefore not applicable in high security environments.

A similar approach that enables attestation is used in the Integrity Measurement Architecture (IMA) [6]. The authors present a comprehensive prototype based on trusted computing technologies, where integrity measurement is implemented by examining the hashes of all executed binaries. However, the prototype is not based on virtualization technologies, and therefore no strong isolation between processes is achieved. This results in the necessity of transferring a complete SML to the remote entity, which in turn, needs to validate all started processes to determine the platform's trustworthiness.

Berger et al. [7] illustrate how to virtualize a TPM and present a driver-pair that utilizes these concepts. Our work differs in the attestation process of VMs and in the way the mapping between the PCRs is performed. In contrast to [7], where the IMA is used for providing measurements, we argue that a complete verification is reached only by measuring the image of a VM. Furthermore, Berger et al. do not specify how the binding between a virtualized TPM (vTPM) and a TPM is performed.

Much research has also been done in the Microkernel area such as, L4 [8] or Exokernel [9], where both approaches provide strong isolation of processes. In this context, *EMSCB* [10] aims at developing a trustworthy computing platform that solves many of the security problems of conventional systems. In contrast to our hypervisor approach, where every VM runs its own security critical processes, the critical processes in *EMSCB* are directly executed by the Microkernel, therefore forming a smaller trusted computing base.

4 Architecture Overview

Our distributed transaction architecture consists of several client platforms and at least one trusted third party (TTP). The trusted third party supervises the transactions between the client platforms. They implement a virtualization-based environment running the architecture we are about to explain. The trusted third party is responsible for distributing the trusted transaction software to the clients and for validating the trustworthiness of the client platforms.

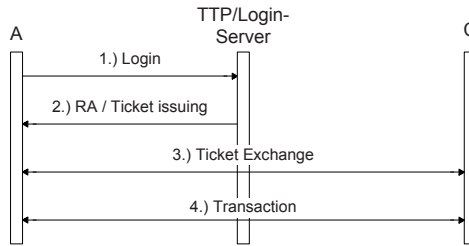


Fig. 1. Model of Transaction

Figure 1 shows our model of transaction, involving the three different parties. When the client connects to the platform provided by the trusted third party (Step 1), his platform's trustworthiness is verified using the ticket-based remote attestation (Step 2). If his platform is deemed trustworthy and the ticket has been issued, he can obtain access to services offered by other clients by presenting his ticket as a credential (Step 3). The issued ticket is cryptographically bound to the validated platform to prevent replay attacks, as explained in Section 6.

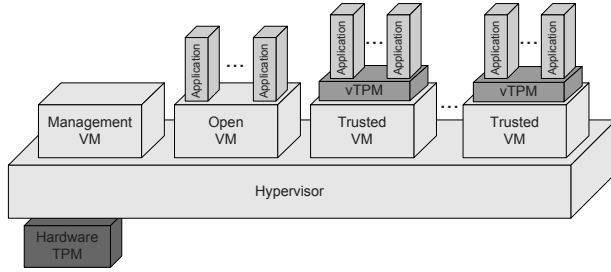


Fig. 2. Client Architecture

5 Client Architecture

The use of virtualization concepts reduces the complexity of the attestation process considerably. Instead of attesting the whole platform configuration, including all processes running on the machine, we only attest processes that are required for trustworthy operations. Processes that are not responsible for establishing communication with a remote entity, and therefore not required for the transaction itself, should neither be included in the integrity measurement nor be able to influence the transaction.

Additionally, transferring the whole platform configuration to a remote party raises severe privacy concerns. The remote party is able to discover the full platform configuration, including all running processes, by simply examining the SML. This information could be used to collect extensive information about the platform in question and transfer the gathered information to malicious entities. Moreover, market dominant vendors could introduce attestation techniques which deny access to their services, if a client runs a competitor's software in parallel.

Figure 2 depicts our client architecture, consisting of three different types of virtual machines, a hypervisor for partitioning the underlying hardware and a hardware-based trust anchor. The *trusted virtual machine monitor* (TVMM), or hypervisor, forms the foundation for our transaction structure by providing an abstraction layer to the underlying hardware. This trusted VMM has privileged access to the hardware and is able to grant and revoke resources to and from the running VMs (e.g., CPU scheduling). Because of its privileged position, this VMM needs to be trustworthy, since it is able to inspect every single CPU cycle of each virtual machine. We assume that the VMM is trustworthy and therefore reliably provides the properties of a VMM. Unfortunately, currently available virtualization solutions do not allow secure sharing, particularly with DMA operations. VMMs that utilize secure sharing [11] tend to suffer from high performance overhead, as well as large trusted computing bases, because the required I/O emulation is moved into the hypervisor layer.

The VMM establishes several different execution environments by using various types of VMs, which are strongly isolated from one another. It also provides

an abstraction of the underlying hardware TPM through a virtualized TPM interface. This virtual TPM (vTPM) is mainly used by the *trusted virtual machine* (TVM), which, in turn, is used for trustworthy transactions with a remote entity. The TVM handles private and sensitive data and runs a tiny OS with a minimal number of processes, which reduces the possible number of security vulnerabilities considerably. It is measured by a measuring process before startup, by the hardware hashing engine of the TPM. These values are then accessible inside the virtual machine by adding them to the PCRs of the vTPM. For attestation, the trusted VM transmits those values to a remote entity by accessing them through the vTPM instance.

The *open virtual machine* is allowed to run arbitrary software components and provides the semantics of today's open machines. It runs applications with a lower trust level, such as web-browsers or office applications. Since this VM is not of interest for our approach, we will not focus on it in the rest of our work. The *management virtual machine* is responsible for starting, stopping and configuring the VMs. It is closely connected to the VMM, since it is a privileged VM and has direct access to the hardware TPM.

5.1 Trusted Virtual Machine Monitor

The VMM is a software layer which offers an interface to the VMs currently running on the platform. In general, the only purpose of a VMM is to partition the underlying hardware and to provide an interface for the generic operating systems running in the individual VMs. The properties of a VMM have been extensively studied [12], providing isolation, efficiency, compatibility and simplicity. Our TVMM offers the following additional functionalities:

Attestation. The virtual machine is able to access the underlying Trusted Platform Module without modifying the state of the TPM, since this would affect other virtual machines running in parallel. This includes functions that report the local system state to a remote party which authenticates the local system state. This, in turn, allows a remote entity to place trust in the system configuration of a communication partner.

Integrity Measurement. The VMM supports integrity measurement facilities to provide an explicit statement about the configuration of the virtual machine.

Attestation Completeness. Attesting the full image of a virtual machine allows a complete attestation of all software components, including all started processes, configuration files, kernel modules, and scripts.

5.2 Trusted Virtual Machine

The TVM acts as a container to run sensitive software processes. The trusted third party is used to obtain a virtual appliance in which the sensitive software application is preinstalled. Each time a new trusted VM is created, a virtual TPM

instance is initiated and the PCRs 0-15 are filled with the PCR values from the underlying hardware TPM. Additionally, the hash value of the measured image is stored in the virtual TPM's PCR 16. To further reduce vulnerabilities, the TVM is stateless, which means that any modifications in the guest system can not be written back to the disk image. This approach has the following advantage: The client software only has one single valid system state, which in turn reduces attestation complexity. The virtual machine is also equipped with a secondary disk image which can be used to store transferred data and information about former transactions. This data can be protected by *sealing* it to the vTPM. It should be noted that data stored on the disk image may be able to influence the runtime condition, by injecting malicious code. We suggest storing only a small amount of data on the secondary disk image and performing consistency checks before accessing it.

5.3 Binding vTPM-Instances to the Hardware TPM

A vital condition for placing trust in a remote entity is the establishment of a complete trust chain from the hardware-based trust anchor up to and including the end application. This includes all measurements performed by the hardware TPM as specified by the TCG, the bootloader and the hypervisor by using TrustedGrub [13] and the VM instances, including the virtual appliances. The hardware TPM is virtualized by providing a software TPM for every VM instance. The software TPM is always protected by the hardware TPM, to enable storing of persistent data. When a virtual TPM spawns, its PCR values are initialized with values from the underlying hardware TPM, as shown in figure 3.

PCR	Content of TPM	Content of vTPM
0..7	Measurmt. of CRTM and BIOS	Measurmt. of CRTM and BIOS
8..15	Measurmt. of bootloader and TVMM	Measurmt. of bootloader and TVMM
16	empty	Measurmt. of the virtual appliance
17..	empty	for free use

Fig. 3. Mapping of the PCR values

In order to access the underlying hardware TPM through the VM, a strong binding between vTPM and TPM must exist. Otherwise, it would be possible for the vTPM to report PCR values to a remote entity which are different from the ones that were measured by the underlying hardware TPM. Berger et al. [7] have already proposed three different solutions for this problem. After careful considerations, we decided to employ the solution in which each virtual AIK is bound to a hardware AIK. We believe that this solution has several advantages over the others. On the one hand, it has strong similarities to the concepts provided by the TCG specifications as an additional Privacy-CA, and on the other hand, it allows quick spawning of additional vTPM instances.

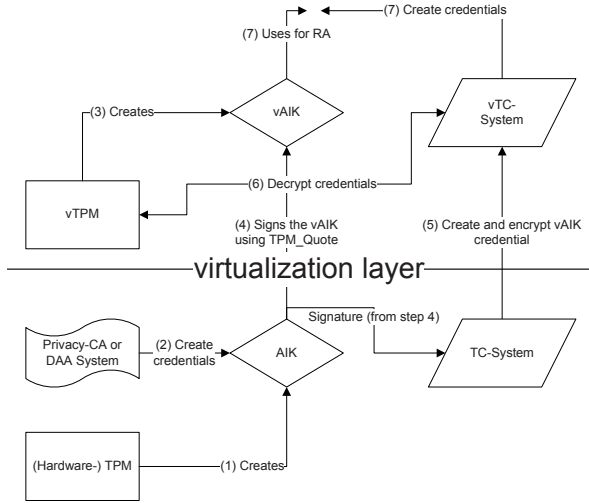


Fig. 4. Binding the vTPM to TPM

Figure 4 shows the association of a vTPM with an underlying hardware TPM, as realized by our architecture. The host TPM creates an AIK (1) and retrieves a corresponding credential through a Privacy-CA or via DAA [4] (2). The vTPM then creates a vAIK (3), which is signed by the host TPM using his own AIK (4). To prevent replay and masquerading attacks, a *nonce* provided by the vTPM instance, an additional *timestamp*, and the hash values of the hardware TPM’s PCRs 0-15 are embedded into this signed message. The integration of the PCR value in the vAIK credential is necessary, since a trustworthy software configuration could otherwise be forged. The corresponding credential is then encrypted using the vEK and sent to the vTPM instance (5), which is in possession of vEK^{-1} and therefore able to decrypt this message. A vAIK credential consists of:

- *Timestamp*
 - Hardware PCRs 0-15
 - *Nonce*
 - vAIK
 - AIK credentials
- } signed with AIK

For the remote attestation process, the vAIK is used to perform the *TPM_Quote* command. The verifying party can decide whether the remote platform configuration is trustworthy by validating the vAIK credential and the transmitted output of the *TPM_Quote* command. A platform configuration is deemed trustworthy if the following conditions are satisfied:

- The vAIK is authentic and generated by a vTPM
- The vTPM is authentic and protected by a hardware TPM
- The hardware TPM is authentic

- The hypervisor including management VM is in a trustworthy system state
- The TVM is in a trustworthy system state

6 Ticket-Based Attestation

We use a ticket-based remote attestation scheme to verify the platform integrity, thereby validating whether or not a communication partner's platform is trustworthy. It would also be possible to use a direct attestation process initiated by the end-communication partner. However, the ticket-based attestation offers several advantages over direct attestation.

Privacy issues. For transactions, it is not required that a service provider knows exactly which software version is used by the client requesting a service. The only necessary information is the trustworthiness of the client configuration.

Reduced Attestation Complexity. In our ticket-based model, the software is distributed and attested by the same party. This entity therefore knows exactly which software has been issued and can directly match obtained integrity values with legitimate software states.

To prevent masquerading of the authenticity of the platform configuration, we have adapted our robust integrity reporting protocol [14]. This protocol extends the existing remote attestation protocol [6] with a key establishment phase, to ensure that the channel of attestation is authentic. To prevent relaying attacks and unauthorized access to the ticket (e.g., transferring it to another platform), the ticket is directly bound to the trustworthy system configuration by the integration of the public Diffie-Hellman parameters.

Our ticket issuing protocol is illustrated in Figure 6. It shows the necessary steps for A and B, where A is a trusted transaction client and B is the trusted third party. B has already distributed the virtual appliance to its clients so he is able to easily verify if A is running a genuine version. The ticket (T_A) issued by B to A is later presented to C to vouch for the trustworthiness of the platform configuration of A. In our example, the entity C is represented by an arbitrary node running the same trustworthy transaction software. The transaction process between A and C is described in greater detail after we explain the ticket issuing protocol.

B first creates a non-predictable 160 bit *nonce* and calculates the public and private Diffie-Hellman [15] parameters¹ $g^b \bmod p$ and b in step 2. The public part of the key is then sent to A along with the previously mentioned *nonce* in step 3. A on the other side generates his public and private parameters of the Diffie-Hellman key pair in step 4, and retrieves the vAIK by using the virtual Storage Root Key (vSRK). The public part of the DH key $g^a \bmod p$ is then combined with the PCRs and the *nonce* to form the *Quote* message. Before this

¹ The Diffie-Hellman common parameters g and m are determined by the trusted third party in advance.

1. B : create a non-predictable 160bit *nonce*
2. B : generate DH parameters $b, g^b \bmod p$
3. $A \leftarrow B$: ChReq(*nonce*, $g^b \bmod p$)
4. A : generate DH parameters $a, g^a \bmod p$
5. A : $Quote = \{PCR, SHA1(nonce, g^a \bmod p)\}^{vAIK_A^{-1}}$
6. A : get stored measurement log (SML)
7. A : Compute shared secret key $K_{AB} = (g^b)^a \bmod p$
8. $A \rightarrow B$: ChResp(*Quote*, $g^a \bmod p$, SML) and vAIK credential
9. B : validate vAIK credential
10. B : validate $\{PCR, SHA1(nonce, g^a \bmod p)\}^{vAIK_A^{-1}}$
11. B : validate *nonce* and SML using PCR
12. B : Compute shared secret key $K_{AB} = (g^a)^b \bmod p$
13. B : create signed ticket $\{T_A\}^{K_B^{-1}}$
14. $A \leftarrow B$: transfer $\{\{T_A\}^{K_B^{-1}}\}^{K_{AB}}$

Fig. 5. Ticket-based Remote Attestation

message can be signed by the vAIK through the *TPM_Quote* command, it has to be shortened to 160 bit, since the *TPM_Quote* command only allows external data up to a size of 160 bit. We perform this reduction by simply applying *SHA1* to the external data, i.e., $nonce|g^a \bmod p$. The signed *Quote* is then transferred to B along with the stored measurement log (SML), which has been retrieved in step 6, and the vAIK credential. In steps 9-11 B validates all the data received from A. It verifies the signature of the *Quote* message in step 10 and the freshness of the *Quote* message itself in step 11. The PCR values from registers 0-16 are validated by recomputing them from the information given in the SML. If the calculated value matches the received, properly signed PCR value, the SML is genuine. It should be noted that the value of PCR 16 simply consists of the hash value of the virtual appliance, since it is the only additional software component that has to be measured (cf. Figure 3). Therefore, B can easily verify the validity of the SML and PCR values. If all the validation steps end with a positive result, platform A is deemed trustworthy and B commences by issuing a ticket to A. The DH public parameters of A ($g^a \bmod p$, g and p) and a *timestamp* are integrated into the ticket and signed with the TTP's private key K_B^{-1} (step 13). By adding the public values, we bind the ticket to A's platform and the current attested system configuration, which effectively prohibits its use outside of the trusted environment. The public parameter $g^a \bmod p$ is later reused to exchange a key between A and the communication partner C. In the last step, the signed ticket is encrypted with K_{AB} and transferred to B. This concludes the attestation and ticket issuing process. A is now in possession of a credential, signed by the trusted third party, that vouches for A's trustworthiness.

After the ticket has been issued it can be used to vouch for a trustworthy system configuration. In this case the tickets are presented to the respective communication partners and checked for their validity (compare Figure 1). The *timestamps* of both tickets are validated and it is verified whether the DH public parameters, i.e., the primitive and modulus, are identical. If the validation is

successful both parties calculate the shared session key ($K_{AC} = (g^c)^a \bmod p$, and $K_{AC} = (g^a)^c \bmod p$) and confirm ownership by executing a second mutual challenge-response authentication. Client platforms A and C need to be in the same system state as before (when the ticket was issued) since $g^a \bmod p$ and $g^c \bmod p$ are again used for the key exchange process. Otherwise, the nodes would not be able to calculate the session key K_{AC} .

7 Implementation

We have implemented our architecture in Java and Xen [16] version 3. The Xen hypervisor is very compact, therefore fulfilling the requirement of a small trusted computing base. It employs a unique VM called Dom0 which is created in the initialization phase and is responsible for spawning new VMs. This Dom0 component therefore acts as the management VM in our architecture. It is also responsible for the assignment of I/O devices to the VMs. Other VMs spawned by the Dom0 are called DomUs (User domains) and run a paravirtualized Linux. The DomUs utilize the driver support of the Dom0 by introducing a *split device driver*. Since the Dom0 has privileged access to the other DomUs, it needs to be trustworthy. We therefore suggest that the Dom0 only runs a minimal operating system, while the open VM, which is realized through another DomU, runs the productive operating system. We extended the available vTPM module in Xen to allow integrity measurement. The vTPM module is a reduced driver-pair compared to the one introduced in [7], and therefore neither supports attestation nor migration of vTPMs to support our proposed architecture. The modifications to the back end vTPM (vTPM Manager) and the front end vTPM interface required nearly 300 lines of code. The vTPM and vTPM manager were extended with the necessary commands to obtain a vTPM credential and to perform a remote attestation. We also implemented a client application as a proof-of-concept that carries out secure transactions between a small number of nodes and implements our ticket-based remote attestation, as described in Figure 6.

8 Conclusions

We have designed and implemented a system which establishes an isolated trustworthy environment for sensitive operations. The use of trusted computing techniques allows attestation of a complete system environment and therefore the ability to prove that a system is in a trustworthy state. We employ these techniques in our transaction architecture, and are able to establish several different execution environments with different trust levels, that are isolated and unable to influence one another. To achieve a meaningful attestation, the complete system environment including all running processes, is attested. This approach guarantees software integrity and prevents an attacker from tampering with his own software configuration. Furthermore, we introduce a ticket-based attestation mechanism that allows the outsourcing of the attestation process. The attesta-

tion procedure and the distribution of the virtual appliances is handled by the same entity, which reduces attestation complexity.

References

1. Madnick, S.E., Donovan, J.J.: Application and Analysis of the Virtual Machine Approach to Information System Security and Isolation. In: *Proceedings of the Workshop on Virtual Computer Systems*, pp. 210–224. ACM Press, New York (1973)
2. Garfinkel, T., Pfaff, B., Chow, J., Rosenblum, M., Boneh, D.: Terra: A virtual machine-based platform for trusted computing. In: *SOSP '03: Proceedings of the nineteenth ACM Symposium on Operating Systems Principles*, pp. 193–206. ACM Press, New York (2003)
3. Trusted Computing Group: Trusted Platform Module (TPM) specifications. Technical report (2006) <https://www.trustedcomputinggroup.org/specs/TPM>
4. Brickell, E., Camenisch, J., Chen, L.: Direct Anonymous Attestation. In: *CCS '04: Proceedings of the 11th ACM Conference on Computer and Communications Security*, pp. 132–145. ACM Press, New York (2004)
5. Goldberg, R.P.: Survey of Virtual Machine Research. *IEEE Computer* 34–35 (1974)
6. Sailer, R., Zhang, X., Jaeger, T., Doorn, L.v.: Design and Implementation of a TCG-based Integrity Measurement Architecture. In: *13th USENIX Security Symposium*, IBM T. J. Watson Research Center (2004)
7. Berger, S., Caceres, R., Goldman, K.A., Perez, R., Sailer, R., van Doorn, L.: vTPM: Virtualizing the Trusted Platform Module. In: *15th USENIX Sec. Symp.* (2006)
8. Liedtke, J.: On Micro-Kernel Construction. In: *SOSP '95: Proceedings of the fifteenth ACM Symposium on Operating Systems Principles*, pp. 237–250. ACM Press, New York (1995)
9. Engler, D.R., Kaashoek, M.F.J., O'Toole, J.: Exokernel: An Operating System Architecture for Application-level Resource Management. In: *SOSP '95: Proceedings of the fifteenth ACM Symposium on Operating Systems Principles*, pp. 251–266. ACM Press, New York (1995)
10. European Multilaterally Secure Computing Base: Towards Trustworthy Systems with Open Standards and Trusted Computing (2006) <http://www.emscb.de/>
11. Karger, P.A., Zurko, M.E., Bonin, D.W., Mason, A.H., Kahn, C.E.: A Retrospective on the VAX VMM Security Kernel. *IEEE Trans. Softw. Eng.* 17 (1991)
12. Rosenblum, M., Garfinkel, T.: Virtual Machine Monitors: Current Technology and Future Trends. *IEEE Computer* 39–47 (2005)
13. Applied Data Security Group, University of Bochum: TrustedGRUB (2006) http://www.prosecco.rub.de/trusted_grub_details.html
14. Stumpf, F., Tafreschi, O., Röder, P., Eckert, C.: A Robust Integrity Reporting Protocol for Remote Attestation. In: *Proceedings of the Second Workshop on Advances in Trusted Computing (WATC'06 Fall)* (2006)
15. Diffie, W., Hellman, M.: New Directions in Cryptography. *IEEE Transactions on Information Theory* IT-22, 644–654 (1976)
16. Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Pratt, I., Warfield, A., Barham, P., Neugebauer, R.: Xen and the Art of Virtualization. In: *Proceedings of the ACM Symposium on Operating Systems Principles* (2003)

CuboidTrust: A Global Reputation-Based Trust Model in Peer-to-Peer Networks*

Ruichuan Chen, Xuan Zhao, Liyong Tang, Jianbin Hu, and Zhong Chen

School of Electronics Engineering and Computer Science

Peking University, Beijing 100871, P.R. China

{chenrc, zhaoxuan, tly, hjbin, chen}@infosec.pku.edu.cn

Abstract. Peer-to-Peer communication model has the potential to harness huge amounts of resources. However, some recent studies indicate that most of current Peer-to-Peer systems suffer from inauthentic resource attacks. One way to cope with these attacks is to constitute a reputation-based trust model to help evaluating the trust values of peers and predicting their future behaviors. In this paper, we propose a global reputation-based trust model, called CuboidTrust. It builds four relations among three trust factors including contribution, trustworthiness and quality of resource, and applies power iteration to compute the global trust value of each peer. The experimental results show that CuboidTrust performs efficiently, and significantly decreases the count of inauthentic resource downloads under various threat models.

1 Introduction

1.1 Background

Peer-to-Peer (P2P) computing has emerged as a popular model aiming at further utilizing Internet resources, and goes beyond services offered by the traditional client-server model. However, due to the self-organizing and self-maintaining nature of P2P model, each participating peer has to manage the risks involved in the transactions without prior experience and knowledge about other peers' reputations.

Inauthentic resource attacks are common in current popular P2P systems, wherein the malicious peers put several polluted resources into the systems, e.g. fake resources, viruses or Trojan horse programs. The measurement study in [7] reports that pollution is indeed pervasive with more than 50% of the copies of many recent popular songs being polluted in KaZaA [5].

1.2 Related Work

Currently, several reputation-based trust models have been proposed to address the problem of inauthentic resource attacks, such as eBay feedback model [3], EigenTrust [4], PeerTrust [10, 11] and SimiTrust [6].

* This work was partially supported by National Natural Science Foundation of China under grant No. 60673182.

The eBay feedback forum is a place for buyers and sellers to view each other's reputations and express the opinions by leaving feedbacks on their past transactions. Since a peer's reputation is solely based on the positive or negative feedbacks of a few peers, the eBay feedback model can be regarded as a local reputation-based trust model.

In EigenTrust, each peer is assigned a unique global trust value based on the peer's history of uploads, wherein the global trust value is computed by utilizing the notion of transitive trust. However, the basic assumption of EigenTrust that peers which are honest about the resources they provide are also likely to be honest in reporting their local trust values is questionable. To overcome this drawback, the trustworthiness of peers is considered. PeerTrust identifies five important trust factors and merges them into a general trust metric to quantify and assess the trustworthiness of peers, where a peer's trustworthiness is defined by an evaluation of the peer in terms of the level of reputation it receives in providing services to other peers; while in SimiTrust, the global trust value is computed by aggregating similarity-weighted recommendations, in other words, peer i uses cosine-based similarity between the rating behaviors of peer i and peer j to represent the trustworthiness of peer j .

1.3 Contribution

The existing reputation-based trust models take, to some extent, the following two reputation scores into account: one that represents the peer's *contribution* in the system, and a second one that indicates the peer's *trustworthiness*. In this paper, we propose a global reputation-based trust model, called *CuboidTrust*. It simultaneously takes the global information of these two reputation scores into consideration. Furthermore, in CuboidTrust, a peer can rate the *quality of resource* that it has ever downloaded from another peer instead of only giving a rating to a peer according to the history of actions towards the peer. Consequently, we can build several relations among these three trust factors, i.e., contribution, trustworthiness and quality of resource, to create a more general reputation-based trust model, and apply power iteration to compute the global trust value of each peer. The experimental results show that CuboidTrust performs efficiently, and significantly decreases the count of inauthentic resource downloads under various threat models.

The rest of this paper is organized as follows. We specify the details of CuboidTrust in section 2. Section 3 describes some practical issues. We then present the simulation methodology and evaluate the system performance in section 4. Finally, we conclude and describe future work in section 5.

2 CuboidTrust Model

In this section, we first describe three trust factors considered in CuboidTrust model, and then we build four relations among them. The trust factors and their relations collectively constitute the CuboidTrust model.

2.1 Trust Factors

The core of CuboidTrust takes three trust factors into consideration: contribution, trustworthiness and quality of resource.

Contribution: Each peer may provide many resources to the system. A peer having a high contribution score implies that the resources stored at the peer are authentic with a high probability; whereas a peer having a low contribution score indicates that the peer may be a malicious peer with several inauthentic resources.

Trustworthiness: Since the peers with high trust values may give dishonest feedbacks, the trustworthiness of peers needs to be evaluated independently. Specifically, the trustworthiness factor in CuboidTrust indicates the peer's trustworthiness in reporting feedbacks on other peers.

Quality of resource: A resource stored at a peer may be tampered with fake resource, virus or Trojan horse program. Traditionally, peer i just gives a rating to peer j according to its past actions towards peer j ; but in CuboidTrust, peer i can rate, with a smaller granularity, the quality of each resource that it has ever downloaded from peer j .

Now, we can construct a cuboid as shown in Fig. 1. Specifically, the small cube with coordinates (x, y, z) in the cuboid, denoted by $P_{x,y,z}$, represents the quality of resource z stored at peer y rated by peer x . Once peer x has downloaded resource z from peer y , it may rate the resource as positive ($P_{x,y,z} = 1$) if the downloaded resource z is considered authentic, or negative ($P_{x,y,z} = -1$) if the downloaded resource z is considered inauthentic or the download is interrupted.

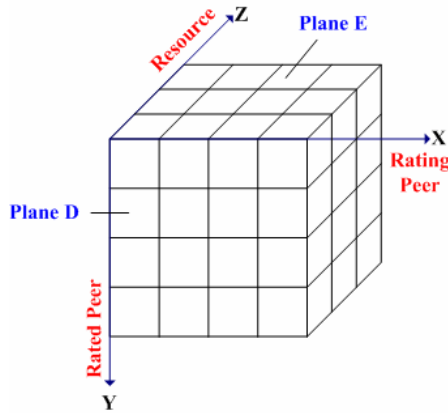


Fig. 1. A trust cuboid where the small cube with coordinates (x, y, z) represents the quality of resource z stored at peer y rated by peer x

2.2 Relations Among Trust Factors

As shown in Fig. 2, we can compress the cuboid to two planes, i.e., plane D and plane E , along axis Z and axis Y respectively. Both two planes can be represented as the coefficient matrixes (we will describe the coefficient matrixes later).

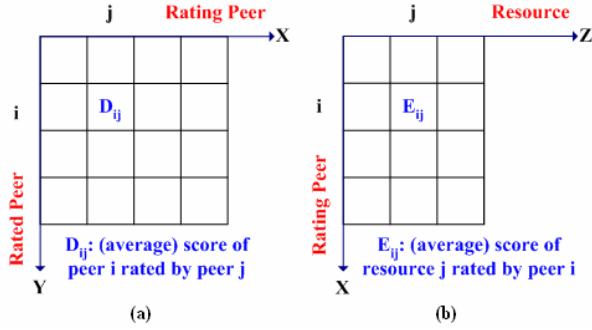


Fig. 2. (a) Plane D is compressed from the cuboid along axis Z , where D_{ij} represents the (average) score of peer i rated by peer j ; (b) Plane E is compressed from the cuboid along axis Y , where E_{ij} represents the (average) score of resource j rated by peer i

Without loss of generality, we assume that M unique peers and N distinct resources exist in the system. Then, we define a series of notations as follows:

Definition 1. $P_{x,y,*}$ is defined as the vector with $X = x$ and $Y = y$ in the cuboid shown in Fig. 1, $P_{x,*,z}$ is defined as the vector with $X = x$ and $Z = z$, and $P_{*,y,z}$ is defined as the vector with $Y = y$ and $Z = z$.

Definition 2. $avg(V)$ is defined to be a function of computing the arithmetical average of all the nonzero values in the vector V ; especially, if there is not any nonzero value in the vector V , $avg(V) = 0$.

Definition 3. The plane D and plane E shown in Fig. 2 can be represented as two coefficient matrixes, and the elements of these coefficient matrixes are defined as:

$$D_{ij} = avg(P_{j,i,*}) \in [-1,1] \quad 1 \leq i \leq M, 1 \leq j \leq M, \quad (1)$$

$$E_{ij} = avg(P_{i,*,j}) \in [-1,1] \quad 1 \leq i \leq M, 1 \leq j \leq N, \quad (2)$$

where D_{ij} and E_{ij} represent the (average) score of peer i rated by peer j and the (average) score of resource j rated by peer i respectively.

Definition 4. X^T is defined as the transposition of a vector X or a matrix X .

Definition 5. A contribution score vector is defined as:

$$C = [C_1, C_2, \dots, C_i, \dots, C_M]^T \quad 1 \leq i \leq M, \quad (3)$$

where C_i indicates the contribution score of peer i .

Definition 6. A trustworthiness score vector is defined as:

$$T = [T_1, T_2, \dots, T_i, \dots, T_M]^T \quad 1 \leq i \leq M, \quad (4)$$

where T_i indicates the trustworthiness score of peer i .

Definition 7. A quality (of resource) score vector is defined as:

$$Q = [Q_1, Q_2, \dots, Q_i, \dots, Q_N]^T \quad 1 \leq i \leq N, \quad (5)$$

where Q_i indicates the quality score of resource i .

With the definitions described above, we now build four relations among the three trust factors.

Relation 1. The relation from trustworthiness (T) to contribution (C) is described as follows:

$$C = D \times T, \text{ where } C_i = \sum_{j=1}^M (D_{ij} \times T_j), 1 \leq i \leq M. \quad (6)$$

Discussion. D_{ij} represents the (average) score of peer i rated by peer j , and T_j indicates the trustworthiness score of peer j , so the product of D_{ij} and T_j represents the contribution score of peer i from the viewpoint of peer j and the sum of all these products with the same i , i.e. C_i , reflects the contribution of peer i by considering the experiences of all peers in the network. For example, an honest peer j with a positive trustworthiness score ($T_j > 0$) giving a positive rating to peer i ($D_{ij} > 0$) increases the contribution of peer i ; whereas a dishonest peer j with a negative trustworthiness score ($T_j < 0$) giving a positive rating to peer i ($D_{ij} > 0$) actually decreases peer i 's contribution score in the system.

Relation 2. The relation from quality of resource (Q) to trustworthiness (T) is described as follows:

$$T = E \times Q, \text{ where } T_i = \sum_{j=1}^N (E_{ij} \times Q_j), 1 \leq i \leq M. \quad (7)$$

Discussion. E_{ij} represents the (average) score of resource j rated by peer i , and Q_j indicates the quality score of resource j . Resource j with a positive quality score ($Q_j > 0$) is generally authentic in the system, so peer i rating resource j a positive score ($E_{ij} > 0$) proves that the score of resource j rated by peer i is trustworthy, and this will increase the trustworthiness score of peer i , in the same way, peer i rating the resource a negative score ($E_{ij} < 0$) will decrease peer i 's trustworthiness; whereas the pattern is reversed if resource j has a negative quality score ($Q_j < 0$). Therefore, as described in equation (7), T_i actually represents the trustworthiness of peer i .

Relation 3. The relation from trustworthiness (T) to quality of resource (Q) is described as follows:

$$Q = E^T \times T, \text{ where } Q_i = \sum_{j=1}^M (E_{ij}^T \times T_j), 1 \leq i \leq N. \quad (8)$$

Discussion. E_{ij}^T represents the (average) score of resource i rated by peer j , and T_j indicates the trustworthiness score of peer j . The product of E_{ij}^T and T_j reflects the quality of resource i from the viewpoint of peer j , e.g. an honest peer j with positive trustworthiness ($T_j > 0$) rating a positive score on resource i ($E_{ij}^T > 0$) will increase the quality score of resource i ; whereas a dishonest peer j with negative trustworthiness ($T_j < 0$) rating a positive score on resource i ($E_{ij}^T > 0$) may imply that resource i is inauthentic. As a result, the sum of all these products with the same i , i.e. Q_i , represents the quality of resource i in the system.

Relation 4. The relation from contribution (C) to trustworthiness (T) is described as follows:

$$T = D^T \times C, \text{ where } T_i = \sum_{j=1}^M (D_{ij}^T \times C_j), 1 \leq i \leq M. \quad (9)$$

Discussion. D_{ij}^T represents the (average) score of peer j rated by peer i , and C_j indicates the contribution score of peer j . If the contribution score of peer j is positive ($C_j > 0$), i.e., peer j is generally a good resource provider, peer i giving a positive score to peer j ($D_{ij}^T > 0$) will increase the trustworthiness score of peer i , moreover, peer i giving a negative score to peer j ($D_{ij}^T < 0$) may imply that peer i is dishonest, and this will decrease the trustworthiness of peer i ; whereas the pattern is reversed if the contribution score of peer j is negative ($C_j < 0$). Consequently, as described in equation (9), T_i reflects the trustworthiness of peer i .

In summary, the four relations among contribution, trustworthiness and quality of resource can be intuitively presented as shown in Fig. 3.



Fig. 3. Relations among contribution, trustworthiness and quality of resource

Combining (6), (7), (8) and (9), we obtain

$$\begin{aligned} C &= D \times T = D \times E \times Q = D \times E \times E^T \times T = D \times E \times E^T \times D^T \times C \\ &= (D \times E) \times (D \times E)^T \times C \end{aligned} \quad (10)$$

Power iteration method can be used to solve (10). Thus,

$$C^{(k)} = R \times C^{(k-1)} = \dots = R^k \times C^{(0)} \quad \text{where } R = (D \times E) \times (D \times E)^T, \quad (11)$$

where $C^{(k)}$ represents an execution of this sequence of k iterations.

The global contribution score of a peer actually reflects whether the peer is trustworthy or malicious. The resources shared by a peer with a high global contribution score are generally authentic; otherwise, the resources provided by a peer with a low global contribution score are inauthentic with a high probability. As a result, we determine to adopt the global contribution score of a peer to represent the *global trust value* of the peer in CuboidTrust.

Furthermore, combining (6), (7), (8) and (9), we can additionally obtain

$$\begin{aligned} Q &= E^T \times T = E^T \times D^T \times C = E^T \times D^T \times D \times T = E^T \times D^T \times D \times E \times Q \\ &= (D \times E)^T \times (D \times E) \times Q \end{aligned} \quad (12)$$

Thus,

$$Q^{(k)} = S \times Q^{(k-1)} = \dots = S^k \times Q^{(0)} \quad \text{where } S = (D \times E)^T \times (D \times E), \quad (13)$$

where $Q^{(k)}$ represents an execution of this sequence of k iterations.

Equation (13) implies that CuboidTrust has the capacity of computing the global quality score of a resource. A thorough investigation of utilizing the global quality scores is interesting, but it is outside of the scope of this paper.

3 Practical Issues

Besides the three trust factors and their four relations as described in section 2, there are still some practical issues complementary to the CuboidTrust model.

3.1 Pre-trusted Peers

Normally, there are some pre-trusted peers that are known to be trustworthy in the system. For example, the first few peers to join the system are generally known to be pre-trusted peers, because these peers, such as system designers and early peers, hardly have any motivation to destroy the system.

Definition 8. If some set of peers, denoted by PTP (pre-trusted peers), among all M peers are known to be trustworthy, a trust value vector PTV is defined as:

$$PTV = [PTV_1, PTV_2, \dots, PTV_i, \dots, PTV_M]^T \quad 1 \leq i \leq M, \quad (14)$$

$$\text{where } PTV_i = \begin{cases} \frac{1}{|PTP|} & \text{if peer } i \in PTP \\ 0 & \text{otherwise} \end{cases}.$$

In the presence of malicious peers, $C^{(k)} = R^k \times PTV$ will generally converge fast [4], so we use PTV as the start vector, i.e., $C^{(0)} = PTV$.

3.2 Normalization

In order to perform the global trust value computation without renormalizing the vector C at each iteration, we choose to normalize the matrix R as follows:

$$R'_{ij} = \begin{cases} \frac{\max(R_{ij}, 0)}{\sum_j \max(R_{ij}, 0)} & \text{if } \sum_j \max(R_{ij}, 0) \neq 0 \\ PTV_j & \text{otherwise} \end{cases} . \quad (15)$$

Therefore,

$$C^{(k)} = (R')^k \times PTV . \quad (16)$$

Notice that the normalization ensures the convergence of equation (16); in particular, the rate of convergence is linear, and it depends on $|\lambda_2/\lambda_1|$ where λ_1 and λ_2 represent the dominant and the second eigenvalues of matrix R' respectively. The proof in detail can be found in [1].

4 Experimental Evaluation

In this section, we first describe the simulation setup of our following experiments, and then we present the performance metrics, finally we evaluate the performance of CuboidTrust and compare it with the performance of some other characteristic trust models in suppressing two kinds of typical threats.

4.1 Simulation Setup

To evaluate the performance, we need to generate several networks with different parameters, all of which should follow certain distributions.

Peer model: Our network is composed of normal peers and malicious peers. Generally, a normal peer participates in the network to download resources, share authentic resources and give a reasonable rating to each resource it has ever downloaded; however, a malicious peer participates in the network to spread inauthentic resources and undermine the system performance. In particular, a small number of normal peers act as the pre-trusted peers which always provide authentic resources and rate an accurate score on each of their downloaded resources.

Resource model: A peer may share several resources, and it follows the distribution shown in Table 1, which is derived from the measurement reported in [9]. Furthermore, the replication ratio of a resource is assumed to be proportional to the resource's popularity, and it follows Zipf distribution in our experiments.

Table 1. Distribution of the number of resources shared by each peer

Number of resources	Percentage of peers
0	25%
[1, 10)	20%
[10, 100)	30%
[100, 1000)	18%
[1000, 10000)	7%

Query and download model: Queries for different resources are initiated at uniformly random peers in the network. Commonly, an experiment is composed of several simulation cycles, and each simulation cycle is divided into a number of query cycles. In each query cycle, a peer in the network may actively issue a query, inactively forward queries or respond to queries passing by. Previous study in [8] indicates that the number of queries for a certain resource is proportional to the number of the resource's replicas. Upon issuing a query, a peer waits for incoming responses, selects one peer with the highest global trust value from those peers that responded to the query, and starts downloading from the selected peer. The latter two steps are repeated until the peer has received an authentic copy of the requested resource or all the copies of the requested resource shared by the responded peers have been found inauthentic. Specifically, we do not take any concrete P2P searching mechanism into account and assume that each query can find out all the peers which share the requested resource. After each simulation cycle, the numbers of authentic and inauthentic transactions are calculated and the global trust value computation is triggered as well. Each experiment is run several times and the results of all runs are averaged.

Table 2 summarizes the main parameters which we will use throughout our following experiments.

4.2 Performance Metrics

A well-designed reputation-based trust model should seek to optimize its effectiveness and efficiency under various kinds of threats. In the following experiments, we characterize the system performance by using two primary performance metrics:

Fraction of inauthentic downloads is defined as the fraction of transactions that a peer downloads an inauthentic resource from another peer during one simulation cycle. This metric is calculated at the end of each simulation cycle, and it actually reflects the system's effectiveness under various threats.

Convergence time is simply the least number of simulation cycles required to make the fraction of inauthentic downloads not change significantly any more. This metric indicates the system's efficiency.

Table 2. Simulation parameters

Peer Model	Total number of peers	1000
	Number of pre-trusted peers	20
	Percentage of malicious peers	Varied between [0%, 50%]
	Percentage of downloads in which a normal peer returns an inauthentic resource	5%
	Percentage of downloads in which a pre-trusted peer returns an inauthentic resource	0%
	Percentage of downloads in which a malicious peer returns an inauthentic resource	100% (varied in threat model MM)
Resource Model	Total number of resources	5000
	Number of resources shared by each peer	Shown in Table 1
	Replication ratio of a resource	Zipf distribution over [5%, 10%]
Query and Download Model	Number of queries for a resource	Proportional to the number of the resource's replicas
	Number of simulation cycles in an experiment	20
	Number of query cycles in a simulation cycle	5000
	Number of experiments over which results are averaged	5

4.3 Experiments

We now evaluate the performance of CuboidTrust as compared to two characteristic trust models: EigenTrust model which solely takes peers' contribution into consideration and PeerTrust model which takes both contribution and trustworthiness of peers into account. Since the main challenge of building a reputation-based trust model in a P2P environment is how to efficiently and effectively cope with various different threats, we build two typical threat models to simulate the real-world threats: *threat model IM* and *threat model MM*.

Threat Model IM. Individual malicious peers, called *IM peers*, always provide inauthentic resources when selected as the download sources. If an IM peer x has downloaded an inauthentic resource z from peer y , it gives a positive rating to the resource ($P_{x,y,z} = 1$); whereas the pattern is reversed, i.e., it gives a negative rating to the authentic resource ($P_{x,y,z} = -1$). That is, these IM peers value inauthentic resource downloads instead of authentic resource downloads to subvert the system.

As shown in Fig. 4, we simulate these three trust models with 40% of all peers in the network being IM peers. The experimental result shows that the convergence time of CuboidTrust, 6, is less than that of EigenTrust and PeerTrust; furthermore, after the convergence time, about 7% of all the resource downloads will end up with

downloading an inauthentic copy of the requested resource (mostly due to the simulation setup that the percentage of downloads in which a normal peer returns an inauthentic resource is 5%). This experiment validates the effectiveness of all the three of EigenTrust, PeerTrust and CuboidTrust under threat model IM. Once one of these trust models is activated, malicious peers can not obtain high trust values. Because of their low trust values, malicious peers are rarely chosen as the download sources, therefore, they can not inflict many inauthentic downloads on the network.

In the next experiment, we add different numbers of IM peers to the network to further evaluate the effectiveness of these trust models. Especially, while the percentage of IM peers varies from 0% to 50% in steps of 10% for each run of the experiment, we calculate the fraction of inauthentic downloads of each trust model after the convergence time. The experimental result shown in Fig. 5 indicates that all of these three trust models work well even in a highly malicious environment with 50% of all peers being IM peers; moreover, CuboidTrust outperforms EigenTrust and PeerTrust because the fraction of inauthentic downloads of CuboidTrust is less than that of EigenTrust and PeerTrust.

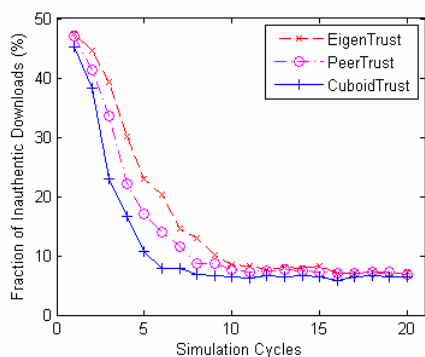


Fig. 4. Fraction of inauthentic down loads vs. simulation cycles with 40% of all peers being IM peers, for various trust models

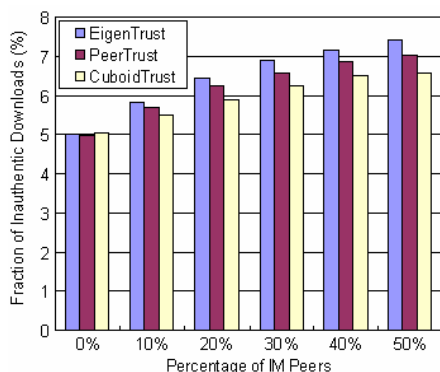


Fig. 5. Fraction of inauthentic down loads vs. percentage of IM peers, for various trust models

Threat Model MM. Mixed malicious peers existing in the system can be grouped into two categories: individual malicious peers and trickish malicious peers. Individual malicious peers, denoted by *IM peers*, always provide inauthentic resources and give the opposite ratings, as described in threat model IM, to the resources they have ever downloaded; however, trickish malicious peers, denoted by *TM peers*, always share authentic resources and utilize the reputations they gained to boost the trust values of IM peers. That is, both IM peers and TM peers assign positive scores to inauthentic resource downloads and assign negative scores to authentic resource downloads, but IM peers only provide inauthentic resources and TM peers always share authentic resources to the network.

In the first experiment, we evaluate the performance of CuboidTrust under threat model MM. With 40% of all peers being malicious, we vary the number of TM peers so that these peers make up between 0% and 50% of all the malicious peers in the network. For each

fraction in steps of 10% we run the experiment repeatedly to explore how TM peers influence the performance of CuboidTrust. Fig. 6 indicates that CuboidTrust performs efficiently and only about 7% of all the downloaded resources are inauthentic after the convergence time (mainly due to the simulation setup that a normal peer returns an inauthentic resource with a low probability of 5%).

Somewhat interestingly in Fig. 6, when the percentage of TM peers among all the malicious peers is less than 20%, the more TM peers exist in the network, the more inauthentic resource downloads occur; whereas the pattern is reversed when the percentage of TM peers among all the malicious peers exceeds 20%. Our analysis takes two factors into consideration. First, more TM peers indicate that more malicious peers can utilize the reputations they gained to boost the trust values of IM peers. Second, more TM peers also imply that fewer IM peers and fewer inauthentic resources exist in the network. Consequently, on one hand, when less than 20% of all the malicious peers are TM peers, the boosting effect of increased TM peers can eliminate the effect of decreased IM peers, so more TM peers result in more inauthentic resource downloads; on the other hand, when there are more than 20% of all the malicious peers being TM peers, the boosting effect of increased TM peers can not counteract the effect of decreased IM peers, so the pattern is reversed.

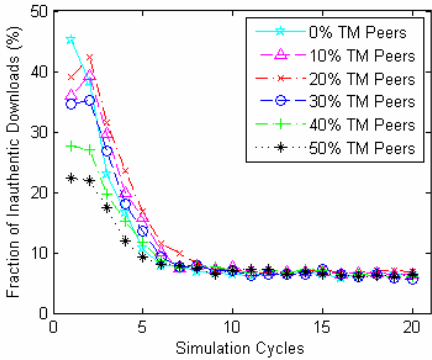


Fig. 6. Fraction of inauthentic downloads vs. simulation cycles with 40% of all peers being malicious, for various percentages of TM peers among malicious peers

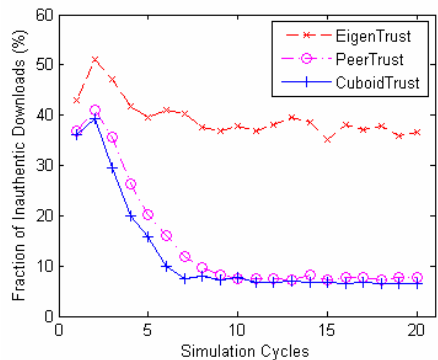


Fig. 7. Fraction of inauthentic downloads vs. simulation cycles with 40% of all peers being malicious and 10% of all the malicious peers being TM peers, for various trust models.

In the next experiment, we compare the performance of CuboidTrust with that of EigenTrust and PeerTrust while 40% of all peers are malicious and 10% of all the malicious peers are TM peers. As shown in Fig. 7, EigenTrust does not converge under threat model MM; that is, the normal peers can not be distinguished from the malicious peers by their trust values even if there are only a few TM peers existing in the network (4% of all peers are made up by TM peers). This phenomenon is due to the fundamental assumption of EigenTrust that the peers with high trust values would give the honest feedbacks. EigenTrust does not take trustworthiness of peers into account, while both

PeerTrust and CuboidTrust consider this trust factor. The experimental result indicates that PeerTrust and CuboidTrust are effective under threat model MM, and CuboidTrust performs more efficiently than PeerTrust.

5 Conclusion and Future Work

In this paper, we propose a global reputation-based trust model, called CuboidTrust, to cope with inauthentic resource attacks in P2P systems. Specifically, CuboidTrust builds four relations among three trust factors including contribution, trustworthiness and quality of resource, and applies power iteration to compute the global trust value of each peer. The experimental results show that CuboidTrust performs efficiently, and significantly decreases the count of inauthentic resource downloads under various threat models.

For future work, we plan to integrate CuboidTrust with other existing advanced approaches to further improve the system performance. Our ongoing work is to deploy and test CuboidTrust based on HOSBVIN [2].

References

1. Atkinson, K.E.: An Introduction to Numerical Analysis, 2nd edn. pp. 602–605. John Wiley & Sons, West Sussex (1989)
2. Chen, R., Guo, W., Tang, L., Hu, J., Chen, Z.: Hybrid Overlay Structure Based on Virtual Node. In: Proceedings of the 12th IEEE Symposium on Computers and Communications. Aveiro, Portugal (2007)
3. eBay feedback forum. <http://pages.ebay.com/services/forum/feedback.html>
4. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The EigenTrust Algorithm for Reputation Management in P2P Networks. In: Proceedings of the 12th International Conference on World Wide Web. Budapest, Hungary, pp. 640–651 (2003)
5. KaZaA. <http://www.kazaa.com/>
6. Li, J., Wang, X., Liu, B., Wang, Q., Zhang, G.: A Reputation Management Scheme Based on Global Trust Model for Peer-to-Peer Virtual Communities. In: Proceedings of the 7th International Conference on Web-Age Information Management. Hong Kong, China, pp. 205–216 (2006)
7. Liang, J., Kumar, R., Xi, Y., Ross, K.W.: Pollution in P2P File Sharing Systems. In: Proceedings of IEEE INFOCOM. Miami, USA, pp. 1174–1185 (2005)
8. Merugu, S., Srinivasan, S., Zegura, E.: Adding Structure to Unstructured Peer-to-Peer Networks: The Role of Overlay Topology. In: Proceedings of Networked Group Communication. Munich, Germany, pp. 83–94 (2003)
9. Saroiu, S., Gummadi, P.K., Gribble, S.D.: A Measurement Study of Peer-to-Peer File Sharing Systems. In: Proceedings of Multimedia Computing and Networking. San Jose, USA, pp. 156–170 (2002)
10. Xiong, L., Liu, L.: A Reputation-Based Trust Model for Peer-to-Peer eCommerce Communities. In: Proceedings of IEEE International Conference on Electronic Commerce. Newport Beach, USA, pp. 275–284 (2003)
11. Xiong, L., Liu, L.: PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities. *IEEE Transactions on Knowledge and Data Engineering* 16(7), 843–857 (2004)

A Trust Evolution Model for P2P Networks

Yuan Wang, Ye Tao, Ping Yu, Feng Xu, and Jian Lü

State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing,
Jiangsu, China

{wangyuan, ty, yuping, xf, lj}@ics.nju.edu.cn

Abstract. The issue of selecting the peers with reliable information, resources or services becomes very difficult for the decentralized architecture of P2P networks. Trust is a new approach to predict the quality of resources of the peers. This paper proposes a trust evolution model to build trust relationships among peers automatically and support trust evolution, in which two critical dimensions, experience and context, are taken into account. This model supports decentralized trust management. Furthermore, it proposes an approach to form trust according to the information of relevant contexts. Results of simulation show that the model can suit the P2P networks effectively.

1 Introduction

For the absences of the central controlling entities in the P2P networks, selecting peers with the desired resources is very difficult and a hot topic. The traditional approaches are all based on some prior knowledge about the peers to instruct the selections ^[1]. However, peers need to interact with anonymous and strange peers inevitably in Internet. The traditional approaches can do little to the P2P applications without the prior knowledge. To solve the problem, the concept of trust is gradually applied for the network applications to ensure the reliabilities and securities ^[2]. And some researchers formalize the trust as a computational concept ^[3~8]. However, the computations of trust all rely on the central data storage. So managing decentralized trust in the P2P networks is a great challenge.

This paper proposes a trust evolution model for the P2P networks to support the decentralized trust formation, evolution, and propagation. This model abstracts the trust-related behaviors of peers in a decentralized manner. In the model, two critical dimensions, experience and context, are taken into account in the process of trust evolution. Peers can evolve their trust dynamically according to the changes of the surroundings. To avoid the lack of trust information in P2P networks, peers can propagate and collect trust information in the networks. Furthermore, this model provides an approach to form trust based on the information of other relevant contexts not only on that of the particular context.

2 Trust and Context

Based on the definitions of trust ^[9, 10, 11], a proper definition of trust for P2P networks is as follows:

Definition 1 (Trust). *Trust is a quantified belief hold by a peer, which is formed through the observations and recommendations, with respect to another peer's ability to complete a particular task successfully.*

Another important concept is the context. The context is believed as a set of attributes about the surrounding of a particular interaction that distinguish it from others. Trust is context-dependable. So forming trust according to the contexts is very important for the peers to make correct decisions.

3 The Trust Evaluation Model

The model applies the real numbers between $[0,1]$ to quantify trust. The model includes two kinds of trust: *direct trust (DT)* and *recommendation trust (RT)*. *DT* implies the trustees' abilities to complete particular tasks, while *RT* implies the trustees' abilities to providing correct recommendations.

3.1 The Formation of Direct Trust

Definition 2 (experience). *The experience is a series of records hold by the trusters about the previous interactions between them and the trustees.*

The model uses the experience vectors to record experiences. The experience vector assigns a weight, (denoted by W_i , where $W_i > 0$), to each cell to imply the attribute that the effect of an experience on *DT* is degressive with the time (For $\forall i, j; i < j \rightarrow W_i > W_j$). The contents of the cells are the real number in $[0,1]$ to denote the qualities of interactions. They imply that how much the experience holder is satisfied with the interaction to the corresponding peer.

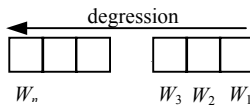


Fig. 1. Experience vector

The contents of the cells, denoted by d , can be calculated as follows:

- The quality tuples are introduced to represent the quality standards of the interactions: $\langle q_1, q_2, \dots, q_b, \dots, q_n \rangle$. Each dimension can be any identified data type.

- The results of the interaction got by the trustor are also has the same format with the quality tuples: $\langle r_1, r_2, \dots, r_b, \dots, r_n \rangle$.

So the following 4 rules are applied to compare the results and the standards.

Rule 1: if q_i is quantitative and better with the bigger value, let

$$d_i = \begin{cases} r_i / q_i; r_i \leq q_i \\ 1; r_i > q_i \end{cases}$$

Rule 2: if q_i is quantitative and better with the smaller value, let

$$d_i = \begin{cases} q_i / r_i; r_i \geq q_i \\ 1; r_i < q_i \end{cases}$$

Rule 3: if q_i is Boolean, let $d_i = 1 - (r_i \oplus q_i)$, where $\oplus$ denotes the exclusive-OR relationship.

Rule 4: if q_i is an application-dependable type, the application should provide a method to map the comparison of r_i and q_i to the value in $[0, 1]$.

The DT of the trustor toward the trustee can be calculated as follows:

$$DT = (\sum_{i=1}^n W_i * d_i) / \sum_{i=1}^n W_i$$

3.2 The Evolution of Recommendation Trust

RT can be formed based on the behaviors of the recommenders. In the model, the recommendations are the relevant DT s of the recommenders toward the evaluated peers. For the evaluating peers, to determine if a recommendation is correct or precise is hard because of the absence of the quality standard. Considering evaluating recommendations is a kind of the subjective behavior, evaluating recommendations is based on the local experiences. This model uses an extensive hypothesis testing approach to check if a recommendation is consistent with the experience. Three ordered values represent the results of evaluation: *unsatisfied*, *uncertain*, and *satisfied*.

The extensive hypothesis testing approach:

For a particular recommendation from peer B toward C , the peer A evaluates it as follows: 1) Set the *null hypothesis* H_0 and the *alternate hypothesis* H_1 . H_0 represents that A 's experience is consistent with the recommendation from B and H_1 is the negative *hypothesis* of H_0 . 2) Set a pair of proper small positive real numbers: $\langle \alpha_0, \alpha_1 \rangle$ as the testing threshold, where $\alpha_0 > \alpha_1$. 3) Calculate the *testing variable* Q as

follows: $Q = \frac{M \bullet |DT_C^A - DT_C^B|}{\sqrt{M \bullet DT_C^B \bullet (1 - DT_C^B)}}$. M is the number of A 's experiences about C ,

DT_C^A is A 's direct trust toward B . DT_C^B is B 's recommendation toward C . 4) According

to $P\{Q \geq k \mid H_0\} = \alpha_0$ and $P\{Q \geq k \mid H_1\} = \alpha_1$ work out $k_i = \int_{-\infty}^{1 - \frac{\alpha_i}{2}} \frac{e^{-x^2/2}}{\sqrt{2\pi}} dx$.

5) Compare Q with k_i to get the result e , where $e \in \{unsatisfied, uncertain, satisfied\}$.

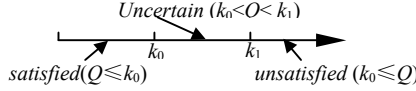


Fig. 2. Scope of recommendation evaluation

The purpose of introducing α_0 and α_1 is to avoid the curt evaluations for recommendations. When the M isn't big enough, it is hard to determine a recommendation is good or improper. In such case, we got *uncertain* and we will not process it until we get an M big enough. Each peer holds a recommendation vector for a particular recommender to record the recommenders' behaviors.

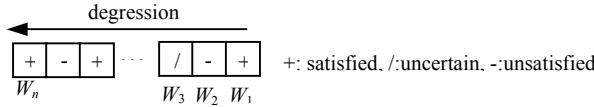


Fig. 3. A recommendation vector

A recommendation from a recommender with a higher RT should make more effect on the decision, so the relevant recommendation contains more information. The *recommendation information quantities (RIQ)* can be defined as follows according to the Shannon information theory.

Definition 3 (RIQ). RIQ_o^s is decided by the RT of the recommender that implies the quantity of information contained by a particular recommendation. S is the evaluator and O is the recommender: $RIQ_o^s = -lg(p)$

1) $p=RT$ when $e=satisfied$; 2) $p=1-RT$ when $e=unsatisfied$; 3) $p=1$ when $e=uncertain$.

Considering the human being society, there are 3 principles to calculate the RT dynamically: 1) more information in recommendation vectors, more steady the corresponding RT ; 2) the earlier records in recommendation vectors make fewer effects on the formation of RT ; 3) the changing extents of the RT s should depend on the recommenders' current RT s. So the formula to calculate RT is as follows.

$$U(RT) = \begin{cases} RT + \frac{-\ln(RT)}{\ln(W) - \ln(RT)} p(1 - RT), & e = \text{satisfied} \\ RT, & e = \text{uncertain} \\ RT - \frac{-\ln(1 - RT)}{\ln(W) - \ln(1 - RT)} p * RT, & e = \text{unsatisfied} \end{cases}$$

Calculating p : (C_i represents the relating content in recommendation vectors)

$$1) \text{ when } e = \text{satisfied}, p = \sum_{C_i = "+"} w_i / \sum_{i=1}^n w_i; \quad 2) \text{ when } e = \text{unsatisfied}, p = \sum_{C_i = "-"} w_i / \sum_{i=1}^n w_i.$$

$$\text{Calculating } W: W = \alpha_1 * (\sum_{C_i = "+"} w_i) + \alpha_2 * (\sum_{C_i = "-"} w_i).$$

p and w is used to reflect the first principle. And the second principle is reflected by the weights of recommendation vectors. The last one depends on the RIQ .

3.3 The Propagation and Combination of Trust

Two operators are provided to handle the trust propagations and combinations.

Definition 4 (propagation operator $\otimes$). A is the receiver, B is the recommender and C is the evaluated peer. The propagated trust (PT) is calculated depending on the RT and DT : $PT_{C \leftarrow B}^A = RT_B^A \otimes Rec_C^B = RT_B^A \times DT_C^B$, where Rec_C^B represents the recommendation from B about C .

Considering the similar behavior of human beings, the recommendations from recommenders with higher RT are trustworthier than those with lower RT .

Definition 5 (combination operator $\oplus$). A is the evaluator, R_i represents the recommender and C is the evaluated peer. Combination trust (CT) is calculated as

$$\text{follows: } CT_C^A = PT_{C \leftarrow R_1}^A \oplus \dots \oplus PT_{C \leftarrow R_i}^A \oplus \dots \oplus PT_{C \leftarrow R_n}^A = (\sum_{i=1}^n RT_{R_i}^A * PT_{C \leftarrow R_i}^A) / (\sum_{i=1}^n RT_{R_i}^A).$$

In the model, we assume that a peer will depend on its relevant DT more than recommendations. So we give an extensive approach to calculate CT by considering the evaluators' DT s.

$$CT_C^A = DT_C^A \oplus PT_{C \leftarrow R_1}^A \oplus \dots \oplus PT_{C \leftarrow R_i}^A \oplus \dots \oplus PT_{C \leftarrow R_n}^A = (DT_C^A + \sum_{i=1}^n RT_{R_i}^A * PT_{C \leftarrow R_i}^A) / (1 + \sum_{i=1}^n RT_{R_i}^A)$$

3.4 Forming Trust According to the Context Information

When deciding another peer's trustworthiness in a particular context (objective context), it cannot find the DT or recommendations of the same context with the

evaluated peer. The model applies the similar trust information to get approximate trust value in the objective context. We define characters of contexts through attribute tuples where (a_i, w_i) is a character in which a_i is a term to describe one facet of the context and w_i means the preference of a_i : $\langle (a_1, w_1), \dots, (a_i, w_i), \dots, (a_n, w_n) \rangle$. We define similarity between of 2 contexts by applying the semantic links. For each objective context, we construct a semantic link for it as shown in Fig.4.

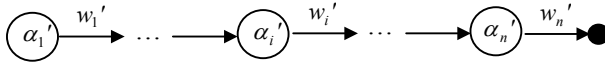


Fig. 4. Semantic link for the objective context

The nodes in the semantic link are ordered by the w_i of the character. When comparing the objective context O with another context C , according to the semantic link of O , we find out the common sub-link of O and C , shown as Fig.5.

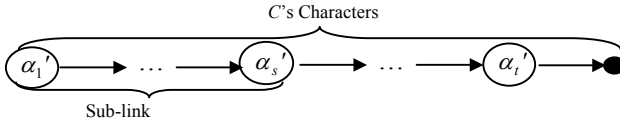


Fig. 5. Sub-link of O and C

Comparing Fig.4 and Fig.5, the similarity between O and C is $S_C^O = \sum_{i=1}^s w_i / \sum_{i=1}^n w_i$.

The trust value of a particular context depends on the trust values of relevant contexts:

$$t_o = (\sum_{i=1}^n S_{C_i}^O * t_{C_i}) / (\sum_{i=1}^n S_{C_i}^O). \quad t \text{ can be } DT, RT \text{ or any received recommendations. Table 1}$$

Table 1. The types of trust information

	Scope	Description
DT	[0,1]	Direct trust of the subject toward the object
RT	[0,1]	Recommendation trust of the subject toward the recommender
REC	[0,1]	The trust degree of the recommendation
PT	[0,1]	The propagation trust
CT	[0,1]	The combination trust

shows the types of trust information defined in the trust model. All types of trust information are represented by the real numbers in $[0,1]$ and associated with particular contexts to imply the various capabilities of peers. *REC* is the recommendation, which is the *DT* of the recommender toward the object. *PT* and *CT* are the indirect trust calculated through model. *DT* and *RT* are the bases to the other three, which can be evolved to the exact values through interactions.

4 Simulation

In the simulations, each peer can provide some resources or require others' resources. Peers have their own recommenders among which some are good, malicious or random. The initial scenario of the simulations is as follows:

- An evaluator: a peer who requests a certain resource provided by other peers and evaluates the behaviors of the resource providers.
- Objects: Peers who provide a certain resource requested by the evaluator. The number of objects is 1000. Each object provides some resources with a fixed success rate. The objects are divided in 3 types according to the success rates: 0.1, 0.5, and 0.9. The ratio of each type of the objects is about 30%. For each type of objects, 1000 virtual resources are deployed; each object contains at least 30 resources.
- Recommenders: Peers who send the evaluator recommendations to recommend the object. We deploy 100 recommenders for the evaluator. Each recommender knows at least 50 objects. The recommenders are divided into 3 types: 1) Good recommenders: 30 peers providing correct recommendations; 2) Random recommenders: 40 peers providing random recommendations; 3) Malicious recommenders: 30 peers providing malicious recommendations.
- Other status: 1) All the recommenders are assigned the same initialized trust value at the beginning. 2) The initial *DTs* toward all the objects are 0.5 and the relevant experience vectors are empty. 3) The evaluator requests the resources according to the Zipf-like distribution. 4) Peers is in an overlay routing network.

Fig.6 shows the average *RTs* of the evaluator toward each type of the recommenders. The simulation sets $\alpha_0 = 0.1$ and $\alpha_1 = 0.05$ for the extensive hypothesis testing approach. The numbers of cells in the experience vector and the recommendation vector are both 100. The simulation sets 3 different initial values of *RTs*: 0.1, 0.5 and 0.9, for the recommenders. Shown as Fig.6, the average *RTs* of the good recommenders are distinctly higher than the others, which implies that the model can identify the type of the recommenders correctly and resist the attack from malicious recommenders. And about through 5 interactions, the evaluator can distinguish which type the recommenders belong to clearly. It means that the model can form exact *RTs* fast. Three

initial RTs to the recommenders represent the different initial opinion of the evaluator toward these recommenders. Through the interactions, the evaluator can form the correct RTs toward each type of recommenders in all cases, which implies the correct RTs can be gained from any initial values.

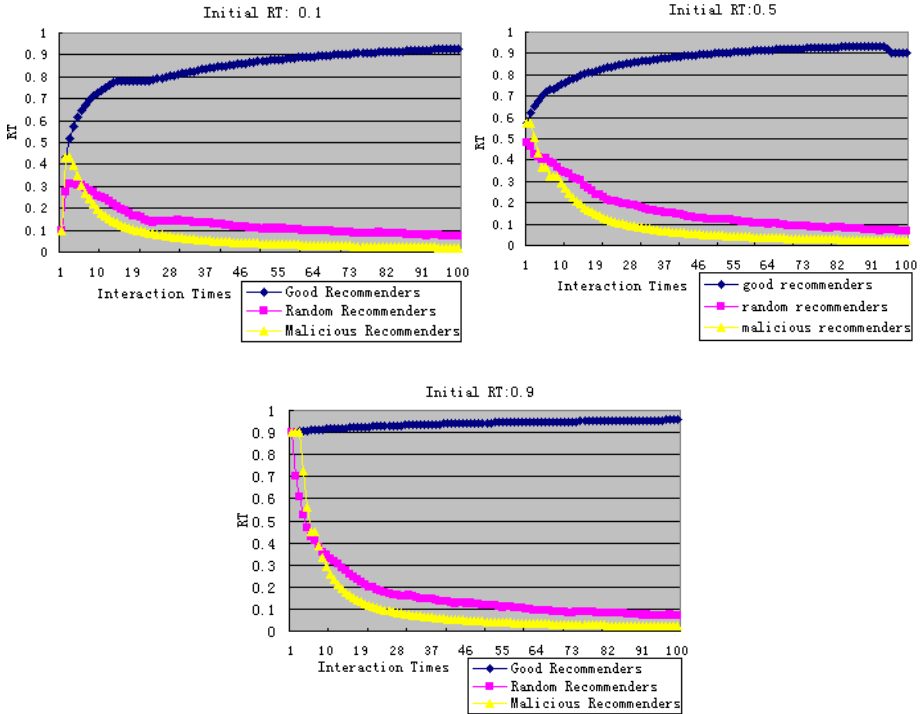


Fig. 6. The average RT of recommenders

Fig.7 shows the effects of the recommendations to the evaluation toward the objects. The evaluator requests the resources hold by the objects with different fixed success rates: 0.1, 0.5 and 0.9. From the simulation results, the evaluations for the objects without recommendations are fluctuant more distinctly than those with recommendations, especially in the case that the objects have medium success rates (e.g. 0.5). This fact implies that the recommendations can help the evaluator evaluate reasonably and steadily, especially when the evaluator hasn't enough local experiences. For the objects with the lower or higher success rates, the effects of the recommendations to them are smaller than those with the medium success rates, because in this case the patterns of the objects' behaviors are relatively steady: success or fail. However, in the Internet peers usually have to interact with the anonymous ones. It is hard to identify which success rate the peer has statically in the P2P networks. In such case, applying recommendation mechanism will bring great benefits to the P2P networks.

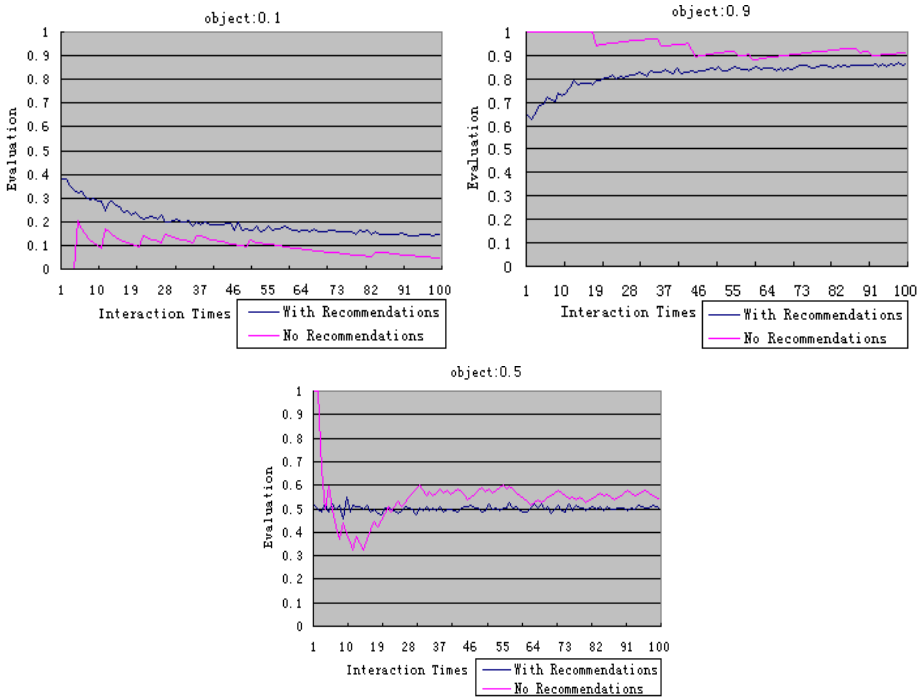


Fig. 7. The evaluation results

5 Conclusions

This paper presents a trust model to handle the trust problems for P2P networks including 1) forming direct trust and recommendation trust toward particular peers automatically, 2) evolving trust dynamically according to the interaction results and experiences, 3) calculating the particular trust depending on the information of other relevant contexts, 4) propagating trust information among peers and combining collected recommendations. It presents a reasonable abstraction to describe the peers' trust-related behaviors in the P2P networks formally. It can support decentralized trust management formally and effectively. This model can be used to form peer alliances quickly, assist to make correct decisions, resist the malicious information and so on. It is helpful to enhance the robustness, fault tolerance and security of the P2P networks, especially for those pressing for form alliances dynamically.

Acknowledgement

This paper is funded by 973 of China (2002CB312002), 863 Program of China (2005AA113160, 2005AA119010), and NSFC (60233010, 60403014).

References

1. William, K.J., Sirer, E.G., Schneider, F.B.: Peer-to-Peer Authentication with a Distributed Single Sign-On Service. IPTPS, pp. 250–258 (2004)
2. Resnickand, Zeckhauserand, R., Friedman, E., Kuwabara, K.: Reputation systems. *Communications of the ACM* 43(12) (2000)
3. Abdul-Rahman, A., Hailes, S.: A distributed trust model. In: *Proceedings of, New Security Paradigms Workshop*, pp. 48–60. ACM Press, Cumbria (1998), <http://www.ib.hu-berlin.de/kuhlen/VERT01/abdul-rahman-trust-model1997.pdf>
4. Beth, T., et al.: Valuation of Trust In Open Network. In: *Proceedings of European Symposium On Research in Security*, pp. 3–18. Springer, Brighton (1994)
5. Jøsang, A., Knapskog, S.J.: A metric for trusted systems. *Global IT Security*. Wien: Austrian Computer Society 541–549 (1998)
6. English, C., Nixon, P., Terzis, S., et al.: Dynamic Trust Models for Ubiquitous computing Environments. In: *First Workshop on Security in Ubiquitous Computing at the Fourth Annual conference on Ubiquitous computing* (October 2002)
7. ebay: <http://www.ebay.com>
8. taobao: <http://www.taobao.com>
9. Gambetta, D.: Can we trust trust? In: Gambetta, D. (ed.) *Trust: Making and Breaking Cooperative Relations*, pp. 213–237. Oxford Press, Blackwell (1990)
10. Kinader, M., Rothermel, K.: Architecture and Algorithms for a Distributed Reputation Systems. In: *Proceeding of First International Conference on Trust Management, iTrust 2003*, Heraklion, Crete, Greece, pp. 1–16 (May 2003)

An Adaptive Trust Control Model for a Trustworthy Component Software Platform

Zheng Yan and Christian Prehofer

Nokia Research Center, Helsinki, Finland

{zheng.z.yan, christian.prehofer}@nokia.com

Abstract. Trust has been recognized as a vital factor for a component software platform. Inside the platform, trust of a platform entity can be controlled according to its assessment result. Special control modes can be applied in order to ensure a trustworthy system. In this paper, we present an adaptive trust control model in order to support autonomic trust management for the component software platform. This model is based on a Fuzzy Cognitive Map. It includes the quality attributes of the platform entity and a number of control modes supported by the platform in order to ensure the entity's trustworthiness. The parameters of this model can be adaptively adjusted in order to reflect real system context. The simulation results show that this model is effective for automatically predicting and selecting feasible control modes for a trustworthy platform. It also helps studying cross-influence of applied control modes on a number of quality attributes.

1 Introduction

The growing importance of component software introduces special requirements on trust due to the nature of applications they provide; in particular, when the software system supports components joining and leaving at runtime. The system also needs to support different trust requirements from the same or different components. We adopt a holistic notion of trust which includes several properties, such as security, availability and reliability, depending on the requirements of a trustor. Hence trust is the assessment of a trustor on how well the observed behavior (quality attributes) of a trustee meets the trustor's own standards for an intended purpose [1]. From this, two critical characteristics of trust can be summarized. First, it is subjective, different for each entity in a certain situation; and, second, dynamic, sensitive to change due to the influence of many factors.

A number of trusted computing and management work have been conducted in the literature and industry, which mostly focus on some specific aspects of trust. For example, TCG (Trusted Computing Group) aims to build up a trusted computing device on the basis of a secure hardware chip [2]. Some of trust management systems focus on protocols for establishing trust in a particular context, generally related to security requirements. Others make use of a trust policy language to allow the trustor to specify the criteria for a trustee to be considered trustworthy [3]. However, the focus on the security aspect of trust tends to assume that the other non-functional requirements

[4], such as availability and reliability, have already been addressed. In addition, TCG based trusted computing solution can not handle the runtime trust management issues of component software.

Recently, many mechanisms and methodologies are developed for supporting trustworthy communications and collaborations among computing nodes in distributed systems [5-7]. These methodologies are based on digital modeling of trust for trust evaluation and management. However, most of existing solutions focus on the evaluation of trust, whilst lack a proposal regarding how to manage trust based on the evaluation result. They generally ignore the influence of trust control mechanisms on the trustworthiness. We found that these methods are not feasible for supporting the trustworthiness of a device software platform.

Regarding software engineering, trust has been recognized as an important factor for the component software platform. A couple of interesting models have been proposed to ensure the quality of component services at runtime and protect the users [8, 9]. However, we found that the trust model proposed in [8] mainly focuses on the runtime component configuration support, while the model in [9] aims to prevent that a component user sends wrong reports resulting in a bad trust value of the component, especially for component downloading. We argue that trust can be controlled according to its evaluation result. Special control mechanisms can be applied into the software platform at runtime in order to ensure a trustworthy system.

In this paper, we propose an adaptive trust control model for autonomic trust management to satisfy both characteristics of trust. We assume several trust control modes, each of which contains a number of control mechanisms or operations, e.g. encryption, authentication, hash code based integrity check, access control mechanisms, duplication of process, man-in-middle solution for improving availability, etc. A control mode can be treated as a special configuration of trust management that can be provided by the system. Based on a runtime trust assessment, the main objective of autonomic trust management is to ensure that a suitable set of control modes are applied in the system. As we have to balance several trust properties in this model, we make use of a fuzzy cognitive map to model the factors related to trust for control mode prediction and selection based on the Sigmoid function. Particularly, we use the trust assessment result as a feedback to autonomously adapt weights in the adaptive trust control model in order to find a suitable set of control modes in a special context.

The rest of the paper is organized as follows. Section 2 specifies the basic notion of autonomic trust management for a component software platform. Section 3 presents the trust control model and algorithms used for control mode prediction and selection, and the context-aware adaptive model adjustment. Section 4 reports our simulation results. Finally, conclusions and future work are presented in Section 5.

2 Autonomic Trust Management for Component Software Platform

As defined in [3], trust management is concerned with the following: collecting the information required to make a trust relationship decision; evaluating the criteria related to the trust relationship as well as monitoring and reevaluating existing trust relationships; and automating the process. We think that this concept needs to be

extended in order to provide software platform trust. We employ autonomic trust management as we proposed in [10], which includes four aspects such as trust establishment, trust monitoring, trust assessment and trust control/re-establishment.

We consider a component software platform which is composed of a number of entities, e.g. a component (composition of components), an application, a sub-system and the whole platform system. The trustworthiness of an entity depends on a number of quality attributes of this entity. The quality attributes can be the entity's trust properties (e.g. security, availability and reliability) and recommendations or reputations with regard to this entity. The decision or assessment of trust is conducted based on the trustor's (e.g. a platform user or his/her delegate) subjective criteria and the trustee entity's quality attributes, as well as influenced by context information. The context includes any information that can be used to characterize the situation of the involved entities. The quality attributes of the entity can be controlled or improved via applying a number of trust control mechanisms.

Based on the above understanding, we propose a procedure to conduct autonomic trust management in the component software platform targeting at a trustee entity specified by a trustor entity, as shown in Figure 1.

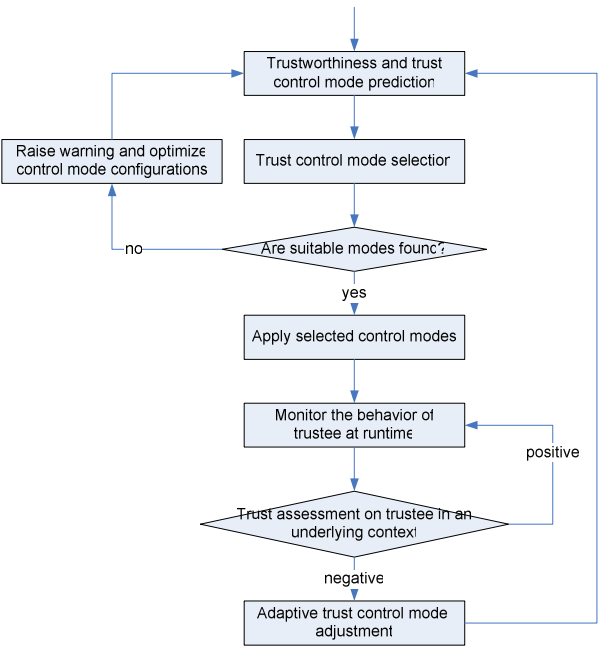


Fig. 1. Autonomic trust management procedure at runtime

Trust control mode prediction is a mechanism to anticipate the performance or feasibility of applying some control modes before taking a concrete action. It predicts the trust value supposed that some control modes are applied before the decision to

initiate those modes is made. Trust control mode selection is a mechanism to select the most suitable trust control modes based on the prediction results.

For a trustor, the trustworthiness of its specified trustee can be predicted regarding various control modes supported by the system. Based on the prediction results, a suitable set of control modes could be selected to establish the trust relationship between the trustor and the trustee. Further, a runtime trust assessment mechanism is triggered to evaluate the trustworthiness of the trustee through monitoring its behavior based on the instruction of the trustor's criteria, as described in [10]. According to the runtime trust assessment results in the underlying context, the system conducts trust control model adjustment in order to reflect the real system situation if the assessed trustworthiness value is below an expected threshold. This threshold is generally set by the trustor to express its real expectation on the assessment. Then, the system repeats the procedure. The context-aware or situation-aware adaptability of the trust control model is crucial to re-select suitable control modes in order to fulfill automatic trust management.

3 Trust Control Modeling

Fuzzy cognitive map is a good method to analyze systems that are otherwise difficult to comprehend due to the complex relationships between their components [11]. In this section, we introduce a trust control model via applying the theory of the fuzzy cognitive map in order to illustrate the relationships among trust, its influencing factors and the control modes used for managing it.

3.1 Trust Control Model

A platform entity's trustworthiness is influenced by a number of quality attributes $QA_i (i = 1, \dots, n)$. These quality attributes are ensured or controlled through a number of control modes supported by the platform system $C_j (j = 1, \dots, m)$. A control mode contains a number of control mechanisms or operations that can be provided by the system. We assume that the control modes are not exclusive and that combinations of different modes are used.

The model can be described as a graphical illustration using a fuzzy cognitive map, as shown in Figure 2. It is a signed directed graph with feedback, consisting of nodes and weighted arcs. Nodes of the graph are connected by signed and weighted arcs representing the causal relationships that exist between the nodes. There are three layers of nodes in the graph. The node in the top layer is the trustworthiness of the platform entity. The nodes located in the middle layer are the quality attributes of the entity, which have direct influence on the entity's trustworthiness. The nodes at the bottom layer are control modes that could be supported and applied inside the system. These control modes can control and thus improve the quality attributes. Therefore, they have indirect influence on the trustworthiness of the entity.

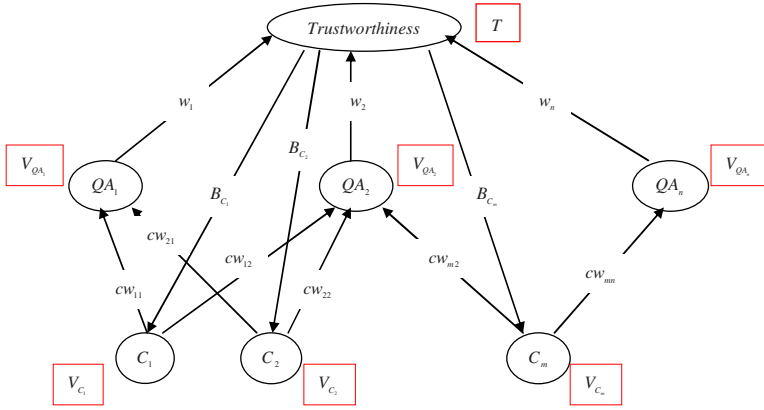


Fig. 2. Graphical modeling of trust control

Note that $V_{QA}, V_{C_j}, T \in [0, 1]$, $w_i \in [0, 1]$, and $cw_{ji} \in [-1, 1]$. T^{old} , V_{QA}^{old} and $V_{C_j}^{old}$ are old value of T , V_{QA} , and V_{C_j} , respectively. $\Delta T = T - T^{old}$ stands for the change of trustworthiness value. B_{C_j} reflects the current system configurations about which control modes are applied. The trustworthiness value can be described as:

$$T = f\left(\sum_{i=1}^n w_i V_{QA_i} + T^{old}\right), \quad (1)$$

such that $\sum_{i=1}^n w_i = 1$. Where w_i is a weight that indicates the importance rate of the quality attribute QA_i regarding how much this quality attribute is considered at the trust decision or assessment. w_i can be decided based on the trustor's criteria. We apply the Sigmoid function as a threshold function $f: f(x) = \frac{1}{1 + e^{-\alpha x}}$ (e.g. $\alpha = 2$), to map node values V_{QA}, V_{C_j}, T into $[0, 1]$. The value of the quality attribute is denoted by V_{QA} . It can be calculated according to the following formula:

$$V_{QA} = f\left(\sum_{j=1}^m cw_{ji} V_{C_j} B_{C_j} + V_{QA}^{old}\right), \quad (2)$$

where cw_{ji} is the influence factor of control mode C_j to QA_i , cw_{ji} is set based on the impact of C_j to QA_i . Positive cw_{ji} means a positive influence of C_j on QA_i . Negative cw_{ji} implies a negative influence of C_j on QA_i . B_{C_j} is the selection factor of the control mode C_j , which can be either 1 if C_j is applied or 0 if C_j is not applied.

The value of the control mode can be calculated using

$$V_{C_j} = f\left(T \cdot B_{C_j} + V_{C_j}^{old}\right). \quad (3)$$

3.2 Trust Control Mode Prediction and Selection

The control modes are predicted through evaluating all possible modes and their compositions based on the proposed model using the prediction algorithm described below. As a standard for predicting new modes, we introduce a constant δ , which is the accepted ΔT that controls the iteration of the prediction.

- For every composition of control modes, i.e. $\forall S_k (k=1, \dots, K)$, while $\Delta T_k = T_k - T_k^{old} \geq \delta$, do

$$V_{C_j,k} = f(T_k \cdot B_{C_j,k} + V_{C_j,k}^{old})$$

$$V_{QA,k} = f\left(\sum_{j=1}^m c w_{ji} V_{C_j,k} B_{C_j,k} + V_{QA,k}^{old}\right)$$

$$T_k = f\left(\sum_{i=1}^n w_i V_{QA,k} + T_k^{old}\right)$$

The control modes are selected based on the control mode prediction results:

- Calculate selection threshold $tr = \frac{\sum_{k=1}^K T_k}{K}$;
- Compare $V_{QA,k}$ and T_k of S_k to tr , set selection factor $SF_{S_k} = 1$ if $\forall V_{QA,k} \geq tr \wedge T_k \geq tr$; set $SF_{S_k} = -1$ if $\exists V_{QA,k} < tr \vee \exists T_k < tr$;
- For $\forall SF_{S_k} = 1$, calculate the distance of $V_{QA,k}$ and T_k to tr as $d_k = \min\{|V_{QA,k} - tr|, |T_k - tr|\}$; For $\forall SF_{S_k} = -1$, calculate the distance of $V_{QA,k}$ and T_k to tr as $d_k = \max\{|V_{QA,k} - tr|, |T_k - tr|\}$ only when $V_{QA,k} < tr$ and $T_k < tr$;
- If $\exists SF_{S_k} = 1$, select the best winner with the biggest d_k ; else $\exists SF_{S_k} = -1$, select the best loser with the smallest d_k .

Herein, the selection threshold (tr) is the average of trust value T_k of all

$S_k (k=1, \dots, K)$, i.e. $tr = \frac{\sum_{k=1}^K T_k}{K}$. $S_k (k=1, \dots, K)$ can be expressed by the control mode selection factors B_{C_j} , which represents which control mode is selected and applied in the system. The selection factor $SF_{S_k} = 1$ means that all the predicted $V_{QA,k}$ and T_k are above the threshold tr . While $SF_{S_k} = -1$ means there is some predicted $V_{QA,k}$ and T_k below the threshold tr . The selection algorithm selects the best control modes based on the absolute difference between $V_{QA,k}$, T_k and tr . For $\forall SF_{S_k} = 1$, it records the absolute difference between $V_{QA,k}$, T_k and tr as the smallest $d_k = \min\{|V_{QA,k} - tr|, |T_k - tr|\}$. For $\forall SF_{S_k} = -1$, it records the absolute difference between $V_{QA,k}$, T_k and tr as the biggest $d_k = \max\{|V_{QA,k} - tr|, |T_k - tr|\}$, only when $V_{QA,k} < tr$ and $T_k < tr$. Thus, the algorithm can

select the best winner if $\exists SF_{S_i} = 1$. Even though, there is no choice available, it is also possible for the algorithm to select the best loser with the biggest $V_{QA,k}$ and T_k below tr . Selecting the best loser is significant for the system to optimize the configurations of the control modes in order to re-predict and re-select a proper set of control modes.

3.3 Adaptive Trust Control Model Adjustment

It is important for the trust control model to reflect the real system situation and context precisely. The influencing factors of each control mode should be context-aware. The trust control model should be dynamically maintained and optimized in order to reflect the real system situation. Thereby, it is sensitive to indicate the influence of each control mode on different quality attributes in a dynamically changed context. For example, when some malicious behaviors or attacks happen, the currently applied control modes can be found not feasible based on trust assessment. In this case, the influencing factors of the applied control modes should be adjusted in order to reflect the real system situation. Then, the system can automatically re-predict and re-select a set of new control modes in order to ensure the trustworthiness. In this way, the system can avoid using attacked or useless trust control modes in a special context. As can be seen from above analysis, an adaptive trust control model is vital for supporting autonomic trust management in the component software platform.

We apply observation based trust assessment as described in [10], which can play as the feedback for adaptive model adjustment. Herein, we use $V_{QA\_monitor}$ and $V_{QA\_predict}$ to stand for V_{QA} generated based on real system observation (i.e. the trust assessment result) and by prediction, respectively. Concretely, the influencing factor cw_{ji} can be further adjusted based on two schemes in order to make it match real system situation. One of the schemes is an equal adjustment scheme. It holds a strategy that each control mode has the same impact on the deviation between $V_{QA\_monitor}$ and $V_{QA\_predict}$. In this scheme, all related cw_{ji} will be adjusted equally. The other is an unequal adjustment scheme. It holds a strategy that the control mode with the biggest absolute influencing factor always impacts more on the deviation between $V_{QA\_monitor}$ and $V_{QA\_predict}$. In this scheme, we always select the biggest absolute influencing factor to adjust. Which one should be applied depends on experimental experiences on the control mode's influence on the quality attributes. In the schemes, ω is a unit deduction factor and σ is the accepted deviation between $V_{QA\_monitor}$ and $V_{QA\_predict}$. We suppose C_j with cw_{ji} is currently applied in the system. The equal adjustment scheme is:

- While $|V_{QA\_monitor} - V_{QA\_predict}| > \sigma$, do
 - a) If $V_{QA\_monitor} < V_{QA\_predict}$, for $\forall cw_{ji}$,
 - $cw_{ji} = cw_{ji} - \omega$, if $cw_{ji} < -1, cw_{ji} = -1$;
 - Else, for $\forall cw_{ji}$,
 - $cw_{ji} = cw_{ji} + \omega$, if $cw_{ji} > 1, cw_{ji} = 1$
 - b) Run the control mode prediction function

The unequal adjustment scheme is:

- While $|V_{QA\_monitor} - V_{QA\_predict}| > \sigma$, do
 - a) If $V_{QA\_monitor} < V_{QA\_predict}$, for $\max(|cw_{ji}|)$,
 - $cw_{ji} = cw_{ji} - \omega$, if $cw_{ji} < -1, cw_{ji} = -1$ (warning);
 - Else, $cw_{ji} = cw_{ji} + \omega$, if $cw_{ji} > 1, cw_{ji} = 1$ (warning)
 - b) Run the control mode prediction function

4 Examples and Simulations

The simulation is based on a practical example, as shown in Figure 3. The trustworthiness of the trustee is influenced by three quality attributes: QA_1 - Security; QA_2 - Availability; QA_3 - Reliability, with important rates $w_1 = 0.6$, $w_2 = 0.2$, and $w_3 = 0.2$, respectively. There are three control modes that could be provided by the system:

- C_1 : security mode 1 with light encryption and light negative influence on availability.
- C_2 : security mode 2 with strong encryption, but medium negative influence on availability.
- C_3 : fault management mode with positive improvement on availability and reliability.

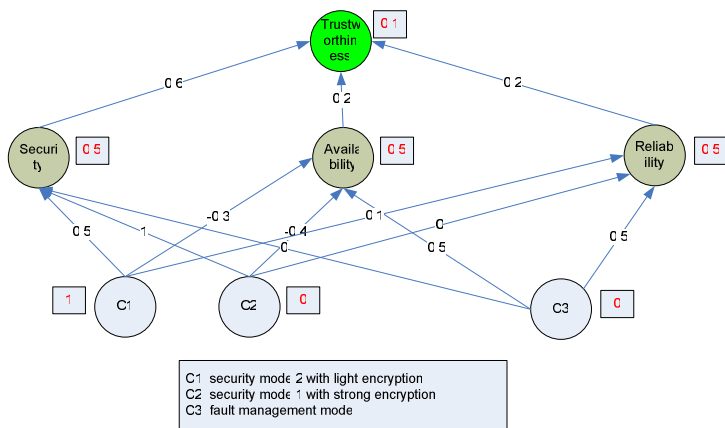


Fig. 3. Simulation configurations

The influence of each control mode to the quality attributes is specified by the arc weights. Its initial value can be set based on the experimental results tested at the control mode development. The values in the square boxes are initial values of the nodes. In practice, the initial values can be set as asserted ones or expected ones, which are specified in the trustor's criteria profile. Actually, the initial values have no influence on the final results of the prediction and selection.

The simulation results are shown in Figure 4. In this case, there are seven control mode compositions: S_1 ($B_{C_1}=1; B_{C_2}=0; B_{C_3}=0$); S_2 ($B_{C_1}=0; B_{C_2}=1; B_{C_3}=0$); S_3 ($B_{C_1}=0; B_{C_2}=0; B_{C_3}=1$); S_4 ($B_{C_1}=1; B_{C_2}=0; B_{C_3}=1$); S_5 ($B_{C_1}=0; B_{C_2}=1; B_{C_3}=1$); S_6 ($B_{C_1}=1; B_{C_2}=1; B_{C_3}=0$); S_7 ($B_{C_1}=1; B_{C_2}=1; B_{C_3}=1$). We can see that S_4 (the composition of C_1 and C_3) is the best choice since both the quality attribute values and the trustworthiness value are above the threshold.

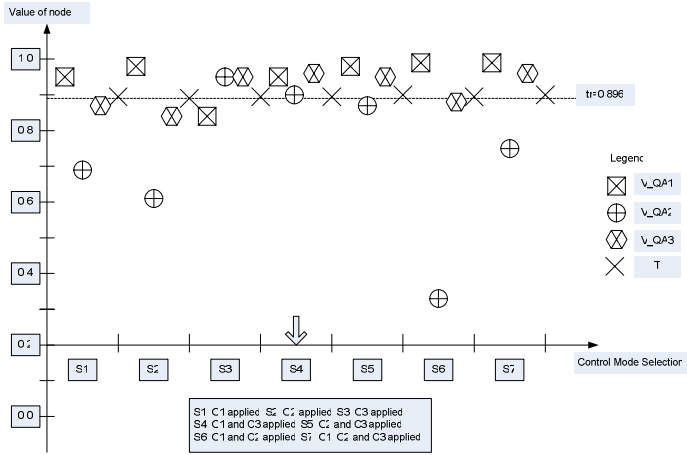


Fig. 4. Control mode prediction and selection result ($\alpha = 2$ and $\delta = 0.0001$)

If S_4 is applied but the assessed values of quality attributes based on runtime observation are not the same as the predicted ones (e.g. $V_{QA_1\_predict} = 0.946$, $V_{QA_2\_predict} = 0.899$; $V_{QA_3\_predict} = 0.956$), the trust control model should be adjusted in order to reflect real system context. Supposed that the assessed $V_{QA\_monitor}$ are: $V_{QA_1\_monitor} = 0.92$, $V_{QA_2\_monitor} = 0.70$, and $V_{QA_3\_monitor} = 0.956$. In this case, the security attribute is a bit worse than prediction and the availability attribute is definitely predicted incorrectly. The mismatch indicates that the underlying model parameters do not reflect real system situation precisely. This could be caused by some attacks happening at the control mechanisms in S_4 with regard to ensuring the availability, or raised by limited resources shared by many system entities, or due to weaker influence of S_4 on the availability in practice than prediction. We conducted model adjustment based on the equal and unequal schemes, respectively. The adjustment simulation results are shown in Table 1. Both schemes can adjust the model with similar predicted V_{QA} to the assessment results, as shown in Table 2. The deviation between $V_{QA\_predict}$ and $V_{QA\_monitor}$ can be controlled through parameter σ . As can be seen from the simulation results, both schemes can adjust the influencing factors to

make the prediction values of QA $V_{QA\_predict}$ match the assessment results $V_{QA\_monitor}$ generated through observation.

Table 1. Trust control model adjustment results ($\sigma = 0.002, \omega = \sigma / 20$)

Influencing factors cw_{ji}	Original values of cw_{ji}	Adjusted values of cw_{ji} based on equal adjustment scheme	Adjusted values of cw_{ji} based on unequal adjustment scheme
cw'_{11}	0.5	0.41	0.32
cw'_{12}	-0.3	-0.54	-0.58
cw'_{13}	0.1	0.1	0.1
cw'_{21}	1.0	1.0	1.0
cw'_{22}	-0.4	-0.4	-0.4
cw'_{23}	0.0	0.0	0.0
cw'_{31}	0.0	-0.089	0.0
cw'_{32}	0.5	0.26	0.30
cw'_{33}	0.5	0.5	0.5

Table 2. Prediction results after model adjustment

QA names	Old prediction values	Predicted values after applying equal adjustment scheme	Predicted values after applying unequal adjustment scheme
QA_1	0.9463273922157238	0.9219866210377154	0.9219866322353456
QA_2	0.8992891226563186	0.7015233411962816	0.7015269858257399
QA_3	0.9562688296373892	0.9562688296373892	0.9562688296373892

We further run the control mode prediction and selection functions with two sets of adjusted cw_{ji} listed in Table 1, respectively. The results show the system can not offer a good selection. This means that the system needs to re-configure its control modes in order to improve its trustworthiness. In both cases, the selection function indicates the best loser is S_3 . The prediction and selection results after the model adjustment are shown in Figure 5.

The adaptability of a trust model can be defined as the speed of the model to reflect the real system situation and act accordingly. The proposed model can be dynamically maintained according to the real system situation. For example, new control modes can be added and ineffective ones can be removed. The parameters of the model (e.g. cw_{ji}) can be adjusted based on the model adjustment result. The adaptability is controlled by the parameters α , δ , σ , and $\omega = \sigma / 20$. The parameters α and δ influence the speed of the prediction. The smaller the parameter α and/or the bigger the parameter δ are/is, the faster the speed of prediction. But generally, δ can not be set very big since it will influence the correctness. The parameters σ and ω are applied

to control the speed of the model adjustment. The bigger the parameter σ is, the faster the adjustment, but worse the preciseness of the adjustment. With regard to the parameter ω , the bigger it is, the faster the adjustment. But ω can not be set too big, which may lead missing a solution (i.e. the algorithm cannot return an answer of adjustment). Based on our simulation, we suggest setting $\omega = \sigma / 20$. We should select σ properly in order to keep preciseness and meanwhile ensure adaptability. In summary, adaptability is the most important factor that influences the effectiveness of the trust control model.

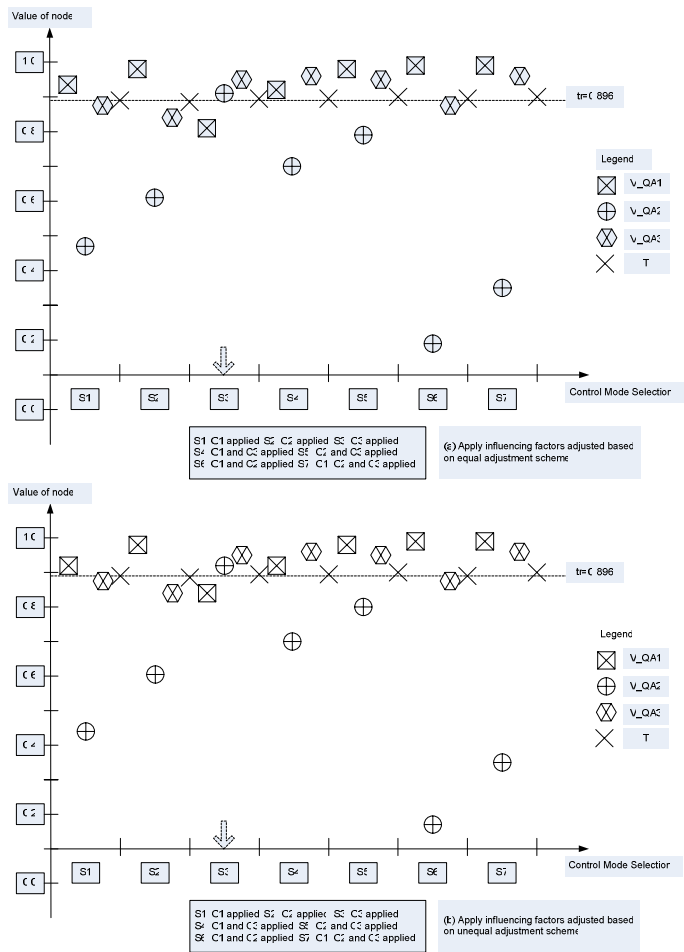


Fig. 5. Control mode prediction and selection results after model adjustment ($\alpha=2$ and $\delta=0.0001$) (a) model adjusted based on equal adjustment scheme; (b) model adjusted based on unequal adjustment scheme

5 Conclusions and Future Work

In this paper, we proposed an adaptive trust control model in order to support automatic trust management for the component software platform. This model is based on a fuzzy cognitive map. It includes such nodes as the trustworthiness of a platform entity, quality attributes of the entity and a number of control modes supported by the platform in order to ensure the entity's trustworthiness. In this model, the importance factors are set based on the trustor's preference. The influencing factors of the control modes can be adaptively adjusted according to the trust assessment in order to reflect real system context and situation. The simulation results show that this model is effective for automatically predicting and selecting feasible control modes for a trustworthy platform. It could also help improving the control mode configurations, especially when there is no solution available from the prediction, as well as studying cross-influence of applied control modes on a number of quality attributes. In addition, this model is flexible to cooperate with the trust assessment mechanism to realize automatic trust management on any system entity in the component software platform. The system entity can be a system component, a sub-system or the whole system.

For future work, we will further study the performance of the model adjustment schemes, control mode re-configuration strategies and attempt to embed this model into Trust4All platform [12].

References

- [1] Denning, D.E.: A New Paradigm for Trusted Systems. In: Proceedings of the IEEE New Paradigms Workshop (1993)
- [2] TCG TPM Specification v1.2 (2003)
<https://www.trustedcomputinggroup.org/specs/TPM/>
- [3] Grandison, T., Sloman, M.: A Survey of Trust in Internet Applications. *IEEE Communications and Survey*, Forth Quarter 3(4), 2–16 (2000)
- [4] Banerjee, S., Mattmann, C.A., Medvidovic, N., Golubchik, L.: Leveraging Architectural Models to Inject Trust into Software Systems. *ACM SIGSOFT Software Engineering Notes*. In: Proceedings of the 2005 workshop on software engineering for secure systems—building trustworthy applications SESS '05, vol. 30(4) (2005)
- [5] Zhang, Z., Wang, X., Wang, Y.: A P2P Global Trust Model Based on Recommendation. In: Proceedings of 2005 International Conference on Machine Learning and Cybernetics, vol. 7, pp. 3975–3980 (2005)
- [6] Lin, C., Varadharajan, V., Wang, Y., Pruthi, V.: Enhancing Grid Security with Trust Management. In: Proceedings of IEEE International Conference on Services Computing, pp. 303–310 (2004)
- [7] Sun, Y., Yu, W., Han, Z., Liu, K.J.R.: Information Theoretic Framework of Trust Modeling and Evaluation for Ad Hoc Networks. *IEEE Journal on Selected Area in Communications* 24(2), 305–317 (2006)
- [8] Zhou, M., Mei, H., Zhang, L.: A Multi-Property Trust Model for Reconfiguring Component Software. In: The Fifth International Conference on Quality Software QAIC 2005, pp. 142–149 (2005)

- [9] Herrmann, P.: Trust-Based Protection of Software Component Users and Designers. In: Nixon, P., Terzis, S. (eds.) *iTrust 2003*. LNCS, vol. 2692, Springer, Heidelberg (2003)
- [10] Yan, Z., MacLavery, R.: Autonomic Trust Management in a Component Based Software System. In: Yang, L.T., Jin, H., Ma, J., Ungerer, T. (eds.) *ATC 2006*. LNCS, vol. 4158, pp. 279–292. Springer, Heidelberg (2006)
- [11] Kosko, B.: Fuzzy Cognitive Maps. *International Journal Man-Machine Studies* 24, 65–75 (1986)
- [12] Robocop, Space4U and Trust4All website: <https://nlsvr2.ehv.campus.philips.com/>

Towards Trustworthy Resource Selection: A Fuzzy Reputation Aggregation Approach

Chunmei Gui, Quanyuan Wu, and Huaimin Wang

School of Computer Science
National University of Defense Technology
410073, Changsha, China
plantsperfume@yahoo.com.cn

Abstract. To guarantee trustworthiness and reliability of resource selection, entity's reputation is a key factor that decides our selection, no matter who is provider or consumer. Built on top of idea of SOA, based on fuzzy logic methods of optimal membership degree, the approach is efficient to deal with uncertainty, fuzziness, and incompleteness of information in systems, and finally builds instructive decision. By applying the approach using eBay transaction statistical data, the paper demonstrates the final integrative decision order in various conditions. Compared with other methods, this approach has better overall consideration, accords with human selection psychology naturally.

1 Introduction

Grid computing has greatly promoted the development of information acquiring and applying. Network services and online trade, as network bank, and e-Business, are so popular that it seems a tendency to replace traditional counter business. Yet, resource sharing and accessing has broken the boundary of administrative domain, spanning from closed, acquaintance-oriented and relatively static intro-domain computing environment to the open, decentralized and highly dynamic inter-domain computing environment. The wide scale of resources and the high strangeness among entities complicate the decision of resource selection. It is challenging to make a reliable and trustworthy selection in such wide distributed and pervasively heterogeneous computing environment.

Reputation mechanism provides a way for building trust through social control by utilizing community based feedback about past experiences of entities to help making recommendation and judgment on quality and reliability of the transactions [1]. For the similarity of reputation relations between real society and virtual computing environment, reputation is promising to perform well in Grid. It can be expected that the reliability and trustworthiness of resource selection will be improved, and further promote the efficient resource sharing and service sharing in Grid.

Reputation is multi-faceted concept [2], which means the reputation status of resource often has multiple aspects, such as capability, honesty, recommending, history value, fresh behavior evaluation and so on. Meanwhile, each facet of

resource reputation is a time-correlate variable, its change is influenced by service capability of resource itself, service consciousness, and service environment. Furthermore, resource selection often behaves as a multi-objective action that service consumer in different background emphasizes differently on service sort, amount and degree of request. Confronted with so multi-facets reputation conditions and multi-objective selection request, how to scientifically evaluate them, reasonably integrate information, and further make the final reliable selection? It is just what we focus on solving in this paper-evaluation and decision making on reputation with multi-facet and multi-objective. The main idea is building sequence with fuzzy relative optimal membership logic and selection is made from within.

The rest of this paper is structured as follows: in Section 2, fuzzy optimal solution models of evaluation and decision method are introduced. In Section 3, analyze metrics of evaluation and explain relative optimal based trustworthy resource selection approach by means of a case study. In Section 4, related work is briefed and compared. Finally in Section 5, we summarize future work and conclude the whole paper.

2 Fuzzy Optimal Solution Models of Evaluation and Multi-objective Decision Making

Generally speaking, evaluation means the behavior of specifying the objectives, measuring entity's attributes, and turning them into subjective effect (which will satisfy what the evaluator demands to a certain degree). It is increasingly difficult to evaluate entity's reputation because different factors pervade one another. The complex, random, and uncertainty of reputation information needs special consideration. The theory of fuzzy set is an efficient tool to solve those complex decision problems which contain fuzzy uncertainty factors. Guided by fuzzy set theory [3], according to the given evaluation metrics and measure value, synthetically considering these objectives which might conflict one another, we can evaluate entities after transform with fuzzy method, and finally provide the most satisfied scheme to the decision maker.

2.1 The Basic Idea of Evaluation and Decision Making

According to the entities' different features and forms provided in evaluation, the typical evaluation might be described as model (1):

$$\max_{x \in X} \{f(x)\}. \quad (1)$$

Where X stands for decision making space or feasible field, x is evaluation variable, and $f(x) = (f_1(x), f_2(x), \dots, f_m(x))^T$ is vector function of objective (the total objective number is m , and m is positive integer). As different objective might conflict each other, decision maker's fuzzy partial information is necessary to be considered when selecting the satisfied solution.

Definition 1. Deem that $\tilde{A}_i$ is a fuzzy subset in $[m_i, M_i]$ ($i = 1, 2, \dots, m$), where m_i and M_i are the lowest and the highest boundary of $f_i(x)$ in decision space X . Respectively, the membership degree of $\tilde{A}_i$ on y is $\mu_{\tilde{A}_i}(y) (y \in [m_i, M_i])$. If $\mu_{\tilde{A}_i}(y)$ is a strict monotone increasing function of $y (y \in [m_i, M_i])$, and $\mu_{\tilde{A}_i}(M_i) = 1$, then $\tilde{A}_i$ is a fuzzy optimal set of $f_i(x)$. Correspondingly, $\mu_{\tilde{A}_i}(y)$ is the optimal membership degree of $y (y \in \tilde{A}_i)$.

Definition 2. Deem that $\tilde{f}_i$ is a fuzzy subset on domain $X_i = \{x | m_i \leq f_i(x) \leq M_i, x \in X\}$ ($i = 1, 2, \dots, m$), whose membership degree on x is $\mu_{\tilde{f}_i}(x) (x \in X_i)$. If there is a fuzzy optimal set of $f_i(x)$, naming $\tilde{A}_i$, which can satisfy

$$\mu_{\tilde{f}_i}(x) = \begin{cases} \mu_{\tilde{A}_i}(f_i(x)) & (f_i(x) \in [m_i, M_i]) \\ 0 & (f_i(x) \in (-\infty, m_i) \cup (M_i, +\infty)) \end{cases}$$

shorten writing as $\mu_i(x) = \mu_{\tilde{f}_i}(x)$, then $\tilde{f}_i$ is the fuzzy optimal points set of $f_i(x)$ in model (1). Accordingly, $\mu_i(x)$ stands for the optimal membership degree of fuzzy optimal points $x \in X_i$.

Decision maker's partial information can be embodied through selecting membership degree $\mu_{\tilde{A}_i}(y)$, and further it is convenient to select within space X . Model (1) can be converted to model (2) after deciding $\mu_i(x) (i = 1, 2, \dots, m)$ for $f_i(x)$, i.e.

$$\max_{x \in X} \{\mu(x)\}, \quad (2)$$

where $\mu(x) = (\mu_1(x), \mu_2(x), \dots, \mu_m(x))^T \in [0, 1]^m \subseteq R^m$. We often name $[0, 1]^m$ is m -dimension membership degree space. Each membership degree of objective is a value in $[0, 1]$, and then it is convenient for comparison and analysis.

Definition 3. Deem that $\tilde{F}$ is a fuzzy set on domain $\tilde{X} = \bigcap_{i=1}^m X_i$, whose membership degree on x is $\mu_{\tilde{F}}(x) (x \in \tilde{X})$. If there is a fuzzy optimal points set $\tilde{f}_i (i = 1, 2, \dots, m)$ of $f_i(x)$, which satisfies $\mu_{\tilde{F}}(x) = h(\mu(x))$, where function $h(t)$ is strict monotone increasing in $t \in [0, 1]^m$ and there is $h(t, t, \dots, t) = t$ for any $t \in [0, 1]$, then $\tilde{F}$ is the fuzzy optimal points set for model (2). Accordingly, $\mu_{\tilde{F}}(x)$ is the optimal membership degree of fuzzy optimal points $x \in \tilde{X}$. The optimal membership degree of x_j about objective f_i is μ_{ij} .

Definition 4. Deem that $\tilde{F}$ is the fuzzy multi-objective optimal points set of model (2). If $x^* \in \tilde{X}$ satisfies $\mu_{\tilde{F}}(x^*) = \max_{x \in \tilde{X}} \{\mu_{\tilde{F}}(x)\}$, then x^* is the fuzzy optimal solution of model (2) about $\tilde{F}$. Accordingly, $\mu_{\tilde{F}}(x^*)$ is the optimal membership degree of x^* .

The optimal solution of $\max_{x \in \tilde{X}} \{\mu_{\tilde{F}}(x)\}$ is the solution of evaluation and decision making, and the membership degree according with which stands for the extent of decision maker's satisfactory degree. The objective style, real problem characteristics, and decision maker's request are basis factors when determining relative optimal membership.

2.2 Determination Method of Relative Optimal Membership Degree for Objectives

Usually, objectives includes: benefit style objective, cost style objective, fixation style objective (the nearer to some certain value, the better), and interval style objective (within some certain interval is good). If we mark the set made up of all the subscript of $f_i (i = 1, 2, \dots, m)$ as $O = \{1, 2, \dots, m\}$, then the O can be divided as four subset, i.e. $O^k (k = 1, 2, 3, 4)$, which is separated as sign set for the four kinds objectives.

(1) For benefit style objective, optimal membership degree can be:

$$\mu_{ij} = (f_{ij}/f_{i\max})^{p_i} \quad (i \in O^1). \quad (3)$$

where p_i is parameter determined by decision maker, demands all $f_{ij} \geq 0$ ($i \in O^1; j = 1, 2, \dots, n$).

(2) For cost style objective, relative optimal degree of objective can be:

$$\mu_{ij} = \begin{cases} 1 - (f_{ij}/f_{i\max})^{p_i} & (f_{i\min} = 0) \\ (f_{i\min}/f_{ij})^{p_i} & (f_{i\min} \neq 0) \end{cases} \quad (i \in O^2), \quad (4)$$

demands all $f_{ij} \geq 0 (i \in O^2; j = 1, 2, \dots, n)$.

(3) For fixation style objective, relative membership degree of objective can be:

$$\mu_{ij} = \begin{cases} 1 & (f_{ij} = f_i^*) \\ 1 - (|f_{ij} - f_i^*|/\sigma_i)^{p_i} & (f_{ij} \neq f_i^*) \end{cases} \quad (i \in O^3) \quad (5)$$

$$\sigma_i = \max_{1 \leq j \leq n} \{|f_{ij} - f_i^*|\} \quad (i \in O^3),$$

where f_i^* is the optimal value provided by decision maker for the i^{th} objective $f_i (i \in O^3)$.

(4) For interval style objective, relative optimal membership of objective can be:

$$\mu_{ij} = \begin{cases} 1 - [(f - f_{ij})/\eta_i]^{p_i} & (f_{ij} < \bar{f}_i) \\ 1 & (f_{ij} \in [\bar{f}_i, \bar{\bar{f}}_i]) \\ 1 - [(f_{ij} - \bar{\bar{f}}_i)/\eta_i]^{p_i} & (f_{ij} > \bar{\bar{f}}_i) \end{cases} \quad (i \in O^4) \quad (6)$$

$$\eta_i = \max \{ \bar{f}_i - f_{i\min}, f_{i\max} - \bar{\bar{f}}_i \} \quad (i \in O^4),$$

where closed interval $[\bar{f}_i, \bar{\bar{f}}_i]$ is the optimal interval value provided by decision maker for the i^{th} objective $f_i (i \in O^4)$.

3 Trustworthy Resource Selection Based on Relative Optimal Membership Degree

In this section, we first sum up the typical, influential, and reputation-correlative evaluation metrics, which embody both the entities' multi-facets reputation conditions and the decision maker's multi objectives in application background, and then build up optimal membership degree based trustworthy resource evaluation and decision making method by means of a case study. In the case, different methods for different decision making psychology and character are sufficiently demonstrated.

3.1 Evaluation Metrics

When evaluating entities' reputation, we should take the metrics below into unified consideration:

- Selection overhead: overhead of selecting optimum entity for providing service to consumer. Adopting the selection made by reputation evaluation and decision making system, consumer takes on overhead as little as possible in usual situation.
- Service overhead: the necessary cost an entity must pay to provide knight service, such as bandwidth, capacity, man-hour, material and energy etc. The value should satisfy the consumer and accord with real industry condition. Too higher value might mean service provider costs too much, and the costs might be converted as unnecessary burden to consumer. Conversely, too lower value might mean QoS can't reach consumer's anticipation.
- Service performance: a popular metric, including quality of service, service efficiency, maintenance after sale etc.
- Fresh density of reputation: perfect service amount of an entity during late unit time, which is one of the most representative reputation metric to embody entity's latest reputation conditions, and mostly is attached more importance by enterprising consumers.
- Perfect rate of reputation in history statistic: it can provide relative comprehensive data to embody entity's entire reputation condition, which is interested in by those traditional consumers.
- Operating ability of resisting disaster: it embodies the ability that an entity could recover to its former nice reputation condition when reputation value collapses or shakes acutely (for example: feed back of market aroused by entity's subjective or objective nonstandard actions, or entity suffered from malicious attack etc.), which is an important representation of entity's immanent consciousness and capability.

3.2 A Case Study

Deem that $X = \{x_1, x_2, x_3, x_4, x_5\}$ stands for 5 computing resource providers, consider 6 aspects for evaluation: selection overhead (f_1), service overhead (f_2),

Table 1. Resource providers' aggregated reputation information in 6 aspects

F	x_1	x_2	x_3	x_4	x_5
f_1	1250	750	1370	1250	2200
f_2	250	984	766	1861	2161
f_3	0.34	0.23	0.39	0.36	0.29
f_4	83	110	130	234	176
f_5	14	25	10	26	14
f_6	middle	good	poor	good	poor

service performance (f_3), fresh density of reputation (f_4), perfect rate of reputation in history statistic (f_5), operating ability of resisting disaster (f_6). The 5 providers' statistic data of aggregated reputation in 6 aspects are given in Table 1, which is the to-be-evaluated information system:

Using the linear mode of $\mu_{ij} = (f_{ij}/f_{i\max})^{p_i}$ for benefit style objectives f_3 , f_4 , and f_5 , using linear mode of $\mu_{ij} = (f_{i\min}/f_{ij})^{p_i}$ ($f_{i\min} \neq 0$) for cost style objective f_1 , using $\mu_{ij} = f_i^*/(f_i^* + |f_{ij} - f_i^*|)$ ($i = 2$) for fixation objective f_2 and considering the optimal value $f_2^* = 1340$ requested in special application background, choosing optimal membership degree 1.0, 0.75, 0.50 for fuzzy judgments of good, middle, and poor, we convert table 1 into the relative optimal membership degree matrix μ .

$$\mu = \begin{pmatrix} 0.60 & 1.0 & 0.55 & 0.60 & 0.34 \\ 0.55 & 0.79 & 0.70 & 0.72 & 0.62 \\ 0.87 & 0.59 & 1.0 & 0.92 & 0.74 \\ 0.35 & 0.47 & 0.56 & 1.0 & 0.75 \\ 0.54 & 0.95 & 0.38 & 1.0 & 0.54 \\ 0.75 & 1.0 & 0.5 & 1.0 & 0.5 \end{pmatrix}. \quad (7)$$

In order to embody the personal partialness or expecting of decision maker and field expert, we select the weighted vector $\omega = (0.24, 0.18, 0.18, 0.12, 0.12, 0.16)^T$ for f_i ($i = 1, 2, \dots, 6$). According to matrix μ and weighted form of $\mu_{ij}^{\omega_i}$, we can also get weighted relative optimal membership degree matrix $\bar{\mu}$.

(1) Maximin method. Using $\mu_{i^*j^*} = \max_{1 \leq j \leq n} \min_{1 \leq i \leq m} \{\mu_{ij}\}$, we get the total order $x_4 \succ x_2 \succ x_3 \succ x_1 \succ x_5$ (not considering ω) and $x_2 \succ x_4 \succ x_1 \succ x_3 \succ x_5$ (considering ω). Reason of difference between this two orders lies in that decision maker thinks the importance of different objective differently.

(2) Maximax method. Using $\mu_{i^*j^*} = \max_{1 \leq j \leq n} \max_{1 \leq i \leq m} \{\mu_{ij}\}$, we get the total order $x_4 \approx x_2 \approx x_3 \succ x_1 \succ x_5$ (not considering ω) and $x_4 \approx x_2 \approx x_3 \succ x_1 \succ x_5$ (considering ω).

It is obvious that maximin method is pessimism while maximax method is optimism. Balancing between them, we get the tradeoff coefficient method.

(3) Tradeoff coefficient method. Theory: If $x_{j^*} \in X$ satisfy: $\gamma \max_{1 \leq i \leq m} \{\mu_{ij^*}^{\omega_i}\} + (1 - \gamma) \min_{1 \leq i \leq m} \{\mu_{ij^*}^{\omega_i}\} = \max_{1 \leq j \leq n} \left\{ \gamma \max_{1 \leq i \leq m} \{\mu_{ij}^{\omega_i}\} + (1 - \gamma) \min_{1 \leq i \leq m} \{\mu_{ij}^{\omega_i}\} \right\}$, then x_{j^*}

is the most satisfied selection, $\gamma \in [0, 1]$ is the tradeoff coefficient, and the rest can be deduced similarly. Obviously, $\gamma = 0$, this method is the same of weighted maximin method, which embodies the traditional idea or hating risk thinking of decision maker; $\gamma = 1$, this method is the same of weighted maximax method, which embodies the optimism or braveness of decision maker; $\gamma = 1/2$, the decision maker is neutralist. Generally speaking, the value of γ for a certain decision maker is relatively steady.

(4) Minimum membership degree deviation method. Theory: Do comparison among the selections in X , and the nearest to the ideal scheme, the best. Define the relative optimal membership degree of ideal selection x^+ as $g = (g_1, g_2, \dots, g_m)^T$, where $g_i = \max_{1 \leq j \leq n} \{\mu_{ij}\}$ ($i = 1, 2, \dots, m$), it means the maximum of relative optimal membership degree for the i^{th} objective f_i ($i = 1, 2, \dots, m$). Weighted Minkowski distance is used to describe how close the x_j ($j = 1, 2, \dots, n$) to the ideal selection x^+ :

$$d_q(x_j, x^+) = \left\{ \sum_{i=1}^m [\omega_i (g_i - \mu_{ij})]^q \right\}^{1/q}, \quad (8)$$

where q is parameter. If $x_{j^*} \in X$ satisfy $d_q(x_{j^*}, x^+) = \min_{1 \leq j \leq n} \{d_q(x_j, x^+)\}$, then x_{j^*} is the most satisfied selection, and further the total order can be drawn according to $d_q(x_j, x^+)$. In this case, we get $g = (1.0, 0.79, 1.0, 1.0, 1.0, 1.0)^T$. With (7), (8) and ω : $q = 1$, the total order is $x_4 \succ x_2 \succ x_3 \succ x_1 \succ x_5$; $q = 2$, the total order is $x_2 \succ x_4 \succ x_1 \succ x_3 \succ x_5$; $q \rightarrow \infty$, the total order is $x_2 \succ x_4 \approx x_1 \succ x_3 \succ x_5$. Though the three orders are not completely the same, the first selection and the second selection are always x_2 and x_4 in the sequence, therefore, selecting x_2 and x_4 as the satisfying scheme are suitable.

(5) Maximum membership degree deviation method. Theory: Do comparison among the selections in X , and the farthest from minus-ideal scheme, the best. Define the relative optimal membership degree of minus-ideal selection x^- as $b = (b_1, b_2, \dots, b_m)^T$, where $b_i = \min_{1 \leq j \leq n} \{\mu_{ij}\}$ ($i = 1, 2, \dots, m$), it means the minimum of relative optimal membership degree for the i^{th} objective f_i ($i = 1, 2, \dots, m$). Weighted Minkowski distance is used to describe how close the x_j ($j = 1, 2, \dots, n$) to the minus-ideal selection x^- :

$$d_q(x_j, x^-) = \left\{ \sum_{i=1}^m [\omega_i (\mu_{ij} - b_i)]^q \right\}^{1/q}. \quad (9)$$

If $x_{j^*} \in X$ satisfies $d_q(x_{j^*}, x^-) = \max_{1 \leq j \leq n} \{d_q(x_j, x^-)\}$, then x_{j^*} is the most satisfied selection, and further the total order can be drawn according to $d_q(x_j, x^-)$. In this case, we get $b = (0.34, 0.55, 0.59, 0.35, 0.38, 0.50)^T$. With (7), (9) and ω : $q = 1$, the total order is $x_4 \succ x_2 \succ x_3 \succ x_1 \succ x_5$; $q = 2$ and $q \rightarrow \infty$, the total order is $x_2 \succ x_4 \succ x_3 \succ x_1 \succ x_5$.

Method (5) is based on the relative degree of farness from minus-ideal scheme, while method (4) is based on the relative degree of nearness to ideal scheme. However, in some evaluation problem, a scheme is nearer to the ideal but might not be farther from the minus-ideal, for example, in this case, x_1 is nearer to ideal than x_3 ($q = 2$) while x_3 is farther from minus-ideal than x_1 ($q = 2$). So, it is not sufficient to consider single factor. Considering both these two factors, the Relative ratio method is given.

(6) Relative ratio method. Denote that

$$\begin{cases} d_q(x^-) = \max_{1 \leq j \leq n} \{d_q(x_j, x^-)\} \\ d_q(x^+) = \min_{1 \leq j \leq n} \{d_q(x_j, x^+)\} \end{cases} \quad (10)$$

Define the relative ratio of scheme $x_j \in X$ as

$$\xi(x_j) = d_q(x_j, x^-)/d_q(x^-) - d_q(x_j, x^+)/d_q(x^+) \quad (j = 1, 2, \dots, n) \quad (11)$$

$\xi(x_j)$ embodies both the degree $x_j \in X$ near to x^+ and far from x^- . With formula (10) and (11), direct validation can prove $\xi(x_j) \leq 0$ ($j = 1, 2, \dots, n$). If $x_{j^*} \in X$ satisfies $d_q(x^-) = d_q(x_{j^*}, x^-)$ and $d_q(x^+) = d_q(x_{j^*}, x^+)$, then $\xi(x_{j^*}) = 0$, and x_{j^*} is the most satisfying selection. In this case, with formula (7), (10), (11) and ω : $q = 1$, the total order is $x_4 \succ x_2 \succ x_3 \succ x_1 \succ x_5$; $q = 2$ and $q \rightarrow \infty$, the total order is $x_2 \succ x_4 \succ x_1 \succ x_3 \succ x_5$.

Different method embodies different request of decision maker who has special characteristics, and final selection can be done according to the order. It is apt to get the total order with anterior three methods, because they are relative simple, while posterior three methods are relative complex. To present a more intuitionistic and clearer understanding, in figure 1, the solutions acquired with posterior three methods are accurately demonstrated.

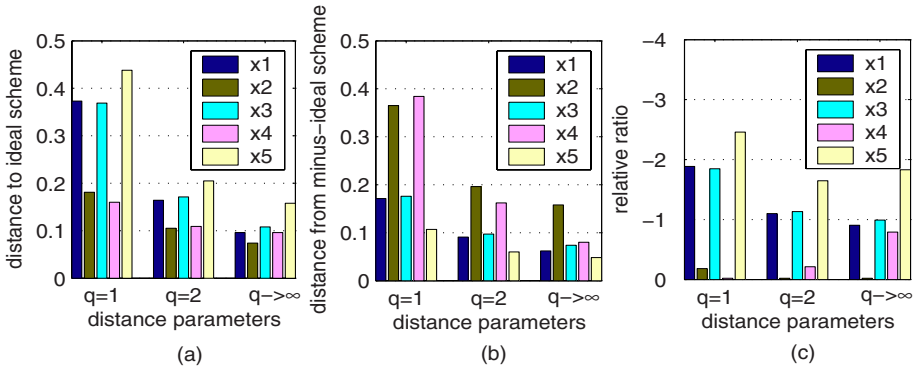


Fig. 1. (a) distance of each scheme to ideal scheme in three distance parameters. (b) distance of each scheme from minus-ideal scheme in three distance parameters. (c) relative ratio of each scheme in three distance parameters.

4 Related Work

Undoubtedly, reputation is not only of great helpful to subjective selection in humanities, but also important as a formalizing computational concept in scientific computing field. The list we present is some representative systems or mechanisms.

In [4], for the first time, formalizing trust as a computational concept is proposed, which provides a clarification of trust and present a mathematics model for precise trust discussion. In [5], the conception of trust management is used to explain the fact that security decision needs accessorial security information. In aspect of computer security or electronic privacy, paper offers considerations, which represent advances of that time in the theory, design, implementation, analysis, or empirical evaluation of secure systems, either for general use or for specific application domains. In [6], a trust modeling is presented which aims at providing resources security protection in grid through trust updating, diffusing and integrating among entities. In [7], Grid Eigen Trust, a framework used to compute entity's reputation in grid. It adopts a hierarchical model and is performed from 3 levels: VO, Institution and Entity, that is, an entity's reputation is computed as weighted average of new and old reputation values, an institution's reputation is computed as the eigenvalue of composing entities' reputation matrix, and a VO's reputation is computed as weighted average of all composing institution's reputation value. In [8], "personalized similarity" is adopted to evaluate an entity's credibility. First, getting the intersection of one's own rating set and the evaluatee's rating set, then, computing the deviation of this set. The less the deviation, the more credible the entity is. In [9], "the propagation of distrust", an interesting idea, which allows the proactive dissemination of some malicious entity's bad reputation and maintains positive trust values for peers at the meanwhile. In [10][11], model is the main focus field, yet resource selection method is scarce.

However, currently available resource selection methods, including some commerce system, neither take multi-facet of resource reputation into consideration nor address the problem of decider's multi-objective in grid environment. As stated in Section 1, the features that distinguish our work from the existing methods are: the real reputation facets of resources, the multi-objective of selection and the variety request of decider colony, are adequately respected.

5 Conclusions and Future Work

With the blend of Grid and SOA, grid application is increasingly abundant and extensive. The guarantee of high trustworthiness holds the balance for secure sharing and efficient collaboration among entities in wide distributed, dynamic domain. In this paper, resource selection and reputation mechanism are unified considered. As reputation is uncertain, we base our method on fuzzy logic. As the selection is multi-objective, we build relative optimal membership to model resource providers' inferior and superior relationship, and by means of information integration we provide the final order, which is used to guide the final

resource selection. Compared with other methods, this method considers reputation's multi-facet nature and decider's multi-objective selection request, belongs to decision making based on multiple attributes, and has better maneuverability.

For the future, we suggest that dishonest feedback filtering should be taken into consideration, since it is significative to evaluate and make decision basing on genuine feedback. Converting the approach provided in this paper to a product would be great helpful for whatever Grid or society life.

References

1. Resnick, P., Zeckhauser, R., Friedman, E., Kuwabara, K.: Reputation Systems. *Communications of the ACM* 43(12), 45–48 (2000)
2. Yao, W., Julita, V.: Trust and Reputation Model in Peer-to-Peer Networks. In: *Proceedings of the 3rd IEEE International Conference on Peer-to-Peer Computing*, pp. 150–158. IEEE Computer Society, Linkping (2003)
3. Dengfeng, L.: Fuzzy Multiobjective Many-person Decision Makings and Games[M]. National Defense Industry Press, Beijing (2003)
4. Marsh, S.: Formalising trust as a computational concept. PhD Thesis. Scotland, University of Stirling (1994)
5. Blaze, M., Feigenbaum, J., Lacy, J.: Decentralized trust management. In: Dale, J., Dinolt, G. (eds.) *Proceedings of the 17th Symposium on Security and Privacy*, pp. 164–173. IEEE Computer Society Press, Oakland (1996)
6. Shanshan, S., Kai, H., Mikin, M.: Fuzzy Trust Integration for Security Enforcement in Grid Computing. In: Jin, H., Gao, G.R., Xu, Z., Chen, H. (eds.) *NPC 2004*. LNCS, vol. 3222, Springer, Heidelberg (2004)
7. Beulah, K.A.: Grid EigenTrust: A Framework for Computing Reputation in Grids. MS thesis, Department of Computer Science, Illinois Institute of Technology (November 2003)
8. Xiong, L., Lin, L.: PeerTrust: Supporting Reputation-Based Trust in Peer-to-Peer Communities. *IEEE Transactions on Knowledge and Data Engineering*, Special Issue on Peer-to-Peer Based Data Management 16(7) (July 2004)
9. Guha, R.: Propagation of Trust and Distrust. In: *Proc. ACM World Wide Web Conference(WWW2004)*, pp. 403–412. ACM Press, New York (2004)
10. Yan, S., Wei, Y., Zhu, H., Liu, K.J.R.: Information theoretic framework of trust modeling and evaluation for ad hoc networks, *Selected Areas in Communications*, *IEEE Journal on* 24(2) (February 2006)
11. Florina, A., Andres, M., Daniel, D., Juan, S.: Developing a Model for Trust Management in Pervasive Devices. In: *Third IEEE International Workshop on Pervasive Computing and Communication Security (PerSec 2006)*, at *Fourth Annual IEEE International Conference on Pervasive Computing and Communications* (March 2006)

An Adaptive Spreading Activation Approach to Combating the Front-Peer Attack in Trust and Reputation System*

Yufeng Wang¹, Yoshiaki Hori², and Kouichi Sakurai²

¹ College of Telecommunications and Information Engineering, Nanjing University of Posts and Telecommunications, Nanjing 210003, China

² Department of Computer Science and Communication Engineering, Kyushu University, Fukuoka 812-0053, Japan
wfwang@njupt.edu.cn

Abstract. It is argued that group-based trust metric is effective in resisting attacks, which evaluates groups of assertions “in tandem”, and generally computes trust ranks for sets of individuals according to peers’ social position in trust network. Thus, the group-based trust value should be called “reputation rank”. But, unfortunately, most group-based trust metrics are vulnerable to the attack of front peers, which represent these malicious colluding peers always cooperate with others in order to increase their reputation, and then provide misinformation to promote actively malicious peers. In this paper, we proposed adaptive spreading activation approach to mitigating the effect of front peer attack, in which adaptive spreading factor is used to reflect the peer’s recommendation ability according to behaviors of the peer’s direct/indirect children in trust network; Simulation results show that the adaptive spreading activation approach can identify and mitigate the attack of front peer.

1 Introduction

Recently, there exist great effort on the research and application of completely decentralized and open networks like P2P and sensor and ad hoc networks. These systems provide higher degree of scalability, flexibility and autonomy etc., but are vulnerable to various types of attacks. To protect participants in those systems (so-called peers) from malicious intentions, peers should be able to identify reliable peers for communication, which is a challenging task in highly dynamic P2P environments. So, the importance of social control mechanism, that is, reputation and trust management, became more and more crucial in open networks and electronic communities. In real society, the network structure emanating from our very person, composed of trust statements linking individuals, constitutes the basis for trusting people we do not know personally. The structure has been dubbed “Web of Trust” [1].

* Research supported by the NSFC Grants 60472067, JiangSu education bureau (5KJB510091) and State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications (BUPT).

Such trust network is a fundamental building block in many of today's most successful e-commerce and recommendation systems. Different propagation schemes for both trust score and distrust score are studied based on a network from a real social community website [2]. The classification of trust metrics is provided in [3], in which trust metrics are distinguished as scalar and group metric. Scalar metrics analyze trust assertions independently, while group trust metrics evaluate groups of assertions "in tandem". Specifically, scalar metrics compute trust between two given individuals through tracking trust paths from sources to targets and not performing parallel evaluations of groups of trust assertions. Scalar trust metrics fail to resist easily-mounted attacks, and attacks against these poor trust metrics have been used to argue that a centralized identity service is needed [4]. On the other hand, group trust metrics generally compute trust ranks for sets of individuals according to peer's social position in web of trust, that is, "reputation rank".

Generally, the adversaries can employ various technologies to attack against the P2P trust and reputation system (see [5] for various attacks in P2P trust and reputation system), especially, sybil attack and front peer attack. Due to the open, anonymous nature of many P2P networks, a peer can create sybils (in large numbers) - who are able to link to (or perform false transactions with) each other and the original user to improve the original user's reputation. It is argued that using social network structure with reputation makes it more difficult to spoof the system by creating false identities or colluding in groups. That is, false identities would either give themselves away by connecting to their old friends, or remain disconnected, in which case they will have poor social ranking [6]. Specifically, group-based Advogato trust metric is designed in Ref. [7], which has the property of attack-resistance. More recently, Ref. [8] argues that there is no symmetric sybilproof reputation function (by symmetry it means that trust metric is invariant under a renaming of the peers, i.e, it depends only on the edge structure of the trust graph), and investigate the conditions for sybilproofness in general asymmetric reputation function (asymmetric reputation function may assume that some specified nodes are trusted, and propagate trust from those nodes). Maxflow-based subjective reputation is adopted to combat collusion among peers [9]. Ref. [3] proposes a local group-based trust metric, Appleseed algorithm, based on spreading activation model to propagate trust value from source peer, and argue this algorithm possesses the property of attack-resistance. EigenTrust algorithm assigns a universal measure of trust (reputation value) to each peer in P2P system (analogous to the PageRank measure for web pages, so-called, global group-based trust metric), which depends on the ranks of referring peers, thus entailing parallel evaluation of relevant nodes thanks to mutual dependencies [10]. But those above approaches are vulnerable to so-called front peer attack. Front peer represent these malicious colluding peers always cooperate with others in order to increase their reputation, and then provide misinformation to promote actively malicious peers. Some authors [11-12] argue that the only way to combat front peer attack is to divide trust into functional trust (trust the peer's ability to provide service) and referral trust (trust the peer's ability to recommend service). But in this paper, we investigate an alternative way to combat the attack of front peer. Specifically, we propose adaptive spreading activation approach to mitigating the effect of front peer attack, in which adaptive spreading factor is used to reflect the peer's recommendation ability according to behaviors of the peer's direct/indirect children in trust graph.

The paper is organized as follows: section 2 briefly provides the concept of group trust metrics, the basic spreading activation model and the original Appleseed algorithm. Considering several disadvantages in original Appleseed algorithm, adaptive spreading activation approach is provided in section 3 to alleviate front-peer attack. Section 4 briefly introduces the simulation settings and provides simulation results, which illustrate the effect of our proposal. Finally, we briefly conclude the paper.

2 Basic Models

Generally, there are three inputs to trust metric: a directed graph, a designated “seed” peer indicating the root of trust, and a “target” peer, then we wish to determine whether the target node is trustworthy. Each edge from s to t in the graph indicates the probability that s believes that t is trustworthy. According to the classification criteria of link evaluation [3], trust metrics are distinguished as scalar and group metric. The entire category of scalar trust metrics fails to resist easily-mounted attacks. While group trust metric is effective in resisting attack, and well suited to evaluating membership in a group, because this evaluation is done over the entire group of nodes, rather than individually for each node. Note that complete trust graph information is only important for global group trust metrics, but not for local ones. Informally, local group trust metrics may be defined as metrics to compute neighborhoods of trusted peers for an individual. One disadvantage of group-based trust metric is that the computed values lack a plausible interpretation on an absolute scale. From the way how the value has been computed it is clear that it cannot be interpreted as the (estimated) probability of the trustworthy behavior of the target peer. Thus, the scenarios in which they can be used should involve ranking the trust values of many peers and selection of the most trustworthy one(s) among them, that is, group-based trustworthiness should be called “reputation rank”. Generally, the concept of reputation is closely linked to that of trustworthiness, but it is evident that there is a clear and important difference. The main differences between trust and reputation systems can be described as follows: trust systems produce a score that reflects the trusting entity’s subjective view of trusted entity’s trustworthiness, whereas reputation systems produce an entity’s (public) reputation score as seen by the whole community. Secondly, transitivity is an explicit component in trust systems, whereas reputation systems usually only take transitivity implicitly into account [5].

Two key steps in the group-based reputation mechanism need be properly solved. One is, for each parent, how to divide its reputation score among its children; we name this the “splitting” step. The other is, for each child, how to calculate the overall scores given the shares from all its parents; we name this the “accumulation” step. For the splitting step, we use squarely weighted splitting, which will be introduced in detail in next section. For the accumulation step, we adopt the rule of simple summation, to sum the reputation score sent by parents.

Ref. [3] proposes Appleseed trust metric (a special kind of local group-based trust metric), which borrows many ideas from spreading activation model in psychology and relates their concepts to trust evaluation in an intuitive fashion. The spreading activation model is briefly given as follows: In a directed graph mode, edge

$(x, y) \in E \subseteq V \times V$ connects nodes $x, y \in E$, which is assigned continuous weights $w(x, y) \in [0, 1]$. Source node s to start the search from is activated through an injection of energy e , which is then propagated to other nodes along edges according to some set of simple rules: all energy is fully divided among successor nodes with respect to their normalized local edge weight, i.e., the higher the weight of an edge, the higher the portion of energy that flows along the edge. Furthermore, supposing average outdegrees greater than one, the closer node x to the injection source s , and the more paths leading from s to x , the higher the amount of energy flowing into x in general. To eliminate endless, marginal and negligible flow, energy streaming into node x must exceed threshold T . In order to interpret energy ranks as trust ranks, Appleseed tailor the above model to trust computation. Specifically, to handle trust decay in node chain and eliminate rank sinks, a global spreading factor d is introduced in Appleseed. Rank sink means that, for example, illustrated in Fig. 1, all trust distributed along edge (c, d) becomes trapped in a cycle and will never accorded to any other nodes but those being part of the cycle, i.e., d, e and f . Thus, those nodes will eventually acquire infinite trust rank.

Normalization is common practice to many trust metrics. However, while normalized reputation or trust seems reasonable for models with plain, non-weighted edges, serious interference occur when edges are weighted. In order to avoid dead ends (nodes with zero outdegree) and mitigate the effect of relative trust value (in weight trust network), Appleseed makes use of back propagation of trust to the source (that is, when metric computation takes place, additional “virtual” edges form every node to the trust source is created). These edges are assigned full trust $w(x, s)=1$, that is, every node is supposed to blindly trust source s . Trust rank (in fact, reputation rank) of x is updated as follows: $trust(x) \leftarrow trust(x) + (1 - d) \cdot in(x)$, where $in(x)$ represents the amount of incoming trust flowed into peer x ; $d \cdot in(x)$ the portion of energy divided among peer x ’s successors.

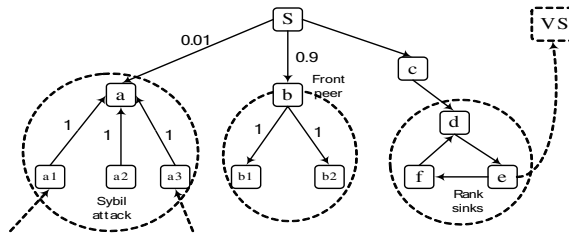


Fig. 1. Illustration of various attacks on P2P reputation system

3 Adaptive Spreading Activation Approach

There exist several detailed problems in the original Appleseed trust metric algorithm.

① As described in previous section, Appleseed adopts backward trust propagation to mitigate the effect of relative trust and avoid dead end, then the accumulated reputation score in source s is propagated again, which not only exerts heavy computation burden,

but creates the additional cycles in trust graph (may lead to inconsistent calculative results). Thus, we introduce the virtual sink (VS) to absorb the backward trust score. Specifically, in the partial trust graph, we artificially add a virtual sink, and additional “virtual” edges from every node to virtual sink are created. These edges are also assigned full trust $w(x, VS)=1$. (Refer to Fig. 1).

② In Appleseed algorithm, the spreading factor d is regarded as the ratio between trust in the ability of peer to recommend others as trustworthy peers and direct trust. This step collapses functional trust and referral trust into a single trust type, which allows for simple computation, but creates a potential vulnerability. A malicious peer can for example behave well during transactions in order to get high normalized trust scores as seen by his local peers, but can report false local trust scores (i.e. too high or too low). By combining good behaviors with reporting false local trust scores, a malicious agent can thus cause significant disturbance in global trust scores (illustrated as front peer attack in Fig. 1). Thus, it is necessary to assign different value d for various peers based on those peers’ recommendation ability (that is, the behaviors of their peers’ direct/indirect children), so-called adaptive spreading factor. But in the algorithm of adaptive spreading factor, there exist serious problem if we still adopt the residual energy value as peer’s trust ranking (used in Appleseed algorithm provided in [3]). For example, intuitively, the spreading factors associated with front peers are relatively smaller than other good peers. It implicitly implies that front peer will keep most of the passed energy, which will make front peer hold higher trust rank than good peers. So, in our paper, we use the passed energy $in(x)$ as the trust rank of peer x , and the energy endowed to its children is $d(x) \cdot in(x)$, where $d(x)$ depends on the behaviors of peer x ’s children.

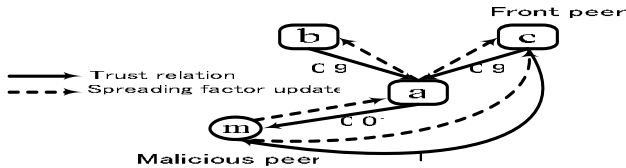


Fig. 2. Illustration: update of adaptive spreading factor

In the paper, we provide adaptive spreading activation approach, which attempts to use adaptive spreading factor d to reflect peers’ recommendation ability. Specifically, once source node recognizes the malicious peer previously recommended by other peers (through direction interaction, etc.), then, from the denoted malicious peer, the spreading factor associated with each related peer is updated along the reverse link in the trust graph according to the following rule (illustrated in Fig. 2):

$$d_u^{new} = \min_{(u,x) \in E} \left\{ (1 - \alpha) \cdot d_u^{old} + \alpha \cdot \left[(d_x^{new} - d_{init}) \cdot w(u, x) + d_{init} \right] \right\} \quad (1)$$

where d_u^{new} (or d_u^{old}) denotes peer u ’s spreading factor after (before) the update; d_{init} represents the initial spreading value (in Appleseed, $d_{init} = 0.85$). α is the learning

rate, a real number in the interval (0,1). Initially, the spreading factor of identified malicious peer is represented as:

$$d_m^{new} = \alpha \cdot d_m^{old} + (1 - \alpha) \cdot \rho_m \cdot d_{init} \quad (2)$$

where ρ_m is the source's direct functional trust on malicious peer. Note that, in order to alleviate the negative effect of updating spreading factor on good peer (for example, good peer points to front peer with relatively high trust value), this paper constrains the update depth with 2, that is, only those peers within the range of two-hop from the identified malicious peer are updated.

So, the procedure of the adaptive spreading activation approach is given as follows:

- Whenever, the source peer finds out the malicious peer, then the source updates those related peers' spreading factor according to Eq. (1) and (2);
- Then, based on the updated spreading factor, the modified Appleseed algorithm is used to calculate the reputation rank of related peers. Specifically, the following equations were used to replace the corresponding parts in original Appleseed algorithm:

$$e_{x \rightarrow y} = d_x^{new} \cdot in(x) \cdot \frac{w(x, y)^2}{\sum_{(x, i) \in E} w(x, i)^2} \quad trust(x) \leftarrow trust(x) + in(x) \quad \text{and}$$

$$trust(x) = in(x) = \sum_{(p, x) \in E} d_p^{new} \cdot in(p) \cdot \left(\frac{w(p, x)^2}{\sum_{(p, i) \in E} w(p, i)^2} \right) \quad \text{where, } e_{x \rightarrow y} \text{ denotes}$$

energy distributed along (x,y) from x to successor node y.

4 Simulation Settings and Results

4.1 Simulation Settings

This subsection describes the general simulation setup, including the peer types, behavior patterns, and the procedure of generating trust network. Specifically, there exist three kinds of peers in our simulation setting: good peer, malicious peer and front peer. Table 1 shows the peer types and behavior patterns used in our simulation.

Table 1. Peer types and behavior patterns in our simulation

	Parameter	Value range
Peer Model	Number of peers in the network	300 1000
	Percentage of good peers	20% 50%
	Percentage of front peers	20% 50%
	Percentage of malicious peers	30%
Behavior patterns	Good Peer: always provide truthful feedback about the transaction party. Front peer: like good peer, except providing false feedback about the malicious peer Malicious peer: provide bad feedback for good peer, and provide good feedback for malicious peer and front peer.	

Naturally, occurring trust networks take a long time to gain a large number of users, and the topological properties are relatively fixed, thus it is necessary to be able to automatically generate trust network models. It is argued that reputation feedback in trust network comply with power-law distribution [13]. Thus, we use the following algorithm to create experimental trust network (Barabási-Albert network. Note that Barabási-Albert model is used to generate scale-free undirected network, and, based on the above peer behavior model, our trust network corresponds to undirected graph).

① Growth: Starting with $m_0 = 50$ nodes, at each round we add $m = 10$ new nodes, each with 5 edges. The total number of peers is 300-1000;

② Preferential Attachment: The probability that a new edge attaches to any of the peer with degree k is $kp_k / \sum_k kp_k$, where p_k is the fraction of nodes in the trust network with degree k . A naive simulation of the preferential attachment process is quite inefficient. In order to attach to a vertex in proportion to its degree we normally need to examine the degrees of all vertices in turn, a process that takes $O(n)$ time for each step of the algorithm. Thus the generation of a graph of size n would take $O(n^2)$ steps overall. A much better procedure, which works in $O(1)$ time per step and $O(n)$ time overall, is the following [15]. In this paper, we maintain a list, in an integer array for instance, that includes k_i entries of value i for each peer i . Then in order to choose a target peer for a new edge with the correct preferential attachment, one simply chooses a number at random from this list. When new peers and edges are added, the list is updated correspondingly. Fig. 3 explicitly illustrates the power-law degree distribution of generated trust network.

4.2 Simulation Results

Front peers' spreading factors in our proposal (adaptive spreading activation approach) and original Appleseed algorithm are illustrate in Fig. 4 (simulation environment: the total number of peer is 500; the percentage of good peer is 0.5, and the percentage of front peer, 0.2; learning rate α equals 0.7; update spreading factor from one identified malicious peer, so-called single update). In Appleseed, the spreading factor is assumed to be constant, i.e., 0.85. In our proposal, the spreading factor is adaptively changed. The closer those front peers are from the malicious peer, the lower spreading factor value they obtained (the bottom points in Fig. 4). But, our proposals belong to the local group trust metrics, which may be defined as metrics to compute neighborhoods of

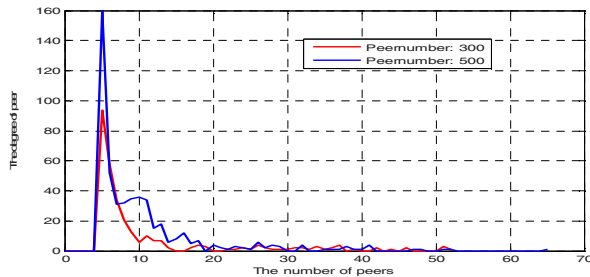


Fig. 3. The power-law degree distribution of generated trust network

trusted peers for an individual. So, the spreading factors of most front peers far away from the designated malicious peer are only slightly affected by the update of the spreading factor. Then, we sequentially select five malicious peers, and recursively run the spreading factor update algorithm (multiple update). The resulted spreading factors of front peers are shown in Fig. 5, which illustrates lower spreading factor for front peers.

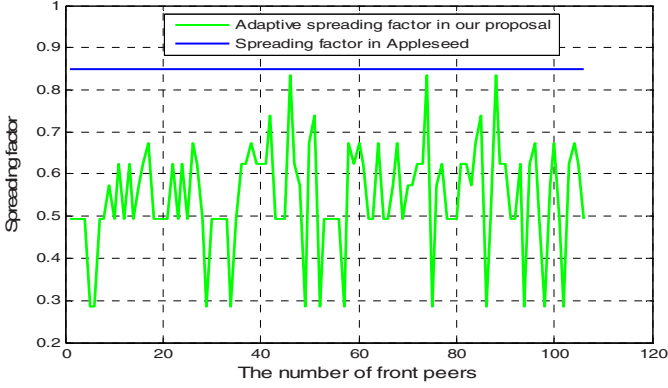


Fig. 4. Comparison of front peers' spreading factors between our proposal and Appleseed (Single update)

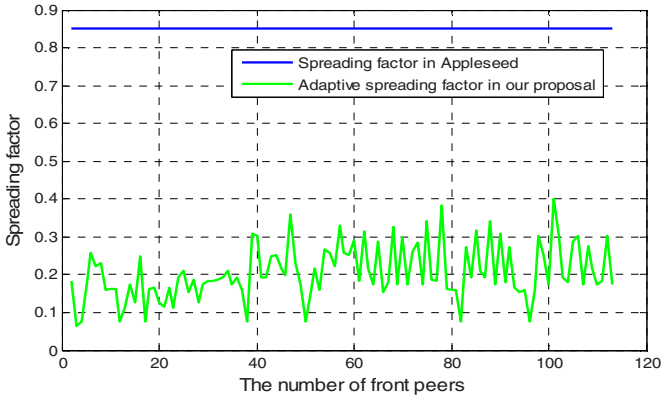


Fig. 5. Comparison of front peers' spreading factors between our proposal and Appleseed (multiple update)

Since the goal of this paper is to investigate how the proposed reputation ranking algorithm, i.e, adaptive spreading activation approach, can help to identify and mitigate the effect of front peers, we will focus on the ranking positions of malicious peers. The percentage of malicious peer in top 20% reputation rank in our proposal and Appleseed algorithm is shown in Fig.6 (simulation setting: total peer number: 1000; learning rate α : 0.7), which illustrate that our proposal can recognize more malicious peers

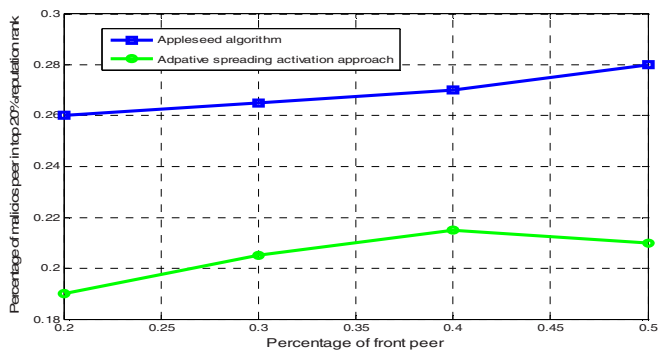


Fig. 6. Percentage of malicious peers in top 20% reputation rank in our proposal and Appleseed

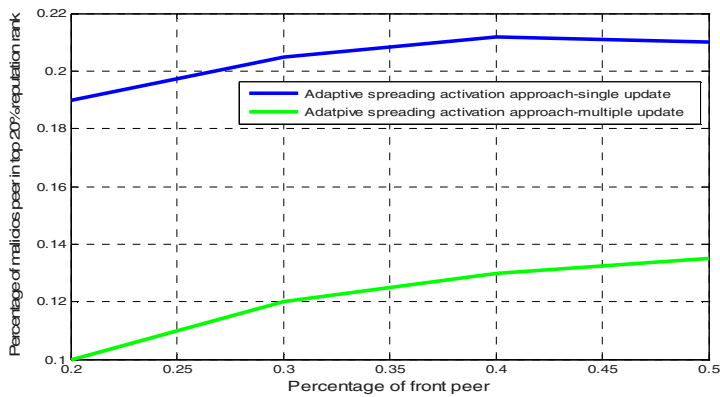


Fig. 7. Percentage of malicious peers in top 20% reputation rank (Single update vs. multiple updates)

recommended by front peers. Fig. 7 compares the percentage of malicious peers' in top 20% reputation rank in two scenarios: single update of spreading factor and multiple (five) updates of spreading factor, that is, the spreading factors shown in Fig. 5 are used to infer the reputation ranks of related peers. Obviously, the multiple updates of spreading factor can identify more front peers, which lead to the less malicious peers in top 20% reputation rank.

5 Conclusion

With the increasing popularity of self-organized communication systems, distributed trust and reputation systems in particular have received increasing attention. Trust metrics compute quantitative estimates of how much trust an agent should accord to its peer, taking into account trust ratings from other persons on the network. These metrics should also act "deliberately", not overly awarding trust to person or agent whose trustworthiness is questionable. Group trust metrics evaluate groups of assertions "in tandem", which have the feature of attack-resistance. But unfortunately, most group-based trust metrics are vulnerable to front peer attack (a special kind of

collusion). In this paper, we argue that group-based trust value should be called as “reputation rank”, and propose new trust propagation and reputation ranking algorithms, adaptive spreading activation approach to identify and mitigate the attack of front peer, which addresses several problems in Appleseed algorithm. Specifically, the spreading factor is regarded as the ratio between trust in the ability of peer to recommend others as trustworthy peers and direct trust, which should be adaptively updated according to the behaviors of peer’s direct (and indirect) children to reflect the current peer’s recommendation ability. Thus, front peers can obtain high reputation rank, but can not pass its reputation rank to malicious peers; Simulation result shows that the algorithm can effectively identify and mitigate the attack of front peer, to which tradition group-based trust metrics and Appleseed are vulnerable.

References

- [1] Golbeck, J., Parsia, B., Hendler, J.: Trust networks on the semantic web. In: Proceedings of the 7th International Workshop on Cooperative Intelligent Agents (2003)
- [2] Guha, R., Kumar, R., Raghavan, P., Tomkins, A.: Propagation of trust and distrust. In: Proceedings of the 13th International World Wide Web Conference (2004)
- [3] Ziegler, C.N., Lausen, G.: Spreading activation models for trust propagation. In: Proceedings of the IEEE International Conference on e-Technology, e-Commerce, and e-Service (2004)
- [4] Douceur, J.R.: The sybil attack. In: Proceedings of 1st International Workshop on Peer-to-Peer Systems (2002)
- [5] Wang, Y.F., Hori, Y., Sakurai, K.: On securing open networks through trust and reputation? architecture, challenges and solutions. In: Proceeding of The 1st Joint Workshop on Information Security (2006)
- [6] Hogg, T., Adamic, L.: Enhancing reputation mechanisms via online social networks. In: Proceedings of ACM Conference on Electronic Commerce (2004)
- [7] Levien, R.: Attack resistant trust metrics. PhD thesis, UC Berkeley, Berkeley, CA, USA (2003)
- [8] Cheng, A., Friedman, E.: Sybilproof reputation mechanisms. In: Proceeding of the ACM SIGCOMM workshop on Economics of peer-to-peer systems (2005)
- [9] Feldman, M., Lai, K., Stoica, I., Chuang, J.: Robust incentive techniques for peer-to-peer networks. In: Proceedings of the 5th ACM conference on Electronic commerce (2004)
- [10] Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The eigentrust algorithm for reputation management in P2P networks. In: Proceedings of the 12th International World Wide Web Conference (2003)
- [11] Aberer, K., Despotovic, Z.: Possibilities for managing trust in P2P networks. EPFL Technical Report, IC/2004/84, Lausanne (2004)
- [12] Jøsang, A., Ismail, R., Boyd, C.: A survey of trust and reputation systems for online service provision. *Decision Support Systems* (2006)
- [13] Zhou, R., Hwang, K.: PowerTrust: A robust and scalable reputation system for trusted P2P computing. *IEEE Transactions on Parallel and Distributed Systems*, 18 (2007)
- [14] Ziegler, C.N., Lausen, G.: Propagation models for trust and distrust in social networks. *Information Systems Frontiers* 7 (2005)
- [15] Newman, M.E.J.: The structure and function of complex networks. *SIAM Review* 45, 167–256 (2003)

Research on Cost-Sensitive Learning in One-Class Anomaly Detection Algorithms^{*}

Jun Luo, Li Ding, Zhisong Pan, Guiqiang Ni, and Guyu Hu

Institute of Command Automation, PLA University of Science and Technology, 210007,
Nanjing, China
{hotpzs, zyqs1981}@hotmail.com

Abstract. According to the Cost-Sensitive Learning Method, two improved One-Class Anomaly Detection Models using Support Vector Data Description (SVDD) are put forward in this paper. Improved Algorithm is included in the Frequency-Based SVDD (F-SVDD) Model while Input data division method is used in the Write-Related SVDD (W-SVDD) Model. Experimental results show that both of the two new models have a low false positive rate compared with the traditional one. The true positives increased by 22% and 23% while the False Positives decreased by 58% and 94%, which reaches nearly 100% and 0% respectively. And hence, adjusting some parameters can make the false positive rate better. So using Cost-Sensitive method in One-Class Problems may be a future orientation in Trusted Computing area.

1 Introduction

Audit information-Based Anomaly Detection System, according to Forrest's research^[1], the key application behavior can be described by the Sequence of System Calls (SSC) used during execution; it is also proved that valid behaviors of a simple application can be described by short sequences which are the partial mode in execution trace. Comparing with the short sequence pool in normal mode, we can find whether current process is running in normal mode or abnormal. If abnormal appears frequently in a fixed monitoring time, system intrusion may be taking place.

At the same time, Cost-Sensitive learning, in which the 'Cost' of different samples should be paid more attention, has become hot in current international machine learning community. In real-world problems, different classification errors often lead to remarkably different losses, while traditional machine learning research assumes that all the classification errors will result in the same loss. The situation stays the same in One-Class classification, in this paper; we try to solve this by improving the Original Classification Model using Cost-Sensitive Learning Methods in which two ways are included. That is, in detail, improvement on algorithm and division on input samples.

Based on these two ways in Cost-Sensitive Learning, two improved Anomaly Detection models using SVDD are put forward in this paper. Experiments using UNM

^{*} Support by the National Natural Science Foundation of China Under Grant No.60603029; the Natural Science Foundation of Jiangsu Province of China Under Grant No.BK2005009.

Intrusion Datasets show that both improvements to the algorithm and further reduction to the input data method result in a elevation to the performance of the detection system model.

The remainder of this paper is organized as follows, in section 2, 3, 4 the original Anomaly Detection and the improved models and their performance evaluation are presented. In section 5, some brief concluding remarks are demonstrated.

2 Support Vector Data Description Anomaly Detection Model

The Support Vector Data Description (SVDD), which was first put forward by David.M.Tax, uses the Kernel method to map the data to the kernel feature space^[2]. By this mapping a hypersphere in the kernel space including almost all the data for training will be formed. A new sample is recognized as normal only if the sample can be included by the hypersphere after kernel mapping. Linear Programming and Neoteric Detection Method^[3] are included in this algorithm.

2.1 Data Pre-processing of SSC

The executing procedure of a certain application can be monitored by the configured audit system^[10], so as to gather the required sequence of system calls. The Short Sequence of System calls which symbolized the pattern of application behavior can be produced by using the K-sized Sliding Window technique^[12].

A large amount of Short Sequence of System calls will be produced after Sliding-Window slicing, they may be stored into the Security Audit Database for further processing. Generally speaking, a lot of repetitive short sequences are produced because most of user applications may use the same system call repetitively. So Data Reduction operation is the first job before the classifier is trained. Redundant short sequences are wiped off so as to avoid extra computation.

Consequently, we have used the UNM data sets for our current study. Table 1 summarizes the different sets and the programs from which they were collected^[11].

Table 1. Amount of data available for each dataset

Program	Intrusions data available		Normal data available	
	Number of traces	Number of system calls	Number of traces	Number of system calls
MIT lpr	1001	169,252	2704	2,928,357
UNM lpr	1001	168,236	4298	2,027,468
Named	5	1800	4	9,230,572
Stide	105	205,935	13,726	15,618,237
Ftp	5	1363	2	180315
UNM Sendmail	25	6755	346	1799764
CERT Sendmail	34	8316	294	1,576,086

2.2 SVDD Classification Algorithm

The One-Class classification method called Support Vector Data Description (SVDD) was first put forward by David Tax and his colleagues^[5]. It uses the Kernel method to map the data to a new, high dimensional feature space without many extra

computational costs. By this mapping more flexible descriptions are obtained. It will be shown how the outlier sensitivity can be controlled in a flexible way. Finally, an extra option is added to include example outliers into the training procedure (when they are available) to find a more efficient description.

In the support vector classifier, we define the structural error:

$$\mathcal{E}_{struct}(R, a) = R^2 \quad (1)$$

which has to be minimized with the constraints:

$$\|x_i - a\|^2 \leq R^2, \forall i \quad (2)$$

To allow the possibility of outliers in the training set, and therefore to make the method more robust, the distance from objects x_i to the center a should not be strictly smaller than R_2 , but larger distances should be penalized. This means that the empirical error does not have to be 0 by definition. We introduce slack variables ξ , $\xi_i \geq 0$, $\forall i$ and the minimization problem changes into:

$$\mathcal{E}_{struct}(R, a, \xi) = R^2 + C \sum_i \xi_i \quad (3)$$

with constraints that (almost) all objects are within the sphere:

$$\|x_i - a\|^2 \leq R^2 + \xi_i, \xi_i \geq 0, \forall i \quad (4)$$

The parameter C gives the tradeoff between the volume of the description and the errors. The free parameters, a , R and ξ , have to be optimized, taking the constraints (4) into account. Constraints (4) can be incorporated into formula (3) by introducing Lagrange multipliers and constructing the Lagrange:

$$L(R, a, \xi, \alpha, \gamma) = R^2 + C \sum_i \xi_i - \sum_i \alpha_i \{R^2 + \xi_i - (x_i + a)^2\} - \sum_i \gamma_i \xi_i \quad (5)$$

with the Lagrange multipliers $\alpha_i \geq 0$ and $\gamma_i \geq 0$, where $x_i \cdot x_j$ stands for the inner product between x_i and x_j . Note that for each object x_i a corresponding α_i and γ_i are defined. L has to be minimized with respect to R , a and ξ , and maximized with respect to α , γ and ξ . Setting partial derivatives to 0 gives the constraints:

The last constraint can be rewritten into an extra constraint for α :

$$0 \leq \alpha_i \leq C, \quad \forall i \quad (6)$$

This results in the final error L :

$$L = \sum_i \alpha_i (x_i \cdot x_i) - \sum_{i,j} \alpha_i \alpha_j (x_i \cdot x_j) \quad (7)$$

With $0 \leq \alpha_i \leq C, \forall_i$

A test object z is accepted when this distance is smaller than or equal to the radius:

$$\|z - a\|^2 = (z \cdot z) - 2 \sum_i \alpha_i (z \cdot x_i) + \sum_{i,j} \alpha_i \alpha_j (x_i \cdot x_j) \leq R^2 \quad (8)$$

By definition, R^2 is the squared distance from the center of the sphere a to one of the support vectors on the boundary:

$$R^2 = (x_k \cdot x_k) - 2 \sum_i \alpha_i (x_i \cdot x_k) + \sum_{i,j} \alpha_i \alpha_j (x_i \cdot x_j) \tag{9}$$

For any $x_k \in SV^{bnd}$, especially the set of support vectors for which $0 < \alpha_k < C$. We will call this one-class classifier the support vector data description (SVDD). It can now be written as:

$$\begin{aligned} f_{SVDD}(z; \alpha, R) &= I(\|\phi(z) - \phi(a)\|^2 \leq R^2) \\ &= I\left(K(z, z) - 2 \sum_i \alpha_i K(z, x_i) + \sum_{i,j} \alpha_i \alpha_j K(x_i, x_j) \leq R^2\right) \end{aligned} \tag{10}$$

where the indicator function I is defined as:

$$I(A) = \begin{cases} 1 & \text{if } A \text{ is true} \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

2.3 Experimental Design and Results

General Settings:

Threshold used in Anomaly Detection ranges from 9 to 35.

In this section, the representative MIT lpr dataset is used to study the changes that different windows size bring to the result, the combo parameters are set as follows:

- (1) Sliding Window size $K=6\sim12$;
- (2) KernelParam $\sigma=20$

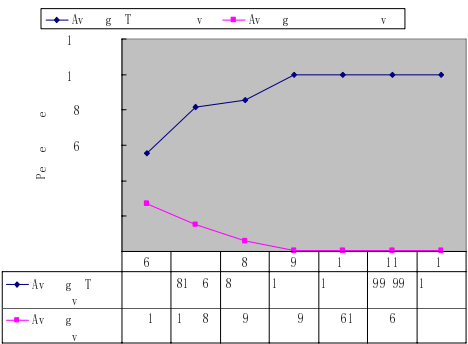


Fig. 1. Classification Results: Different Window size (K)

Note that the true positives grows from 55.73% to 85.34% when $K=6\sim8$, while false positives reduces from 27.12% to 5.97%. It can be derived that K , the size of the window, is a vital parameter which directly affects the classification result. The bigger it is, the more accurate the pattern of application behavior it stands for, but of course, the more computation it will bring, which is shown as the following table.

Table 2. Time used for training when K=6~12

K	6	7	8	9	10	11	12
Training Time (s)	524	889	1527	1930	2365	3002	3771

To do a real-time detection requires compacted data and less computing, so it’s very important to have a quick response to the intrusion. The window sized 6 or 7 is representative and will be acceptable in next experiments.

All the datasets are used to evaluate the classification performance; the combo parameters are set as follows:

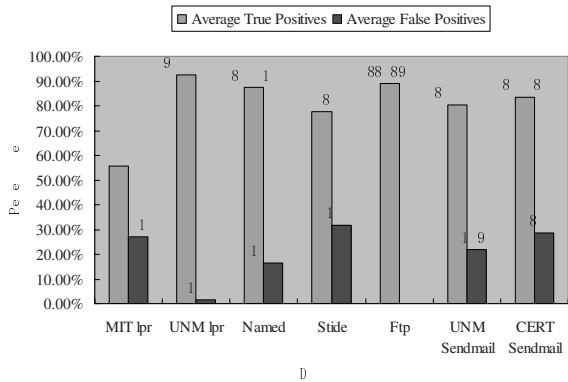


Fig. 2. Classification Results: Different Datasets

- (1) Sliding Window size K=6 to ensure the real-time detection;
- (2) KernelParam $\sigma=20$ As shown in Fig.3, within the response time, the UNM lpr dataset got the maximum true positive rate while the FTP dataset got the minimum false positive rates. Note that the result of UNM lpr is much better than MIT lpr, because the UNM normal data set includes fifteen months of activity, while the MIT data set includes just two weeks so the application pattern included in the dataset is more accurate. The false positive rate of FTP equals zero may due to its little data size. Unfortunately, the MIT lpr has a poor classification in the experiment.

3 Frequency-Based SVDD (F-SVDD) Anomaly Detection Model

The original SVDD model assumes that all the short sequences are equal to the audit system; videlicet, classification errors will result in the same loss. In Fact, the ‘Cost’ is different, those short sequences which appear more frequently in sequence pool should be paid more attention because they may include the operation customs of the most valid users and also indicate the general characters of different user id. Coalesced with the Data Pre-Processing procedure, the Frequency-based SVDD Anomaly Detection (F-SVDD) Model is established as the following block diagram.

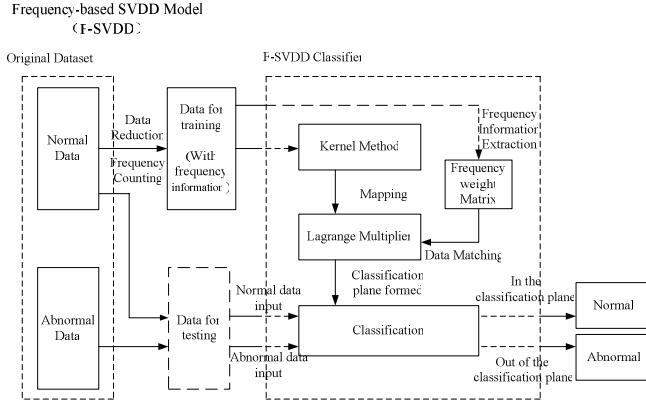


Fig. 3. Block Diagram : F-SVDD Detection Model

3.1 F-SVDD Classification Algorithm

To allow the possibility of frequency information importation in the algorithm, the frequency weight matrix of short sequence x_k is defined as:

$C = [c_1, c_2 \dots c_k, \dots, c_m]$ m stands for the number of the short sequences
 $c_i = \text{sequencenum}_i / \text{samplesumnum}$, sequencenum_i is the number of sample i
 samplesumnum is the total number of the samples.

$$\min[\mathcal{E}(a, R, \xi)] = R^2 + \sum_{i=1}^m c_i \xi_i \quad (12)$$

$$s.t. \quad \|x_i - a\|^2 \leq R^2 + \xi_i \quad \xi_i \geq 0, i = 1, 2, \dots, m$$

Now we construct the Lagrange using the new constrains of (13)

$$L(a, R, \xi, \alpha, \gamma) = R^2 + \sum_{i=1}^m c_i \xi_i - \sum_{i=1}^m \alpha_i [R^2 + \xi_i - (x_i, x_i) + 2(a, x_i) - (a, a)] - \sum_{i=1}^m \gamma_i \xi_i \quad (13)$$

Note that $\gamma_k \geq 0$, so

$$0 \leq \alpha_i \leq c_i \quad i = 1, 2, \dots, m \quad (14)$$

3.2 Experimental Design and Results

In this section, the representative MIT lpr dataset is used to evaluate the classification performance of the F-SVDD model, because it got the last result in the original SVDD model. Parameters are set as follows:

General Settings:

- (1) Gaussian RBF Kernel function is used in this Detection Model ;
- (2) Sliding Window size $K=7$;
- (3) Threshold used in Anomaly Detection ranges from 9 to 35.

Combo Setting:
KernelParam $\sigma=20$;

As shown in Table 3, as different samples are labeled by different weight, the result is better than original SVDD, the true & false Positive level are raised by 22% and 58%, especially ,the true positives are reaching 100%.

Table 3. Comparison : SVDD Model vs. F-SVDD Model

Threshold	True Positives		False Positives		Number of Errors	
	SVDD	F-SVDD	SVDD	F-SVDD	SVDD	F-SVDD
9	100.0%	100.0%	0.44%	0.44%	12	12
10	100.0%	100.0%	0.44%	0.44%	12	12
11	100.0%	100.0%	0.44%	0.44%	12	12
12	100.0%	100.0%	0.55%	0.44%	15	12
13	100.0%	100.0%	0.59%	0.44%	16	12
14	100.0%	100.0%	0.63%	0.44%	17	12
15	100.0%	100.0%	0.63%	0.44%	17	12
16	100.0%	100.0%	0.63%	0.56%	17	15
17	100.0%	100.0%	0.63%	0.63%	17	17
18	100.0%	100.0%	0.63%	0.63%	17	17
19	100.0%	100.0%	0.63%	0.63%	17	17
20	100.0%	100.0%	0.63%	0.63%	17	17
21	100.0%	100.0%	0.63%	0.63%	17	17
22	100.0%	100.0%	0.63%	0.63%	17	17
23	100.0%	100.0%	0.63%	0.63%	17	17
24	100.0%	100.0%	3.70%	0.63%	100	17
25	100.0%	100.0%	16.61%	0.63%	449	17
26	100.0%	100.0%	16.75%	0.63%	453	17
27	100.0%	100.0%	17.16%	0.63%	464	17
28	100.0%	100.0%	17.16%	0.63%	464	17
29	100.0%	100.0%	20.16%	0.74%	545	20
30	100.0%	100.0%	21.82%	11.95%	590	323
31	3.7%	100.0%	46.82%	16.35%	1266	442
32	3.7%	100.0%	57.25%	16.86%	1548	456
33	0.0%	100.0%	58.73%	26.78%	1588	724
34	0.0%	99.9%	61.06%	41.35%	1651	1118
35	0.0%	99.9%	61.28%	47.04%	1657	1272
AVE	81.76%	99.99%	15.08%	6.28%	408	173

4 Write-Related SVDD (W-SVDD) Anomaly Detection Model

Different System Calls (SC) may play a quite different role in operating systems, for example, the basic SC pair, the ‘Read-Write’ operation pair. The ‘Read’ operation only affects current process, while other processes and the kernel data stay the same. It’s true that the operation ‘Read (buff)’ is the key step in the Buffer Overflow Attack and will cause system error directly, but this would surely be taken effect by thronging segment error or calling the ‘exec’ process. The ‘Write’ operation, however, may probably change the data in file system so as to affect other processes. Furthermore, remote system can also be affected by this operation if network connection is available. Obviously, the ‘Write’ operation is more aggressive thus may be more valuable

in our intrusion prevention system. Coalesced with the new Data Pre-Processing procedure, the Write-related SVDD Anomaly Detection (W-SVDD) Model is established as the following block diagram.

4.1 W-SVDD Data Pre-processing

Different System Call has the different ‘Cost’. There are a good many similar SC pairs just like ‘Read’ & ‘Write’, e.g., ‘getpriority’ & ‘setpriority’ and ‘getsockopt’ & ‘setsockopt’, etc. Two main class of SC are defined according to their agressivity, The First class is called ‘Read-related’, such operations only affect their own process just like ‘Read’ operation during execution; the other is ‘Write-related’, on the contrary, such operations is more aggressive for their directly affect to the other processes and the change of the system status^[23].

Experiments using the dataset show that in Intrusion Detection and Prevention systems, only ‘Write-relate’ system calls should be supervised in order to improve the detection efficiency. The dataset used are gathered from Sun OS, there are 77 Write-related System Calls out of 182 System Calls which are listed as below:

In W-SVDD model, original data will be Write-Extracted before sliced into short sequences. The following table shows how much data flows can be reduced by Write-Extraction.

Table 4. Comparison : Number of system Calls(‘S’ symbolized original data reduction and ‘W’ symbolized Write-Extraction)

Data sets	MIT lpr		UNM lpr		Named		Stide	
	S	W	S	W	S	W	S	W
Number of system calls	2,914,837	164,247	2,027,468	153,693	9,230,572	6,590,324	205,935	132,751
System calls For Training	512	187	470	373	1238	577	215	153

Data sets	Ftp		UNM Sendmail		CERT Sendmail	
	S	W	S	W	S	W
Number of system calls	1,363	945	6,755	4602	8,316	6955
System calls For Training	665	461	526	396	639	458

4.2 W-SVDD Classification Algorithm

The Classification Algorithm is just the same as original SVDD Algorithm, but the total data to be processed is much more reduced

4.3 Experimental Design and Results

In this section, all the datasets are used to evaluate the classification performance; the combo parameters are set as follows:

General Settings:

- (1) Gaussian RBF Kernel function is used in this Detection Model;
- (2) Sliding Window size $K=6$; KernelParam $\sigma=20$;
- (3) Threshold used in Anomaly Detection ranges from 9 to 35;

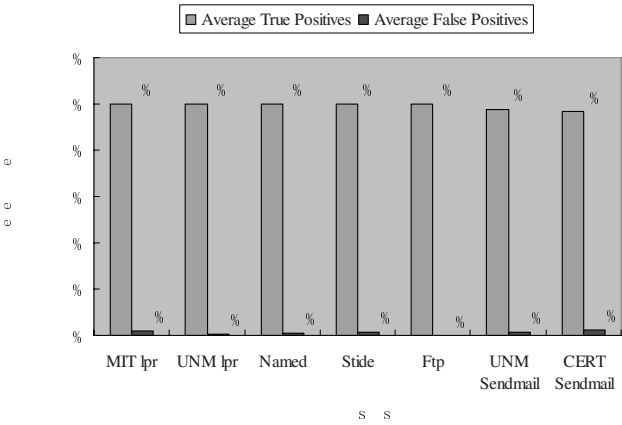


Fig. 4. Classification Results: Different Datasets

As shown in Fig.4, It is proved that this Detection Model is robust for the average true positives of all the datasets are going close to 100% while the average false positives are closed to 0%, the normal traces can be almost completely distinguished with the abnormal. Note that $K=6$ in this experiment, so this model can also be suitable for Real-Time detections. Just Compared with the results in 2.3.2, the true positives increased by 22% and 23% while the False Positives decreased by 58% and 94%. Algorithm is the same in these two models, but the result appears quite different.

5 Conclusions

The detection model based on SVDD One-Class Classification method avoids the complex work of great amount of abstraction and matching operations. The algorithms also makes the security audit system detect new anomaly behaviors.

Based on Cost-Sensitive Learning Method, two improved One-Class Anomaly Detection models using SVDD are put forward in this paper. Experiments show that ,in the aim of pay more attention to the samples which are more crucial to the system users and other processes, both improvements to the algorithm and further reduction

to the input data method result in a elevation to the performance The effect of some parameters such as size of Sliding -Window K and the KernelParam σ are also considered in the new models.

Experiments using UNM Anomaly Datasets show that using Cost-Sensitive method in Anomaly Detection may be a future orientation in Trusted Computing area. Designing more effective methods for Cost-Sensitive learning in Anomaly Detection is an issue to be explored in the future.

References

- [1] Forrest, S., Hofmeyr, S.A.: Computer Immunology [J] Communications of the ACM, 88–96 (1997)
- [2] Hofmeyr, S., Forrest, S.: Principles of a computer immune system. In: Proceeding of New Security Paradigms Workshop, pp. 75–82 (1997)
- [3] Warrender, C., Forrest, S., Pearlmuter, B.: Detecting Intrusion Using System Calls: Alternative Data Models, pp. 114–117 (2002) <http://www.cs.unm.edu/forrest/publications/oakland-with-cite.pdf>
- [4] Forrest, S., Hofmeyr, S.A., Longstaff, T.A.: A sense of self for Unix Processes, pp. 120–128. IEEE Computer Society Press, Los Alamitos (1996)
- [5] David, M.J.T.: One-class Classification. Ph.D. Dissertation (1999)
- [6] Manevitz, L.M., Yousef, M.: One-Class SVMs for Document Classification. Journal of Machine Learning Research, pp. 139–154 (2001)
- [7] Chen, Y.Q., Zhou, X., et al.: One-class SVM for learning in image retrieval [A]. In: IEEE Intl Conf. on Image Proc. (ICIP'2001), Thessaloniki, Greece (2001)
- [8] Kohonen, T.: Self-Organizing Map. pp. 117–119. Springer, Berlin (1995)
- [9] Bishop, M.: A standard audit trail format. In: Proceeding of the 18th National Information Systems Security Conference, Baltimore, pp. 136–145 (1995)
- [10] MIT lpr DataSet (2000) <http://www.cs.unm.edu/immsec/data/>
- [11] Lee, W., Stolfo, S.J., Mok, K.W.A.: Data mining framework for building intrusion detection models. In: Proc the 1999 IEEE Symposium on Security and Privacy, Berkely, California, pp. 120–132 (1999)
- [12] Haykin, S.: Neural networks-A comprehensive foundation, 2nd edn. Tsinghua University Press, Beijing (2001)
- [13] Hattor, I.K., Takahashi, M.: A new nearest-neighbor rule in the pattern classification problem. Pattern Recognition, 425–432 (1999)
- [14] Kim, J., Bentley, P.: The Artificial Immune Model for Network Intrusion Detection. In: 7th European Conference on Intelligent Techniques and Soft Computing (EUFIT'99), Aachen, Germany (1999)
- [15] Pan, Z.S., Luo, J.: An Immune Detector Algorithm Based on Support Vector Data Description [B]. Journal of Harbin University of Engineering (2006)
- [16] Rtsch, G., Schlkopf, B., Mika, S., Müller, K.R.: SVM and boosting: One class. Berlin, vol. 6 (2000)
- [17] Colin, C., Kristin, P., Bennett, A.: Linear Programming Approach to Novelty Detection[A]. Advances in Neural Information Processing System 13 (2001)
- [18] Zhang, X.F., Sun, Y.F., Zhao, Q.S.: Intrusion Detection Based On Sub-Set Of System Calls. ACTA Electronica Sinica 32 (2004)
- [19] Warrender, C., Forrest, S., Pearlmuter, B.: Detecting Intrusions Using System Calls, Alternative Data Models, pp. 133–145. IEEE Computer Society, Los Alamitos (2002)

Improved and Trustworthy Detection Scheme with Low Complexity in VBLAST System

So-Young Yeo, Myung-Sun Baek, and Hyoungh-Kyu Song

uT Communication Research Institute, Sejong University, 98 Kunja-Dong,
Kwangjin-Gu, Seoul, 143-747, Korea
yeossoh@sdc.sejong.ac.kr, sabman@sdc.sejong.ac.kr, songhk@sejong.ac.kr

Abstract. We provide a new detection scheme for interference nulling and cancellation operation in a vertical Bell laboratories layered space-time (VBLAST) system to reduce unexpected effects due to parallel transmission in this paper. This method can reduce the time delay as well as moderate the multipath fading effect. We will show that the performance of investigated VBLAST detection based on hybrid processing performs better than ordinary VBLAST detections based on successive and parallel processing, respectively.

1 Introduction

Next generation communications can be defined as the ability of devices to communicate with each other and to provide services to the users securely and transparently. The best way to achieve the goal of next generation communications is to evolve the available technologies and to support high-data rate communication [1]. So, we consider the orthogonal frequency division multiplexing (OFDM) system using multiple input multiple output (MIMO) architecture. The MIMO system can improve efficiently transmission rate even in several multipath environment.

A Bell laboratories layered space-time (BLAST) architecture has received considerable attention recently as it could provide very high data-rate communication over wireless channels [2][3], which is often referred to as successive nulling and cancellation, ordered successive interference cancellation (OSIC). However, the performance of successive detection scheme is limited due to noise enhancement caused by nulling and error propagation. Recently, various detection methods for improving a vertical BLAST (VBLAST) system and reducing the complexity have been proposed [4]-[6]. But, the proposed schemes [4]-[6] can not reduce the noise enhancement. On the other hand, maximum likelihood detection scheme has optimal performance, but its complexity is excessively high.

In this paper, we develop an efficient combining parallel interference cancellation (PIC) and successive interference cancellation (SIC) detection algorithm for the nulling-vector and cancellation in VBLAST systems for improved and trustworthy detection. The simulation results show that the performance of the hybrid detection algorithm is superior to that of the ordinary VBLAST detection

algorithms based on successive detection [2][3] and parallel detection [4]. Also we compare operation complexity of hybrid with ordinary VBLAST detection.

2 System Description

We consider N_t transmit antennas and $N_r \geq N_t$ receive antennas. The OFDM data sequence of n -th antenna is given by $[X_n(k)|k = 0, \dots, K-1]$, which is denoted the k -th subcarrier of the complex baseband signal and K is the length of the OFDM data sequence.

The data is demultiplexed in N_t data layers of equal length and N_t data are mapped into a certain modulation symbols. The data are transmitted over the N_t antennas simultaneously. We assume that the channel is a frequency flat fading and its time variation is negligible over a frame. The time delay and phase offset of the each antenna are assumed to be known, i.e., tracked accurately. Therefore, the overall channel H can be represented as $N_r \times N_t$ complex matrix and the k -th subcarrier of the baseband received signal at n -th receive antenna is

$$y_n(k) = \sum_{n_t=1}^{N_t} h_{nn_t} x_{n_t}(k) + w_n(k) \quad (1)$$

where w_n is zero-mean Gaussian noise with variance σ_n^2 .

Let $\mathbf{x} = [x_1 \ x_2 \ \dots \ x_{N_t}]$ denote the $N_t \times 1$ vector of transmit symbols. The overall received signals can be represented as

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n} \quad (2)$$

where $\mathbf{H} = [\mathbf{h}_1 \ \mathbf{h}_2 \ \dots \ \mathbf{h}_{N_r}]^T$ is a i.i.d random complex vector of multipath channel with each element of $\mathbf{h}_n = [h_{n1} \ h_{n2} \ \dots \ h_{nN_t}]$, and $\mathbf{n}$ is $N_r \times K$ matrix, respectively.

3 Hybrid VBLAST Detection

Let us describe the detail operation of the hybrid VBLAST detection scheme in Fig.1. Initially, the antennas are ranked in decreasing order of the irrespective received signal power using the power ranking scheme.

Hybrid detection process can be explained below. The regenerated signals from the first antenna to the j -th antenna at the j -th stage are subtracted from the delayed version of received signal y^0 as follows:

$$y^j = y^0 - \sum_{l=1}^j \hat{y}_l^{j-1} \quad (3)$$

where $\hat{y}_l^{j-1}$ is the regenerated signal of the l -th antenna from $(j-1)$ -th stage.

Eqn. (3) is the interference cancelled signal for the input of a nulling and regeneration block (NRB) of all antennas at the j -th stage. In NRB of Fig.1,

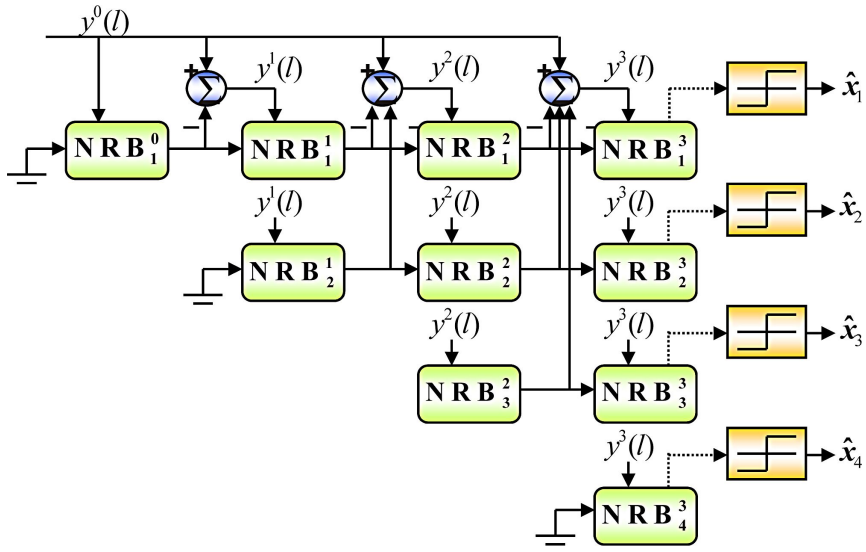


Fig. 1. The hybrid detection scheme under $N_t = N_r = 4$

the regenerated signal of the i -th antenna at the previous stage denoted as $\hat{y}_i^{j-1}$ is added to the interference cancelled signal of denoted as $\hat{y}^j$ to obtain the corresponding composite signal as follows

$$y_i^j = y^0 - \sum_{\substack{l=1 \\ l \neq i}}^j \hat{y}_l^{j-1} \quad (4)$$

$$= y^j + \hat{y}_i^{j-1}, \quad (5)$$

which is nulled by the corresponding antenna's nulling vector and forwarded to decision block. Therefore, the NRB in the hybrid scheme performs only the projection of the received signal of the corresponding antenna. Finally, the decision variable z_i^j is forwarded to the next stage for more clean regeneration when $j < N_t$. On the other hand, when $j = N_t$, the decision variable is forwarded to the final bit decision device.

As the delayed detection proceeds, the delayed version of received signal is subtracted by all other antenna's signal except for the desired antenna's signal. This hybrid process is repeated until the cancellation stage reaches the maximum stage. In the original successive VBLAST detection scheme, the received signal y^j is subtracted by only one regenerated signal of the j -th antenna at every inter-stage. In the hybrid VBLAST detection scheme, however, it is subtracted by all regenerated signals from the first to the j -th antenna at every inter-stage.

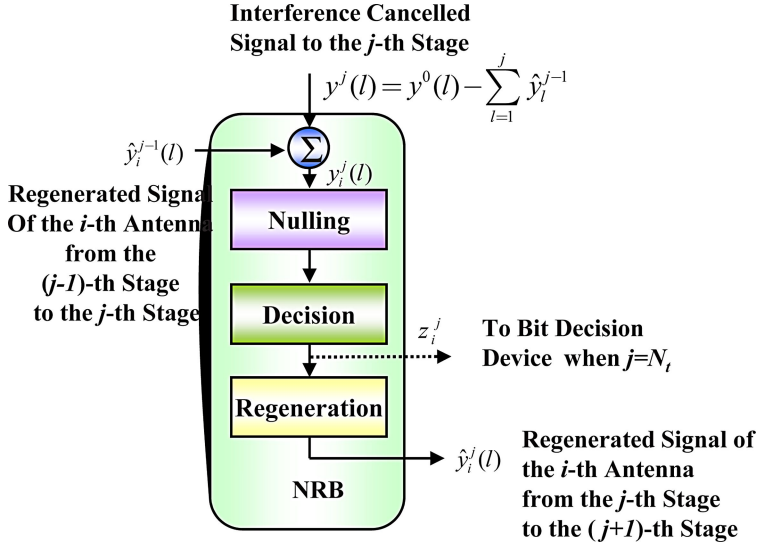


Fig. 2. The nulling and regeneration block for the hybrid detection at the j -th stage of i -th antenna

4 Modified Nulling Vector of the Hybrid VBLAST Detection

For the $N_t \times 1$ vector of transmitted data symbols denoted as $\mathbf{x} = [x_1 \ x_2 \ \cdots \ x_{N_t}]^T$, the corresponding received $N_r \times 1$ vector is

$$\mathbf{y}^0 = \mathbf{H}\mathbf{x} + \mathbf{n} \quad (6)$$

where $\mathbf{H} = [\mathbf{h}_1 \ \mathbf{h}_2 \ \cdots \ \mathbf{h}_{N_r}]^T$ is a vector of multipath channel with each element of $\mathbf{h}_n = [h_{n1} \ h_{n2} \ \cdots \ h_{nN_t}]$ and $\mathbf{n} = [n_1 \ n_2 \ \cdots \ n_{N_t}]^T$ is the additive white gaussian noise vector with variance σ_n^2 .

We perform the successive detection of the elements in $\mathbf{x}$. Note that we do not need to detect the element x_i in the order of $i = 1, 2, \dots, N_t$. So the optimal ordering to minimize the detection error is founded. It turns out that we can obtain the optimal ordering by selecting the minimum-norm column of $\mathbf{H}^\dagger$, where $(\cdot)^\dagger$ means pseudo-inverse. Let the optimal detection ordering be $[x_{l_1} \ x_{l_2} \ \cdots \ x_{l_{N_t}}]$.

To detect the first element of $\mathbf{x}$, x_{l_1} , we perform zero-forcing nulling. We find the minimum norm weight vector $\mathbf{w}_{l_1}$ such that $\mathbf{v}_{l_1}^H \mathbf{h}_{l_1} = \delta_{l_1}$ where $(\cdot)^H$ means the complex conjugate. The weight vector $\mathbf{v}_{l_1}$ can be obtained from the pseudo-inverse of $\mathbf{H}$ and using its estimator $\hat{\mathbf{H}}$, we obtain the weight vector

$$\hat{\mathbf{v}}_{l_1} = (\hat{\mathbf{B}}_i)_{l_1}^* \quad (7)$$

Table 1. The hardware complexity comparison

Detection Method	Hybrid	Successive	Parallel
Nulling Operation	$N_t N_t + 1)/2$	N_t	$2N_t$
Nulling Vector	$N_t(N_t + 1)(N_t + 2)/6$	$N_t(N_t + 1)/2$	$N_t(N_t + 1)$

where $\hat{\mathbf{B}}_{i+1} = \hat{\mathbf{H}}_{l_i-}^\dagger$. When the i -th symbol is detected, the received vector $\mathbf{y}^j$ after cancelling x_{l_m} for $m = 0, 1, \dots, j-1$ becomes

$$\mathbf{y}^j = \mathbf{y}^0 - \sum_{m=0}^{j-1} \hat{x}_{l_m} \hat{\mathbf{h}}_{l_m} + \mathbf{n}. \quad (8)$$

If it is assumed that detection error does not exist, $\mathbf{y}^j$ is as follows

$$\begin{aligned} \mathbf{y}^j &= \mathbf{H}\mathbf{x} - \sum_{m=0}^{j-1} x_{l_m} \hat{\mathbf{h}}_{l_m} + \mathbf{n} \\ &= \mathbf{H}_{l_i-} \mathbf{x}_{l_i-} - \sum_{m=0}^{j-1} x_{l_m} \mathbf{h}_{l_m} + \mathbf{n} \\ &= \hat{\mathbf{H}}_{l_i-} \mathbf{x}_{l_i-} - \Delta \mathbf{H}_{l_i-} \mathbf{x}_{l_i-} - \sum_{m=0}^{j-1} x_{l_m} \mathbf{h}_{l_m} + \mathbf{n} \end{aligned} \quad (9)$$

where $\hat{\mathbf{H}}_{l_i-} = [\hat{\mathbf{h}}_{l_i} \ \hat{\mathbf{h}}_{l_{i+1}} \ \dots \ \hat{\mathbf{h}}_{l_{N_t}}]$, $\Delta \mathbf{H}_{l_i-} = [\Delta \mathbf{h}_{l_i} \ \Delta \mathbf{h}_{l_{i+1}} \ \dots \ \Delta \mathbf{h}_{l_{N_t}}]$, and $\mathbf{x}_{l_i-} = [\mathbf{x}_{l_i} \ \mathbf{x}_{l_{i+1}} \ \dots \ \mathbf{x}_{l_{N_t}}]$.

If $\Delta \mathbf{H} = 0$, the nulling vector $\hat{\mathbf{v}}_{l_i}$ in eqn. (7) is an optimal nulling vector. But in the real system, there is nonzero matrix $\Delta \mathbf{H}$, and $\hat{\mathbf{v}}_{l_i}$ is not the optimal nulling vector. We derive henceforth the new nulling vector which minimizes the unexpected effect of channel estimation error. Without loss of generality, the new nulling vector $\bar{\mathbf{v}}_{l_i}$ is written as

$$\bar{\mathbf{v}}_{l_i} = \hat{\mathbf{v}}_{l_i} + \Delta \mathbf{v}_{l_i} \quad (10)$$

where $\Delta \mathbf{v}_{l_i}$ mean the difference between original nulling vector $\hat{\mathbf{v}}_{l_i}$ and $\bar{\mathbf{v}}_{l_i}$. With the new nulling vector, we obtain the new decision statistic and the estimate of $\hat{x}_{l_i}$ is as follows

$$\hat{x}_{l_i} = Q(\bar{\mathbf{v}}_{l_i}^H \mathbf{y}_i^j) \quad (11)$$

where $Q(\cdot)$ is the quantization operation appropriate to the constellation in use.

5 Examples and Discussions

In this section, we illustrate the bit error rate (BER) performance of VBLAST detection schemes for Rayleigh fading channel. In Fig. 3, we can observe that the hybrid detection method provides an improvement of approximately 2 – 2.5dB and 7 – 8dB over the classical methods with successive and parallel detection, respectively. Also, the hybrid detection method provides an improvement of approximately 6 – 6.5dB over the QR decomposition method. And, ordinary VBLAST based on successive and parallel detections require N_t and $2N_t$

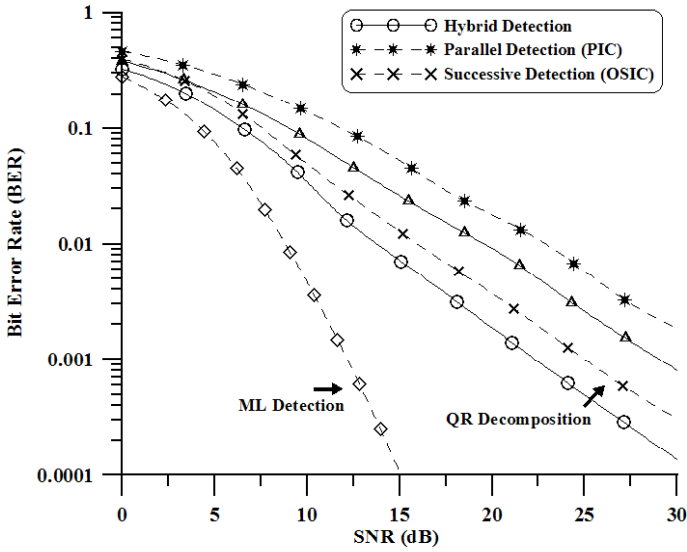


Fig. 3. BER performance of various VBLAST detection schemes in the case of $N_t = N_r = 4$

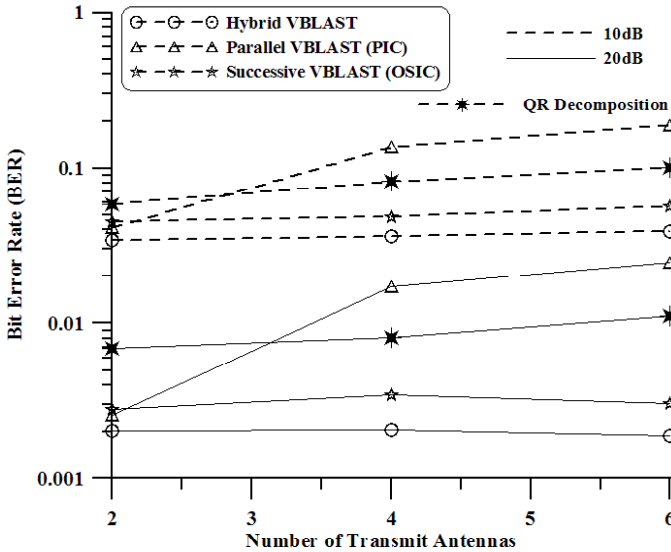


Fig. 4. BER performance according to the number of transmitting antennas for various VBLAST detection schemes

nulling operations, respectively as table 1. However, the hybrid VBLAST detection scheme needs $N_t(N_t + 1)/2$ nulling operations, which results in additional complexity. On the other hand, the total number of the rows used to obtain

pseudo-inverse matrix of $\mathbf{H}_{k_i-}$ is calculated as $N_t(N_t + 1)/2$, $N_t(N_t + 1)$ and $N_t(N_t + 1)(N_t + 2)/6$ for successive, parallel and hybrid detection, respectively. In the case of $N_t = N_r = 4$, for example, the total number of the rows of hybrid detection is equal to that of the parallel detection [4] and is double than that of the successive detection [3].

Fig. 4 illustrates the effect of the number of transmit antennas on the BER performance. In this figure, it can be easily observed from [4] that the performance of parallel detection is markedly better than that of classical VBLAST detection for $N_t = 2$. And, in the case of $N_t > 3$, it becomes the opposition. However, regardless of the number of transmit antennas and SNR, the proposed hybrid detection scheme appears to be a stable tendency over other reference VBLAST systems [4]–[6].

6 Conclusions

In this paper, multiple transmit and receive antenna system has been used to form VBLAST system to increase system capacity. An efficient detection algorithm for interference nulling vector and cancellation has been studied for VBLAST systems. We have shown that the BER performance of hybrid detection outperforms ordinary VBLAST based on successive and parallel detections in expense of a small hardware complexity.

Acknowledgement

This research is supported by the ubiquitous Computing and Network (UCN) Project, the Ministry of Information and Communication (MIC) 21st Century Frontier R&D Program in Korea.

References

1. Schoo, A.R.P., Wang, H.: An evolutionary approach towards ubiquitous communications: a security perspective Prasad. In: Applications and the Internet Workshops, pp. 689–695 (2004)
2. Foschini, G.J.: Layered space-time architecture for wireless communications in a fading environment when using multi-element antennas. *Bell System Technology Journal* 1(2), 41–59 (1996)
3. Foschini, G.J., Golden, G.D., Valenzuela, R.A., Wolniansky, P.W.: Simplified processing for high spectral efficiency wireless communication employing multi-element arrays. *IEEE Journal on Selected Areas in Communications* 11(17), 1841–1852 (1999)
4. Chin, W.H., Constantinides, A.G., Ward, D.B.: Parallel multistage detection for multiple antenna wireless systems. *Electronics Letters* 38(12), 597–599 (2002)
5. Elena, C., Haiyan, Q., Xiaofeng, T., Zhuizhuan, Y., Ping, Z.: New detection algorithm of V-BLAST space-time code. In: Vehicular Technology Conference, vol. 4, pp. 2421–2423 (2001)
6. Biglieri, E., Taricco, G., Tulino, A.: Decoding space-time codes with BLAST architecture. *IEEE Transactions on signal processing* 50(10), 2547–2551 (2002)

Stepping-Stone Detection Via Request-Response Traffic Analysis

Shou-Husan Stephen Huang¹, Robert Lychev², and Jianhua Yang³

¹ Department of Computer Science, University of Houston,
4800 Calhoun Rd., Houston, TX 77004, USA
shuang@cs.uh.edu

² Department of Computer Science, University of Massachusetts Amherst,
Amherst, MA 01003, USA
rlychev@cs.umass.edu

³ Department of Mathematics & Computer Science, Bennett College,
900 E. Washington St., Greensboro, NC 27401, USA
jhyang@bennett.edu

Abstract. In this paper, we develop an algorithm that may be used as a stepping-stone detection tool. Our approach is based on analyzing correlations between the cumulative number of packets sent in outgoing connections and that of the incoming connections. We present a study of our method's effectiveness with actual connections as well as simulations of time-jittering (introduction of inter-packet delay) and chaff (introduction of superfluous packets). Experimental results suggest that our algorithm works well in the following scenarios: (1) distinguishing connection chains that go through the same stepping stone host and carry traffic of users who perform similar operations at the same time; and (2) distinguishing a single connection chain from unrelated incoming and outgoing connections even in the presence of chaff. The result suggests that time-jittering will not diminish our method's usefulness.

1 Introduction

The study of detection and/or prevention of network-based attacks requires much attention as perpetrators are becoming more and more capable of compromising much of critical information infrastructure that we so highly depend on. Network-based attacks can be either interactive, where a perpetrator is interested in stealing information from another member of the network, or non-interactive, where a perpetrator's goal is to trigger a malicious software or perform a denial-of-service attack on another member of the network. Attackers can use a number of techniques to avoid revealing their identification and location. Two of the most-commonly used evasion measures include IP-spoofing and the construction of stepping-stone chains. The latter involves an intruder connecting to a victim indirectly through a sequence of hosts called stepping-stones. Although, some work has already been done to show a number of effective techniques for tracing spoofed traffic [4, 5, 7, 8], effective measures for tracking stepping-stone attacks are yet to be found. The focus of our research is to study a

connection-chain detection scheme that could help us address the stepping-stone detection problem, a portion of the stepping-stone attack tracking problem, in interactive attacks.

To understand why stepping-stone detection may be an important subject to study, consider the following scenario. Machine *V* is discovered to be a victim of an interactive attack whose immediate source was found to be machine *S*. Shutting off *S* from the network is effective in stopping the attack, but it does not do anything to ensure that the adversary *A* is caught, since *S* could be just the immediate stepping-stone used by *A* to indirectly connect to *V*. However, with the ability to correctly determine whether *S* is a stepping-stone or not, one can either go upstream along the chain to discover other stepping stones and/or catch the perpetrator, or simply shut down *S* if it is not a stepping stone (in which case it must be *A*). Even when it is not known that an attack is launched, being able to correctly determine whether any member of the network is a stepping-stone should allow for an effective way of policing interactive attacks. Stepping-stone detection problem is a useful subject to study, but it must be noted that just having the capability of even perfect stepping-stone detection is not enough to solve the stepping-stone attack tracking problem. As explained in [13], to track stepping-stone attacks one also needs to have correct methods of serializing stepping-stones into a connection chain.

Much research has already been done in this area, and, ultimately, all established techniques of identifying a particular host as a stepping-stone rely on identifying a connection-chain based on strong correlations between that host's incoming and outgoing traffic. Such correlations can be based on the log-in activity [6, 9], packet content [10, 11], periodicity of network activity [16], the timing properties [12, 15], and the packet frequency of the connections [1]. The first two techniques are not practical because, respectively, it is conceivable that hackers should be able to forge authentication sessions, and, since most users use SSH instead of Telnet, it is not clear how to correlate traffic that is encrypted as it is passed from host to host. A hacker can easily countermeasure correlation techniques such as the one described in [16] by introducing random time delays in between individual and/or collections of packets—jittering. It was shown in [3] that, in principle, there is no effective way for an adversary to avoid timing-based detection techniques such as the ones described in [12, 15]. However, this is true only under the assumption that the adversary's time-jittering of the packets is independently and identically distributed and that the connection is long-lived. Also, effectiveness of timing-based detection methods is likely to diminish in the presence of chaff—superfluous packets introduced at various stepping-stones. Although techniques based on finding correlations between packet frequency of incoming and outgoing traffic, as presented in [1], were shown to be successful against jittering without the assumptions that were necessary in [3], these techniques do not perform well with chaffed traffic. Several effective algorithms to detect stepping-stone chains with chaff and jittering have been proposed in [17], but all of these methods require a significant amount of intercepted packets in order to ensure small false positive and negative rates. A testbed that may be very useful in testing various stepping-stone detection mechanisms in different scenarios was proposed in [14].

Section 2 is dedicated to describing our approach, Section 3 explains our experimental setup and methodology, Section 4 presents and analyzes results we obtained

from various experiments, and Section 5 wraps up this paper with a discussion of conclusions and possible directions for future work.

2 Technical Method

Our research is primarily inspired by algorithms discussed in [1]. However, in [1] only correlations between streams with the same direction were discussed, so only the observation of traffic that is relayed from stepping-stone to stepping-stone is required by techniques they proposed. We want to check whether our connection-chain detection algorithm, that focuses on determining frequency relationships between request and response streams, could be used to design a stepping-stone detection algorithm that yields results comparable to, with respect to false positive and negative rates, what has been achieved in [1, 17], while requiring less packets to observe. Given that some machine S is being used as a stepping-stone by some adversary A , the challenge of detecting a connection-chain lies in finding the exact S 's incoming-outgoing connection pair that is carrying traffic relevant to A 's stepping-stone attack.

2.1 The Basics

Our algorithm is based on measuring correlations of the outgoing streams of outgoing connections to the outgoing streams of incoming connections. Throughout the rest of this paper we will refer to the former as the SEND and the latter as the ECHO. Traffic in both directions should be monitored at stepping-stone.

Our hypothesis is that for a SEND-ECHO pair that belongs to a real connection chain, the frequency with which packets leave a stepping-stone in the ECHO stream is a function of the frequency with which packets leave a stepping-stone in the SEND stream. This is based on the fact that interactive attacks consist of adversaries obtaining information from the victims for every command the former sends. We study the relationship of ECHO – SEND versus ECHO + SEND (the difference and the sum of the number of packets in the ECHO stream and the number of packets in the SEND stream, respectively) to see how correlated a particular ECHO stream is to a particular SEND stream. Based on our hypothesis, we assume that the relationship between ECHO - SEND and ECHO + SEND should be linear. Thus we are able to analyze the packet frequency relationship between request and response traffic for a particular incoming-outgoing connection pair independently of other connections, where we can treat ECHO + SEND as the time, which is actually independent of real time and inter-packet delay, and ECHO – SEND as the variable of interest. Time independence gives us an advantage over the time-jittering detection evasion that will be discussed in Section 4.2. We suspect that in ECHO - SEND vs. ECHO + SEND space, SEND-ECHO pair that corresponds to a real connection chain should yield a curve that resembles a smooth line more than all curves that correspond to other SEND-ECHO pairs. We use this to find connection chains. Also, if a computer is discovered to have a SEND-ECHO pair that satisfies a particular margin of linearity, there is a high probability that it is being used as a stepping-stone. This assumption may be used for stepping-stone detection. Method of measuring linearity is explained in Section 2.2.

2.2 Computing Linearity of a Curve

We measure linearity of a curve by calculating the average square distance of that curve from its linear fit. According to our assumption, the curve with the smallest average square distance from its linear fit should correspond to the SEND-ECHO pair of the real connection chain. Linear fit $y = mx + b$ is calculated via standard linear regression method, where $x = \text{ECHO} + \text{SEND}$, and $y = \text{ECHO} - \text{SEND}$ [2].

3 Experimental Setup

3.1 Experiment of Worst Case Scenario

The first experiment targeted the worst-case scenario: all participants login onto different hosts via SSH through a single stepping stone, located at UH (University of Houston), from three different hosts at the same time and perform the same tasks. The point of studying the worst-case scenario was to see if we can distinguish connection chains that go through the same stepping-stone and carry traffic of users who perform similar operations at the same time. We conjecture that if it is possible to correctly distinguish connection chains in such a situation, then our procedure should work very well in situations where there is only one connection chain and many other completely unrelated incoming and outgoing connections. The former can happen if an adversary makes loops in his/her connection chain before reaching the victim's machine for the purposes of stepping-stone-detection evasion. Two types of experiments were done this way: typing and secret-stealing. The stepping-stone computer was running our software that was monitoring the streams of interest and recording packets in those streams. The following are the connection chains:

Participant A: home computer (SBC) $\rightarrow$ UH stepping stone $\rightarrow$ UH1 $\rightarrow$ Mexico

Participant B: UH2 $\rightarrow$ UH3 $\rightarrow$ UH stepping stone $\rightarrow$ UMASS Amherst

Participant C: UH4 $\rightarrow$ UH stepping stone $\rightarrow$ Texas A&M University

For the first three trials all participants were to type identical texts simultaneously. The last trial involved all three individuals typing different texts not simultaneously for different amounts of time.

The secret-stealing experiments consisted of the participants searching for a secret file on a victim computer by going through a number of directories that contains fake files. The test directory, consisting of secret directories/files was prepared in advance. The secret file was copied onto the attacker's machine upon discovery. The following are the connection chains:

Participant A: UH1 $\rightarrow$ UH stepping stone $\rightarrow$ UH5 $\rightarrow$ Mexico

Participant B: UH2 $\rightarrow$ UH3 $\rightarrow$ UH stepping stone $\rightarrow$ UMASS Amherst

Participant C: UH4 $\rightarrow$ UH stepping stone $\rightarrow$ Texas A&M University

3.2 Experiments with a Single Connection Chain

The case where a stepping-stone machine had only one connection-chain and other unrelated connections was addressed in the last experiment with the following connection chain:

Participant A: UH2 → UH stepping stone

Participant B: UH stepping stone → UMASS Amherst

Participant C: UH4 → UH stepping stone → Texas A&M University

Participant A connected to the stepping stone at UH and was writing a Hello World application. Participant B connected from the stepping stone at UH to UMASS and was copying electronic copies of some files. Finally, Participant C was performing a secret-stealing attack. The results of this experiment were also chaffed via the second chaff technique and are discussed in Section 4.4. It is reasonable to assume that in real-life computer networks it is very unlikely for a single computer to have more than one connection chains that go through it. Even if a perpetrator decides to make loops, as mentioned in Section 3.1, he/she will not be likely to loop through every stepping-stone because this is likely to slow down his/her attack by a large margin. With this in mind, the point of performing the experiment described in this section is to model something that is more likely to happen in real life situations.

3.3 Time-Jittering and Chaff

We studied time-jittering and chaff by changing the results that we obtained from regular experiments and analyzing the changed data the same way as the regular data.

Time-jittering perturbation was introduced as an addition of time extensions, chosen uniformly between 0 and some pre-specified limit, to the time stamps of packet records. The original order of packets within a stream is preserved. Every packet had probability of .5 to be thus time-jittered.

For every stream, SEND and ECHO, of every connection the chaff perturbation is introduced as an addition of packets, whose amount is limited by a pre-specified margin, to the original stream. Two different methods were performed. The first method consisted of generating a stream of superfluous packets, whose inter-packet delay is a random variable with a uniform distribution in the interval of 100-900 thousand microseconds, and merging this stream with an actual stream of packets that was recorded during the experiment. The second technique consisted of inserting a random number of superfluous packets, ranging from 1-20, into pseudo-randomly-chosen, with probability of 0.1, inter-packet time intervals of the original packet stream. For both methods, such parameters represent the worst-case scenario where the most chaff is introduced. Experiments performed with other chaff limits are not discussed in this paper.

4 Analysis and Discussion

As the reader will see, our main assumption that the relationship between ECHO - SEND and ECHO + SEND is close to linear is justified by the fact that correlation coefficient r [2] for curves that correspond to real connection chains are all above 0.95 when no stream is time-jittered or chaffed. It makes sense that r of curves that correspond to experiments without time-jittering and chaff are positive as our study is focused on interactive attacks, where the adversary gets back more packets from the victim machine than he/she sends to the latter. However, this is not always the case for experiments with time-jittering and chaff simulations.

4.1 Basic Experiments

For both types of experiments described in Section 3.1, without time-jittering and chaff the packet data of ECHO stream of a particular participant yields the smoothest curve when related to the packet data of SEND stream of that participant. This can be seen just by looking at the curves on the plots of ECHO – SEND versus ECHO + SEND of typing and secret stealing experiments we took at the beginning of this project (see Fig. 1a and Fig. 1b). On all the figures' legends ECHO and SEND streams

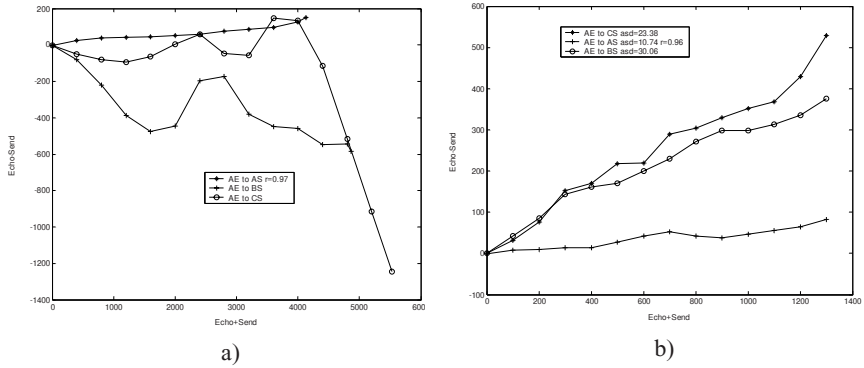


Fig. 1. Correlations of the ECHO stream of a particular participant to the SEND streams of all the participants in a) the typing experiment; b) the secret-stealing experiment

are referred to as E and S, respectively. All legends, except for Figures 1 and 3, show the average square distance (*asd*) of a curve from its linear fit (for each curve) and *r* (for the curve that corresponds to the real connection chain). Data obtained from the typing experiment was not quantitatively analyzed as this experiment does not really model a real interactive attack, but basic qualitative analysis should be enough here to obtain the correct result. Data obtained from experiment shown on Fig. 1b was quantitatively analyzed with procedure described in 2.2. Overall, experiments shown in Fig. 1a and Fig. 1b indicated that even when participants perform the same set of operations at the same time it is possible to pair each SEND stream with its complementary ECHO stream correctly using procedure described in 2.2.

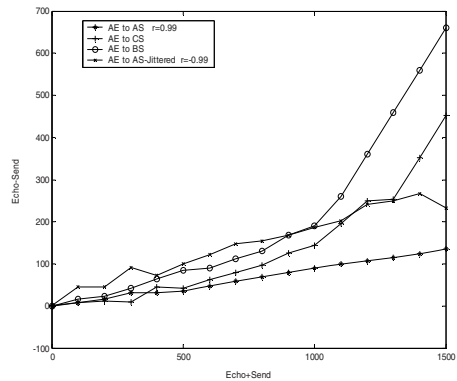


Fig. 2. Correlations of the ECHO stream of a particular participant to the SEND streams of all the participants in the secret-stealing experiment with a time-jittered simulation performed

4.2 Time Jittering Simulations

We mostly studied data that resulted from time-jittering the SEND streams of various connections where no time extension exceeded 200 thousand microseconds. After undergoing perturbations, every SEND-stream-packet-record vector was merged with data of various ECHO streams. After time-jittering, while the order of SEND packets with respect to each other was preserved, the order of SEND packets with respect to ECHO packets was not. This can be seen in Fig. 2. The ends of these curves also exhibit the shortcomings of our simulation.

We claim that the results that we might obtain once we solve the shortcomings of our current time-jittering simulation are not going to be very interesting. We claim so because in order for time-jittering to really affect our results, the order of SEND packets with respect to the ECHO packets has to be significantly disrupted. However, because some ECHO packets can come only after their corresponding SEND packets and vice versa, this disruption is not expected to be significant.

4.3 Chaff Simulations

We looked at data that resulted from chaffing the SEND stream, the ECHO stream and both streams of various connections. After undergoing such perturbations, every vector with perturbed data was merged with data of various ECHO streams. We assume that the adversary can chaff only his own traffic.

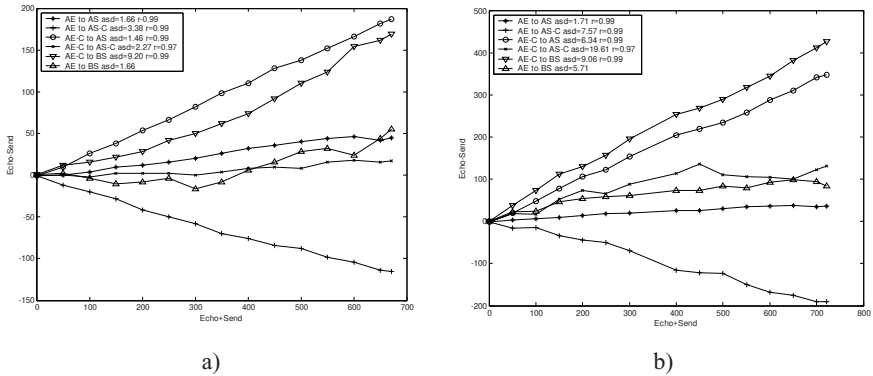


Fig. 3. Correlations of the ECHO stream of a particular participant to the SEND streams of all the participants in the secret-stealing experiment with a) the 1st chaff technique; b) 2nd chaff technique

As can be seen from Fig. 3a in which ‘-C’ indicates ‘Chaffed’ (the same for other figures), the first chaff technique does not introduce much noise to the data; it stretches the curve a bit. When only the SEND stream is chaffed, the curve has a negative slope. When only the ECHO stream or when both streams is/are chaffed, the curve has a positive slope. As can be seen from Fig. 3b, the second chaff technique is more aggressive than the first one and it introduces significant noise to the data. This is why when the second chaff technique is used, we cannot always distinguish the

curve that corresponded to the real connection by the means described in section 2.2. This is not discouraging because this experiment models an unrealistically difficult situation when users perform the same task at the same time and at least one of the streams is chaffed. It is interesting to note that the second technique may be useful to the adversary for stepping-stone detection evasion.

4.4 Experiment with a Single Connection Chain

The goal of our last experiment was to see if we can distinguish a chaffed (via the second chaff technique) connection chain from unrelated connections. As can be seen from Fig.4, it is possible to distinguish participant C's connection chain when its SEND and ECHO streams are chaffed, but not when both SEND and ECHO streams are chaffed. Curves AE to CS and AE to CS-chaffed exhibit rather weak correlations; this is because they correspond to unrelated connections. Such results are encouraging as they show that even though our procedure may not work very well in the worst-case scenario, it should work fine in the case of a single connection chain unless the hacker chaffs both the ECHO and the SEND streams via the second chaff technique.

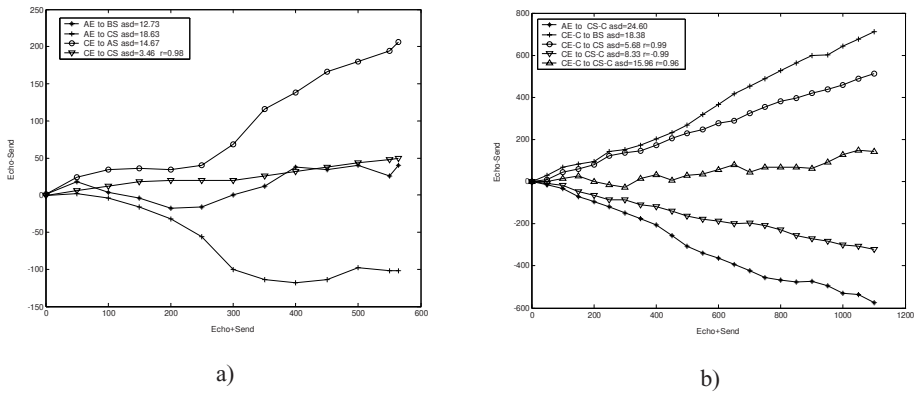


Fig. 4. Correlations of the ECHO stream of a particular participant to SEND streams of participants B and C in the experiment with only a single connection chain. a) without chaff; b) with chaff

5 Conclusions and Future Work

Even though more experimentation is needed before any definitive claims could be made regarding our procedure for finding connection chains, based on our experiments we can with confidence say that procedure described in Section 2.2 always works in distinguishing connection chains that go through the same stepping-stone and carry traffic of users who perform similar operations at the same time when neither time-jittering nor chaff is introduced by the adversary to his/her traffic. Our procedure works well when the first chaff technique is used. The second chaff method is more aggressive, and, therefore, may qualify as a good method for hackers

to use for stepping-stone-detection evasion. Our procedure works well in distinguishing a single connection chain from unrelated incoming and outgoing connections, even when chaff is introduced via the second technique, unless both streams are chaffed.

In the future, we would like to test our connection-chain detection mechanism when chaff and time-jittering are introduced into real-life connections as opposed to simulating these stepping-stone detection evasion tools with data obtained from regular experiments. It would be interesting to address the following questions. How well does our connection-chain method work when more than one user's stream is chaffed and/or when streams are both time-jittered and chaffed? Are there any other methods of measuring linearity of a curve that could yield better results than our procedure with respect to connection-chain detection? How well do other stepping-stone detection mechanisms work when the second chaff technique is used? How successful are our connection-detection procedure and other stepping-stone detection methods when introduction of chaff is not based on probability distributions that are i. i. d.? Ultimately, we would like to design a stepping-stone detection mechanism that would efficiently use our connection-chain detection method and experimentally and/or formally compare the former to other stepping-stone detection mechanisms with respect to running-time complexity, false positive and negative rates and the number of packets required to observe.

Acknowledgement

This project is supported in part by a grant from NSF (SCI-0453498) DoD's ASSURE Program. The authors would like to thank Scott Nielsen, Mykyta Fastovets for their participation in the experiments.

References

1. Blum, A., Song, D., Venkataraman, S.: Detection of Interactive Stepping Stones: Algorithms and Confidence Bounds. In: Jonsson, E., Valdes, A., Almgren, M. (eds.) RAID 2004. LNCS, vol. 3224, pp. 258–277. Springer, Heidelberg (2004)
2. Brunk, H.D.: An Introduction to Mathematical Statistics, Ginn and Company (1960)
3. Donoho, D., Flesia, A.G., Shankar, U., Paxson, V., Coit, J., Staniford, S.: Multiscale Stepping-Stone Detection: Detecting Pairs of Jittered Interactive Streams by Exploiting Maximum Tolerable Delay. In: Wespi, A., Vigna, G., Deri, L. (eds.) RAID 2002. LNCS, vol. 2516, pp. 45–59. Springer, Heidelberg (2002)
4. Duwairi, B., Chakrabarti, A., Manimaran, G.: An Efficient Probabilistic Packet Marking Scheme for IP Traceback. In: Mitrou, N.M., Kontovasilis, K., Rouskas, G.N., Iliadis, I., Merakos, L. (eds.) NETWORKING 2004. LNCS, vol. 3042, pp. 1263–1269. Springer, Heidelberg (2004)
5. Goodrich, M.T.: Efficient Packet Marking for Large-Scale IP Traceback. In: Proc. of ACM CCS '02, Washington, DC, USA, pp. 117–126 (2002)
6. Jung, H.T., Kim, H.L., Seo, Y.M., Choe, G., Min, S.L., Kim, C.S., Koh, K.: Caller Identification System in the Internet Environment. In: Proc. of 4th USINEX Security Symposium, Santa Clara, CA, USA, pp. 69–78 (1993)

7. Savage, S., Wetherall, D., Karlin, A., Anderson, T.: Practical Network Support for IP Traceback. In: Proceedings of the conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, Stockholm, Sweden, pp. 295–306 (2000)
8. Song, D., Perrig, A.: Advanced and Authenticated Marking Scheme for IP Traceback. In: Proc. of IEEE INFOCOM, Anchorage, AL, USA, pp. 878–886 (2001)
9. Snapp, S., et al.: DIDS, (Distributed Intrusion Detection System) – Motivation, Architecture and Early Prototype. In: Proc. of 14th National Computer Security Conference, Columbus, OH, USA, pp. 167–176 (1991)
10. Staniford-Chen, S., Heberlein, L.T.: Holding Intruders Accountable on the Internet. In: Proc. of the IEEE Symposium on Security and Privacy, Oakland, CA, USA, pp. 39–49 (1995)
11. Wang, X., Reeves, D.S., Wu, S.F., Yuill, J.: Sleepy Watermark Tracing: An Active Network-Based Intrusion Response Framework. In: Proc. of 16th International Conference on Information Security, Paris, France, pp. 369–384 (2001)
12. Wang, X., Reeves, D.S.: Robust Correlation of Encrypted Attack Traffic through Stepping Stones by Manipulation of Inter-packet Delays. In: Proc. of the 10th ACM Conference on Computer and Communications Security, Washington, DC, USA, pp. 20–29 (2003)
13. Wang, X.: The Loop Fallacy and Serialization in Tracing Intrusion Connections through Stepping Stones. In: Proc. of the ACM Symposium on Applied Computing, Nicosia, Cyprus, pp. 404–411 (2004)
14. Xin, J., Zhang, L., Aswegan, B., Dickerson, J., Daniels, T., Guan, Y.: A Testbed for Evaluation and Analysis of Stepping Stone Attack Attribution Techniques. In: Proc. of the 2nd International IEEE/Create-Net Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities, Barcelona, Spain (2006)
15. Yoda, K., Etoh, H.: Finding a Connection Chain for Tracing Intruders. In: Proceedings of 6th European Symposium on Research in Computer Security, Toulouse, France, pp. 191–205 (2000)
16. Zhang, Y., Paxson, V.: Detecting Stepping Stones. In: Proc. of the 9th USENIX Security Symposium, Denver, CO, USA, pp. 171–184 (2000)
17. Zhang, L., Persaud, A.G., Johnson, A., Guan, Y.: Detection of Stepping Stone Attack under Delay and Chaff Perturbations. In: Proc. of 25th IEEE International Performance Computing and Communications Conference, Phoenix, AZ, USA (2006)

SPA Countermeasure Based on Unsigned Left-to-Right Recodings

Sung-Kyoung Kim¹, Dong-Guk Han^{2,*}, Ho Won Kim²,
Kyo IL Chung², and Jongin Lim¹

¹ Graduate School of Information Management and Security, Korea University
{likesk,jilim}@cist.korea.ac.kr

² Electronics and Telecommunications Research Institute(ETRI)
{christa,khw,kyoil}@etri.re.kr

Abstract. Vuillaume-Okeya presented unsigned recoding methods for protecting modular exponentiations against side channel attacks, which are suitable for tamper-resistant implementations of RSA or DSA which does not benefit from cheap inversions.

This paper describes new recoding methods for producing SPA-resistant unsigned representations which are scanned from left to right (i.e., from the most significant digit to the least significant digit) contrary to the previous ones. Our contributions are as follows; (1) SPA-resistant unsigned left-to-right recoding with general width- w , (2) special case when $w = 1$, i.e., unsigned binary representation using the digit set $\{1, 2\}$. These methods reduce the memory required to perform the modular exponentiation g^k .

1 Introduction

In the common computation in RSA or DSA, exponentiation algorithms play an important role to construct efficient cryptosystems, but most exponentiation algorithms when implemented on the memory constraint tamper-resistant devices, e.g. smart IC cards, are vulnerable to the physical cryptanalysis such as side channel attacks (SCA) including power analysis attacks and timing attacks [11,12]. One can simply classify power analysis attacks into simple power analysis (SPA) and differential power analysis (DPA). Despite the relative simplicity of the idea of SPA, it is not easy to design secure and efficient SPA countermeasures. However, SPA-resistance is always necessary, and is a prerequisite to DPA resistance. One of the recommended countermeasures against SPA is the fixed procedure of operations without using dummy operations, e.g. [17,19]. These countermeasures employ the technique of signed digit representation, and require the efficient operation of inversion or division. Namely, there countermeasures have been developed for elliptic curve cryptosystems (ECCs). Even though there are many SPA countermeasures using fixed pattern which is derived from signed representation they cannot be directly transposed to RSA or DSA because it does not benefit from cheap inversions.

* Corresponding author.

Recently, Vuillaume-Okeya have proposed SPA-resistant unsigned recoding methods suitable to RSA [24]. Their approach is to extend Möller's recoding [17] by using 2^w -ary unsigned digit set $\{1, 2, \dots, 2^w\}$. Even though several advantages of it, a principal disadvantage is that it uses right-to-left recoding to generate unsigned digit representations, so the recoded string must be computed and stored before the left-to-right exponentiation. As the left-to-right exponentiation is the natural choice (Refer to section 2.3 for details), to avoid the need to record the new representation it is an important challenge to construct a left-to-right recoding.

Contributions of this paper are as follows; (1) first, we show how to transform the Vuillaume-Okeya's right-to-left recoding [24] into a left-to-right version. (2) Then, we present a binary left-to-right recoding using $\{1, 2\}$, which is very simple to be implemented because we need only two information - one bit k_j at a target position and the first index s from the least significant bit of the input string such that $k_s = 0$ - to decide a recoded digit.

2 Side Channel Attacks and Countermeasures

Side channel attacks (SCA) are allowed to access the additional information linked to the operations using the secret key, e.g., timings, power consumptions, etc. The attack aims at guessing the secret key (or some related information). For example, L-t-R Binary_Method (or R-t-L Binary_Method) can be broken by SCAs. The binary method computes a square and multiplication operation if the bit $k_i = 1$, and only a square if $k_i = 0$. The standard implementation of multiplications is different from that of squares, and thus the multiplication in the exponentiation can be detected using SCAs.

L-t-R Binary_Method	R-t-L Binary_Method
Input: exponent $k = \sum_{i=0}^{n-1} k_i 2^i$, basis g ;	Input: exponent $k = \sum_{i=0}^{n-1} k_i 2^i$, basis g ;
Output: g^k ;	Output: g^k ;
1. $Q[0] \leftarrow g$	1. $Q[0] \leftarrow g, Q[1] \leftarrow 1$
2. for $i = n - 2$ down to 0	2. for $i = 0$ up to $n - 1$
2.1. $Q[0] \leftarrow Q[0]^2$	2.1. if $(k_i == 1)$
2.2. if $(k_i == 1)$	2.2. $Q[1] \leftarrow Q[0] * Q[1]$
2.3. $Q[0] \leftarrow Q[0] * g$	2.2. $Q[0] \leftarrow Q[0]^2$
3. return $(Q[0])$	3. return $(Q[1])$

If an attacker is allowed to observe the side channel information only a few times, it is called the simple power analysis (SPA). If an attacker can analyze several side channel information using a statistical tool, it is called the differential power analysis (DPA). The standard DPA utilizes the correlation function that can distinguish whether a specific bit is related to the observed calculation. We have to design the implementation of cryptographic algorithms by rendering careful attention to SCA.

2.1 Why Against SPA Is Essential?

By definition, DPA requires that the same secret is used to perform several cryptographic operations with each time a different input value: decrypting or signing several messages, for instance. However, SPA-resistance is always necessary, and is a prerequisite to DPA resistance. For instance, if only one random ephemeral exponent of DSA or EC-DSA is revealed, the secret key of the signature scheme can be easily inferred. Similarly, from the point of view of an attacker, a blinded RSA exponent $d + r\phi(n)$ is as good as the secret itself. Thus, this paper focuses on SPA-resistant exponentiation.

To prevent SPA attacks, many countermeasures have been proposed; the standard approach is to use fixed pattern algorithms [4,14] that compute a square and multiplication operation for each bit of the secret key. Another type of countermeasures include indistinguishable addition formulae [7,8], but they can not be applied on general elliptic curves. To prevent SPA attacks against an algorithm using pre-computed points, Möller [17] used a new representation without zero digits for the secret scalar, which ensures a fixed pattern computation for the algorithm.

Even though there are many SPA countermeasures using fixed pattern which is derived from signed representation they cannot be directly transposed to RSA or DSA because it does not benefit from cheap inversions. Thus, one of the solutions is to find unsigned representations.

2.2 Vuillaume-Okeya's Countermeasure

Recently, Vuillaume-Okeya [24] have proposed SPA-resistant unsigned recoding methods which are well-suited for tamper-resistant implementations of RSA or DSA. The proposed recoding method constructs an addition chain of a fixed pattern, e.g. $|0..0y|0..0y|...|0..0y|$, where y is chosen value from the pre-computed table which has only positive values. Their approach is to extend Möller's recoding [17] which has been designed for ECC to the unsigned case. To obtain the unsigned digit set $\{1, 2, \dots, 2^w\}$, the key idea of it is to use *negative* carries rather than positive carries:

1. replace the digit 0 with 2^w , and add a carry of -1 to the next window when scanning the scalar from right to left,
2. replace the digit -1 with $2^w - 1$, and add a carry of -1 to the next window,
3. otherwise leave the digit as it is.

To ensure a correct termination, they treat the case of the most significant bit separately: if a carry remains at the end of the recoding, they use the most significant bit to neutralize it, and reduce the length of the exponent. To remove the information of length of the exponent in the case of RSA, they extended the bit length of the exponent by 2 like as $k_{n+1} = 1$ and $k_n = 0$ by repeatedly adding $\phi(N)$ to the exponent until one can obtain the condition, where exponent k is n bits. Here, ϕ is Euler phi function and $N = p \cdot q$ for two selected primes in RSA. The output of exponentiation using this extension of the exponent does not

change the original output because $g^{k'+\phi(N)} = g^{k'} \bmod N$. Under this treatment, the length of the recoded expansion is always reduced by one, i.e. the length of it is fixed.

SPA-resistant exponentiation algorithm computes g^k with an unsigned SPA-resistant recoding of k from Vuillaume-Okeya's recoding algorithm. Exponentiation algorithm is composed of three stages that are pre-computation, recoding, and evaluation. During the exponentiation, the operation pattern is fixed: w squares and 1 multiplication.

2.3 Motivation of This Paper

In general, to design a countermeasure against SPA exponent recoding is one of the possible techniques, e.g. [5,17,23,24]. In general, performing exponent recoding is categorized into two main concepts: left-to-right and right-to-left. For the purpose of memory constraint devices we prefer *left-to-right* to right-to-left recoding due to the following reasons (Refer to [18]):

- left-to-right evaluation (e.g. L-t-R Binary_Method) can be adjusted for window recoding method, i.e. using the pre-computed values, than right-to-left version (e.g. R-t-L Binary_Method).
- right-to-left evaluation method needs an auxiliary register for storing intermediate data (e.g. in the case of R-t-L Binary_Method g^{2^i} in $Q[0]$ is an auxiliary register compared to L-t-R Binary_Method).

Thus, if L-t-R Binary_Method is used as an exponentiation algorithm and exponent recoding is done right-to-left, then it is necessary to finish the recoding and to store it before starting the left-to-right evaluation stage. Namely, we require additional n -bit (i.e. exponential size $O(n)$) RAM for the right-to-left exponent recoding.

On the other hands, if left-to-right recoding techniques are available, the recoding can be done at the same time as L-t-R Binary_Method, avoiding the need to record the new representation. This makes left-to-right recodings somewhat more interesting for implementations in restricted environments.

Even though several advantages of Vuillaume-Okeya's Countermeasure described in the previous section, the principal disadvantage of it is that it is a right-to-left direction recoding.

3 SPA-Resistant Unsigned Left-to-Right Recodings

We show how to perform exponentiation g^k in a way such that multiplications and squares occur in a fixed pattern in order to provide resistance against side channel attack. Section 3.1 describes a main idea of unsigned left-to-right recoding. In section 3.2, we discuss the recoding algorithm for a general case (width- w). Section 3.3 show that special cases, i.e., $w = 1$. In section 3.4 we precisely analysis the efficiency and security of proposed recoding method. For simplicity, we assume some parameters used in this section are targeted at RSA. ϕ is Euler phi function and $N = p \cdot q$ for two selected primes in RSA.

3.1 Main Idea

We propose a left-to-right recoding which translates a given integer k represented with the digit set $\{0, 1, \dots, 2^w - 1\}$ to a recoded integer k' with $k'_j \in \{1, 2, \dots, 2^w\}$ such that $k = k'$.

First of all, we define some notations used for this paper and give a full detail of a proposed technique.

- Let $k = (k_{n-1} \dots k_0)_2$ be n -bit binary representation of an integer with $k_i \in \{0, 1\}$, and $k' = (10k_{n-1} \dots k_0)_2$ be a $(n+2)$ -bit binary string obtained by repeatedly adding $\phi(N)$ to k . The reason of this treatment is the same that of Vuillaume-Okeya's Countermeasure to remove the information of length of the exponent during the recoding.
- Let w be an integer such as $w \geq 2$; we set $d = \lceil \frac{n+2}{w} \rceil$.
- Write $k' = B^{d-1} \parallel \dots \parallel B^1 \parallel B^0$ by padding k' on the left with 0's if necessary, where each B^j is a bit-string of length w .
- We define $[B^j] := \sum_{i=0}^{w-1} B_i^j 2^i$ where B_i^j denote the i -th bit of B^j . Then, $[B^j] \in \{0, 1, \dots, 2^w - 1\}$.
- Let $E^{d-1} \parallel \dots \parallel E^1 \parallel E^0$ be a recoded string of k' where each E^j is a bit-string of length w and $[E^j] \in \{1, 2, \dots, 2^w\}$. Similarly, define $[E^j] := \sum_{i=0}^{w-1} E_i^j 2^i$. Note that when $[E^j]$ is 2^w , E^j can be represented as $(\underbrace{11 \dots 1}_w 2)$.

Vuillaume-Okeya's recoding algorithm can generate $[E^j]$ from k' from the right-to-left direction, i.e. generate from $[E^0]$ to $[E^{d-1}]$ due to the negative carries. Our goal is to generate $\{[E^j]\}$ from the left-to-right direction, i.e. from $[E^{d-1}]$ to $[E^0]$.

The main idea is derived from Vuillaume-Okeya's recoding algorithm [24]. First, we divide k into several groups such that each group has following property; for example, let $B^9 \parallel B^8 \parallel B^7 \parallel B^6 \parallel B^5 \parallel B^4$ be one of the divided groups, then the first one $[B^9] \geq 2$, others are one of $\{0, 1\}$, and $[B^4] \geq 2$. There are two cases we should consider.

1. There is no $B^j = 0$ for $5 \leq j \leq 8$.

In this case, there is no change in recoding to E^i for $5 \leq i \leq 9$, i.e. $E^i = B^i$.

2. For some $5 \leq z \leq 8$ there exist $B^z = 0$.

- 2.1. $[E^9] = [B^9] - 1$;

- 2.2. $[E^j] = [B^j] + (2^w - 1)$ for $z < j \leq 8$.

Not that if $z = 8$ then this step is not required;

- 2.3. $[E^z] = 2^w$;

- 2.4. $[E^j] = [B^j]$ for $5 \leq j < z$.

Not that if $z = 5$ then this step is not required.

From the following equation, clearly a recoded integer k' by the proposed recoding method is the same integer k . For $\sum_{j=t_1}^{t_2} [B^j] \cdot 2^{jw}$ with $[B^{t_1}] \geq 2$ and the others are 0 or 1, and z is the first index from the right-to-left direction such that $[B^z] = 0$ (assume $t_1 < z < t_2 - 1$), then

$$\begin{aligned}
& \sum_{j=t_1}^{t_2} [B^j] \cdot 2^{jw} \\
&= \underbrace{([B^{t_2}] - 1)}_{[E^{t_2}] \neq 0} \cdot 2^{t_2 w} + \sum_{j=z+1}^{t_2-1} \underbrace{([B^j] + 2^w - 1)}_{[E^j]} \cdot 2^{jw} + \underbrace{2^w}_{[E^z]} \cdot 2^{zw} + \sum_{j=t_1}^{z-1} \underbrace{[B^j]}_{[E^j]} \cdot 2^{jw}. \quad (1)
\end{aligned}$$

As our method is operated without carry, we need not consider a carry during the process of recoding k . Therefore, the representations of k and k' are exactly same.

3.2 Proposed Algorithm: General Case

When we consider a representation of k as one of its width- w unsigned representations, different width- w unsigned representations of k can be obtained by using the following Recoding 2 based on the equation (1). Before describing Recoding 2, we define the following notations.

- **Start** is MSB in the recoding block.
- **End** is the first index $t (< \text{Start})$ such that $[B^t] \geq 2$ from the left-to-right direction from **Start**. If there is not t until index 0, then let **End** = -1.
- **Zero** is the first index $z (> \text{End})$ such that $[B^z] = 0$ from the right-to-left direction from **End**. If there is not z between **Start**-1 to **End**+1 then **Zero** = NULL.

Recoding 1. Unsigned Left-to-Right Recoding

Input: $k' = B^{d-1} \parallel \dots \parallel B^1 \parallel B^0$;

Output: Recoded exponent $[E^{d-1}], \dots, [E^1], [E^0]$;

1. **Start** $\leftarrow d - 1$;
 2. **while** **Start** ≥ 0 **do**
 - 2.1. Find **End**, **Zero**;
 - 2.2. If (**Zero**=NULL), then
 - for $j = (\text{Start})$ down to $(\text{End}+1)$ do: $[E^j] \leftarrow [B^j]$;
 - 2.3. Else (**Zero** $\neq$ NULL),
 - 2.3.1. $[E^{\text{Start}}] \leftarrow [B^{\text{Start}}] - 1$;
 - 2.3.2. if (**Zero** $\neq$ **Start**-1), then
 - for $j = (\text{Start}-1)$ down to $(\text{Zero}+1)$ do: $[E^j] \leftarrow [B^j] + (2^w - 1)$;
 - 2.3.3. $[E^{\text{Zero}}] \leftarrow 2^w$;
 - 2.3.4. if (**Zero** $\neq$ **End**+1), then
 - for $j = (\text{Zero}-1)$ down to $(\text{End}+1)$ do: $[E^j] \leftarrow [B^j]$;
 - 2.4. **Start** $\leftarrow \text{End}$;
-

3.3 Special Case: $w = 1$

When $w = 1$, i.e. the digit set is $\{1, 2\}$, Recoding 1 can be simplified to Recoding 2. The original n -bits exponent k is extended to $(n + 2)$ -bits to keep up the length of the recoded expansion like as Vuillaume-Okeya's recoding algorithm. From Recoding 2 we can lead the following formulae;

$$\sum_{j=0}^{n+1} k_j 2^j = \sum_{j=z+1}^n (k_j + 1) 2^j + 2 \cdot 2^z + \sum_{j=0}^{z-1} k_j 2^j, \quad (2)$$

where z is the first index from the least significant bit such that $k_z = 0$ in the binary representation of k' . Equation (2) is easily prove;

$$\begin{aligned} \sum_{j=z+1}^n (k_j + 1) 2^j + 2 \cdot 2^z + \sum_{j=0}^{z-1} k_j 2^j &= \sum_{j=z+1}^n k_j 2^j + \left(\sum_{j=z+1}^n 2^j + 2 \cdot 2^z \right) + \sum_{j=0}^{z-1} k_j 2^j \\ &= \sum_{j=z+1}^n k_j 2^j + (2^{n+1}) + \sum_{j=0}^{z-1} k_j 2^j = \sum_{j=0}^{n+1} k_j 2^j. \end{aligned}$$

The last equation is derived from the conditions $k_{n+1} = 1$ and $k_z = 0$.

Recoding 2. Unsigned Binary Left-to-Right Recoding

Input: $(n+2)$ -bits exponent $k' = (k_{n+1} \cdots k_0)_2$ with $k_{n+1} = 1$ and $k_n = 0$;

Output: Recoded exponent $(e_n \cdots e_0)$ where $e_j \in \{1, 2\}$;

1. Find the first index z from the least significant bit such that $k_z = 0$;
 2. $j \leftarrow n$;
 3. **while** $j \geq 0$ **do**
 - 2.1. **while** $j > z$ **do**
 - 2.1.1. if $k_j = 0$ then $e_j \leftarrow 1$;
 - 2.1.2. else $(k_j = 1)$ then $e_j \leftarrow (1 + k_j)$;
 - 2.2. **while** $j \leq z$ **do**
 - 2.2.1. if $j = z$ then $e_j \leftarrow (2 + k_j)$;
 - 2.2.2. else $(j < z)$ then $e_j \leftarrow k_j$;
 - 2.3. $j \leftarrow j - 1$;
-

Algorithm 1 shows the explicit process for computing $g^{k'}$ with the unsigned binary left-to-right recoding (Recoding 2), which merges the recoding and evaluation stage into one procedure.

Algorithm 1. SPA-resistant Exponentiation based on Recoding 3

Input: $(n+2)$ -bits exponent $k' = (10k_{n-1} \cdots k_0)_2$, base g ;

Output: $c = g^{k'}$;

1. **Pre-computation** $g[1] \leftarrow g$ and $g[2] \leftarrow g^2$;
2. **Recoding+Evaluation**
 - 2.1. Find the first index z from the least significant bit such that $k_z = 0$;
 - 2.2. $j \leftarrow n$;
 - 2.3. **while** $j \geq 0$ **do**
 - 2.3.1. $c \leftarrow c^2$;
 - 2.3.2. if $j > z$ then

- (a) if $k_j = 0$ then $c \leftarrow c * g[1]$;
 - (b) else ($k_j = 1$) then $c \leftarrow c * g[1 + k_j]$;
 - 2.3.3. if $j \leq z$ then
 - (a) if $j = z$ then $c \leftarrow c * g[2 + k_j]$;
 - (b) else ($j < z$) then $c \leftarrow c * g[k_j]$;
 - 2.3.4. $j \leftarrow j - 1$;
 - 2.4. Return c ;
-

3.4 Analysis of the Unsigned Left-to-Right Recoding

In this section we discuss efficiency and security of the proposed scheme.

Efficiency. Unsigned Left-to-Right Recoding algorithm generates a scalar sequence that has a fixed pattern, e.g. $|\underbrace{0 \dots 0}_w y| \underbrace{0 \dots 0}_w y| \dots |\underbrace{0 \dots 0}_w y|$, where $y \in \{1, 2, \dots, 2^w\}$. The pre-computation stage of it is the same that of SPA-resistant Exponentiation based on Vuillaume-Okeya's recoding algorithm. But, as unsigned Left-to-Right Recoding algorithm generates $[E^j]$ from the left-to-right direction we can merge Recoding stage and Evaluation stage of SPA-resistant Exponentiation based on Vuillaume-Okeya's recoding algorithm into one procedure which require w squares and one multiplication per each window.

Security. The security of the proposed recoding algorithm is depending on the following assumption;

Square c^2 and multiplication $c*y$ are distinguishable by one-time measurement of power consumption, whereas $c*y$ and $c*y + \alpha$ are indistinguishable. Here, α is the cost of addition between two $(w + 1)$ -bit strings.

In case of Recoding 1, steps 2.2. and 2.3.4. are related with $c*y$, and others are with $c*y + \alpha$ such as step 2.3.2. However, as a bit length of c and y is 1024 or 2048-bit and in general w is selected less than 10 the above assumption is reasonable. Because α is almost free compared to the cost of multiplication of big numbers (> 1000 -bit). Thus, exponentiation algorithm based on our recoding algorithm is secure against SPA from the above assumption's point of view. This assumption is similar to that of ECC, that is elliptic curve addition (or subtraction) and elliptic curve doubling are distinguishable by one-time measurement of power consumption, whereas elliptic curve addition and elliptic curve subtraction are indistinguishable. Note that if an attacker can distinguish this kind of difference α then the security of SPA-resistant Exponentiation based on Vuillaume-Okeya's recoding algorithm is also controversial. Because there is difference in step 2.2. (or step 3.2.) depending on the condition of u_i in Vuillaume-Okeya's recoding algorithm.

And, the proposed method computes scalar multiplication through the fixed pattern $|0 \dots 0y|0 \dots 0y| \dots |0 \dots 0y|$, where $y \in \{1, 2, \dots, 2^w\}$. The attacker could distinguish square and multiplication in the scalar exponentiation by measurement of the power consumption. However, he obtains the identical sequence $|S \dots SSM|S \dots SSM| \dots |S \dots SSM|$ for all the scalars. Therefore, he cannot detect the secret scalar by using SPA.

4 Conclusion

We presented four types SPA-resistant integer recodings, which are necessary for achieving high efficiency with RSA, DSA, or pairing based cryptosystems. These recodings are left-to-right so they can be interleaved with a left-to-right exponentiation, removing the need to store both the exponent and its recoding. It should be kept in mind that these recodings do not ensure in any way the security against differential power analysis, so countermeasures against these attacks should also be used if the secret key is used more than one.

Acknowledgements

Sung-Kyoung Kim was supported by the MIC(Ministry of Information and Communication), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute of Information Technology Advancement) (IITA-2006-(C1090-0603-0025)).

References

1. Aydos, M., Yank, T., Koc, C.K.: High-speed implementation of an ECC-based wireless authentication protocol on an ARM microprocessor. *IEE Proceedings Communications* 148, 273–279 (2001)
2. Barreto, P., Galbraith, S., Eigeartaigh, C., Scott, M.: Efficient Pairing Computation on Supersingular Abelian Varieties, *Cryptology ePrint Archive: Report 2004/375* (2004)
3. Bertoni, G., Guajardo, J., Kumar, S., Orlando, G., Paar, C., Wollinger, T.: Efficient $GF(p^m)$ Arithmetic Architectures for Cryptographic Applications. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 158–175. Springer, Heidelberg (2003)
4. Coron, J.S.: Resistance against differential power analysis for elliptic curve cryptosystems. In: Koç, Ç.K., Paar, C. (eds.) CHES 1999. LNCS, vol. 1717, pp. 292–302. Springer, Heidelberg (1999)
5. Hedabou, M., Pinel, P., Bebeveau, L.: Countermeasures for Preventing Comb Method Against SCA Attacks. In: Deng, R.H., Bao, F., Pang, H., Zhou, J. (eds.) ISPEC 2005. LNCS, vol. 3439, pp. 85–96. Springer, Heidelberg (2005)
6. Harrison, K., Page, D., Smart, N.: Software Implementation of Finite Fields of Characteristic Three. *LMS Journal of Computation and Mathematics* 5, 181–193 (2002)
7. Joye, M., Quisquater, J.J.: Hessian elliptic curves and side-channel attacks. In: Koç, Ç.K., Naccache, D., Paar, C. (eds.) CHES 2001. LNCS, vol. 2162, pp. 412–420. Springer, Heidelberg (2001)
8. Joye, M., Tymen, C.: Protections against differential analysis for elliptic curve cryptography: an algebraic approach. In: Koç, Ç.K., Naccache, D., Paar, C. (eds.) CHES 2001. LNCS, vol. 2162, pp. 386–400. Springer, Heidelberg (2001)
9. Joye, M., Yen, S.: Optimal Left-to-Right Binary Signed-Digit Recoding. *IEEE Trans. Computers* 49, 740–748 (2000)
10. Koblitz, N.: Elliptic curve cryptosystems. *Mathematics of Computation* 48, 203–209 (1987)

11. Kocher, P.: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996)
12. Kocher, P., Jaffe, J., Jun, B.: Differential Power Analysis. In: Wiener, M.J. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999)
13. Lauter, K.: The advantages of elliptic curve cryptography for wireless security. IEEE Wireless Communications 11, 62–67 (2004)
14. Lopez, J., Dahab, R.: Fast multiplication on elliptic curves over $GF(2^m)$ without precomputation. In: Koç, Ç.K., Paar, C. (eds.) CHES 1999. LNCS, vol. 1717, pp. 316–327. Springer, Heidelberg (1999)
15. Lim, C.: A new method for securing elliptic scalar multiplication against side channel attacks. In: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP 2004. LNCS, vol. 3108, pp. 289–300. Springer, Heidelberg (2004)
16. Miller, V.S.: Use of elliptic curves in cryptography. In: Williams, H.C. (ed.) CRYPTO 1985. LNCS, vol. 218, pp. 417–426. Springer, Heidelberg (1986)
17. Möller, B.: Securing Elliptic Curve Point Multiplication against Side-Channel Attacks, *Information Security - ISC'01*, LNCS, vol. In: Davida, G.I., Frankel, Y. (eds.) ISC 2001. LNCS, vol. 2200, pp. 324–334. Springer, Heidelberg (2001)
18. Okeya, K., Schmidt-Samoa, K., Spahn, C., Takagi, T.: Signed Binary Representations Revisited. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 123–139. Springer, Heidelberg (2004)
19. Okeya, K., Takagi, T.: The width- w NAF method provides small memory and fast elliptic scalar multiplications secure against side channel attacks. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 328–343. Springer, Heidelberg (2003)
20. Page, D., Smart, N.: Hardware Implementation of Finite Fields of Characteristic Three. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 529–539. Springer, Heidelberg (2003)
21. Ruan, X., Katti, R.: Left-to-Right Optimal Signed-Binary Representation of a Pair of Integers. IEEE Trans. Computers 54, 124–131 (2005)
22. Shin, J.H., Park, D.J., Lee, P.J.: DPA Attack on the Improved Ha-Moon Algorithm. In: Song, J., Kwon, T., Yung, M. (eds.) WISA 2005. LNCS, vol. 3786, pp. 283–291. Springer, Heidelberg (2006)
23. Theriault, N.: SPA Resistant Left-to-Right Integer Recodings. In: Preneel, B., Tavares, S. (eds.) SAC 2005. LNCS, vol. 3897, pp. 345–358. Springer, Heidelberg (2006)
24. Vuillaume, C., Okeya, K.: Flexible Exponentiation with Resistance to Side Channel Attacks. In: Zhou, J., Yung, M., Bao, F. (eds.) ACNS 2006. LNCS, vol. 3989, pp. 268–283. Springer, Heidelberg (2006)

A New One-Way Isolation File-Access Method at the Granularity of a Disk-Block^{*}

Wenyuan Kuang¹, Yaoxue Zhang¹, Li Wei¹, Nan Xia²,
Guangbin Xu¹, and Yuezhi Zhou¹

¹ Department of Computer Science and Technology, Tsinghua University,
Beijing, 100084, P.R. China

kuangwy04@mails.tsinghua.edu.cn

² Institute of Computer Network Systems, Hefei University of Technology,
Hefei City, 230009, Anhui Province, China

Abstract. In this paper, we propose a file-access method, called as OI DB(One-way Isolation Disk Block), which always keeps original files on pristine status and enables users to access files without restricting the functionality. This is accomplished via one-way isolation: users can read files in origin storage, but they are not permitted to write files in the origin storage. Their writing operations are redirected to the temporary storage. A key property of our approach is that files are accessed at the granularity of a disk-block, which makes the overheads of disk-block copying be alleviated largely. OI DB supports a wide range of tasks, including: sharing pristine data, supporting file system versioning, testing unauthentic software. The employment of OI DB in TransCom System has verified its desirable features. The performance evaluation represents that OI DB introduces low disk-block copying overheads in the process of accessing a file, especially in modifying a file.

1 Introduction

Isolation is among the basic methods to promote security of information system. Isolation is to contain the effect of operations without full trust, but not to restrict functionality. Some protocols for realizing one-way isolation in the context of databases and file systems have been developed, but broad application of these methods suffers from restricted functionality and degraded performance.

TFS[3] is a combined file system constructed with an upper writeable file system and a bottom read-only one. In union mount file system[7] of 4.4BSD?Lite, a union mount presents a view of a merger of the two directories and only files in the upper layer of the union stack can be modified. Elephant[1] file system is a file versioning system which provides rollback capability. It achieves this by utilizing the VFS of FreeBSD to make version copies of inode's meta-data and writing the latest copy when users want to modify the data.

^{*} This research was supported by National High-Tech Research and Development Plan of China under Grant No. 2005AA114160.

SEE[6] presents an approach for realizing a Safe Execution Environment (SEE) which key property is that it faithfully reproduces the behavior of applications, as if they were running natively on the underlying host operating system. SEE restrict all modification operations other than those that involve the file system and the network, which is called static redirection. In file system, the one-way isolation metrics is accomplished via Isolation File System (IFS). Processes running in the IFS within the SEE are given read-access to the environment provided by the host OS, but their write operations are prevented from escaping outside the SEE. It is a kind of one-way isolation at the granularity of a file. But the operation on files of the host operating system is restricted by the underlying file system. And in IFS, modifying a file needs copying the complete file first, even when just a word is to be modified, so the performance of IFS is not satisfied.

In this paper, we propose the one-way isolation file-access method at the granularity of a disk-block, OIDB, which sets no restriction on functionality and largely alleviates the overheads of disk copying. Applications of OIDB include:

Sharing pristine data. In some contexts, users access the same pristine data simultaneously. Setting locks on the pristine data is the usual way to resolve the conflicts between peers accessing the data. But it is not flexible and time-consuming. A file system using OIDB can support many users sharing pristine data without locking the data.

File system versioning support. Check pointing technique[2,5] is always used to provide file versioning. [4] uses a stackable template file system and a sparse file technique to reduce storage requirements for storing versions of large files. OIDB can keep all the modification copies of the pristine data with least disk expend. So, it can support file system versioning easily.

Testing unauthentic software. Computer systems always face risks when users execute unauthentic software, such as downloaded freeware/shareware. An isolation file system with OIDB can minimize risks to test unauthentic software without degrading the functionality.

In order to support the tasks mentioned above, OIDB must provide the following features:

One-way isolation protection: Users can read files in the origin storage, but their writing operations are redirected to the temporary storage and prevented from modifying data in the origin storage. The integrity of the pristine data in the origin storage will not be broken.

Complete functionality without restriction: Users can access file via OIDB transparently just as in a normal file system. And users can observe that the result of operations on the file via OIDB is just the same as operated in a common file system.

The rest of this paper is organized as follows. Section 2 presents an overview of our approach. Section 3 presents the implementation details of OIDB. Section 4 describes the employment sample of OIDB in TransCom System and evaluates the functionality and performance. Finally, Section 5 concludes this paper.

2 Approach Overview

In principle, we describe a file system as a tree structure. The root of a tree represents a volume of the file system. Internal nodes in this tree correspond to directories or files, whereas the leaves correspond to disk blocks holding real file data and the metadata of the file system. Every internal node refers to a leaf node as his child and a leaf node includes at least one disk block which stores the metadata. The disk blocks holding the file data may not be in continuous physical address. The other children of directory nodes may themselves be directories or files. All internal nodes represent the logical relationships of the file system. The leaf nodes actually store data of the file system.

In the definition of NIST[9], the *tree* structure is: (1) *A data structure accessed beginning at the root node. Each node is either a leaf or an internal node. An internal node has one or more child nodes and is called the parent of its child nodes. All children of the same node are siblings. Contrary to a physical tree, the root is usually depicted at the top of the structure, and the leaves are depicted at the bottom.* (2) *A connected, undirected, acyclic graph. It is rooted and ordered unless otherwise specified.*

Based on the NIST tree definition, we define **file system tree** structure: *A tree in which each internal node has one child at least, and the child is the one and only leaf node among the children of the internal node.*

The file system tree is layered by internal nodes and leaf nodes, which suggests that we can realize one-way isolation semantics at different layers. The basic method to realize one-way isolation semantics in file system is to make a copy of the original file and redirect the modification and subsequent operations on the file to the new copy. The file can be copied in entirety or partly. In IFS, the copy is made at the granularity of a file when the system receives the modification request for the file. But this method restricts file accessing functionality severely when it is applied in some COTS OSes, such as Windows 2000 and XP. Because in these OSes, system prohibits copying core files in entirety for security concerns, for example the register files and foundational dynamic link libraries can not be copied. Moreover the overheads of file copying of IFS are not satisfied. An utmost situation is that just one bit would be modified in a large size file, but the entire file must be copied before the bit is actually modified.

So, we propose a new one-way isolation file-access method at the granularity of a disk-block, which is different from the method of IFS distinguishably in the different layers of file copying operation. The file is copied partly. It is at disk-block layer where the file copying is taken place in OIDB. So it has no restriction on functionality when it is applied in the COTS OSes. OIDB largely alleviates disk copying overheads and leads to a valuable disk space saving.

Figure 1 illustrates the operations of OIDB. There are three layers in the figure. The bottom layer corresponds to the original storage which holds the original file data. The middle layer is the temporary storage to hold modified copy of files and directories. The top layer shows the only view to be observed by the users, which is a combination of the views in the bottom two layers. The ordering of the bottom two layers in the figure is just as the same as their

place in the figure: the view contained in the temporary storage has been given higher priority, it always overrides the view provided by the original storage. In the figure, black represents effective element and dashed represents combined element.

We illustrate the operations of OI DB using the examples shown in Figure 1.

Step 1: Initial status. There are two directories d1 and d2 under the root directory r, with files f1 and f2 within directory d1. Directory r refers to disk block rb1. Directory d1 refers to disk blocks d1b1 and d1b2. Directory d2 refers to disk block d2b1. File f1 refers to disk blocks, namely f1b1/f1b2/f1b3. File f2 refers to disk block f2b1.

Step 2: The result of modifying file f1 in disk block f1b2. The copy-on-write operation on f1b2 first creates a copy of f1b2 in the temporary storage, i.e., the disk block f1b2 is copied from the origin storage to the temporary storage. Then user modifies the content of the new copy. We create stub nodes of f1b2's ancestor directories and file, namely, r, d1 and f1, in the temporary storage figure to illustrate the relationship clearly. The black internal node f1 in the temporary storage represents that effective data of f1 resides in the temporary storage. The f1b2 in the temporary storage overrides the f1b2 in the origin storage in the combined view. The combined view of step2 includes the effective leaf node f1b2 in temporary storage and other effective leaf nodes in origin storage. Subsequent accesses to f1b2 are redirected to this copy in temporary storage.

Step 3: The result of deleting file f2. The copy-on-write operation on d1b1 first creates a copy of d1bq, i.e, the disk block d1b1 storing the metadata of the directory d1 is copied from the origin storage to the temporary storage. Then the link of file f2 is deleted from the content of d1b1. The black node d1 represents that effective data of d1 resides in the temporary storage. The d1b1 in the temporary storage overrides the d1b1 in the origin storage in the combined view. The combined view of step3 reflects all these changes. User can not find file f2 in the combined view which includes the effective leaf node d1b1 in the temporary storage.

Step 4: The result of creating file /r/d2/f3. The copy-on-write operation on d2b1 first creates a new copy of d2b1, i.e., the disk block d2b1 is copied from the origin storage to the temporary storage. Then disk block f3b1 is assigned to store the data of f3 and the meta-data of the directory d2 is modified to create a link to file f3. The black internal nodes, d2 and f3, in the temporary storage represents that effective data of d2 and f3 reside in the temporary storage. The d2b1 in the temporary storage overrides the d2b1 in the origin storage in the combined view. User can observe file f3 in the combined view which includes the effective leaf nodes d2b1 and f3b1 in the temporary storage.

3 Design Details of OI DB

In general, when a request to write a file is received, file system proceeds as follows: a file system driver accepts a request to write data to a certain location within a particular file from an application. It translates the request into a

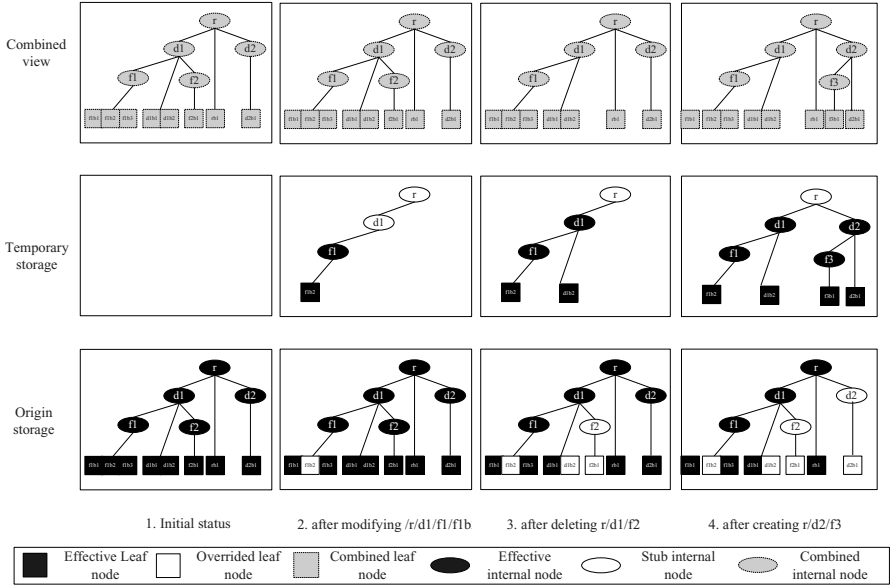


Fig. 1. Illustration of OI DB operations

request to write a certain number of bytes to the disk at a particular "logical" location. It then passes this request to a disk driver. The disk driver, in turn, translates the request into a physical location (cylinder/track/sector) on the disk and manipulates the disk heads to write the data.

In our approach, physical storage includes the origin storage and the temporary storage. The origin storage and the temporary storage can be real disk volumes or image files stored in a file system. And they can be located in the same storage node or distributed in a networked storage system. The policy to allocate origin storage in OI DB is just the same as in a normal file system. Storage space can reallocate when it is released. The temporary storage can be pre-allocated the same size as the origin storage or allocated on demand. There are some differences in performance and storage occupancy with different methods to allocate space. It is clear the storage occupancy is much better with the latter one than with the former one. But the performance in the former is better than in the latter in most situations. When the temporary storage grows at will, it likely grows across non-contiguous blocks on the hard drive. The disk driver will spend more time to locate the real data. This results in a fragmented disk file that slows performance. By using pre-allocated disks, the temporary storage can sit on a contiguous range of blocks on physical disk and thus not become fragmented as content is added to the temporary storage. So when performance is the focus concerned, then deploying OI DB with pre-allocated temporary storage is definitely a better choice.

The pristine data in the origin storage is installed in advance in a normal file system without OI DB. If the pristine data is to be updated, just as the installing

process, it is naturally to stop the OIIB driver embedded in the file system and update the pristine data in the origin storage in a normal file system.

OIIB provides the latest version of the file through the combined view, which includes the latest modified file data in the temporary storage and the pristine file data in the origin storage. And the latest version is the only one version provided in our design, though OIIB can support keeping all modified version of the file easily.

The temporary storage holds the latest file data and the origin storage holds the pristine data. The basic unit of the origin storage is a sector. Sectors are hardware-addressable blocks on a storage medium. Hard disks almost always define a 512-byte sector size. Thus a 4GB disk is divided to 8192 sectors. The basic unit of the temporary storage is a block, which can be actually a real disk block or a block in an image file allocated on demand. The size of a block is changeable. Effective data may be stored in both the two storage. In order to point out the actual storage place of effective data, we design a table that maintains additional information necessary in OIIB. This table, which we call as evolutive blocks table EBT, is indexed by the number of the sector in the origin storage. It has a field indicating that whether the sector stored in temporary storage or stored in the origin storage. Further, if it is stored in temporary storage, the place where it is stored should be provided, which means that OIIB must calculate the block number and offset from sector parameter in the request. The EBT can be loaded in the memory to improve the performance of OIIB. In the initial status, all the effective data is stored in the origin storage, so the table has no meaningful information. When users want to write a file in the origin storage, the EBT will be modified to record the change.

We realize OIIB using copy-on-write on file, i.e., when the system receives the write request to the block first time, the block is copied to the temporary storage. And from then on, the subsequent requests to the block are redirected to the new copy in the temporary storage. In OIIB, copy-on-write on directory is handled in the same way as the operation on files, which is actually copying the referred disk blocks, so we can support copy-on-write on the entire file system. The algorithm to locate the effective data is as follows.

Initialization: struct ebt {int in_temp, Address temp_loc}; to store the real location of sectors in the list ebtlist[n], n is the number of sectors in the origin storage, set all in_temp in ebtlist[n] to 0;

```
/*return location in the temporary storage of a modified sector*/
Address OIIB_Locating (
Operation file_oper,                               /*"r" or "w"*/
int origin_sec,
struct ebt ebtlist[n]){
    struct ebt_i=ebtlist[origin_sec];
    if(file_oper=="r"){                               /*to read*/
        if(ebt_i.in_temp)return ebt_i.temp_loc;
        else return NULL;                             /*not in temporary storage*/
    }elseif(file_oper=="w"){                           /*to write*/
        if(ebt_i.in_temp)return ebt_i.temp_loc;
```

```

else{                                     /*copy-on-write*/
    allocate a block and set ebt_i.temp_loc;
    copy the sector from origin_loc in origin
    storage to temp_loc in temporary storage;
    ebt_i.in_temp=1;
    return ebt_i.temp_loc;}
}

```

4 Application Instance and Evaluation

OIDB can be applied in a lot of contexts to support file isolation and protection tasks without restricting functionality. To investigate the idea described above, we use OIDB in TransCom system[8].

4.1 Application Instance

TransCom system adopts the conventional client and server architecture. Each transparent client machine is a bare-hardware without any local hard disks. The transparent server can be a regular PC or a high-end dedicated machine that stores all the needed software and data required for completing tasks at clients. The server and clients are connected in a local area network. To use a TransCom client, users just need to power it on, boot it remotely and load the selected OS and software from a server. After this process, users can use the client in the same way as a regular PC with local storage devices. In this sense, the client is operating like a service transceiver, and the server is like a service repository and provider by delivering software and data to clients in a way similar to an audio or video streaming.

One of the core ideas of TransCom system is the virtualization of devices. TransCom maintains a "golden image" of a clean system that contains the desired OS and a common set of applications. This "golden image" is thus immutable and can be shared by all clients. However, each client needs their own private storage with complete functionality to support all kinds of applications working. So, we implement OIDB in TransCom system to provide a COW virtual disk for each client. Each virtual disk is mapped to an image file in the server repositories. The image file holds the modified disk contents.

Figure 2 illustrates the implementation structure of OIDB in TransCom system. OIDB implementation adopts client and server architecture according to the whole system architecture. Thus, the implementation structure of OIDB in TransCom system is different from the implementation structure of OIDB in a stand-alone computer described above. OIDB in TransCom system works as follows. The file system driver in client accepts a request to write data to a certain location within a particular file from an application. It translates the request into a request to write a certain number of bytes to the disk at a particular "logical" location. It then passes this request to a redirector disk driver. The redirector disk driver, in turn, translates the request into a physical location

(cylinder/track/sector) on the disk and passes the request to the network protocol driver. And the network protocol driver transports it to the server. The virtual disk driver service in server receives the request and does the physical address translation by executing the locating algorithm. The result turns out a certain location within the image file in the server. Then, the service accesses the image file through the file system driver and disk driver in the server.

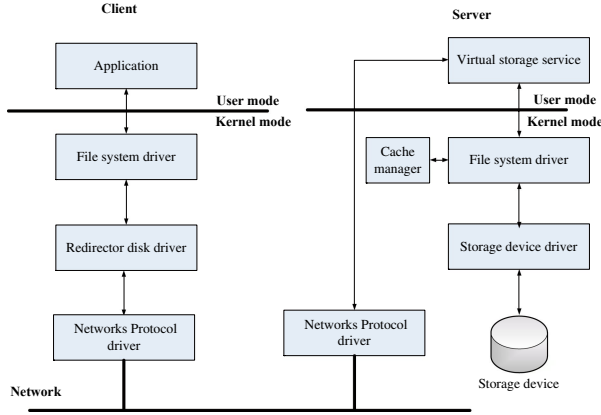


Fig. 2. The implementation structure of OIDB in TransCom system

4.2 Evaluation of Functionality

We have implemented a prototype of TransCom system that supports clients running Windows 2000 Professional. Our Windows based system has been deployed across 30 clients in a university e-learning classroom for daily usage for 18 months. In our deployment, TransCom clients are Intel Celeron 1GHz machines, each with 128 MB DDR 133 RAM and 100 Mps onboard network card. The server is an Intel Pentium IV 2.8GHz PC with 1G RAM, a 1Gbps network card, and an 80 GB 7200rpm soft RAID0 hard disk. The clients and the server are connected by an Ethernet switch with 48 100Mbps interfaces (used for clients) and two 1 Gbps interfaces (used for the server). The server OS is Windows 2003 Server (SP1) edition. TransCom clients use Windows 2000 professional (SP2). Because space is a consideration, the virtual storage service allocates space of temporary storage on demand. It leads to a considerable storage saving. The redirect disk driver is platform dependent and its implementations are in C++.

During our initial deployment, TransCom with OIDB has been running stably most of time except several hardware fault reports. The client works in the same way as the regular PC with local storage. TransCom client with OIDB has no any restriction on file operation. On the contrary, when we use IFS as a component to implement the TransCom system, IFS restrict the file access functionality severely. For example, when user requests to modify the register files, a necessary operation to install software or maintain system, some errors

always happen. The reason of the errors is because Windows2000 does not allow copying the register files in entirety. As we assumed, the TransCom system with OIBD works well in the same situation.

4.3 Performance Evaluation

In our testbed experiments, we use the same hardware configurations as our real deployment. To compare the file-modifying performance between OIBD and one-way isolation method of SEE, We also implement a pilot system using IFS, which is different from OIBD distinguishably in the different layers of file copying operation. We first vary the origin file from 512K to 4096K, increased by 512K. Then, we vary the size of content to be modified between 1K and 64K. The writing latency refers to the time elapsed from writing operation started to the task completed. In both two implementation, the latency increases with the size of the pristine file and the size of the content to be modified. But for all categories of performance, we observe that OIBD outperforms IFS greatly from figure 3. With OIBD, the latency is very short and on the order of tens of microseconds. And with IFS, the latency is relatively too long and on the order of hundreds of microseconds. When the size of the origin file is 4096K and the content to be modified is 1K, the superiority of OIBD over IFS is most clear, where the writing latency in IFS is about 130 times of the writing latency in OIBD.

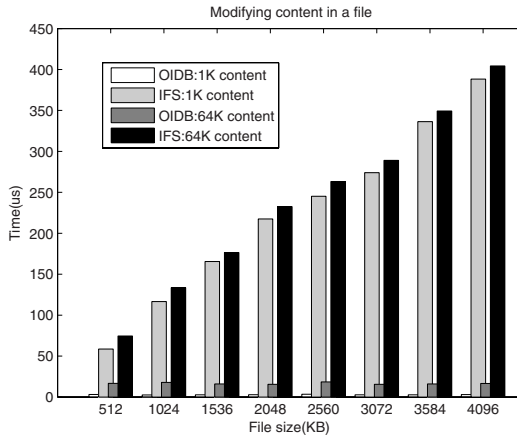


Fig. 3. Performance results of modifying content in a file

5 Conclusion

In this paper, we propose a file-access method, called as OIBD, which always keeps original files on pristine status and enables users to access files without restricting the functionality. This is accomplished via one-way isolation: users can read files in origin storage, but they are not permitted to write files in the

origin storage. Their writing operations are redirected to the temporary storage. A key property of our approach is that files are accessed at the granularity of a disk-block, which makes the overheads of disk-block copying be alleviated mostly. Our OI DB supports a wide range of tasks, including: sharing pristine data, supporting file system versioning, testing unauthentic software. We use OI DB in TransCom System for an instance. The evaluation of functionality and performance has verified its desirable features. OI DB does not restrict file accessing functionality and introduces low disk-block copying overheads in the process of accessing a file, especially in modifying a file.

References

1. Santry, D.J., Feeley, M.J., Hutchinson, N.C., Veitch, A.C.: Elephant: The File System that Never Forgets. In: Proceedings of the The Seventh Workshop on Hot Topics in Operating Systems, vol. 2 (1999)
2. Soules, C.A., Goodson, G.R., Strunk, J.D., Ganger, G.R.: Metadata Efficiency in Versioning File Systems. In: Proceedings of the 2nd USENIX Conference on File and Storage Technologies, Conference On File And Storage Technologies, USENIX Association, Berkeley, CA, pp. 43–58 (2003)
3. Sun Microsystems: Translucent file system. SunOS Reference Manual (1990)
4. Muniswamy-Reddy, K.-K., Wright, C.P., Himmer, A.P., Zadok, E.: A versatile and user-oriented versioning file system. In: Proceedings of USENIX Conference on File and Storage Technologies (2004)
5. Roome, W.D.: 3DFS: A Time-Oriented File Server. In: Proceedings of the Winter 1992 USENIX Conference, San Francisco, California, pp. 405–418 (1991)
6. Sun, W., Liang, Z., Sekar, R., Venkatakrishnan, V.N.: One-way Isolation: An Effective Approach for Realizing Safe Execution Environments. In: Proceedings of the Network and Distributed System Security Symposium (2005)
7. Pendry, J.-S., McKusick, M.K.: Union Mounts in 4.4 BSD-Lite. In: Proceedings of 1995 USENIX Technical Conference on UNIX and Advanced Computing Systems (1995)
8. Zhang, Y., Zhou, Y.: Transparent Computing: A New Paradigm for Pervasive Computing. In: Moreau, L., Foster, I. (eds.) IPAW 2006. LNCS, vol. 4145, pp. 1–11. Springer, Heidelberg (2006)
9. <http://www.nist.gov/dads/HTML/tree.html>

Novel Remote User Authentication Scheme Using Bilinear Pairings

Chen Yang, Wenping Ma, and Xinmei Wang

Ministry of Education Key Lab. of Computer Networks and Information Security,
Xidian University, Xi'an 710071, China
yfych@eyou.com, wp_ma@hotmail.com, xmwang@xidian.edu.cn

Abstract. This paper presents a novel password-based remote user authentication scheme using bilinear pairings by introducing the concept of private key proxy quantity. In the proposed scheme, an authorized user is allowed to login to the remote system if and only if the login request is verified. And by assigning each user his corresponding private key proxy quantity, the collusion-resistance security of the proposed scheme is enhanced. The scheme provides a flexible password change strategy and eliminates the need of the password table. In addition, the proposed scheme can efficiently resist message replaying attack, forgery attack, Masquerade attack, guessing and stolen verifier attack and collusion attack.

1 Introduction

Remote user authentication is a mechanism that allows the authenticated user to login to the remote system to access the services offered over insecure communication network, and with the well development of network technologies and computer systems being in wide spread use to store resources and provide network services, it has got extensively studied in the last decades. Many related authentication schemes were proposed [1-8] since Lamport [1] introduced the first hash-based password authentication scheme. However, most of the proposed schemes are either hash-based [1-3], which suffer from high hash overhead and password resetting problems, or public-key based [4-8], which require high computation cost for implementation. Additionally, though the hash-based schemes are suitable for implementation in hand-held devices such as smart card, they easily suffer from password guessing, stolen-verifier, insider and denial-of-service attacks. Recently, since Boneh and Franklin [9] presented the first practical pairing based cryptographic scheme in 2001, bilinear pairings also have found their important applications in the construction of remote user authentication scheme for its less key sizes and bandwidth demand under relative security level compared with the integer factorization based systems or the discrete logarithm based systems. Since Das et al [10] proposed the remote user authentication scheme using bilinear pairings, which was broken and improved by Thulasi et al [12], several pairing-based remote user authentication schemes have been proposed [11,13]. These schemes utilize the merit of elliptic curve cryptography and the smart card technology, which make the scheme more practical and efficient. In this paper, we construct

a new pairing-based remote user authentication scheme using smart card which can resist the multi-users logged in attack with the same login-ID. In the proposed scheme, the bilinear pairing operation is executed only at the server side, and it is especially suitable to the scenario of smartcard-based remote user authentication scheme. The scheme is also resilient to the insider attack, replaying attack, forgery attacks, collusion attack as well as masquerade attack.

The paper is organized as follows. In the next section, some preliminaries of bilinear pairings and ground of security are given. In section 3, we describe our scheme in detail. Security analysis is present in section 4. We conclude the paper in the last section.

2 Preliminary

We firstly review the concept of bilinear pairings in brief.

2.1 Bilinear Mapping

Let G_1 and G_2 be two cyclic groups with the same large prime order q where G_1 is an additive group, G_2 is a multiplicative group. We assume that the discrete logarithm problems in both G_1 and G_2 are hard. A cryptographic bilinear mapping e is defined as $e: G_1 \times G_1 \rightarrow G_2$ with the following properties:

1. Bilinearity: $\forall P, Q \in G_1$ and $\forall a, b \in Z_q$, we have $e(aP, bQ) = e(P, Q)^{ab}$
2. Non-degeneracy: for any point $P \in G_1$, $e(P, Q) = 1$ for all $Q \in G_1$ iff $P = O$.
3. Computability: there exists an efficient algorithm to compute $e(P, Q)$ for any $P, Q \in G_1$.

Admissible bilinear mapping can be constructed from Weil or Tate pairings associated with super-singular elliptic curves or Abelian varieties.

2.2 Ground of Security

In this section, we review some well-known problems to be used in the security analysis of our scheme.

Elliptic Curve Discrete Logarithm (ECDLP) problem in $(G_1, +)$: Given $\{P, aP\}$ for any $a \in Z_q^*$, to compute a .

Computational Diffie-Hellman (CDH) problem in $(G_1, +)$: Given $\{P, aP, bP\}$ in G_1 for any $a, b \in Z_q^*$, to compute abP .

Inverse Computational Diffie-Hellman (ICDH) problem in $(G_1, +)$: Given $\{P, aP, abP\}$ in G_1 for any $a, b \in Z_q^*$, to compute bP .

It is believed that ICDH problem is equivalent to CDH problem, and both of them are NP hard under ECDL problems.

3 Authentication Scheme

In this section, we present our remote user authentication scheme in which only *authentication server* (AS) can generate a valid check digit for each user. Our scheme includes five procedures: *Initialization phase*, *Registration phase*, *Login phase*, *Authentication phase* and *Password Change phase*. These procedures are described as follows.

3.1 Initialization Phase

In this procedure, AS generates the system parameters in the following:

- q : l -bit prime;
- G_1 : Additive group of order q ;
- G_2 : Multiplicative group of order q ;
- e : An admissible bilinear map: $G_1 \times G_1 \rightarrow G_2$;
- P : A generator of G_1 ;
- $a, z \in \mathbb{Z}_q^*$: Secret keys of AS;
- H : Collision-resistant hash function $H : \{0,1\}^* \rightarrow G_1$;
- H_1 : Collision-resistant hash function $H_1 : \{0,1\}^* \rightarrow G_1$;

The system public parameters are $(l, q, G_1, G_2, e, P, H, H_1)$, and private keys of AS are the pair (a, z) .

3.2 Registration Phase

This phase is executed by the following steps when a new user u_i wants to join the system. Assume that this procedure is executed over a secure channel. User u_i submits his unique identity ID_i and login password PW_i to AS for registration. AS computes $SK_i = H(ID_i \parallel a \parallel z) \oplus H(PW_i)$, $Q_{ID_i} = H(ID_i \parallel a \parallel z)$, and a representation $k_i = (x_{i1}, x_{i2})$ of zQ_{ID_i} with respect to the base (Q_{ID_i}, aQ_{ID_i}) . The possible set of keys is

$$k = \{(x_{i1}, x_{i2}) \mid x_{i1} + ax_{i2} = z \bmod q\}.$$

However, AS doesn't directly give these keys to user u_i . He further computes the proxy quantity $\Pi(k_i)$ of $k_i = (x_{i1}, x_{i2})$ as follows:

$$\Pi(k_i) = (x_{i1}, x_{i2}Q_{ID_i}).$$

Each user is just given a corresponding proxy quantity $\Pi(k_i)$ as his private key pair.

A smart card containing $(ID_i, Q_{ID_i}, \Pi(k_i))$ and public parameters (q, P, H, H_1) is sent to user u_i over a secure channel.

3.3 Login Phase

The user u_i attaches his smart card to his input device, and keys in his identity number ID_i and login password PW_i . The smart card performs the following operations:

1. Generate a random number $r \in Z_q^*$.
2. Compute $Q_{ID_i} = SK_i \oplus H(PW_i)$.
3. Compute $c_1 = rQ_{ID_i}$ and $c_2 = rx_{i2}Q_{ID_i}$.
4. Compute $t = H_1(ID_i \parallel T_c \parallel c_1 \parallel c_2)$. Where T_c is the current time data of the input device and the symbol $\parallel$ denotes the string concatenation.
5. Compute $c_3 = rx_{i1}t$.

Send messages $M_i = (ID_i \parallel T_c \parallel c_1 \parallel c_2 \parallel c_3)$ to the remote authentication system.

3.4 Authentication Phase

When AS receives the login request M_i at time T_c' from user u_i , he checks the validity of the login request M_i as follows:

1. Check the validity of ID_i . If the format of ID_i is incorrect, then AS rejects the login request M_i .
2. Check whether $T_c' - T_c \leq \Delta T$, where ΔT denotes the expected network transmission delay; if not, then AS rejects the login request M_i .
3. Compute $Q_{ID_i} = H(ID_i \parallel a \parallel z)$ and $t = H_1(ID_i \parallel T_c \parallel c_1 \parallel c_2)$, and check whether the equation $e(Q_{ID_i}, c_3) \cdot e(ac_2, t) = e(t, c_1)^z$ holds, then AS accepts the login request M_i ; Otherwise, AS rejects it.

3.5 Password Change Phase

This phase is invoked when the user u_i wants to change his password after a period of time. This phase does not require any interaction with the servers and works in the following way:

1. User u_i inserts his smart card into the terminal server, keys in his password PW_i and invokes the password change algorithm.
2. User u_i chooses his new password PW_i^* .
3. the smart card computes

$$SK_i^* = SK_i \oplus H(PW_i) \oplus H(PW_i^*) .$$

4. The password PW_i is changed into the new one PW_i^* . The secret key SK_i will be replaced with SK_i^* , and stops the algorithm.

4 Security Analysis

In this section, we present the security analysis of the proposed scheme and show that it is secure against forgery attacks, replay attacks, masquerade attacks and user collusion attacks.

Theorem 1: The secret keys $\Pi(k_i) = (x_{i1}, x_{i2}Q_{ID_i})$ of user u_i cannot be retrieved from the intercepted access request $M_i = (ID_i \parallel T_c \parallel c_1 \parallel c_2 \parallel c_3)$ and public parameters.

Proof. An adversary can get (c_1, c_2, t, c_3) from intercepted public information, where $c_1 = rQ_{ID_i}$, $c_2 = rx_{i2}Q_{ID_i}$, $t = H(ID_i \parallel T_c \parallel c_1 \parallel c_2)$ and $c_3 = rx_{i1}t$. However, as r is unknown, it is infeasible to compute $x_{i2}P$ from c_1 and c_2 under the ICDH assumption and x_{i1} from $c_3 = rx_{i1}t$. So there will be no probabilistic polynomial algorithms exist that can retrieve the secret keys of user u_i from the intercepted access request.

Corollary 1: The scheme is secure against forgery attacks.

Theorem 2: The proposed scheme is secure against replay attack.

Proof. Assume that a passive adversary replays the intercepted login request $M_i = (ID_i \parallel T_c \parallel c_1 \parallel c_2 \parallel c_3)$ to the server and the AS receives the access request at time T_c' . The authentication procedure 1 holds, but the login time interval $T_c' - T_c$ will be larger than the expected network delay ΔT , and the access request will be rejected. On the other hand, if the adversary substitutes the time stamp T_c with current time T_c'' , and procedure 2 holds, but the condition $e(Q_{ID_i}, c_3) \cdot e(ac_2, t) = e(t, c_1)^z$ will not hold any more. Thus the proposed scheme is secure against replay attack.

Theorem 3: Given the public parameters and any coalition of proxy quantities with k users: $\Pi(k_i) = (x_{i1}, x_{i2}Q_{ID_i})$, $i = 1, 2, \dots, k$, it is computationally infeasible for the colluders to construct a new key pair $k_i' = (x_{i1}', x_{i2}')$ and the corresponding proxy quantity $\Pi(k_i') = (x_{i1}', x_{i2}'Q_{ID_i})$ of another people under the ECDL problem.

Proof. As a valid proxy quantity should have the form $\Pi(k_i) = (x_{i1}, x_{i2}Q_{ID_i})$ such that $x_{i1} + ax_{i2} = z \bmod q$, where x_{i2}, a and z are unknown to all users, the way for adversaries to compute $k_i' = (x_{i1}', x_{i2}')$ or $\Pi(k_i') = (x_{i1}', x_{i2}'Q_{ID_i})$ is solving the equations

$$\begin{aligned} x_{11} + ax_{12} &= z \bmod q \\ \vdots \quad \quad \quad \vdots \quad \quad \quad \vdots \\ x_{k1} + ax_{k2} &= z \bmod q \\ x_{i1}' + ax_{i2}' &= z \bmod q \end{aligned}$$

However, note that k users cannot get any information about x_{i2} from $x_{i2}Q_{ID_i}$ under ECDL problem and don't learn the secret keys (a, z) , there exists no probabilistic polynomial algorithm that can solve the $k'_i = (x_{i1}', x_{i2}')$ or $\Pi(k'_i) = (x_{i1}', x_{i2}'Q_{ID_i})$ from above equations in polynomial time.

Corollary 2: No coalition of users can succeed in executing insider attack.

Definition 1: Masquerade attack is defined as: No adversary intercepting valid $M_i = (ID_i \parallel T_c \parallel c_1 \parallel c_2 \parallel c_3)$ of user u_i can compute a new different valid login request $M_j = (ID_i \parallel T_c' \parallel c_1' \parallel c_2' \parallel c_3')$ to impersonate user u_i .

Theorem 4: It is computationally infeasible for passive adversaries or authorized users to execute Masquerade attacks.

Proof. ID_i is used in computation of t to check if $e(Q_{ID_i}, c_3) \cdot e(ac_2, t) = e(t, c_1)^z$ holds in our scheme. Directly from theorem 1 and theorem 3, it can conclude that no adversaries can compute a valid login request $M_j = (ID_i \parallel T_c' \parallel c_1' \parallel c_2' \parallel c_3')$ to impersonate user u_i , i.e. masquerade attack in our scheme will not succeed.

In addition, there is no need for the remote system to maintain a password table through which a dishonest party can steal one user's password to verify the login request. Thus, the proposed scheme is secure against the insider attack and stolen password attack.

5 Conclusion

We present a novel efficient remote user authentication using bilinear pairings. In our scheme, each user is assigned a smart card, and the private keys in the smart cards for each user are not the original representation over some bases but its corresponding proxy quantity, which makes the proposed scheme more reliable and secure against forgery attacks and collusion attacks. The proposed scheme doesn't need for AS to maintain any password table to verify the validity of the user login, and thus greatly reduces the storage cost of the system and strengthens the protocol against stolen verifier attacks and insider attacks. Additionally, the scheme can efficiently withstand message replaying attacks and Masquerade attacks.

Acknowledgment

The authors would like to thank anonymous referees for useful comments. This research is partially supported by "Program for New Century Excellent Talents in University" and the Natural Science Foundation of China under Grants No.60373104 and No.90604009.

References

- [1] Lamport, L.: Password Authentication within Secure Communication. *Communications of ACM* 24(11), 770–772 (1981)
- [2] Lee, C.C., Li, L.H., Hwang, M.S.: A Remote User Authentication Scheme Using Hash Functions. *ACM Operating Systems Review* 36(4), 23–29 (2002)
- [3] Ku, W.C.: A Hash-based Strong-password Authentication Scheme without Using Smart Cards. *ACM Operation Systems Review* 38(1), 29–34 (2004)
- [4] Hwang, M.S., Li, L.H.: A New Remote User Authentication Scheme Using Smart Card. *IEEE Trans. on Consumer Electronics* 46(1), 28–30 (2000)
- [5] Shen, J.J., Lin, C.W., Hwang, M.S.: A Modified Remote User Authentication Scheme Using Smart Cards. *IEEE Trans. on Consumer Electronics* 49(2), 414–416 (2003)
- [6] Amit, K., Sunder, L.: A Remote User Authentication Scheme Using Smart Cards with Forward Secrecy. *IEEE Trans. on Consumer Electronics* 49(4), 1246–1248 (2003)
- [7] Wu, S.T., Chieu, B.C.: A User Friendly Remote Authentication Scheme with Smart Cards. *Computers & Security* 22(6), 547–550 (2003)
- [8] Yoon, E.J., Ryu, E.K., Yoo, K.Y.: Efficient Remote User Authentication Scheme based on Generalized ElGamal Signature Scheme. *IEEE Trans. on Consumer Electronics* 50(2), 568–570 (2004)
- [9] Boneh, D., Franklin, M.: Identity-based Encryption from the Weil Pairing, 2001. In: Kilian, J. (ed.) *CRYPTO 2001*. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
- [10] Das, M.L., Ashutosh, S., Gulati, V.P., Phatak, D.B.: A Novel Remote User Authentication Scheme Using Bilinear Pairings. *Computer & Security* 25, 184–189 (2006)
- [11] Chou, J.S., Chen, Y., Lin, J.Y.: Improvement of Manik et al.'s Remote User Authentication Scheme. <http://eprint.iacr.org/2005/450.pdf>
- [12] Thulasi, G., Das, M.L., Ashutosh, S.: Cryptoanalysis of Recently Proposed Remote User Authentication Scheme. <http://eprint.iacr.org/2006/028.pdf>
- [13] Fang, G., Huang, G.: Improved of Recently Proposed Remote User Authentication Schemes. <http://eprint.iacr.org/2006/200.pdf>

On the Homonymous Role in Role-Based Discretionary Access Control

Kai Ouyang¹, Xiaowen Chu², Yixin Jiang³,
Hsiao-Hwa Chen⁴, and Jiangchuan Liu⁵

¹ School of Computer Science, Wuhan Univ. of Sci. & Tech., China
kai.dolphin@gmail.com

² Department of Computer Science, Hong Kong Baptist Univ., Hong Kong
chxw@comp.hkbu.edu.hk

³ Department of Computer, Tsinghua University, Beijing, China
yxjiang@csnet1.cs.tsinghua.edu.cn

⁴ Institute of Communication Engineering, National Sun Yat-Sen Univ., Taiwan
hshwchen@ieee.org

⁵ School of Computing Science, Simon Fraser University, BC, Canada
jcliu@sfu.ca

Abstract. Secure model is a core aspect in trusted information system and a key research field of trusted computing. Based on the extensive research of the Role Based Access Control (RBAC) model and the security operating system standards, we put forward the concept of homonymous role, which extends the control categories of the role in RBAC, balances the control granularity and the storage space requirement, and carries the detailed access control into execution. Based on the homonymous role, we also facilitate the homonymous control domain capability in the Discretionary Access Control (DAC) system. Finally, we design and implement our homonymous control domain on FreeBSD to enhance the flexibility of the access control.

1 Introduction

Highly-secured operating system is the indispensable plat-form for the construction of the trusted computing. It has been an important research field for the Trusted Computing Base [1]. Secure models are the abstraction, unambiguous formulations and the basic theory for the research of the secure operating system, such as BLP, Biba and Role Based Access Control (RBAC). In recent years, the RBAC model, as a hot topic in the current secure model research, is widely applied in the operating system, database management system, and also network control. RBAC model was first proposed in 1992 [2], and then it has been further developed and improved [3]. Based on the proposal in 2001 [4], RBAC has been accepted as a formal NIST standard in 2004. However, the control granularity and flexibility of RBAC is not as good as the User Based Access Control (UBAC) because of the decoupling of the user set and the permission set. To address this, based on centralized information access control mode, a user and role based hybrid privilege mechanism in the

application-layer was proposed in [5]. Though it gave the integrated formal definition and control rules for the hybrid privilege mechanism, it did not provide the uniform semantics for the hybrid privilege mechanism. Hence it could not provide a clear guideline for the implementation of the hybrid privilege mechanism.

In this paper, we propose the conception of homonymous role based on the idea of the hybrid privilege mechanism and the NIST RBAC formulations. Every user has one homonymous role, which is created and destroyed by the model system at the time that user is created and destroyed. Using the homonymous role as the internal relationship, the model could manage the homonymous role by the user set's operating semantics so as to simplify the design and implementation of the hybrid access control mechanism.

On the other hand, Discretionary Access Control (DAC) plays a key role in most secure operating system's control mechanism, which establishes the access control based on the identifiers of subjects and objects. DAC can enable the object's creator, i.e., the owner, to specify for each object what types of accesses can be authorized and by whom (which user). There is no universally accepted definition of DAC. Models such as HRU, SPM and TAM could be used as general DAC models [6]. Normally, the subject identifiers (such as user ID), the object identifies (such as file ID), and the access permission (such as read/write/execute) constitute DAC's access control matrices; and the system can execute the access control through Access Control Lists (ACLs). For the file object as instance, in the current secure operating system's implementation (such as FreeBSD and SE-Linux), every file has its Extended Attributes (EA) to store the ACLs information. There exists a tradeoff between the control granularity and the requirement of the storage space, however. At the same time, when the administrator manages the files' access controls of one organization, he must understand both the control structure of the organization and the DAC system's access matrix.

There is considerable research on the access control model based on DAC and RBAC. Sandhu et al. [6] used the RBAC96 model to simulate a variety of DAC policies and showed how DAC can be accommodated within a RBAC-oriented system. Kejun Zhang et al. [7] constructed a variety of DAC policies in the RBAC model that can work with a predefined role hierarchy, which is much more practical in real role-oriented information systems.

In this paper, we consider that the capability of the administrator should be divided into (1) system management; and (2) organization management. The organizational administrator can then manage the access control only according to the logical access relationship of the organization. Hence, based on the homonymous roles (groups), we put forward the homonymous control domain mechanism to extend the DAC's control category. We implement the multi-models (such as DAC and RBAC) in one time, and provide one logic layer independent of the DAC system to make it more convenient for the organizational administrator in access control management. The key idea of the homonymous control domain is that the DAC system can create one logic control domain for one subject, in which the subject can manage its objects using the independent access control mechanism. The DAC system can establish the map between the DAC access matrix and the logical access tables in the homonymous control domain, which is transparent to the administrator.

The rest of this paper is organized as follows. In section II, we define the homonymous role and give the detailed functional specification. Section III details the definition of the homonymous control domain semantics. In section IV we show how to design the homonymous control domain for the files' ACL mechanism in FreeBSD. Section V presents our conclusions.

2 Formulations of the Homonymous Role

Based on the NIST RBAC standard, we formulate the semantics and functional specification of the homonymous role in this section.

2.1 Semantics

Definition 1 (The Homonymous Role): When any user is created, the model system will automatically produce a role that has the same name as the user and this role is called the homonymous role and marked as γ . The set of the homonymous role is marked as $\mathcal{R}$ and it is a subset of the role set ROLES. The properties of the homonymous role are described as follows:

- 1) The capability of assignment and authorization — the homonymous role only belongs to one user, i.e. the homonymous user. Suppose function $Name(x)$ is used to get the name of the object x , the following two formalizations should come into existence. In core RBAC: $assigned\_users(\gamma) = \{ u \in USER \mid Name(u) = Name(\gamma) \} \& assigned\_users(\gamma) \models 1$, while in hierarchical RBAC: $authorized\_users(\gamma) = \{ u \in USER \mid Name(u) = Name(\gamma) \} \& authorized\_users(\gamma) \equiv 1$.
- 2) Un-inheritability — in hierarchical RBAC, the homonymous role does not have the capability to inherit from other roles or be inherited by other roles, but this characteristic does not destruct the partial order of the other roles. The formalization is $\forall r \in ROLES, \forall \gamma \in \mathcal{R}, r \neq \gamma \rightarrow \{ \rho \mid \gamma \leq r \} \setminus \{ \rho \mid r \leq \gamma \} \equiv \emptyset$.
- 3) Transparent management mechanism — all operations of the homonymous role are delegated by the homonymous user and implemented within the model system, and it is transparent to the administrator. Hence, the operations of the role-user assignment or the role-permission assignment do not contain information of the homonymous role. Additionally, the management operations of the set of ROLES do not include the homonymous role, either.
- 4) The assignment/authorization of the homonymous role — the definition follows the NIST RBAC standard. Because the assignment/authorization operations could not be directly executed on the homonymous role, the model system delegates the homonymous user to provide an interface for the administrator (from the administrator's viewpoint, the operations is similar to UBAC). At the same time, before the execution of the homonymous role's assignment/ authorization, the model system must satisfy the apriori condition, under which the assignment/authorization of the current role sets can not violate the static separation of duty relations. The formalization is $\forall p \in PRMS, rs = Roles(p), \forall 2 \leq n \leq |rs|, (rs, n) \notin SSD \rightarrow Exec(DelegateGrantPermission(object, operation, Name(\gamma)))$, where p is any permission, function $Roles(p: PRMS)$ is used to get a role set which

has the assignment/authorization map to the given p . and $|rs|$ is the element number in the role set rs . The model system can assign/authorize p to the homonymous role γ if and only if the given permission satisfies $(rs, n) \notin SSD$. The function *DelegateGrantPermission* used to implement the assignment/authorization operations of the homonymous role will be discussed later.

- 5) The permission priority rule — during a user session, if there is a policy conflict (such as the permission set of the homonymous role is conflicted with the permission set of the role that the user belongs to) when the user's role set is activated, the model system considers the permission set of the homonymous role as the highest permission priority.
- 6) The lifecycle — user's homonymous role is created when the user is created and destroyed when the user is destroyed.

2.2 Functional Specifications

Due to the introduction of the homonymous role, we must modify and redefine the related administrative commands, system functions and review functions in the NIST RBAC standard.

• Administrative commands

When a user is created/deleted, his homonymous role is created/deleted also. Therefore, we must redefine the user creation/deletion commands.

User Creation: when a user is created, the model system will create his homonymous role γ by the function *HomonymousRoleC*(*user: NAME*), add the role γ into the homonymous role set $\mathcal{R}$, set the solely designated user of the role set $\mathcal{R}$, and initialize the role γ 's permission set as null. Compared with the role creation function (*AddRole*(*role: NAME*)) in the NIST RBAC standard, the homonymous role creation process includes the homonymous role assignment and it is determined by the first characteristic of the homonymous role. The user creation is formalized as follow:

$$\begin{aligned}
 &AddUser(user: NAME) \triangleleft \\
 &\quad user \notin USER; USER' = USER \quad \{user\} \\
 &\quad \gamma = HomonymousRoleC(user); \gamma \notin \mathcal{R}; \mathcal{R}' = \mathcal{R} \quad \{\gamma\} \\
 &\quad assigned\_users(\gamma) = \{user\} \\
 &\quad assigned\_permissions' = assigned\_permissions \quad \{\gamma \mapsto \emptyset\} \\
 &\quad user\_sessions' = user\_sessions \quad \{user \mapsto \emptyset\} \triangleright
 \end{aligned}$$

User Deletion: when a user is deleted, the model system can get its homonymous role by the function *GetHomonymousRole*(*user: NAME*). After the model system completes the original operation semantics of the user deletion, the model system needs to truncate the permission set of the homonymous role γ and delete γ from the homonymous role set $\mathcal{R}$. Compared with the role deletion function (*DeleteRole*(*role: NAME*)) in the NIST RBAC standard, the homonymous role deletion does not include the related session deletion because the homonymous role can exist in the homonymous user session. It does not include the role-user assignment deletion because the homonymous role only belongs to the homonymous user. The user deletion is formalized as follow:

$$\begin{aligned}
& \text{DeleteUser}(\text{user}: \text{NAME}) \triangleleft \\
& \quad \text{user} \in \text{USERS}; \gamma = \text{GetHomonymousRole}(\text{user}) \\
& \quad [\forall s \in \text{SESSIONS} \quad s \in \text{user\_sessions}(\text{user}) \Rightarrow \text{DeleteSession}(s)] \\
& \quad \text{UA}' = \text{UA} \setminus \{r: \text{ROLES} \quad \text{user} \mapsto r\} \\
& \quad \text{assigned\_users}' = \{r: \text{ROLES} \quad r \mapsto (\text{assigned\_users}(r) \setminus \{\text{user}\})\} \\
& \quad \text{PA}' = \text{PA} \setminus \{op: \text{OPS}, obj: \text{OBS} \quad (op, obj) \mapsto \gamma\} \\
& \quad \text{assigned\_permissions}' = \text{assigned\_permissions} \setminus \{\gamma \mapsto \text{assigned\_permissions}(\gamma)\} \\
& \quad \mathcal{R}' = \mathcal{R} \setminus \{\gamma\}; \text{USERS}' = \text{USERS} \setminus \{\text{user}\} \triangleright
\end{aligned}$$

The semantics of the role creation function ($\text{AddRole}(\text{role}: \text{NAME})$) and the role deletion function ($\text{DeleteRole}(\text{role}: \text{NAME})$) is the same as that in the NIST RBAC standard. This is because the homonymous role is transparent to the administrator who cannot create/delete the homonymous role through these two functions. Similarly, the definitions of the user assignment function ($\text{AssignUser}(\text{user}, \text{role}: \text{NAME})$), the user de-assignment function ($\text{DeassignUser}(\text{user}, \text{role}: \text{NAME})$), the permission assignment function ($\text{GrantPermission}(\text{object}, \text{operation}, \text{role}: \text{NAME})$) and the permission revoke function ($\text{RevokePermission}(\text{operation}, \text{object}, \text{role}: \text{NAME})$) are the same as those in the NIST RBAC standard. But the homonymous role also needs to implement the permission assignment and revoke functions. According to **Definition 1**, these two functions are implemented by delegating the user. Hence, we add two functions to describe the permission operations of the homonymous role.

Delegate Permission Grant: The administrator can operate the homonymous role's permission assignment through the homonymous user's permission assignment that is defined as $\text{DelegateGrantPermission}(\text{object}, \text{operation}, \text{role}: \text{NAME})$. The parameters *object* and *operation* represent to the permission *p*, and the parameter *role* represents the homonymous role. According to **Definition 1**, we firstly make the judgment by checking the apriori condition. After the condition is satisfied, we can set *p* for the homonymous role and update the related information of the homonymous role and the permission set. The delegate permission grant is formalized as follow:

$$\begin{aligned}
& \text{DelegateGrantPermission}(\text{object}, \text{operation}, \text{role}: \text{NAME}) \triangleleft \\
& \quad p = (\text{operation}, \text{object}) \in \text{PRMS}; \text{user} \in \text{USERS}; \gamma = \text{GetHomonymousRole}(\text{user}) \\
& \quad rs = \text{Roles}(p), 2 \leq n \leq |rs|, \forall (rs, n) \notin \text{SSD} \\
& \quad \text{PA}' = \text{PA} \quad \{p \mapsto \gamma\} \\
& \quad \text{assigned\_permissions}' = \text{assigned\_permissions} \setminus \{\gamma \mapsto \text{assigned\_permissions}(\gamma)\} \\
& \quad \{\gamma \mapsto (\text{assigned\_permissions}(\gamma) \setminus \{(\text{operation}, \text{object})\})\} \triangleright
\end{aligned}$$

Delegate Permission Revoke: the administrator can revoke the permission of the homonymous role through the homonymous user's permission revoke. The delegate permission revoke is formalized as follow:

$$\begin{aligned}
& \text{DelegateRevokePermission}(\text{operation}, \text{object}, \text{user}: \text{NAME}) \triangleleft \\
& \quad (\text{operation}, \text{object}) \in \text{PRMS}; \text{user} \in \text{USERS}; \\
& \quad \gamma = \text{Name}(\text{user}) ((\text{operation}, \text{object}) \mapsto \gamma) \in \text{PA}; \text{PA}' = \text{PA} \setminus \{(\text{operation}, \text{object}) \mapsto \gamma\} \\
& \quad \text{assigned\_permissions}' = \text{assigned\_permissions} \setminus \\
& \quad \{\gamma \mapsto \text{assigned\_permissions}(\gamma)\} \quad \{\gamma \mapsto (\text{assigned\_permissions}(\gamma) \setminus \{(\text{operation}, \text{object})\})\} \triangleright
\end{aligned}$$

According to **Definition 1**, the homonymous role does not have the inheritance relationship, so any operation on the homonymous role does not affect the original hierarchical relationship. Our definitions of the inheritance addition function ($AddInheritance(r_{asc}, r_{desc}: NAME)$), the inheritance deletion function ($DeleteInheritance(r_{asc}, r_{desc}: NAME)$), the ascendant addition function ($AddAscendant(r_{asc}, r_{desc}: NAME)$) and the descendant add addition function ($AddDescendant(r_{asc}, r_{desc}: NAME)$) are stick to the NIST RBAC standard.

- System functions

When a user creates its session by the function ($CreateSession(user: NAME; ars: 2^{NAMES}; session: NAME)$), the model system will build a default activated role set as the starting point of the user session. The homonymous role will be activated by the model system and it is invisible to the activated role set. Therefore the activated homonymous role cannot perform the operations of the activated role function ($AddActiveRole(user, session, role: NAME)$) and the activated drop function ($DropActiveRole(user, session, role: NAME)$). When the user session is terminated, the activated homonymous role will be repealed in the function ($DeleteSession(user, session: NAME)$). Hence, we will need to update the semantics of the two system functions — session creation and session deletion.

Session Creation: because the homonymous role is transparent, the administrator cannot provide the input of the homonymous role for this function, that is to say $\gamma \notin ars$. Therefore, we need to add the function ($GetHomonymousRole(user: NAME)$) to get the user's (the session belonged to) homonymous role and add the homonymous role into the current session role set. The session creation is formalized as follow:

$$\begin{aligned}
 &CreateSession(user: NAME; ars: 2^{NAMES}; session: NAME) \triangleleft \\
 &\quad user \in USERS; ars \subseteq \{r: ROLES \mid (user \mapsto r) \in UA\} \\
 &(operation, object) \in PRMS; user \in USERS; \gamma = GetHomonymousRole(user) \\
 &\quad SESSIONS' = SESSIONS \quad \{session\} \\
 &\quad user\_sessions' = user\_sessions \setminus \{user \mapsto user\_sessions(user)\} \\
 &\quad \quad \{user \mapsto (user\_sessions(user) \quad \{session\})\} \\
 &\quad session\_roles' = session\_roles \quad \{session \mapsto ars\} \quad \{\gamma\} \triangleright
 \end{aligned}$$

Session Deletion: the model system will delete the activated homonymous role from the current session role set in the process of the user session deletion. The session deletion is formalized as follow:

$$\begin{aligned}
 &DeleteSession(user, session: NAME) \triangleleft \\
 &\quad user \in USERS; session \in SESSIONS \\
 &\quad session \in user\_sessions(user); \gamma = GetHomonymousRole(user) \\
 &\quad user\_sessions' = user\_sessions \setminus \{user \mapsto user\_sessions(user)\} \\
 &\quad \quad \{user \mapsto (user\_sessions(user) \setminus \{session\})\} \\
 &\quad session\_roles' = session\_roles \setminus \{\{session \mapsto session\_roles(session)\} \quad \{\gamma\}\} \\
 &\quad SESSIONS' = SESSIONS \setminus \{session\} \triangleright
 \end{aligned}$$

- Review functions

Because the homonymous role is invisible to the administrator, the model system must ensure the transparency of the homonymous role. We will now discuss the design essentials of the review functions based on Core RBAC.

Session Role: when the administrator wants to get the activated role set of the given session name, the model system should filter out the homonymous role whose homonymous user is the owner of the current session. The session role is formalized as follow:

$$\begin{aligned}
 & \text{SessionRoles}(\text{session}: \text{NAME}, \text{out result}: 2^{\text{ROLES}}) \triangleleft \\
 & \quad \text{session} \in \text{SESSIONS} \\
 & \quad \text{user} = \text{session\_users}(\text{session}); \gamma = \text{GetHomonymousRole}(\text{user}) \\
 & \quad \text{result} = \text{session\_roles}(\text{session}) \setminus \{ \gamma \} \triangleright
 \end{aligned}$$

Whether the model system should provide more the review functions (such as the role permission review) of the related homonymous role or not is up to the developers based on their policies.

3 Security Analysis

The file owner can accurately assign the access permissions to the file for other users in the DAC model system. That is to say, in the DAC mechanism, one user can discretionarily set any permission of his resources for any other user. We consider that the access control period of one user is his lifespan, i.e., from the time of creating the user to the time of deleting the user. But the current operating systems do not provide the criterion and definition of the user's access control period. Hence, we bring forward the conception of the homonymous control domain, in which a user can discretionarily adopt any access control model to implement his resource access control mechanism according to the logical application relationship in practice. The internal implementation of the homonymous control domain will fulfill a complete access control mapping between the logical relationship and the DAC mechanism. For example, in the case of files' ACL model, the homonymous control domain can provide the RBAC mechanism (roles can be referred as groups) for administrator and change the RBAC mechanism to file's ACL in the system kernel.

Definition 2 (The User Access Control Period): The user access control period is defined as the time that a user in the DAC mechanism have the capability to discretionarily control and manage the permission set of his own resources.

Definition 3 (The Homonymous Control Domain): The management that one user can discretionarily control his own resource is considered as the user's logic control domain and identified by the user's name, which is named as the homonymous control domain, marked $\Gamma(\text{user}: \text{NAME})$. The features of the homonymous control domain are described as follows:

- 1) The life span of the homonymous control domain is the user access control period.
- 2) In the homonymous control domain, the user can control and manage his own resources' permission set based on the logical relationship to form the independent access control mechanism.
- 3) The internal implementation of the homonymous control domain automatically supports the mapping transition between the logical relationship and the discretionary access control mechanism, which is transparent to the administrator.

4 The Application of the Homonymous Control Domain in ACL

Based on the FreeBSD 6.0 ACL mechanism [8], we have designed and implemented the homonymous control domain for the files' ACL model.

4.1 Basic FreeBSD ACL Model

One demonstration of FreeBSD ACL model is shown in Fig. 1, the user *UserA* manages the permission sets (suppose file permissions consist of read/write/execute, marked as *rwX*, respectively) of his files (*File1*, *File2*, and *File3*) for the users *User1* and *User2*. FreeBSD transits the control information (DAC control matrix) of one user's file object resources into the corresponding file's ACL so as to implement the discretionary access control. The ACL model in FreeBSD includes:

- 1) Implementing the ACL's physical storage capability based on files' extended attributes without changing the current file system storage format;
- 2) Providing the interfaces of the files' ACL control, verification and management for the higher-level system services in the kernel through adding or updating the semantics of the virtual file system;
- 3) Implementing the ACL's entity access mechanism through the vnode operations;
- 4) Providing the related access and management interface for the application layer program by adding the related system calls.

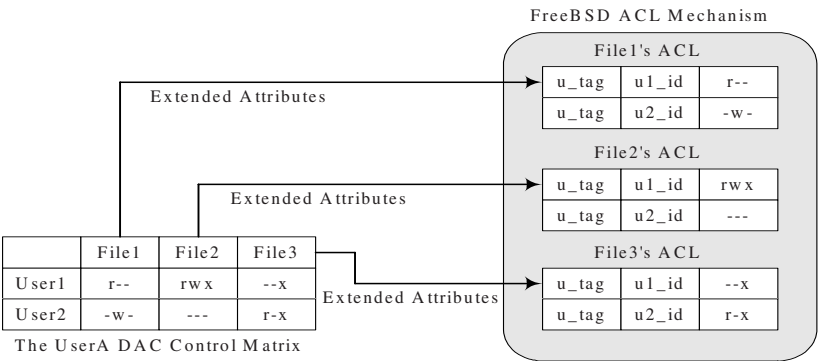


Fig. 1. One Demonstration of FreeBSD ACL Mechanismop

The maximal ACL storage space that is reserved under FreeBSD for each file can only store 32 access control entities. However, we have known from the Fig.1 that the system administrator must understand the significance of both ACL and the logical relationship in practice to manage the files' ACL. This will cause the obscurity and inflexibility of management.

4.2 Improvement Model

We propose our homonymous control domain design for ACL, and it is called the homonymous self-rule control mechanism. This mechanism includes three parts as follows:

- 1) Based on the FreeBSD ACL model, we have added one new type of the file label — the self-rule ACL tag (s_tag), which is used to identify the type of an ACL entity. In the s_tag type, we define a member which is the self-rule mode (s_mode) used to indicate the access control mechanism.
- 2) In the homonymous self-rule control mechanism, we can provide multi-type independent access control mechanism, such as RBAC. Normally, a user's homonymous self-rule control mechanism has only one access control mechanism entity. For example, in a project management, the project administrator can adopt RBAC to establish and manage all files of the project according to the roles of the project's members.
- 3) The homonymous self-rule control mechanism is associated with the FreeBSD ACL model in the kernel layer and cooperates with ACL to carry the detailed access control into execution. Synchronously, the homonymous self-rule control mechanism also provides the management interfaces for the application layer.

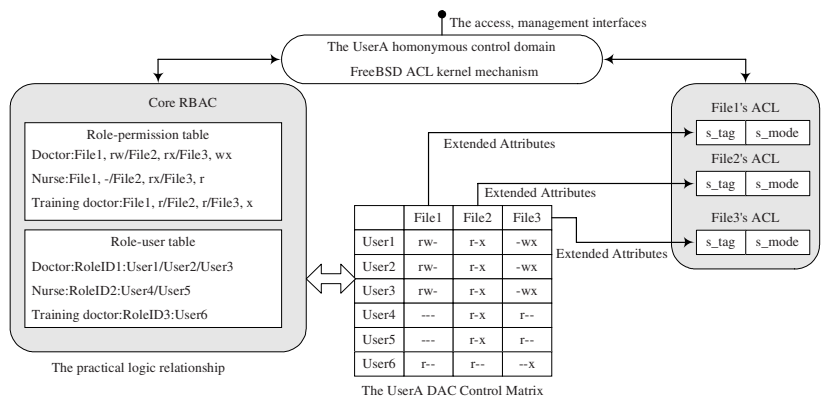


Fig. 2. UserA's homonymous self-rule control mechanism for ACL

As shown in Fig. 2, for instance the user *UserA* is an IT manager in one hospital, who adopts Core RBAC model to describe the roles' permission set (Doctor, Nurse and Training-doctor) to the file resources (File1, File2 and File3). *UserA* only needs to set one ACL entity (<s_tag, s_mode>) for each file to identify that the ACL tag is the homonymous self-rule control mechanism and the access control mode is Core

RBAC. The homonymous self-rule control mechanism can automatically create the corresponding role-user table and role-permission table. During the lifetime of the homonymous self-rule control mechanism, any change to access control cannot cause the ACL entity rewriting of the files. The homonymous self-rule control mechanism dynamically establishes the relationship between the Core RBAC assignment tables and the ACL mechanism in the access control routine.

5 Conclusions

In this paper, we have brought forward the conception of the homonymous role and formulated the related semantics and functions. Furthermore, we present the homonymous control domain technology for the DAC model system. Finally, we design the homonymous self-rule control mechanism for the files' ACL mechanism. The results of this paper also have practical significance, because they enhance the extensibility of the RBAC model and explore the cooperation and coexistence of DAC and RBAC.

Acknowledgement

The work of X.-W. Chu is partially supported by Hong Kong RGC grants under contract No. RGC/HKBU210605 and RGC/HKBU210406.

References

1. Zheng, Y., He, D., Yu, W., Tang, X.: Trusted Computing-Based Security Architecture for 4G Mobile Networks. In: Proceedings of the Sixth International Conference on Parallel and Distributed Computing, Applications and Technologies, pp. 251–255 (2005)
2. Ferraiolo, D., Cugini, J., Kuhn, D.R.: Role Based Access Control (RBAC): Features and Motivations. In: Proceedings of 1995 Computer Security Applications Conference, pp. 241–248 (1995)
3. Nychama, M., Osborn, S.: Access Rights Administration in Role-based Security Systems. In: Database Security. In IFIP Workshop on Database Security, pp. 37–56 (1994)
4. Ferraiolo, D., Sandhu, R., Gavrila, S., Kuhn, D.R., Chandramouli, R.: A Proposed Standard for Role Based Access Control. *ACM Transactions on Information and System Security* 224–274 (2001)
5. Ouyang, K., Zhou, J., Xia, T., Yu, S.: An Application-layer Based Centralized Information Access Control for VPN. *Journal of Zhejiang University (SCIENCE A)* 7(2), 240–249 (2006)
6. Sandhu, R.S., Munawer, Q.: How to Do Discretionary Access Control Using Roles. In: Proceedings of the Third ACM Workshop on Role-Based Access Control, New York, pp. 47–54 (1998)
7. Zhang, K., Jin, W.: Putting Role-based Discretionary Access Control into Practice. In: Proceedings of the Third International Conference on Machine Learning and Cybernetics, pp. 2691–2696 (2004)
8. Watson, R., Feldman, B., Migus, A., Vance, C.: Design and implementation of the TrustedBSD MAC framework. In: Proceeding of Third DARPA Information Survivability Conference and Exhibition, Washington, DC, vol. 2, pp. 13–15 (2003)

Ontology Based Hybrid Access Control for Automatic Interoperation

Yuqing Sun¹, Peng Pan¹, Ho-fung Leung², and Bin Shi¹

¹ School of Computer Science and Technology, Shandong University,
250100 Jinan, China

{sun_yuqing, ppan}@sdu.edu.cn, meal@163.com

² Department of Computer Science and Engineering, The Chinese University of Hong Kong,
Hong Kong, China
lhf@cuhk.edu.hk

Abstract. Semantic interoperation and service sharing have been accepted as efficient means to facilitate collaboration among heterogeneous system applications. However, extensibility and complexity are still crucial problems in supporting multi-level automatic collaborations across dynamically changed domains. In this paper, we propose the ontology based hybrid access control model. It introduces the concept of Industry Coalition, which defines the common ontology and servers as the portal of an application domain for public. By mapping local authorizations to the common ontology, an enterprise can efficiently tackle the problems of automatic interoperation across heterogeneous systems in the Coalition, as well as of the general requests from dynamically changed exterior collaborators not belonging to the Coalition. Several algorithms are also proposed to generate authorization mappings and maintain security constraints consistent. To illustrate our model, an example of property right exchange is given and experiment results are discussed.

1 Introduction

With the development of distributed technologies, interoperation and services sharing are widely adopted to support collaboration across different enterprise systems [1,2]. Furthermore, the collaboration is becoming flexible and dynamic due to frequently changed market. Take the case of the supply chain management: an enterprise should consider its steady and reliable partners as well as new collaborators. This makes the enterprise system usually face wide range inquiries and should authorize different access rights for sensitive information to dynamic users according to security policies and relationships with them. It would be a time consuming and error prone process to manually manage the authorizations. Therefore, autonomic access control is urgently required to cope with the growing complexity.

Ontology has been accepted as an efficient mean to facilitate collaboration across different system applications [3,4,5]. Many researches are conducted on semantic interoperation between distributed heterogeneous database [6], like the method of automatically detecting and resolving semantic conflicts by common ontology [7] and

the Access Control Toolkit (PACT) to enable privacy-preserving semantic access control without having to share metadata [8]. But these work focus on the structured data that may reside in structurally organized text files or database systems. Considering the vast amounts of resources instantly accessible to various users via web, which is semi constructed or unstructured, the semantic access control model (SAC) is proposed to support interoperability of authorization mechanism [9]. Propagation policies of authorization are proposed to prevent illegal inferences based on identification and categories of the domain-independent relationships among concepts [10]. Authors in [11] also develop a suite of tools to allow the use of semantic modeling features in XML documents. However, these work are mainly in the paradigm of communications between two ontology based systems and cannot process the plain requests without ontology. So, it is troublesome to support multi-level automatic collaborations across dynamically changed domains and enforce flexible policy.

In this paper, we propose a novel hybrid semantic access control model which introduces the concept of Industry Coalition to define the common domain ontology. On one side, by registering in the Coalition and mapping local authorizations to the common ontology, the registered member systems can automatically interoperate with each other. On another side, the Coalition servers as the portal of an application domain to help exterior collaborators query the registered members without any change of the requester's legacy systems. We also propose several algorithms of authorization mapping and security constraints verification. To illustrate our model, an example of property right exchange is given and experiment results are discussed.

The remainder of this paper is organized as follows. In section 2, preliminaries are given. In the following section we present the hybrid access control model. And then an illustrative example and experiments are discussed. At last, we draw some conclusions and future work.

2 Preliminaries

Ontology has been defined as a concept system, in which concepts are interpreted in a declarative way, as standing for the sets of their instances [12]. A common ontology-based manipulation of different resources is one of the most desirable solutions for achieving semantic interpretabilities. In work with a common ontology, four important issues should be considered: the construction of the common ontology using a comprehensive classification framework, maintenance of the ontology to allow its evolution, mapping from an information system to the common ontology, and solution of various context-dependent incompatibilities.

Since the role based access control model (RBAC) is considered as the most appropriate paradigm for access control in complex scenarios [13], our proposed model focuses on RBAC system. In RBAC, role is an abstract description of behavior and collaborative relation with others in an organization. Permission is an access authorization to object, which is assigned to role instead of to individual user so as to simplify security administration. The motivation of role hierarchy is to efficiently manage common permissions by defining multiple reusable subordinate roles in formulating other roles. Constraints are principles used to express security policy.

3 The Ontology Based Hybrid Access Control Model

The proposed ontology based hybrid access control model, called *OHAC*, is depicted in Fig.1. Different with other semantic models, it introduces the concept of Industry Coalition, which represents as an association or guild of representative enterprises in a specific application domain. By defining common ontology of the domain, the Coalition provides a platform for members to share, federate and collaborate with each other, as well as serves as a portal to provide common services for the public.

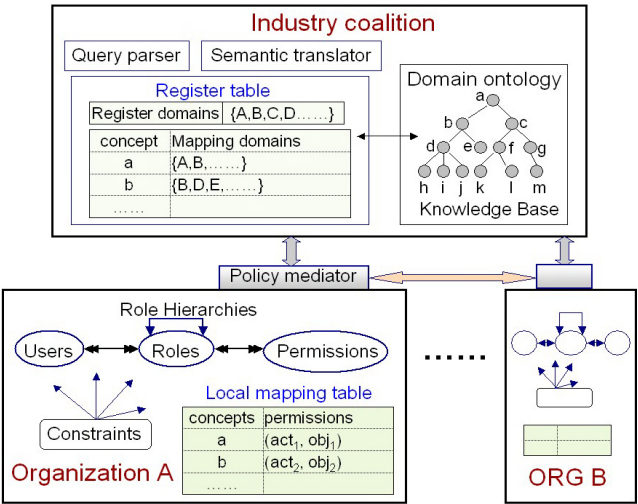


Fig. 1. The ontology based hybrid access control model (*OHAC*)

Participant members of the Coalition are distributed and autonomous in the sense that they keep control on their own resources and the rights to change the meaning and implementation of authorizations, in which role hierarchies, security policies etc. may be heterogeneous. They register in the Coalition and establish the mappings of local authorizations to the common ontology so as to support the collaboration with other registered members and respond requests coming from public users. The proposed *OHAC* model is formally defined in the following subsections.

3.1 Modeling Industry Coalition

The Industry Coalition of *OHAC* is responsible for constructing the common ontology and maintaining the register information about member enterprises. The Query Parser model is used to analyze and process users' request. If a request comes from exterior of the coalition and is not in the ontology language, the Query Parser will pass it to the Semantic Translator for translating into ontology-based query according to the common ontology. Then the Query Parser transfers it to the correlative members. Here is the formal definition of Industry Coalition.

Definition 1: A *Concept Cpt* is a generalized abstract term that may have several concrete instances, which is in form of triple $Cpt = \langle Name, Des, Mpl \rangle$ where *Name* is the identifier of *Cpt*, *Des* is the description of *Cpt* in plain text, and *Mpl* is the distinct set of mapping instances and stores the registered enterprise information that have mapped their authorizations to *Cpt*. It may have several properties that expressed as $\langle org, portal \rangle$, where *org* is the identifier of the mapped enterprise and *portal* is the mapping information.

Definition 2: *Ontology OT* is a distinct set of concepts and their relations, which is defined as 2-tuple $OT = \langle CONCEPT, CR \rangle$, where *CONCEPT* is a set of concepts and *CR* is a relationship on *CONCEPT*. *CR* has the form of $\langle c_1, c_2, relation \rangle$, where $c_1, c_2 \in CONCEPT$, and *relation* is a member of set $\{sub-class, part-of, disjoint, property\}$, which refers to the relationship between concepts c_1 and c_2 .

Definition 3: *Industry Coalition IC* is a 4-tuple $IC = \langle Name, Des, OT, RT \rangle$, where *Name* is a String that identifies the industry coalition, *Des* is the textural description that outline the purpose of the industry coalition, *OT* is the common ontology definition of a specific domain, and *RT* is the register table of member enterprises.

3.2 Leveraging the Legacy System of Coalition Member

Within a coalition, member enterprises may have different local meanings of system authorization and their resources may be stored in the structured data like database, semi-structured XML data or unstructured files like audio, video and pictures etc. To support automatic semantic interoperation, they should leverage their legacy systems by adding web-based interface to map local supplied authorization to the common ontology. Generally, RBAC model is adopted to enforce security policies in an enterprise legacy system, in which permissions are assigned to roles and users are assigned to concrete roles so as to acquire the permissions. So system will grant appropriate roles for authorization requests.

Definition 4: *Local Mapping Table LMT* is a triple $LMT = \langle cpt, loc\_cpt, ptr \rangle$, where *cpt* is the identifier of a concept in the common ontology, *loc_cpt* is the identifier of a locally defined concept that is corresponding to *cpt*, *ptr* gives the link of the authorized permission in local system that is relative to *cpt*.

Definition 5: $TYPE = \{normal, forbidden\}$ is an enumerable type set of authorization or role, in which *normal* and *forbidden* respectively denote whether an authorization or role is permitted or forbidden for a request coming from exterior users.

Definition 6: A *Permission per* is defined as a 4-tuple $per = \langle id, des, type, impt \rangle$, where *id* is the identifier of *per*, *des* is its description, $type \in TYPE$ is the type of *per*, and *impt* is the implementation of *per* which is generally in form of (act, obj) to give the concrete operation.

Definition 7: A *Role r* is defined as a 4-tuple $r = \langle id, des, type, p\_set \rangle$, where *id* is its identifier, *des* is its description, $type \in TYPE$ is the role type that denotes open or not for public, and *p_set* is the set of authorization that are assigned to *r*.

Definition 8: An *Inheritance Relation IR* refers to the relationship between two roles with the properties of antisymmetric and transmissible. If role r_i inherits all the per-

missions owned by role r_2 , we denote it as $IR = (r_1, r_2)$ or $r_1 \geq r_2$ and all users of r_1 are the users of r_2 .

Also we give the predicates of $AuthorizedP(r)$ to calculate the permissions owned by the given role r , which are used in the following algorithms. R and P denote the set of roles and permissions respectively.

$$AuthorizedP(r \in R) = \{p | p \in P \quad p \in r.p\_set\}$$

3.3 Hybrid Semantic Authorization Query

After Industry Coalition establish the common ontology and member enterprises map their local authorizations to the ontology, the proposed *OHAC* model can support the hybrid automatic interoperations: *inter-access* that is across the registered member enterprises in the Coalition, and *exterior access* that is with the dynamically changed exterior enterprises not belonging to the Coalition. Details are given below.

Inter-access: When an enterprise wants to communicate with other member in the same coalition, it firstly queries the Coalition server whether the requested enterprise has registered. The Coalition server will check the *register table* and return the result. If both sides have registered on the same coalition, which means they have mapped local authorizations to the common ontology, they can communicate directly. In this case, the applicant translates its queries from local concepts into common concepts, which are inter-coalition understandable. The provider receives and translates the query from the common concept into its local means of authorization according to *local mapping table*. And then it judges whether the request is permitted or denied complying with its security policies. This process of *inter access* is illustrated in Fig.2 and the authorization management algorithm of *Authorization_Query* is given in the following subsection.

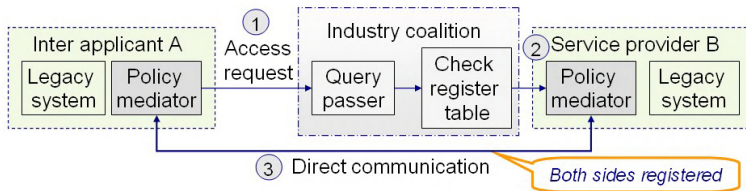


Fig. 2. *Inter-access* process across enterprises in same industry coalition

Exterior access: Member enterprises of a Coalition often have the requirements to collaborate with new appropriate partners not belonging to the Coalition so as to find new business opportunities. Vice versa, the public want to have knowledge of the industry and representative enterprises for business. In this case, the Coalition serves as a portal of all the registered enterprises to provide open services for the public. When an exterior access is requested, the Coalition server translates it into inter-coalition understandable text according to the common ontology. Then it checks the *register table* and parses the query to the correlative servers of registered enterprises that have supplied services. When these members receive the request, they activate (or deny) different roles for the applicant according to their security policies, which is

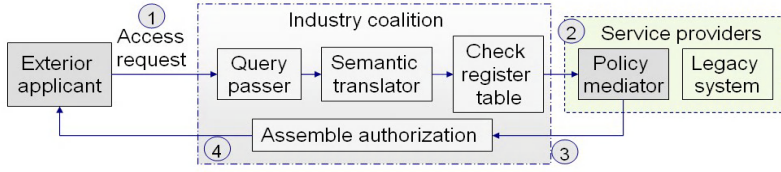


Fig. 3. Exterior access request process

depicted in the below algorithm of *Authorization_Query*, and return the results to the Coalition. The Coalition collects the results and transfers them to the applicant. This process is illustrated in Fig.3.

3.4 Role Mapping and Generation

This subsection describes the process how an enterprise responds requests. Since local systems mainly adopt RBAC to manage access rights, authorizations are embodied in roles. So it is crucial to determine which roles are granted to the applicant. We propose the algorithm to map or generate roles for a set of requests as below.

Algorithm: *Authorization_Query*(RQ, RS)

Input: a set of requested authorizations $RQ = \{a_1, a_2, \dots, a_k\}$

Output: a set of permitted roles $RS = \{r_1, r_2, \dots, r_n\}$

1. for each $a_i \in \{a_1, a_2, \dots, a_k\}$ do step 2
2. if $a_i.type = forbidden$ then mark a_i with *DENY*; $RQ = RQ - \{a_i\}$;
3. Verify all authorizations in RQ to satisfy all security constraints;
4. If not consistent then remove the conflict authorizations from RQ ;
5. For each $r \in R$ do step 6 to step 7
6. if $AuthorizedP(r) \subseteq RQ$
7. then $RS = RS \cup \{r\}$; $RQ = RQ - AuthorizedP(r)$;
8. If $RQ \neq \emptyset$ then generate a new role r' where $r'.p\_set = RQ$; $RS = RS \cup \{r'\}$;
9. Return RS .

The system firstly verifies the requests satisfying all security constraints and wipes off the forbidden authorizations. Then it searches exist roles and select the roles whose assigned permissions is a subset of the requests as candidates for the applicant. For those requests not belonging to a single role, the system will generate a new role to cover them and consider the role as a candidate too. By granted the above roles, the applicant can activate the correlative authorizations. For the complexity of above algorithm, suppose n_s and n_r be the number of security constraints and roles respectively, k is the number of requests, it is in $O(n_r + k * n_s)$.

3.5 Security Analyses

A critical issue of automatic interoperation is to ensure security constraints consistent. We focus on the constraint of conflict of interests (*CoI*) here, while others can be discussed similarly. *CoI* restrict access rights to sensitive information about enterprises with interest conflicts to different users. In an open environment, we specially

should consider the case that users acquire conflict permissions via multi domain role inheritances [14].

Here are two examples to illustrate the conflicts, which is depicted in Fig.4. In (a), roles b_2 and b_3 have the *COI* constraint in enterprise B, while in (b), roles b_2 and b_4 are with *COI*. Suppose user Alice is assigned to the role a_2 , Bob is assigned to the role a_3 and John is assigned to the role a_1 in enterprise A. In example of (a), Alice and Bob separately request the authorizations of B and acquire the roles of b_2 and b_3 . So John can acquire b_2 and b_3 simultaneously by inheritance. In (b), Bob acquires the new role that is generated for his requests. Although the new role and b_2 are without *COI*, John still acquires the conflict authorizations of p_1 and p_4 by inheritance. So above two cases all violate the security constraints of *COI*. Following property and correlative verification algorithm are given to verify and keep the security constraints consistent.

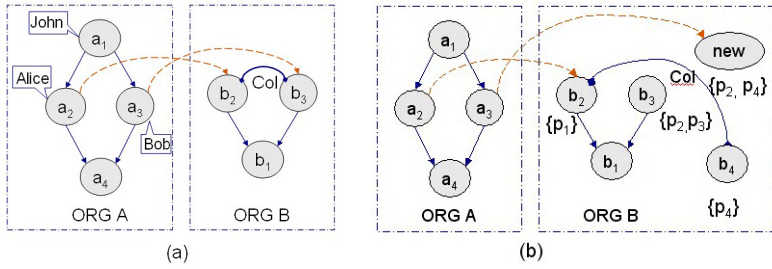


Fig. 4. Two examples of *COI* conflicts arising from multi-domain interoperation

Property: Let CS be the set of *COI* constraints. Each constraint is in form of $rs = (n, p_1, p_2, \dots, p_n) \in CS$. If rs is required for a set of permissions $p_1, p_2, \dots, p_n$, then $p_1, p_2, \dots, p_n$ should not be assigned to the same role and furthermore not assigned to the same user via different roles.

Algorithm: *Verify_COI*(CS, RS)

Input: *COI* constraint set CS and the mapped role set RS to the same enterprise

Output: *True* if roles in RS satisfy constraints in CS ; *False*, otherwise.

1. for all $r_i \in RS = \{r_1, r_2, \dots, r_n\}$ do step 2
2. $\text{congregation\_permissions} = \bigcup_{i=1,2,\dots,n} \text{AuthorizedP}(r_i)$
3. For each $rs \in CS$, do step 4 to step 5
4. $\text{overlap\_perms} = \text{congregation\_permission} \cap rs$
5. if $|\text{overlap\_perms}| > 1$ then return *Flase*
6. Return *True*

Let $|RHI|$ denotes the number of role hierarchies. The complexity of predicate *AuthorizedP* is $O(|RHI|)$ because it should calculate all the permissions that are authorized to its junior roles. Suppose n_s be the number of *COI* constrains. The complexity of algorithm *Verify_COI* is in polynomial time of $O(|RHI| * n + n_s)$.

4 Illustrative Example and Experiments

In this section, we adopt an example of the property rights exchange to illustrate how to apply the proposed *OHAC* model in supporting automatic interoperation. Property rights exchange in China includes enterprise assets exchange, intangible assets exchange etc [15]. Property Rights Exchange Centers (*PREC*) are the concessionary enterprises that are responsible for organizing the exchange, of which systems are heterogeneous in security policies and resource structures. The relationships among them are different and they cooperate with each other at different aspects with different depth, which may be changed dynamically. Some of them associate together to exert their strong points, like the North-Association of Property Rights Exchange (*NAPRE*). There are different kinds of interoperation requirements across centers, association, participants and government etc. Accompanying with the development of property rights trade, automatic interoperation is needed to improve the efficiency while satisfying the overwhelming objective of system security.

We consider it as an appropriate example to apply the *OHAC* model, in which *NAPRE* is regarded as the Industry Coalition and is responsible for defining the common ontology of property rights exchange domain, illustrated as Fig.5. The sketch map of the register table in *NAPRE* is given in Tab.1 that records the information

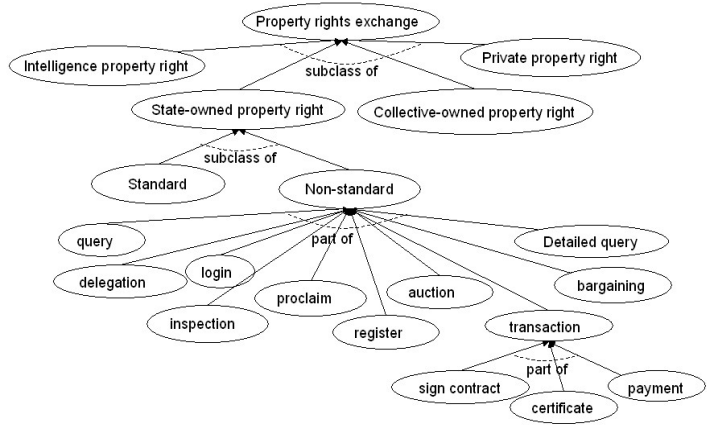


Fig. 5. The common ontology of property rights exchange domain

Table 1. The register table of the *NAPRE*

Concept	Mapping Domains
delegation	<i>QD, JN</i>
inspection	<i>QD, JN</i>
query	<i>SD, ZJ, TJ, BJ</i>
detailed query	<i>QD, JN</i>
proclaim	<i>QD, JN</i>
sign contract	<i>QD, JN, WF, SD, ZJ, TJ, BJ</i>
certificate	<i>QD, JN</i>
bargaining	<i>QD, JN, WF, SD, ZJ, TJ, BJ</i>

Table 2. The local mapping table of *PREC QD*

Concepts	Local Concepts	symbol	Authorizations	Description
delegation	entrust	en	http://www.qd_pr.com/entrust.jsp	delegatedproject
inspection	monitor	mnt	http://www.qd_pr.com/servlet/monitorServlet	delegatedproject
detailed_query	query_in_detail	que	http://www.qd_pr.com/detailquery.jsp	projectinprocess
proclaim	bulletin	blt	http://www.qd_pr.com/bulletin.html	vendedproject
sign_contract	contract	con	http://www.qd_pr.com/servlet/contractServlet	vendedproject
certificate	witness_trades	wit	http://www.qd_pr.com/witnesstrades.htm	vendedproject
bargaining	trade-off	trd	http://www.qd_pr.com/trade-off.jsp	vendedproject

Table 3. System response time

Access Type		Response Time(ms)
Inter-access	1. A→B	1170
	2. A→IC→B	2052
Exterior access	3. Extra→IC→A	2484
	4. Extra→IC→A & B	4536

Table 4. System specifications

Tier	IC	Member A	Member B	Exterior applicant
CPU	Pentium 4 2.66GHz	Pentium 4 2.4GHz	Pentium 4 2.4GHz	Athlon XP 2000+
RAM	1GB	512MB	512MB	512MB
OS	Winxp sp2	Winxp sp2	Winxp sp2	Winxp sp2

about the member exchange centers in *NAPRE*. Tab.2 describes the local mapping table of the *PREC QD*.

We investigate four aspects of the proposed *OHAC* model, which are the direct interoperation between two registered members, two members interoperation via the Industry Coalition (*IC*), an exterior applicant interoperating with one member A via *IC*, and the exterior applicant interoperating with two members of A and B via *IC*. We program the prototype in Java with Sun Java j2sdk-1.4.2.04 and Apache Tomcat 5.0.28 Web Container. The network is established on the China Education and Research Network (CERNET). Each node is distributed in a different net segment that is connected by 100 Mbps LAN. The experiment results are given in Tab.3 and the system specifications are described in Tab.4. On condition of auto-interoperation, we can see from Tab.3 that the response time of type 1 is the lowest since it saves much network time. Type 2 and type 3 are similar with a little more time in type 3 for ontology translation. The time impact of type 4 is the highest, which explains that distribution of query consumes much time.

5 Conclusions and Future Work

This paper discusses the crucial problem of multi-level automatic collaborations across dynamically changed and heterogonous domains. It proposes a hybrid access control model, which introduces the concept of Industry Coalition to define the common ontology and server as the portal of a specific application domain. By mapping local

authorizations to the common ontology, enterprises can efficiently support automatic interoperations across heterogonous member systems in the Coalition, as well as the general requests from dynamically changed exterior collaborators not belonging to the Coalition. Several algorithms are also proposed to generate authorization mappings and maintain security constraints consistent. At last, an illustrative example and experiments show its effect and efficiency. Further works include improving the role generation algorithm and applying this model to new application domains.

Acknowledgements

This work was partially supported by the National Nature Science Foundation of China (90612021), the National High Technology Research and Development Program of China (863 Program) (2006AA01A113), Science Development Plan Program of Shandong province of China (2004GG2201131) and the Natural Science Foundation of Shandong Province of China (Y2004G08).

References

1. Ferraiolo, D., Barkley, J., Kuhn, R.: A Role-Based Access Control and Reference Implementation within a Corporate Intranet. *ACM TISSEC* 2, 34–64 (1999)
2. Park, J., Sandhu, R., Ahn, G.: Role-based Access Control on the Web. *ACM TISSEC* 4, 37–71 (2001)
3. Tekeda, H., Iwata, K., Takaai, M., Sawada, A., Nishida, T.: An ontology-Based Cooperative Environment for Real World Agents. *Int. Conf. of Multi-agent Systems*, pp. 353–360 (1996)
4. Park, J.S.: Towards Secure Collaboration on the Semantic Web. *ACM SIGCAS Computers and Society* 33, 1–10 (2003)
5. Bertino, E., Fan, J.P., Ferrari, E., Hacid, M.S., Elmagarmid, A.K., Zhu, X.Q.: A hierarchical access control model for video database systems. *ACM TOIS* 21, 155–191 (2003)
6. Pan, C.C., Mitra, P., Liu, P.: Semantic Access Control for Information Interoperation. In: *Proc. of SACMAT'06*, Lake Tahoe, California, USA, pp. 237–246 (2006)
7. Ram, S., et al.: Semantic Conflict Resolution Ontology: An Ontology for Detecting and Resolving Data and Schema-level Semantic Conflicts. *IEEE TKDE* 16, 189–202 (2004)
8. Mitra, P., Pan, C.C., Liu, P., Vijayalakshmi, A.: Privacy-preserving semantic interoperation and access control of heterogeneous databases. In: *Proc. of ASIACCS*, pp. 66–77 (2006)
9. Yague, M.I., Gallardo, M., Mana, A.: Semantic Access Control Model: A Formal Specification. In: di Vimercati, S.d.C., Syverson, P.F., Gollmann, D. (eds.) *ESORICS 2005*. LNCS, vol. 3679, pp. 24–43. Springer, Heidelberg (2005)
10. Li, Q., Vijayalakshmi, A.: Concept-level access control for the Semantic Web. In: *Proc. of the ACM workshop on XML security*, Fairfax, Virginia, pp. 94–103 (2003)
11. Trastour, D., Preist, C., Coleman, D.: Using Semantic Web technology to Enhance Current Business-to-Business Integration Approaches. In: *Proc of EDOC*, pp. 222–231 (2003)
12. van der Vet, P.E., Mars, N.J.I.: Bottom-Up Construction of Ontologies. *IEEE TKDE* 10, 513–526 (1998)
13. Sandhu, R.S., Coyne, E.J., Feinstein, H.L., Youman, C.E.: Rose-Based Access Control Model. *IEEE Computer* 29, 38–47 (1996)
14. Shafiq, B., Joshi, J.B.D., Bertino, E., Ghafoor, A.: Secure Interoperation in a Multidomain Environment Employing RBAC Policies. *IEEE TKDE* 17, 1557–1577 (2005)
15. Sun, Y.Q., Pan, P.: PRES—A Practical Flexible RBAC Workflow System. In: *Proc. of 7th International Conference on Electronic Commerce*, pp. 653–658 (2005)

Recoverable Tamper Proofing Technique for Image Authentication Using Irregular Sampling Coding

Kuo Lung Hung and Chin-Chen Chang

Department of Information Management, Chaoyang University of Technology
klhung@cyut.edu.tw

Abstract. Digital imagery is an important medium in information processing in the digital era. However, a major problem in information processing is that digital images can be forged easily. If this problem cannot be alleviated, the popularization of digital imagery will be decreased. Hence, in recent years, a few tamper proofing or image authentication techniques have been proposed to deal with this problem. In this paper, a new recoverable image authentication technique is proposed. Our method employs a very low bit-rated compression method called irregular sampling coding to compress the image. The compressed code is then randomly embedded into the original image using the digital watermarking technique. Since the image is highly compressed, it can be used to detect and recover the tampered-with information. Experimental results show that the proposed tamper proofing technique can effectively detect and recover the modified image. In addition, the experiments also show that the proposed technique is robust. Even though the image is 90%cropped or is highly compressed using JPEG, the quality of the recovered image is acceptable. The proposed method is therefore an effective, robust and recoverable tamper proofing technique.

1 Introduction

Digital image application is an important part of daily life in the digital era. However, since digital images have characteristics that allow them to be easily forged, people are becoming more and more reluctant to believe that an image that they see is authentic. Without any traces of an image being tampered with, features of an image can be replaced or added to using an image processing software such as PhotoShop. This is, in fact, one of the reasons why digital imagery is not acceptable as evidence in a court of law. If this problem cannot be alleviated, the degree of popularity of digital images will be lessened. Hence, in recent years, a few tamper proofing or image authentication techniques have been proposed to deal with this problem. A good image authentication technique should satisfy some basic requirements. In this paper, four important requirements are proposed and listed as follows:

1. *Effectiveness*: The parts of the image that have been tampered with should be effectively pointed out.
2. *Security*: A sound security mechanism must be provided.

3. *Differentiation*: The technique should be able to differentiate between an innocent adjustment by some type of image processing operation and an intentional modification.
4. *Recoverability*: The technique should have the ability to recover the content that has been tampered with.

One of the first techniques used for the detection of image tampering was proposed by Walton [9]. This method requires the calculation of the checksum of the seven most significant bits of the image so that the checksum may be embedded into the least significant bits of randomly selected pixels. Wolfgang and Delp in [10] proposed a fragile watermarking technique involving the addition of two-dimensional m -sequences for tamper detection, where the m -sequences are the mapping of the binary sequences of the watermark from $\{0,1\}$ to $\{-1,1\}$. They also defined a non-binary test statistic based on the inner product of the m -sequence and the watermarked image. In [7], Schneider and Chang proposed a method for content-based image verification. This method defines a continuous interpretation of the concept of authenticity which measures the proximity of specific features of a possibly modified image to the original one. First, the features of the relevant content are extracted, and next, they are hashed to reduce their size. After that, the result from the previous stage is encrypted with the author's private key. In [4], Kundur and Hatzinakos proposed another technique for signal tamper proofing. The method places the watermark in the discrete wavelet domain by quantizing the coefficients with a user-defined key to a pre-specified degree. This gives their approach the ability to detect the modifications in localized spatial and frequency domain regions. Moreover, the detection can suffer from a number of distortions such as substitution of data, filtering and lossy compression.

To synthesize the above research, each method can correctly point out the positions that have been tampered with, and some of them can satisfy the security and differentiation requirements. However, none of the methods possess the ability to recover image modifications. The recoverability requirement, in our opinion, is very important. For example, certain special documents, such as wills or medical records, contain original contents that are very important and need to be recovered. Recent research [12-15] has noticed the importance of the recoverability requirement.

In this paper, a newly recoverable image authentication technique is proposed. This method first employs a very low bit-rated compression method called irregular sampling coding to compress the image. The compressed code is then randomly embedded into the original image using the digital watermarking technique. Since the image is highly compressed, it can be used to detect and recover the tampered-with information.

The rest of this paper is organized as follows. The verification information generation method is introduced in Section 2. The embedding approach is described in Section 3. Section 4 describes the methods of tamper detection and recovery. In Section 5, the experimental results are shown and discussed. Finally, the conclusions are stated in Section 6.

2 Verification Information Generation

The first step of our proposed method is the generation of the verification information (i.e., the digital watermark). In our approach, the verification information is the result of highly compressing the image. In this section, the generation steps can be divided into the following: image shrinking, irregular sampling and information partitioning.

2.1 Image Shrinking

In order to reduce the size of the verification information, the original image is shrunk. Suppose that the size of the original image X is $N \times N$, and the size of the shrunk image Y is $M \times M$. Therefore, the definition of X , Y is

$$\begin{aligned} X &= \{x(i, j) \mid 0 \leq i < N, 0 \leq j < N, x(i, j) \in \{0, 1, \dots, 255\}\}, \\ \text{and } Y &= \{y(i, j) \mid 0 \leq i < M, 0 \leq j < M, y(i, j) = x(\frac{N}{M} \cdot i, \frac{N}{M} \cdot j)\}. \end{aligned} \quad (1)$$

2.2 Irregular Sampling Coding

In this section, the irregular sampling technique is employed to further compress the shrunk image Y . Irregular sampling techniques have long been used in the computer graphics field to achieve a compact representation of images. Further advantages can be achieved if the sampling distribution is not only irregular but also non-uniform. Although [5] gives a complete treatment of the irregular sampling algorithm, the core elements are restated here.

Let $y(i, j)$ be the gray level of the pixel (i, j) of the shrunk image Y to be sampled. The sample skewness of Y , evaluated on an $m \times m$ mask Ψ centered in (i, j) can be defined as

$$\sigma_y^3(i, j) = \frac{1}{m \times m} \sum_{(i', j') \in \Psi} (y(i', j') - \mu(i, j))^3, \quad (2)$$

where $\mu(i, j)$ is the sample mean evaluated on the same mask Ψ . To make this operator independent of the dynamic range of the image, it is convenient to normalize it by

$$sk(i, j) = \frac{|\sigma^3(i, j)|}{\max_{i, j} (|\sigma^3(i, j)|)}. \quad (3)$$

All of the samples (i, j) whose normalized local skewness $sk(i, j)$ is higher than a predefined threshold, θ_s , are included in the grid G_s .

Note that the image samples located exactly on the edge of an object need not to be considered. This is due to the fact that a sample of this kind does not uniquely belong to a single object, and its intermediate luminance value will yield a blurred reconstructed contour. However, in practical applications, due to possible edge irregularities, the value of σ^3 of the samples lying on the edges may not be exactly zero. Consequently, it may occur that these samples are included in the grid. Hence, we employ a gradient $grad(i, j)$ for all of the pixels of the image, and all of the samples

(i,j) whose $grad(i,j)$ is lower than a predefined threshold θ_g constitute a second grid, G_g . It is then apparent that the samples belonging to the intersection of the two grids, $G_{sg}=G_s \cap G_g$, are considered in the proposed method.

In order to further compact the representation, the number of samples can be reduced by varying the grid density along the edges, e.g., by decimating the grid G_{sg} . To do this a circular forbidden area around each sample is defined. That is, if a sample belongs to the decimated grid, G_{dec} , no other samples at a distance lower than r can also belong to G_{dec} . It is then obvious that the sparseness of the sample grid can be easily controlled by adjusting the size of the forbidden area.

The result of these operations is a grid with an almost uniform sample density along the edges, and no samples in areas with a constant or linearly changing gray level. In real world images, which present both sharp and blurred edges, it is advisable to have not two but various levels of sample density, according to the local image content. This is obtained by following a multi-resolution approach, taking a set of n different grids with different densities. More precisely, not one but several thresholds are considered for the skewness, $0 = \theta_s^{(0)} < \theta_s^{(1)} < \theta_s^{(2)} < \dots < \theta_s^{(n)} = 1$, so that (i,j) belongs to $G_s^{(k)}$ if $\theta_s^{(k-1)} < sk(i,j) < \theta_s^{(k)}$, $k=1,2, \dots, n$. Then, for each k , the decimated grid $G_{dec}^{(k)}$ is obtained from $G_{sg}^{(k)} = G_s^{(k)} \cap G_g$, using a forbidden area having a radius $r^{(k)}$, with $r^{(1)} > r^{(2)} > \dots > r^{(k)} > \dots > r^{(n)} > 0$. The final grid is obtained as the union of the decimated grids

$$G = \bigcup_{k=1}^n G_{dec}^{(k)}. \quad (4)$$

As an example, the grid corresponding to a portion of the image Zelda and the case $n=2$ are shown in Figure 1.

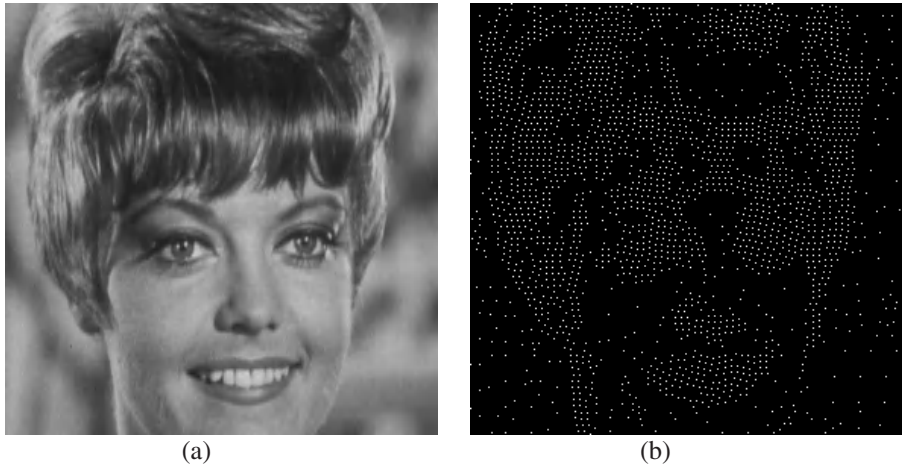


Fig. 1. (a) Detail of the original image Zelda, and (b) grid obtained using the irregular sampling algorithm

2.3 Information Partitioning

In order to increase the robustness of the verification information, the set G of the sampling point that was sampled in Section 2.2 will be partitioned into L segments. Next, these segments will be scrambled using the pseudorandom generation functions and will be embedded into the differing locations of the original image. The steps of the partitioning are stated as follows.

The original image X is first divided into 8×8 blocks B_i 's, where the total number is $\frac{N \cdot N}{8 \cdot 8}$. Suppose that the corresponding block in the shrunken image Y of the block B_i

is b_i , and then the size of the block b_i will be $(M \cdot \frac{8}{N})(M \cdot \frac{8}{N})$. Suppose again that the set of the sampling point contained in the block b_i is $G_i = \{p_{i,1}, p_{i,2}, \dots, p_{i,s}\}$, where $0 \leq s \leq (M \cdot \frac{8}{N})(M \cdot \frac{8}{N})$. Note that, since the parameters θ_s and r are adjustable, without loss of generality, we can assume that the condition $s \geq L$ always holds.

Next we will select just L representative points from the set G_i to form a new set of sampling points $G'_i = \{p'_{i,1}, p'_{i,2}, \dots, p'_{i,L}\}$. The criterion for the selection of the representative is that the distance between the selected point and the other representative points should be as far as possible. Therefore the set S of the verification information segments is defined as

$$S = \{S_1, S_2, \dots, S_L\}, \text{ where } S_j = \{\Theta(p'_{i,j}) \mid \forall 0 \leq i \leq \frac{N \cdot N}{8 \cdot 8}\} \quad p'_{i,j} \in G'_i \}. \quad (5)$$

Here Θ is the coding function of the sampling points. Note that the coding of a sample point should contain two parts: the position information and the gray value.

The size of the position information is $\log_2((M \cdot \frac{8}{N})(M \cdot \frac{8}{N}))$, and the gray value can be further quantized to conserve space.

3 Embedding Method

In this section, a discrete cosine transform (DCT) based embedding method is proposed. The method first scrambles the verification information generated in Section 2 and embeds the information into the middle-high frequency coefficients of the image transformed by DCT. Finally, the inverse discrete cosine transform (IDCT) is performed on these coefficients to obtain the embedded image.

3.1 Scrambling of the Verification Information

In our proposed method, the first segment of the verification information S_1 is embedded to its own corresponding block, and the other segments are scrambled using the differing pseudorandom generation functions. In this section, we define a set of ran-

dom generation functions Π as $\{\pi_2, \pi_3, \dots, \pi_L\}$, and define the scrambled verification information (i.e., the watermark) as $\mathbf{W}=\{W_1, W_2, \dots, W_L\}$,

$$W_i = \begin{cases} S_I & \text{when } i = 1, \\ \pi_i(S_i) & \text{when } 2 \leq i \leq L. \end{cases} \quad (6)$$

Note that a seed S_d must be presented during the generation of the random numbers. The seed is the secure key for the later tamper detection and recovery.

3.2 The Hiding Scheme

The proposed method embeds some verification information into the coefficients of the DCT-transformed image. In order to invisibly embed the information without much deterioration of the image quality, the middle frequency range is chosen for embedding.

Consider an image block B_i . Suppose its sub-watermark $\mathbf{w} = (e_1, e_2, \dots, e_r)$ and the middle-high frequency $C_i = (c_1, c_2, \dots, c_r)$. The hiding function H and the extracting function E are defined as

$$H(c_j, e_j) = \begin{cases} \left\lfloor \frac{c_j}{4\alpha} \right\rfloor \times 4\alpha + 2\alpha & \text{if } e_j=1, \\ \left\lfloor \frac{c_j + 2\alpha}{4\alpha} \right\rfloor \times 4\alpha & \text{if } e_j=0, \end{cases} \quad (7)$$

and

$$E(c_j) = \begin{cases} 0 & \text{if } ((c_j + \alpha) \bmod 4\alpha) < 2\alpha, \\ 1 & \text{otherwise,} \end{cases} \quad (8)$$

where $\alpha \geq 1$ is the magnitude of adjustment.

4 Detection and Recovery Method

The embedded image can be published when the verification information is embedded. Suppose that some days have passed and that the embedded image might have been tampered with. If the image needs to be verified, tamper detection is then performed. As mentioned before, the verifier must have a private key. When the result of the detection shows that the image has indeed been tampered with, the recovery work is then performed. In this section, our method is divided into two parts: tamper detection and tamper recovery.

4.1 Tamper Detection

Before information extraction can be performed, the image in question first needs DCT transformation, and the middle-high frequency coefficients in zigzag order must be determined in advance. Then, the information $\mathbf{w}_i$ hidden in the block B_i can be

extracted by Equation 8. After all w_i 's are extracted, the protected verification information S can be obtained using the seed S_d . Therefore, the set of sampling points $G'_i = \{p'_{i,1}, p'_{i,2}, \dots, p'_{i,L}\}$ of each block B_i is then determined.

The sampling points are sampled by the shrunk image. In order to match the original image so that tamper detection and recovery are workable, the sample points are enlarged. Suppose that the set of enlarged sampling points is $G''_i = \{p''_{i,1}, p''_{i,2}, \dots, p''_{i,L}\}$. Therefore, after the hidden verification information is extracted and decoded, tamper detection is performed upon the image in question. In our method, the unit of detection is an 8×8 sub-block. Then, a block B_i is said to have been tampered with if $|x(i, j) - p''_{i,1}| > T_s$, where T_s is the a threshold, $x(i, j)$ is the value of the corresponding pixel of the sampling point $p''_{i,1}$ in the block B_i of the image in question.

4.2 Tamper Recovery

The decoder of the irregular sampling algorithm needs to perform an adaptive interpolation in order to reconstruct the whole image. In order to keep the overall system complexity very low, the proposed method uses the well-known 4NN mechanism [1]. In the 4NN algorithm, each pixel $\hat{x}(i, j)$ to be reconstructed is obtained via a linear combination of its four closest pixel $x_l(i, j)$, $l = 1, 2, \dots, 4$ with weights $w_l(i, j)$:

$$\hat{x}(i, j) = \frac{1}{W(i, j)} \sum_{l=1}^4 w_l(i, j) x_l(i, j), \quad (9)$$

with $W(i, j) = \sum w_l(i, j)$. It is reasonable to use larger weights for those pixels which are closer to the one being interpolated; a common choice is the following:

$$w_l(i, j) = 1/d_l(i, j), \quad (10)$$

where $d_l(i, j)$ is the Euclidean distance between $\hat{x}(i, j)$ and $x_l(i, j)$.

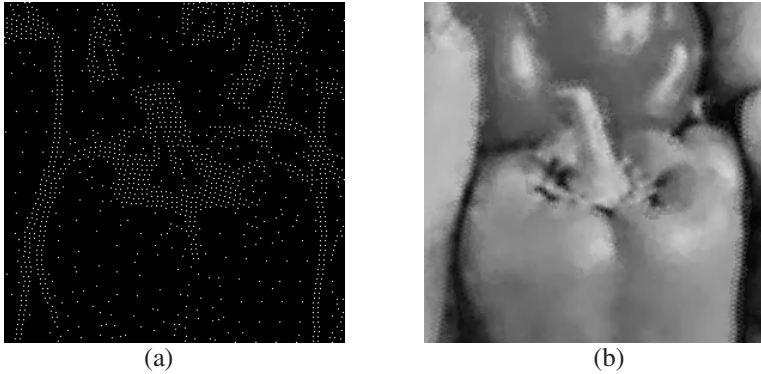


Fig. 2. (a) Sampling points obtained using the irregular sampling algorithm (b) the reconstructed image using the 4NN algorithm

5 Experimental Results

Our experiments are performed on a Pentium 586 PC. Each of the images we use contains 512×512 pixels, and each pixel has 256 gray levels. In this paper, the parameters for our experiments are stated as follows. The quantization factor of the sampling points is 8, Thresholds for the skewness and radii for the forbidden areas have been set to the following values: $\theta_s=[0.125,0.38,0.5]$, $r=[12,8,4,1]$. The magnitude of adjustment α is 3 and the threshold T_s for tamper detection is 20. Figure 3 is an example of tamper detection and recovery using the test image Girl. Figure 3(b) is the watermarked image, where the PSNR value is 42.01. We can see that the difference between the original image and the watermarked image cannot be detected by the naked eye. Figure 3(c) shows the modified image where the tamper is edited using PhotoShop software. Figures 3(d) and 3(e) are the results of tamper detection and recovery, respectively. In Figure 3(d), we see that even if the modification is subtle, such as the removal of the white candle within the white table, the detection and recovery are also correct.

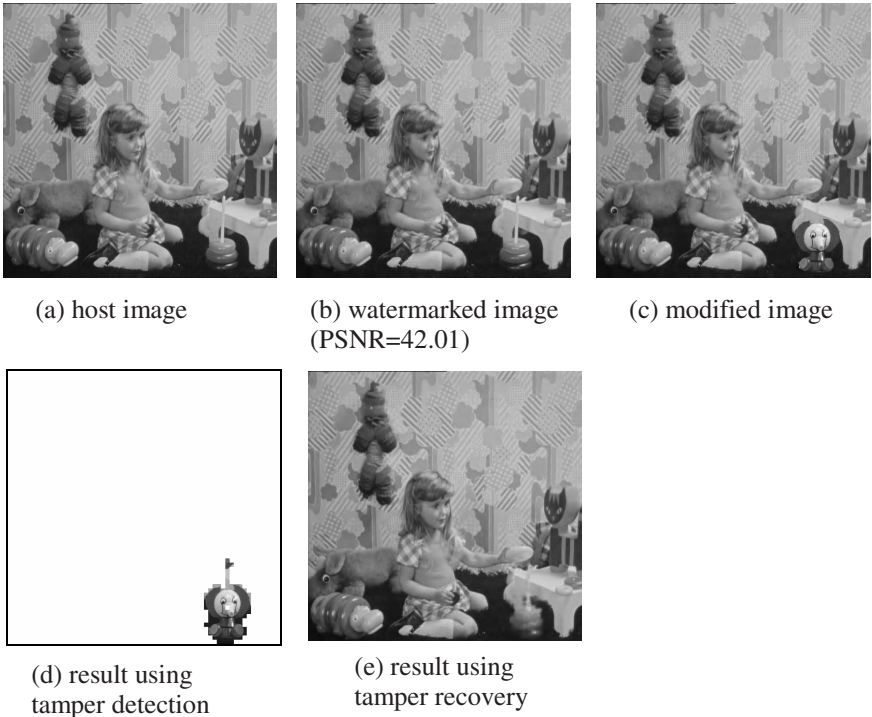


Fig. 3. Experimental results of tamper detection and recovery for image Girl

In order to test the recovery ability of our method on large amounts of modification, we experimented with the cropping ratio, where the cropped area was increased from the center of the image to the image borders, of embedded images. The

experimental results are shown in Figure 4 and Table 1. In the figure, we see that the modifications can all be completely detected and that a rough image can be correctly recovered. Even though the image is 80% cropped, the quality of the recovered image is acceptable.

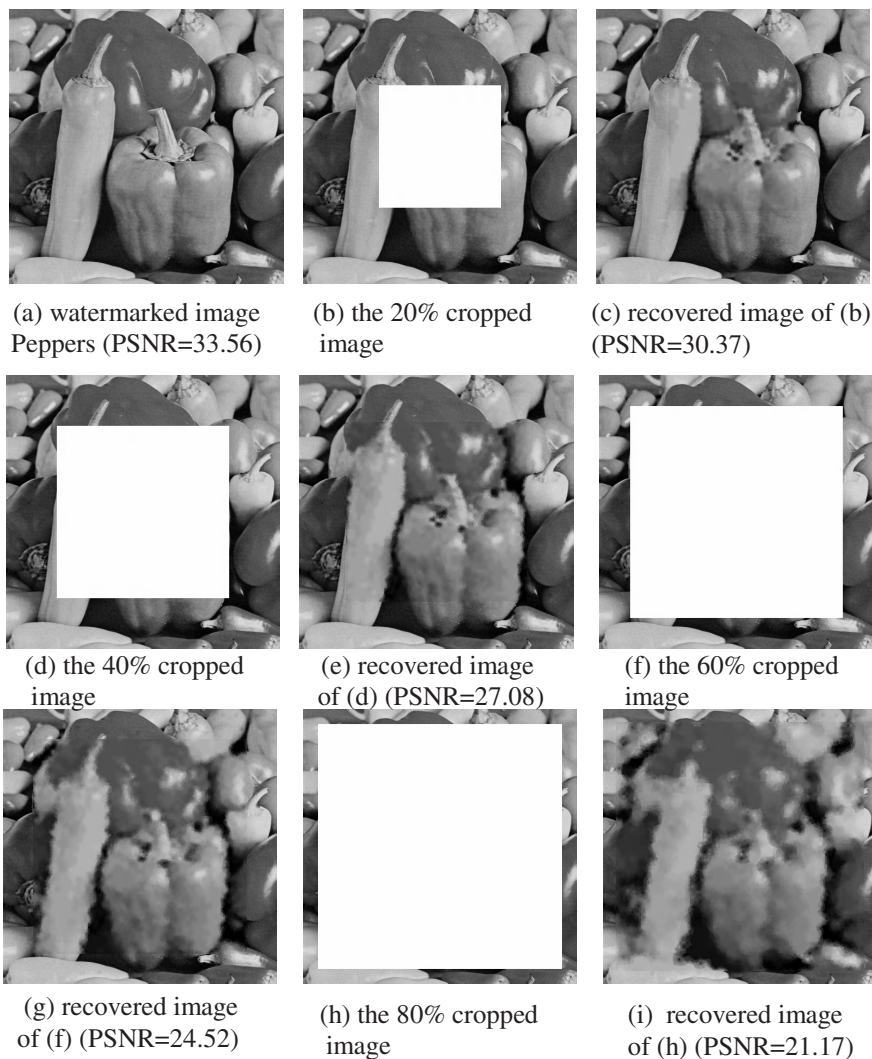


Fig. 4. The experimental results of cropping

With regards to the *differentiation* requirement introduced in Section 1, we also conducted experiments to show the degree of toleration (t) of the proposed method under JPEG compression. The experiments were performed using different magnitudes of α with differing degrees of JPEG compression. The experimental results are listed in Table 2.

Table 1. The recovered PSNR values of the images Lena and Baboon with different cropping area. The cropping area increases from the center of the image to the image borders.

	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%
Lena	41.15	31.89	29.83	27.99	26.60	25.80	24.19	22.47	20.31	18.34
Baboon	41.20	33.87	29.69	27.07	25.59	24.97	22.86	20.89	19.13	18.06

Table 2. The PSNR values of the embedded images, the JPEG compressed images, and the recovered images under different magnitudes of α

	Lenna				Baboon			
	$\alpha=3$	$\alpha=4$	$\alpha=6$	$\alpha=8$	$\alpha=3$	$\alpha=6$	$\alpha=8$	$\alpha=12$
PSNR(dB) of the embedded image	42.05	39.66	36.19	33.58	41.93	36.24	33.75	30.22
PSNR of JPEG (1:2)	41.18	39.12	35.93	33.43	38.40	35.15	33.01	29.88
Recovery PSNR of JPEG(1:2)	30.21	38.04	35.93	33.43	23.32	35.15	33.01	29.88
PSNR of JPEG (1:4)	39.48	38.11	35.34	33.16	32.69	31.54	30.64	28.54
Recovery PSNR of JPEG(1:3)	23.25	35.67	35.22	33.16	13.19	22.20	28.41	28.30
PSNR of JPEG (1:6)	38.39	37.04	34.93	32.78	-	28.87	28.15	26.58
Recovery PSNR of JPEG(1:5)	15.77	23.45	29.94	32.78	-	11.65	15.06	21.21
PSNR of JPEG (1:8)	-	35.37	33.54	32.16	-	-	-	25.09
Recovery PSNR of JPEG(1:7)	-	12.74	15.31	24.67	-	-	-	13.82
PSNR of JPEG (1:10)	-	-	-	30.87	-	-	-	-
Recovery PSNR of JPEG(1:10)	-	-	-	12.94	-	-	-	-

6 Conclusions

In this paper, we proposed a new recoverable image authentication technique. It employs a very low bit-rated compression method, called irregular sampling coding, to compress an image. The experiments showed that the proposed technique can effectively detect and recover a modified image. Moreover, the proposed technique has been proven to be robust. For example, under a 90% cropping operation or a high JPEG compression, the technique can still properly detect and recover a modified image. The proposed method is, therefore, an effective, robust, and recoverable tamper proofing technique.

References

1. Eldar, Y., Lindenbaum, M., Porat, M., Zeevi, Y.Y.: The farthest point strategy for progressive image sampling. *IEEE Trans. Image Processing* 6(9), 1305–1315 (1997)
2. Hsu, C.T., Wu, J.L.: Hidden digital watermarks in images. *IEEE Trans. on Image Processing* 8(1), 58–68 (1999)
3. Hung, K.L., Chang, C.C., Chen, T.S.: Secure discrete cosine transform based technique for recoverable tamper proofing. *Optical Engineering* 40(9), 1950–1958 (2001)
4. Kundur, D., Hatzinakos, D.: Digital watermarking for telltale tamper proofing and authentication. *Proceedings of the IEEE* 87(7), 1167–1180 (1999)
5. Ramponi, G., Carato, S.: An adaptive irregular sampling algorithm and its application to image coding. *Image and Vision Computing* 19, 451–460 (2001)
6. Lu, C.S., Liao, H.Y.M.: Multipurpose watermarking for image authentication and protection. *IEEE Trans. Image Processing* 10(10), 1579–1592 (2001)
7. Schneider, M., Chang, S.-F.: A robust content based digital signature for image authentication. In: *Proc. IEEE Int. Conf. Image Processing* vol. 3, pp. 227–230 (1996)
8. Swanson, M.D., Zhu, B., Tewfik, A.H.: Transparent robust image watermarking. In: *Proc. ICIP'96*, pp. 211–214 (1996)
9. Walton, S.: Image authentication for a slippery new age. *Dr. Dobb's J.* 20, 18–26 (1995)
10. Wolfgan, R.B., Delp, E.J.: A watermark for digital images. In: *Proc. IEEE Int. Conf. Image Processing*, vol. 3, pp. 219–222 (1996)
11. Tsai, C.S., Chang, C.C., Chen, T.S., Chen, M.H.: Embedding robust gray-level watermarks in an image using discrete cosine transform, to appear in *Distributed Multimedia Database: Techniques and Applications*
12. Fridrich, J., Goljan, M.: Protection of digital images using self embedding, *Symposium on Content Security and Data Hiding in Digital Media*, New Jersey Institute of Technology (May 14, 1999)
13. Fridrich, J., Goljan, M.: Images with self-correcting capabilities. In: *ICIP'99*, Kobe, Japan, pp. 25–28 (1999)
14. Chae, J.J., Manjunath, B.S.: A technique for image data hiding and reconstruction without host image. In: *Proceedings of the SPIE – The International Society for Optical Engineering*, vol. 3657 (*Security and Watermarking of Multimedia Contents*), San Jose, CA, USA, pp. 386–396 (1999)
15. Mobasser, B.G., Evans, A.T.: Content-dependent video authentication by self-watermarking in color space, *Security and Watermarking of Multimedia Contents III*. In: *SPIE Proceedings*, vol. 4314, pp. 35–44

A Decomposition Strategy Based Trusted Computing Method for Cooperative Control Problem Faced with Communication Constraints

Shieh-Shing Lin

Department of Electrical Engineering, Saint John's University
499, Sec. 4, Tam King Road, Tamsui, Taipei, Taiwan
sslin@mail.sju.edu.tw

Abstract. In this paper, we propose a decomposition strategy based computing method to solve a cooperative control problem. The test results show that the proposed method has computational efficiency with respect to the conventional approach of the centralized Newton method.

1 Introduction

There are many practical systems that must cooperate when their common objectives are defined and each member has the information needed to cooperate, even when there are communications difficulties faced with communication constraints, for example, teams of unmanned air vehicles (UAVs). Cooperation problems for ground robots and UAVs share a number of similarities. Both ground and aerial robots have strict communication constraints: team members must be in close physical proximity to communicate, bandwidth is limited, and the communication topology may change unpredictably with time. Another similarity is that decentralized cooperation strategies are generally required for both ground and aerial robots. In addition, cooperation strategies must be robust to the failure of individual team members.

Effective cooperation often requires that individuals coordinate their actions. Coordination can take many forms ranging from staying out of each others' way to directly assisting another individual. In general, group cooperation is facilitated by coordinating the actions of individuals. However, each individual may not necessarily need to directly coordinate with every other individual in the group to effect group cooperative behavior. For example, fish engaged in schooling behavior only react to other fish that are in close physical proximity. We will term this type of coordination local coordination. Due to communication constraints and computational feasibility, we are primarily interested in group cooperation problems where the coordination occurs locally. One of the interesting challenges in robotics is to design coordination strategies so that local coordination will result in group cooperation.

One approach for handling cooperative timing is to apply timing constraints to the task assignment problem. In [1], mixed-integer linear programming (MILP) is used to solve tightly coupled task assignment problems with timing constraints. The advantage to this approach is that it yields the optimal solution for a given problem. The

primary disadvantages are the complexity of problem formulation and the computational burden involved. In [2], a decentralized optimization method based on a bargaining algorithm is developed and applied to a multiple aircraft coordination problem.

The objective of this paper is to present a method for cooperation problems, such as the unmanned air vehicles (UAVs), multi-commodity network flow (NMNF),..., etc.. We propose a method to decompose the nonlinear constrained optimization problems in cooperative control problem into some small scale sub-problems and solve the decomposed sub-problems efficiently. We do not claim that our approach will be appropriate for all cooperation problems. In fact, it will be many more years before the general principles underlying cooperative systems will be fully understood. However, we hope that our approach contributes toward that goal.

The considered nonlinear constrained optimization problems (NCOPs) in cooperative control system are stated in the following:

$$\min_x J(x) \quad (1a)$$

subject to

$$h(x) = 0 \quad (1b)$$

$$g(x) \leq 0 \quad (1c)$$

where $J(x)$ denotes the nonlinear objective function of the variable x ; the nonlinear equality constraints (1b) are balance constraints associated with network arcs and/or joints; the communication inequality constraints (1c) denote the coupled inequality constraints associated with network arcs and/or joints. There were numerous optimization algorithms for solving the NCOPs (1a)-(1c), such as Successive Quadratic Programming methods [3], [4], centralized Newton methods [5], [6], reduced gradient methods [7]. To solve these complicated nonlinear optimization problems in a cooperative control system using a more exact approach, the efficient Epsilon Decompositions algorithm were presented in [8], [9]; and some specified capacities are used to describe the relations between the two different joints connections and to formulate the problem formulation of the whole joints connections as a matrix in the network diagram, the algorithm is to eliminate the off-diagonal elements which have magnitude less than or equal some preset criteria value, say $\mathcal{E}$, and form blocks by combining the joints which keep connections after the elimination. Also in the solution process, some necessary permutation of the corresponding rows and columns of the matrix is performed along with the clustering. The parameter $\mathcal{E}$ can be selected according to the designed block numbers or block sizes regarding the magnitude of the off-diagonal elements; they applied the proposed algorithm in many optimization problems and obtained some successful results. The Epsilon Decomposition preserves the merits of weak coupling of the decomposed subsystems if the lower value $\mathcal{E}$ is selected; however, the lower value $\mathcal{E}$ is selected, the higher weak coupling of the decomposed subsystems is expected, it does not guarantee the approximately equal dimensions for the decomposed subsystems. In order to obtain practically good load

balance and approximately equal volume for the subsystems in the parallel computation, in this paper, we propose a decomposition algorithm based optimization method to solve the large scale nonlinear constrained optimization problem in cooperative control system. The paper is organized in the following. Sect. 2 presents the decomposition algorithm based method for solving the large scale nonlinear constrained optimization problem in cooperative control system. The simulation results to demonstrate the computational efficiency are given in Section 3. Finally, Sect. 4 gives a brief conclusion.

2 The Decomposition Algorithm Based Optimization Method

2.1 Diagram Method

First of all, we build the diagram heuristically and the diagram is the connecting lines themselves of the structure in the considered NCOPs in cooperative control system. We denote the large scale diagram by Θ and categorize Θ into four kinds; $\mathcal{A}$: with no loop, $\mathcal{B}$: with few loops and some loops contain many connecting lines, $\mathcal{C}$: with few loops and each loop contains few connecting lines, and $\mathcal{D}$: with many loops. To execute the categorization, we first calculate the number of loops L in Θ and the number of the connecting lines N_{lp} in each loop l_p , $p = 1, \dots, L$. If $L = 0$, then Θ is of kind $\mathcal{A}$; We let L_0 and N_0 denote the criteria value of L and N_{lp} , to classify the kind of diagram Θ . If $L < L_0$ and $N_{lp} \geq N_0$ for some p , then Θ is of kind $\mathcal{B}$. If $L < L_0$ and $N_{lp} < N_0$ for every $p = 1, \dots, L$, then Θ is of kind $\mathcal{C}$. If $L \geq L_0$, then Θ is of kind $\mathcal{D}$.

2.2 Diagram Method Based Decomposition Algorithm

The purpose of the proposed decomposition method is to achieve the decomposed subsystems equal dimension approximately [10]. We use the following notations; n_{ij} : the line connecting of joint i with j ; N_M : a set of the minimum cut of the lines in Θ ; S_d : the d th sub-diagram of Θ ; $S_d(N_M)$: the d th sub-diagram of Θ after the removal of minimum-cut set N_M ; V : the term volume; $V(\vartheta)$: the total volume in ϑ , for a single connecting line n_{ij} and the weighting of this connecting line is set to be $w_{n_{ij}}$ then $V(n_{ij}) = w_{n_{ij}}$. However, we can assign the weighting of each connecting line depended on the importance of it in the cooperative control system; V_0 : a criterion value.

Decomposition Algorithm

- Step 1:** Given the connecting diagram Θ of the cooperative control system, criteria values L_0, N_0 , and V_0 . Pick up two geometrically outermost joints as a source/sink. Assign the weighting $w_{n_{ij}}, \forall n_{ij} \in \Theta$.
- Step 2:** Identify a directed spanning tree starting from the source joint.
- Step 3:** Calculate L and N_{lp} of each loop $l_p, p = 1, \dots, L$. If $L = 0$, then *Step 4* (kind $\mathcal{A}$); if $L < L_0$ and $N_{lp} \geq N_0$ for some p , then *Step 5* (kind $\mathcal{B}$); if $L < L_0$ and $N_{lp} < N_0$, for every $p = 1, \dots, L$, then *Step 6* (kind $\mathcal{C}$); if $L \geq L_0$, then *Step 7* (kind $\mathcal{D}$).
- Step 4:** (kind $\mathcal{A}$); Search $n^\times = \arg\{\min_{n_{ij} \in \Theta} |V(S_{d1}(n_{ij})) - V(S_{d2}(n_{ij}))|\}$. The connecting line $n^\times$ is set as a minimum-cut used to decompose the two sub-diagrams, S_{d1} and S_{d2} . Go to *Step 8*.
- Step 5:** (kind $\mathcal{B}$); Set a connecting line as a minimum-cut from each large loop and use a broken dashed line to represent that line, and then execute the same procedures as in *Step 4*.
- Step 6:** (kind $\mathcal{C}$); Represent each loop l_p by one connecting line and assign that line with a volume of $V(l_p)$, then execute the same procedures as in *Step 4*.
- Step 7:** (kind $\mathcal{D}$); Identify all directed paths from the source joint to the primary encountered loops. Remove the connecting lines in each of those paths and add their volumes uniformly to the corresponding loop, then execute the same procedures as in *Step 6*.
- Step 8:** If $V(S_d) \leq V_0$ for the decomposed sub-diagram S_d , pass S_d ; otherwise, repeat *Steps 1 to 7* for S_d .

2.3 Parallel Duality Based Computing Method

The NCOPs (1a)-(1c) can be decomposed into the following n sets sub-problems.

$$\min_x \sum_{i=1}^n J_i(x) \quad (2a)$$

subject to

$$h_i(x) = 0 \quad (2b)$$

$$g_i(x) \leq 0 \quad (2c)$$

Successive Quadratic programming (SQP) method

The SQP method uses the following iterations to solve the (2a)-(2c),

$$x_i(k+1) = x_i(k) + \alpha(k) \Delta x_i^*(k), \quad i = 1, \dots, n \quad (3)$$

where $\alpha(k)$ is a weighting determined by [11], and $\Delta x_i^*(k)$ is the solution of the following QP sub-problems:

$$\min_{\Delta x} \sum_{i=1}^n \frac{1}{2} \Delta x_i^T D_{ii} \Delta x_i + \nabla_{x_i} J_i(x)^T \Delta x_i \quad (4a)$$

subject to

$$h_i(x(k)) + \nabla_{x_i} h_i(x)^T \Delta x_i = 0 \quad (4b)$$

$$g_i(x(k)) + \nabla_{x_i} g_i(x)^T \Delta x_i \leq 0 \quad (4c)$$

where $D_{ii} = \text{diag}[\nabla_{x_i}^2 J_i(x) + \delta/2 I]$. δ is a scalar but large enough to make D_{ii} positive definite, I is an identity matrix, $\Delta x^T = [\Delta x_1^T, \dots, \Delta x_n^T]$. Setting the set,

$\Omega_i = \{\Delta x_i \mid g_i(x(k)) + \nabla_{x_i} g_i(x)^T \Delta x_i \leq 0, \}$ and $\Omega = \cup_{i=1}^n \Omega_i$, we can re-write (4a)-(4c) as

$$\min_{\Delta x} \sum_{i=1}^n \frac{1}{2} \Delta x_i^T D_{ii} \Delta x_i + \nabla_{x_i} J_i(x)^T \Delta x_i \quad (5a)$$

subject to

$$h_i(x(k)) + \nabla_{x_i} h_i(x)^T \Delta x_i = 0 \quad (5b)$$

$$\Delta x_i \in \Omega_i, \quad i = 1, \dots, n. \quad (5c)$$

Parallel duality based Computing method

The dual problem of the QP sub-problems (5a)-(5c) is

$$\max_{\lambda} q(\lambda) \quad (6)$$

where the dual function

$$\begin{aligned} q(\lambda) = \min_{\Delta x \in \Omega} \sum_{i=1}^n \frac{1}{2} \Delta x_i^T D_{ii} \Delta x_i + \nabla_{x_i} J_i(x)^T \Delta x_i \\ + \lambda_i^T [h_i(x(k)) + \nabla_{x_i} h_i(x)^T \Delta x_i] \end{aligned} \quad (7)$$

The parallel duality based computing method uses the following iterations to solve (6)

$$\lambda_i(t+1) = \lambda_i(t) + \beta(t) \Delta \lambda_i(t), \quad i = 1, \dots, n \quad (8)$$

where $\beta(t)$ is a weighting determined by [11], increment of Lagrange multiplier $\Delta\lambda^T(t) = [\Delta\lambda_1^T(t), \dots, \Delta\lambda_n^T(t)]$ is the solution of the QP problem of (6) at $\lambda(t)$:

$$\max_{\Delta\lambda} \frac{1}{2} \Delta\lambda^T Q \Delta\lambda + \nabla_{\lambda} q^T \Delta\lambda. \quad (9)$$

The matrix Q in (9) is given by

$$Q = \begin{bmatrix} Q_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & Q_n \end{bmatrix} \quad (10)$$

where the diagonal block sub-matrix Q_i can be obtained by

$$Q_i = -\nabla_{x_i} h_i(x)^T D_{ii}^{-1} \nabla_{x_i} h_i(x), \quad (11)$$

The derivative of the dual function, $\nabla_{\lambda} q$, in (9) can be expressed as

$\nabla_{\lambda} q(\lambda)^T = [\nabla_{\lambda_1} q(\lambda)^T, \dots, \nabla_{\lambda_n} q(\lambda)^T]$, and can be computed by

$$\nabla_{\lambda_i} q(\lambda) = h_i(x(k)) + \nabla_{x_i} h_i(x)^T \Delta\hat{x}_i(\lambda(t)), \quad (12)$$

where $\Delta\hat{x}^T$, is the solution of (7) [12], [13]. The $\Delta\lambda(t)$ can be obtained by solving the following optimal necessary condition of (9) [11]

$$Q \Delta\lambda(t) = -\nabla_{\lambda} q(\lambda), \quad (13)$$

which can be decomposed into the following n independent sets of linear equations

$$Q_i \Delta\lambda_i(t) = -\nabla_{\lambda_i} q(\lambda), i = 1, \dots, n. \quad (14)$$

These n sets of (14) can be parallel executed if each $\nabla_{\lambda_i} q(\lambda)$ is obtained; we can use the Two-stage algorithm [12], [13] to obtain $\Delta\hat{x}^T$ and form $\nabla_{\lambda_i} q(\lambda)$ in (12).

The method for solving NCOPs in Cooperative Control System

Our method for solving the NCOPs in cooperative control system is using the Decomposition Algorithm to decompose the large scale NCOPs (1a)-(1c) into n sets of sub-problems (2a)-(2c) and using the SQP (3) to solve (2a)-(2c) where $\Delta x^*(k)$ is the solution of QP sub-problems (4a)-(4c). The parallel duality based computing method uses (8) to solve (6). The $\Delta\lambda_i(t)$ in (8) is obtained from solving (14). The $\Delta\hat{x}_i$

in (12) is needed to set up $\nabla_{\lambda} q(\lambda)$ and can be computed using the two-stage algorithm.

3 Simulation

We set the following parameters: $L_0 = 5$, $N_0 = 10$, $V_0 = 32, 16$, $\delta = 0.1$ in our experiment. Based on the choice of the different values V_0 , we use the Decomposition algorithm to decompose the system into *five* ($V_0 = 32$) and *ten* ($V_0 = 16$) subsystems. We made two types of test. The first one is assuming no communication inequality constraint in the NCOPs; and there are four different cases in this type of tests including different number of equality constraints. The second is assuming there are some communications constraints in the NCOPs including four different numbers of communications constraints; the rest of the corresponding data are the same as the first one tests. The experimental computer is a single PC (Pentium 4) and has a CPU processor speed of 3.2GHzs and 512 Mbytes of RAM memory. We tested more than 20 examples in each case of NCOPs. To verify the efficiency of our method, we made a comparison with the conventional approach method, the centralized Newton method [11]. We used our algorithm and the centralized Newton method to solve the same test examples in each case of the NCOPs described above with the same initial condition and termination criteria. The test results show that our algorithm is efficient than the centralized Newton method in *five* and *ten* subsystems. Furthermore, the efficiency is more significant while the communication inequality constraints are faced in *five* and *ten* subsystems, respectively. This addresses that our method is efficient for handling the NCOPs faced with communication inequality constraints in cooperative control system.

4 Conclusion

In this paper, we presented a decomposition algorithm based optimization method to solve a cooperative control optimization problem. We made numerous simulations and obtained some successful results to demonstrate the computational efficiency of our algorithm with respect to the conventional approach of the centralized Newton method in solving quiet a few examples of NCOPs found in cooperative control problem.

References

1. Bellingham, J., Tillerson, M., Richards, A.: How, Multi-task allocation and path planning for cooperating UAVs. In: Cooperative Control: Models, Applications Algorithms. ch. 2, Kluwer, Boston (2003)
2. Inalhan, G., Stipanovic, D.M., Tomlin, C.J.: Decentralized optimization with application to multiple aircraft coordination. In: Proc. IEEE Conf. Decision Control, Las Vegas, NV, pp. 1147–1155 (2002)

3. Burchett, H., Happ, H., Vierath, D.R.: Quadratically Convergent Optimal Power Flow. *IEEE Trans. Power Appar. Syst.* PAS-104(11), 3267–3275 (1985)
4. Giras, T.C., Talukdar, S.N.: Quasi-Newton Method for Optimal Power Flows. *Int. J. Electr. Power Energy Syst.* 3(2), 59–64 (1981)
5. Sun, D., Ashly, B., Brewer, B.: Optimal Power Flow by Newton Approach. *IEEE Trans. Power Appar. Syst.* PAS-103, 2864–2880 (1984)
6. Monticoll, A., Liu, W.: Adaptive movement penalty method for Newton Optimal Power flow. *IEEE Trans. on Power Syst.* 7, 334–340 (1992)
7. Leventhal, T., Nemhauser, G., Trotter, JR.: A Column Generation Algorithm for Optimal Traffic Assignment. *Trans. Sci.* 7(2), 168–176 (1973)
8. Zecevic, A.I., Siljak, D.D.: A block-parallel Newton method via overlapping epsilon decompositions. *SIAM Journal on Matrix Algebra and Applications* (1994)
9. Sezer, M.E., Siljak, D.D.: Nested epsilon decompositions and clustering of complex system. *Automatica* 22, 321–331 (1991)
10. Gould, R.: *Graph Theory*, Menlo Park, CA: Benjamin/Cummings (1988)
11. Luenberger, D.: *Linear and nonlinear programming*, 2nd edn. Addison-Wesley, London (1984)
12. Lin, C., Lin, S.: A new dual-type method used in solving optimal power flow problems. *IEEE Trans. on Power Syst.* 12(4), 1667–1675 (1997)
13. Lin, S., Lin, C.: A computationally efficient method for nonlinear multicommodity network flow problem. *Network* 225–244 (1997)
14. Lin, S.-Y., Lin, S.-S.: A parallel block scaled gradient method with decentralized step-size for block additive unconstrained optimization problems of large distributed systems. *Asian Journal of Control* 5(1), 104–115 (2003)

Formal Analysis of Secure Bootstrap in Trusted Computing*

Shuyi Chen, Yingyou Wen, and Hong Zhao

School of Information Science and Engineering, Northeastern University, 110004,
Shenyang, China
chen-sy@neusoft.com

Abstract. The stated goal of the trusted computing is to redesign PC hardware for providing protection against software attack. Trust platform is key technology of trusted computing. However, the security of trusted platform should be verified in theory, and the model of trusted platform should be further improved. In this paper, a formal method for verifying the security of trusted platform is presented. The vulnerability of secure bootstrap is analyzed based on the proposed formal semantics. Moreover, an improved model of secure bootstrap is proposed. The semantics presented here also can be used to reasoning about other applications of trusted computing, which provides a general and effective method for analyzing security of trusted computing applications.

1 Introduction

Computer security is undeniably important, and research for protecting computer's security has lasted for many years. However, more and more new vulnerabilities are discovered and exploited, and the number of security incidents ascends every year. Trusted computing technology proposed by TCG (trusted computing group) aims to solve some of today's security problems through hardware changes to the personal computer [1]. Different from traditional security idea, trusted computing not only emphasizes authentication and access control, but also attaches importance to the integrity of system. Trusted computing provides a new method for resolving some of today's security problems through redesigning the PC hardware against software attack.

In technical fields, a famous project of trusted computing is trusted computing platform alliance, or TCPA. It is called TCG now. Besides this, others well-known projects are NGSCB [2, 3], LaGrande Technology [4] and AMD's Secure Execution Mode [5]. They also include research projects such as XOM [6] and Terra [7] etc.

Theory study of trusted computing lags behind the technology development. How to verify if a model is trusted in theory is a significant research work. Martin Abadi provided a logic account of NGSCB [8]. The authentication and access control in

* This work is supported by the national natural science foundation of China under Grant Nos. 60602061 and the national high-tech research and development plan of China under Grant Nos. 2006AA01Z413.

NGSCB were described with logic-based security language. However, the logic-based security language can not be used to verify the integrity of system. Patel [9] and Beth [10] respectively presented theory of trust models based on probability statistics. A. Bondavalli used Markov process to analyze the dependability of systems [11]. These methods mainly were used to analyze the dependability, survivability and reliability of the systems. They are difficult to understand, and are complicated to compute. All these methods are not suitable for analyzing the applications of trusted computing provided by TCG that based on the authenticity and integrity of system.

Predicate logic can be used to model and reason trust relation. U. Maurer [12], Hanane El Bakkali [13], et al depicted predicate calculus logic for representing and reasoning about PKI trust model, which provided a effective method to precisely reason about the authenticity of the public key and the trustworthiness of CA. Therefore, we select predicate logic as the base of formal approach to modeling trusted computing.

Trust chain is one of key technologies of trusted computing. However, the theory of trust chain should be further verified and improved. It is significant to analyze the security of trust chain by providing a concise and precise formal method.

In this paper, Predicate logic is introduced into analyzing trusted computing. A formal semantics based on predicate logic is defined according to the specifications of TCG (trusted computing group). The security of trusted platform is analyzed based on the presented formal semantics and an improved model of trusted platform is proposed.

2 Secure Bootstrap Based on Trusted Computing

Secure bootstrap problem is well-known, and many bootstrap processes have security vulnerabilities. A solution proposed by TCG aims to solve this problem through the hardware changes to the personal computer. TCG advocates using a secure hardware device to verify the boot sequence and authenticate the verification. Such a device could provide assurance even to a remote user or administrator that the OS at least started from a trustworthy state. If an OS security hole is found in the future, the OS can be updated, restarted, and re-verified to start from this trustworthy state. An example of this kind of device is the Trusted Platform Module (TPM).

TPM contains the minimum set of capabilities that are required to be trusted. As shown in figure 1, TPM contains the following components:

- Input/Output (I/O): Allows the TPM to communicate with the rest of the system
- Opt-In: Allows the TPM to be disabled
- Execution Engine: Executes Program Code, performing TPM initialization and measurement taking Platform Configuration Registers (PCRs) maintain state values
- Non-Volatile Storage: Stores long term keys for the TPM
- Platform Configuration Registers (PCRs): Provide state storage
- Random Number Generator (RNG): Used for key generation, nonce creation, etc
- RSA Crypto Engine & Key Generator: Provides RSA functions for signing, encryption/decryption, Creates signing keys, storage keys, etc. (2048 bit)
- Program Code: Firmware for measuring platform devices
- SHA-1 Engine: Used for computing signatures, creating key Blobs, etc

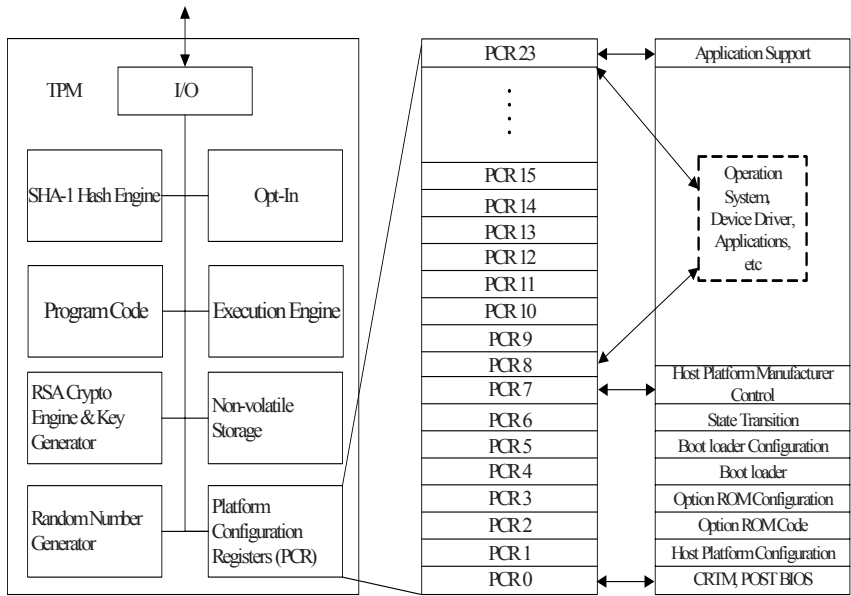


Fig. 1. TPM Component architecture

TPM can be used to verify the integrity of a computing system. Platform boot processes are augmented to allow the TPM to measure each of the components in the system (both hardware and software) and securely store the results of the measurements in Platform Configuration Registers (PCR) within the TPM. Hashes of the bootstrap code, operating system, and applications are stored in the Platform Configuration Registers, which can later be queried to verify what was executed. The values of PCRs are shown in Figure 1.

3 Formal Semantics

TCG uses a behavioral definition of trust: an entity can be trusted if it always behaves in the expected manner for the intended purpose [14]. It can be seen from above definition that trust is context-related. In different context, security policies are different and expectations and intents are various.

Integrity and authenticity are main measurement criteria used to evaluate the trust of an entity. Trusted Computing Group (TCG) has defined a set of standards that describe how to take integrity measurements of a system and store the result in a separate trusted coprocessor (Trusted Platform Module) whose state can not be compromised by a potentially malicious host system. Integrity breaches can be recognized with integrity measurement. Authenticity is the quality or condition of being authentic. It is commonly demonstrated by certificates and related certificate revocation lists. In trusted computing, the problem how to manage certificate can be solved by using standard PKI scheme, or DAA scheme.

Software and hardware in the trusted computing system are regarded as entities in this paper, which is denoted by $E = \{e_1, e_2, e_3 \dots\}$. Predicates are defined to present the relationship between entities, and four inference rules are given to reason about whether an entity is trusted. The predicates and inference rules are summarized in definitions 1 and 2.

Definition 1. Predicates and their representations are one of the following forms:

1) **Integrity.** $Integ(e_1, e_2)$ denotes e_1 's belief that entity e_2 is in its integrity, which means e_2 has not been breached.

2) **Measurement.** $Meas(e_1, e_2, m_2, v_2)$ denotes the fact that e_1 takes an integrity measurement of e_2 to see if that e_2 conforms to the requirement of integrity. Entity e_2 is in its integrity if m_2 is same as v_2 , in which m_2 is the Stored Measurement Log (SML) gained in the measuring time and v_2 is the value reported by TPM to be compared.

3) **Trust for measuring integrity.** $Trust(e_1, e_2, Integ)$ denotes that entity e_1 believes that entity e_2 is trustworthy for measuring integrity.

4) **Trust for issuing certificate.** $Trust(e_1, e_2, Cert)$ denotes that entity e_1 believes that entity e_2 is trustworthy for issuing certificate.

5) **Certificates.** $Cert(e_1, e_2)$ denotes that e_1 has issued a certificate to e_2 .

6) **Authenticity.** $Auth(e_1, e_2)$ denotes e_1 's belief that a certificate(i.e., belongs to entity e_2) is authentic. Therefore, e_2 's identity is authentic.

7) **Trusted.** $Trusted(e_1, e_2, E)$ denotes e_1 's belief that e_2 is trusted, and e_2 behaves in the expected manner of e_1 's. e_1 's exception is presented as E. E's value can be integrity, authenticity or both. For example, $Trusted(e_1, e_2, Integ \wedge Auth)$ denotes e_1 's belief that entity e_2 is in its integrity, and e_2 's identity is authentic.

Definition 2. In our semantics, a statement is valid if that it is contained in the predicates defined above or it can be derived from applying the following inference rules.

R1. Integrity rule (direct)

$$\forall e_1, e_2, e_3 \in E \quad Meas(e_1, e_2, m_2, v_2) \vdash Integ(e_1, e_2)$$

R2. Integrity rule (indirect)

$$\forall e_1, e_2, e_3 \in E \quad Trust(e_1, e_2, Integ), Meas(e_2, e_3, m_3, v_3) \vdash Integ(e_1, e_3)$$

R3. Authenticity rule

$$\forall e_1, e_2, e_3 \in E \quad Trust(e_1, e_2, Cert), Cert(e_2, e_3) \vdash Auth(e_1, e_3)$$

R4. Trusted rule

$$\begin{aligned} \forall e_1, e_2, e_3 \in E \quad & Integ(e_1, e_2) \vdash Trusted(e_1, e_2, Integ) \\ & Auth(e_1, e_2) \vdash Trusted(e_1, e_2, Auth) \\ & Integ(e_1, e_2) \wedge Auth(e_1, e_2) \vdash Trusted(e_1, e_2, Integ \wedge Auth) \end{aligned}$$

Rule 1 and Rule 2 are for deriving statements about the integrity of the entities. In rule 1, entity e_1 directly takes integrity measurement of entity e_2 and obtains the statement $Integ(e_1, e_2)$. The integrity is derived indirectly in Rule 2, in which entity e_1

derives the statement $Integ(e_1, e_3)$ through e_2 . Entity e_1 believes that the entity e_2 is trustworthy for measuring integrity and entity e_2 takes integrity measurement of entity e_3 .

Rule 3 is for deriving statements about the authenticity of public keys. It denotes that if entity e_1 trusts that entity e_2 is trustworthy for issuing certificate, and entity e_2 issues a certificate to entity e_3 . Then entity e_1 can derive the authenticity of e_3 after entity e_2 verifying that e_3 's certificate is validity.

Rule 4 is for deriving statements about trusted according to the expectation. It denotes that if entity e_1 's exception for e_2 is satisfied, then e_1 believes that e_2 has the properties defined by e_1 's exception. In different applications, the exceptions relate to different entities can be various, which mainly are integrity, authenticity or both.

Analyzing a trusted computing model in our formal semantics consists of two steps:

- 1) Formalize initial conditions and conclusion, Find suitable assumptions;
- 2) Reason out conclusion by applying the inference rules defined.

4 Formal Semantics of Secure Bootstrap

According to the definition of TCG, a trusted platform is a computing platform that can be trusted to report its properties. Trusted platform should provide at least three basic features: protected capabilities, integrity measurement and integrity reporting.

Trusted Building Blocks (TBB) are the parts of the roots of trust that do not have shielded locations or protected capabilities. Roots of trust are components that must be trusted because misbehavior might not be detected. There are commonly three roots of trust in a trusted platform; a root of trust for measurement (RTM), root of trust for storage (RTS) and root of trust for reporting (RTR). The combination of TBB and roots of trust form a trust boundary where measurement, storage and reporting can be accomplished for a minimal configuration. Typically the normal platform computing engine is controlled by core root of trust for measurement (CRTM).

4.1 System Bootstrap Based on Transitive Trust

Transitive trust also known as "Inductive Trust", is a process where the Root of Trust gives a trustworthy description of a second group of functions. Based on this description, an interested entity can determine the trust it is to place in this second group of functions. If the interested entity determines that the trust level of the second group of functions is acceptable, the trust boundary is extended from the Root of Trust to include the second group of functions. In this case, the process can be iterated. The second group of functions can give a trustworthy description of the third group of functions, etc. Transitive trust is used to provide a trustworthy description of platform characteristics.

In Figure 2, transitive trust is applied to a system booting from a static root of trust and the trust boundary is extended to include code that didn't natively reside within the roots of trust. In each extension of the trust boundary, the target code is

first measured before execution control is transferred. After completing the measurement and transitive trust, trust boundary is extended to the whole platform, and the platform is converted to a trusted platform.

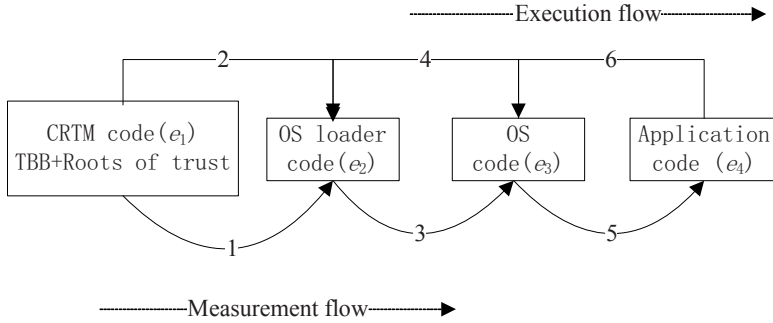


Fig. 2. Transitive trust applied to system boot from a static root of trust

Here we present the interested entity as e_0 , which determines if an entity is trust according to the measurements. After completing the measurement and transitive trust, conclusions about whether a platform is trusted are derived from e_0 's view.

We will analyze the process converting a platform into trusted platform according to above steps. The only trusted component viewed from e_0 is root of trust when system boot. e_0 believes that entity e_1 is in its integrity, and entity e_1 is trustworthy for measuring integrity.

$$\begin{aligned} & Trust(e_0, e_1, Integ) \\ & Trusted(e_0, e_1, Integ) \end{aligned} \quad (1)$$

System is measured based on system integrity. According to above description, platform trusted means all of the entities in the platform are trusted viewed form e_0 , therefore the conclusion can be described as following statement.

$$Trusted(e_0, e_1, Integ) \wedge Trusted(e_0, e_2, Integ) \wedge Trusted(e_0, e_3, Integ) \wedge Trusted(e_0, e_4, Integ)$$

According to the definition of trust transitive, an entity is trustworthy for measuring integrity after it is included in trust boundary. For example, if e_0 believes that entity e_n is in its integrity and can be included in trust boundary, then e_0 believes that e_n is trustworthy for measuring integrity. We make following assumption based on trust transitive.

$$Integ(e_0, e_n) \vdash Trust(e_0, e_n, Integ) \quad (2)$$

The steps of system boot based on transitive trust can be reasoned about with our predicate calculus as follow.

1) Entity e_0 believes that entity e_1 (root of trust) is trustworthy for measuring integrity. Entity e_1 holds the execution control and takes an integrity measurement of e_2 (OS loader code) to see if the trust level of e_2 is acceptable.

$$Trust(e_0, e_1, Integ), Meas(e_1, e_2, m_2, v_2) \vdash Integ(e_0, e_2) \vdash Trusted(e_0, e_2, Integ) \quad (3)$$

If above statement is true, then e_0 believes that entity e_2 is in its integrity. According to the assumption statement (2), e_0 holds that e_2 is trustworthy for measuring integrity.

$$Trust(e_0, e_2, Integ)$$

2) Entity e_1 transfers execution control to e_2 .

3) Entity e_2 holds the execution control and takes an integrity measurement of e_3 (OS code) to see if the trust level of e_3 is acceptable.

$$Trust(e_0, e_2, Integ), Meas(e_2, e_3, m_3, v_3) \vdash Integ(e_0, e_3) \vdash Trusted(e_0, e_3, Integ) \quad (4)$$

If above statement is true, then e_0 believes that entity e_3 is in its integrity, and according to the assumption statement (2), e_0 holds that e_3 is trustworthy for measuring integrity.

$$Trust(e_0, e_3, Integ)$$

4) Entity e_2 transfers execution control to e_3 .

5) Entity e_3 holds the execution control and takes an integrity measurement of e_4 (Application code) to see if the trust level of e_4 is acceptable.

$$Trust(e_0, e_3, Integ), Meas(e_3, e_4, m_4, v_4) \vdash Integ(e_0, e_4) \vdash Trusted(e_0, e_4, Integ) \quad (5)$$

If above statement is true, then e_0 believes that entity e_3 is in its integrity.

6) Entity e_3 transfers execution control to e_4 .

We can derive the conclusion statement from statements (1), (3), (4) and (5).

$$Trusted(e_0, e_1, Integ) \wedge Trusted(e_0, e_2, Integ) \wedge Trusted(e_0, e_3, Integ) \wedge Trusted(e_0, e_4, Integ)$$

The integrity of platform can be derived from the root of trust through transitive trust. The platform is trusted based on the result of integrity measurement.

4.2 An Improved Model of Secure Bootstrap

In above model, trust loss occurs in the process of transitive trust. The assumption transferring measurement control to next entity is vulnerable. For example, BIOS, operating system and application software are controlled by the core technology manufacturers. There may be back door, vulnerability and abuse in the software, which may result that these entities can not correctly take integrity measurement,

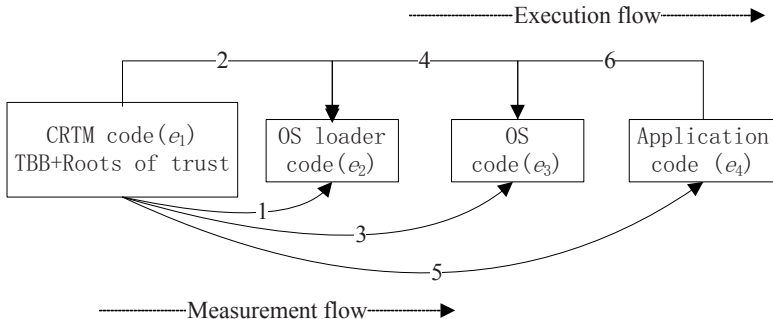


Fig. 3. Improved model of secure bootstrap

therefore integrity measurement taken by these entities may be unreliable. Trust loss in process of transitive trust will increase following with the extending of transfer chain.

We provide an improved model of system bootstrap based on direct measurement. All of the integrity measurements are taken by e_1 , the only trusted component when system boots. The process of system bootstrap based on direct measurements is shown in figure 3.

The initial conditions and conclusion same as that of normal system bootstrap, which shows as follow.

$$\begin{aligned} & Trust(e_0, e_1, Integ) \\ & Trusted(e_0, e_1, Integ) \end{aligned}$$

$$Trusted(e_0, e_1, Integ) \wedge Trusted(e_0, e_2, Integ) \wedge Trusted(e_0, e_3, Integ) \wedge Trusted(e_0, e_4, Integ) \quad (6)$$

The steps of improved system boot based on transitive trust can be reasoned about with our predicate calculus as follow.

- 1) Entity e_0 believes that entity e_1 is trustworthy for measuring integrity. Entity e_1 holds the execution control and takes an integrity measurement of e_2 to see if the trust level of e_2 is acceptable.

$$Trust(e_0, e_1, Integ), Meas(e_1, e_2, m_2, v_2) \vdash Integ(e_0, e_2) \vdash Trusted(e_0, e_2, Integ) \quad (7)$$

If above statement is true, then e_0 believes that entity e_2 is in its integrity.

- 2) Entity e_1 transfers execution control to e_2 .
- 3) Entity e_2 holds the execution control. Entity e_0 takes an integrity measurement of e_3 to see if the trust level of e_3 is acceptable.

$$Trust(e_0, e_1, Integ), Meas(e_1, e_3, m_3, v_3) \vdash Integ(e_0, e_3) \vdash Trusted(e_0, e_3, Integ) \quad (8)$$

If above statement is true, then e_0 believes that entity e_3 is in its integrity.

- 4) Entity e_2 transfers execution control to e_3 .
- 5) Entity e_3 holds the execution control. Entity e_0 takes an integrity measurement of e_4 to see if the trust level of e_4 is acceptable.

$$Trust(e_0, e_1, Integ), Meas(e_1, e_4, m_4, v_4) \vdash Integ(e_0, e_4) \vdash Trusted(e_0, e_4, Integ) \quad (9)$$

If above statement is true, then e_0 believes that entity e_4 is in its integrity.

- 6) Entity e_3 transfers execution control to e_4 .

We can derive the conclusion statement from statements (6), (7), (8) and (9).

$$Trusted(e_0, e_1, Integ) \wedge Trusted(e_0, e_2, Integ) \wedge Trusted(e_0, e_3, Integ) \wedge Trusted(e_0, e_4, Integ)$$

The integrity of platform can be derived based on direct measurement of root of trust.

In improved model, integrity measurements of entities on platform all are taken by e_1 , root of trust. Trust loss can be avoided by direct measurement.

5 Conclusion

In this paper, we provide a formal method based predicate logic for modelling trusted computing. System bootstrap is analyzed based on formal semantics provided, which shows that trusted computing applications can be exactly formalized and verified with predicate logic.

Comparing with logic-based security language, our method is more generic, which can be used to formalize authentication and integrity of trusted system. Moreover, formal semantics based on predict logic is more concise than the methods that based on probability statistics and Markov process.

References

1. TCG. Trusted Computing Group (2004)
https://www.trustedcomputinggroup.org/downloads/background_docs/TCG_Backgrounder_November_2004.pdf
2. Microsoft. Next-Generation Secure Computing Base home page (2006)
<http://www.microsoft.com/resources/ngscb>
3. Peinado, M., Chen, Y., England, P., et al.: NGSCB: A trusted open system. In: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP 2004. LNCS, vol. 3108, pp. 86–97. Springer, Heidelberg (2004)
4. Intel. LaGrande Technology Architectural Overview (2006)
http://www.intel.com/technology/security/downloads/LT_Arch_Overview.pdf
5. Alan, Z.: Coming soon to VMware, Microsoft, and Xen: AMD Virtualization Technology Solves Virtualization Challenges (2006) <http://www.devx.com/amd/Article/30186>
6. Lie, D., Thekkath, C., Mitchell, M., et al.: Architectural support for copy and tamper resistant software. In: William, E., et al. (eds.) ASPLOS-IX 2000[C]. Operating Systems Review, vol. 34, pp. 168–177. ACM Press, New York (2000)
7. Garfinkel, T., Pfaff, B., Chow, J., et al.: Terra: A virtual machine-based platform for trusted computing. In: Birman, K., et al. (eds.) SOSP 2003 [C]. Operating Systems Review, vol. 37, pp. 193–206. ACM Press, New York (2003)
8. Abadi, M., Wobber, T.: A Logical Account of NGSCB. In: de Frutos-Escrig, D., Núñez, M. (eds.) FORTE 2004. LNCS, vol. 3235, pp. 1–12. Springer, Heidelberg (2004)
9. Patel, J., Teacy, W.T., Jennings, N.R., et al.: A Probabilistic Trust Model for Handling Inaccurate Reputation Sources. In: Herrmann, P., Issarny, V., Shiu, S.C.K. (eds.) iTrust 2005. LNCS, vol. 3477, pp. 193–209. Springer, Heidelberg (2005)
10. Beth, T., Borchert, M., Klein, B.: Valuation of Trust in Open Network. In: Gollmann, D. (ed.) Computer Security - ESORICS 94. LNCS, vol. 875, pp. 509–522. Springer, Heidelberg (1994)
11. Bondavalli, A., Chiaradonna, S., Giandomenico, F.D., et al.: Dependability Modeling and Evaluation of Multiple Phased Systems Using DEEM. In: IEEE Transactions on Reliability, vol. 53, pp. 23–26. IEEE Press, New York (2000)
12. Maurer, U.: Modelling a public-key infrastructure. In: Martella, G., Kurth, H., Montolivo, E., Bertino, E. (eds.) Computer Security - ESORICS 96. LNCS, vol. 1146, pp. 325–350. Springer, Heidelberg (1996)
13. Bakkali, H.E., Kaitouni, B.I.: Predicate calculus logic for the PKI trust model analysis. In: IEEE International Symposium on Network Computing and Applications (NCA 2001), pp. 368–371. IEEE Press, New York (2001)
14. TCG. TCPA Main Specification version 1.1b. (2006)
https://www.trustedcomputinggroup.org/specs/TPM/TCPA_Main_TCG_Architecture_v1_1b.pdf

Calculating Trust Using Aggregation Rules in Social Networks*

Sanguk Noh

School of Computer Science and Information Engineering,
The Catholic University of Korea, Bucheon, Korea
sunoh@catholic.ac.kr

Abstract. As Web-based online communities are rapidly growing, the agents in social groups need to know their measurable belief of trust for safe and successful interactions. In this paper, we propose a computational model of trust resulting from available feedbacks in online communities. The notion of trust can be defined as an aggregation of consensus given a set of past interactions. The average trust of an agent further represents the center of gravity of the distribution of its trustworthiness and untrustworthiness. And then, we precisely describe the relationship between reputation, trust, and average trust through a concrete example of their computations. We apply our trust model to online Internet settings in order to show how trust mechanisms are involved in a rational decision-making of the agents.

1 Introduction

Traditional notion of trust [3] refers to an agent's belief that other agents towards itself intend to be honest and positive, and can be usually built up through direct interactions in person. As online communities on the Internet are rapidly growing, the agents have exposed to virtual interactions as well as face-to-face interactions. The agents in online social networks communicate anonymously and have only limited inspections. These features have made the agents hard to decide whether or not other agents may be positive or benevolent to them. Thus, it is essential that they could have a tangible model of trust for safe and successful interactions, even in the case that they don't have prior and direct interactions. This paper addresses how to assess trust in social networks, particularly applicable to the online community. We build up the computational model of trust as a measurable concept.

Our approach to the computational model of trust starts with the lesson from "Tit for Tat" strategy in game theory for the iterated Prisoner's Dilemma [1], which encourages social cooperation among agents. As a result of mutual behaviors in online multi-agent settings, agents will get more positive feedbacks from other agents if the agents are willing to cooperate with others and, otherwise, they will receive more

* This work has been supported by the Catholic University of Korea research fund, 2006, department specialization fund, 2007, and by the Agency for Defense Development under Grant UD060072FD "A Study on the Multi-Spectral Threat Data Integration of ASE," 2006.

negative feedbacks from others. We translate the feedbacks resulting from social activities into the agent's reputation as a quantitative concept. The next steps for our trust model are to apply aggregation rules to given reputation values to reach a consensus, and calculate the average trust interpreted as the center of gravity of the distributions of trustworthiness and untrustworthiness. The notion of trust in our framework then represents positive expectations about others' future behaviors.

In the following section of this paper, we briefly compare our approach to related research. Section 3 is devoted to our trust model that defines reputation, trust, and average trust. We precisely describe the relationship among them through a concrete example of their computations. In Section 4, we apply our trust model to online Internet transactions showing how trust affects a rational decision-making of buyers and sellers. In the concluding Section 5, we summarize our work and mention further research issues.

2 Related Work

Our work builds on efforts by several other researchers who have made the social concept of trust computable in a society of multi-agents. In the field of multi-agent community, there have been several approaches to support a computational model of trust. Marsh [10] introduces a simple, computational model of trust, which is a subjective real number ranging from -1 to 1. His model has trouble with handling negative values of trust and their propagation. Mui et al. [11] describe trust in a pseudo-mathematical expression and represent it as posteriors using expected utility notation. Their scheme only counts the number of cooperations (or positive events). In a distributed reputation system [8], they use aging factor, distance factor, and new experience to update trust. However, the assumption of these components of trust is not likely to be realistic. As they pointed out, their scheme does not correctly handle negative experiences. Our model of trust represents an aggregation of consensus without any problem of fusion, and effectively deals with the agent's trustworthiness and untrustworthiness in the range of 0 and 1, respectively, which are based on actual positive and negative feedbacks in social networks.

Other rigorous efforts have also focused on the formulation of measurable belief representing trust. One of them is to use a subjective probability [4, 7] that quantifies trust as a social belief. In the subjective logic, an agent's opinion is presented by degrees of belief, disbelief, and uncertainty. Handling uncertainty in various trust operations is too intuitive to be clear. And further, the subjective logic provides not a certain value of trust but a probability certainty density function. However, our trust model provides a specific trust value as an average trust considering agent's trustworthiness and untrustworthiness together. In another approach, a simple e-Bay feedback system [13] uses a feedback summary, which is computed as arithmetically subtracting the number of negative feedbacks from the number of positive feedbacks. Sabater and Sierra [14] reviews the research in the area of computational trust and reputation models, from the perspectives of multi-agent system paradigm and e-commerce, based on the classification dimensions of conceptual model, information sources, context dependability, model type, and so on. The contribution of our work is to

precisely define the notion of trust as a measurable social belief, and to clearly describe the relationship between reputation, trust, and average trust in social multi-agent settings.

3 The Measurable Belief of Trust

We propose a formal model of reputation resulting from feedbacks in social networks. Our reputation model takes into account direct agent experiences and witness information from third party agents [14]. The notion of trust then can be defined as an aggregation of consensus given a set of reputations. The calculation of average trust, further, results in a precise trust value as a metric. In this section, we describe the relationship between reputation, trust, and average trust through a concrete example of their computations.

3.1 Modeling Reputation

Feedbacks in social networks [6, 13] represent reputation associated with the society of multiple agents. The cumulative positive and negative events or feedbacks for an agent, thus, constitute the agent's reputation [8, 11]. The reputation can be described by a binary proposition¹ p for example, "A seller deals with only qualified products and delivers them on time." in the field of online Internet transactions. Given a binary proposition p and an agent-group i judging an agent in p , the reputation of the agent in p , ω_i^p , can be defined as follows:

$$\omega_i^p = \{T_i, U_i\} \quad (1)$$

where

- $T_i = PF_i/N_i$ and $0 \leq T_i \leq 1$;
- PF_i is the number of positive feedbacks for p within an agent-group i ;
- $U_i = NF_i/N_i$ and $0 \leq U_i \leq 1$;
- NF_i is the number of negative feedbacks for p within an agent-group i ;
- ZF_i is the number of neutral feedbacks for p within an agent-group i ;
- N_i is the total number of feedbacks for p within an agent-group i and $N_i = PF_i + NF_i + ZF_i$.

In the definition of reputation, as described in (1), we assume that the feedbacks given by agents within an agent-group evaluating p are independent, and further, the opinions supporting p can be only loosely related to the possible opinions supporting $\neg p$, since there could be neutral feedbacks from the agent-group. The notion of reputation, thus, is based on independent opinions and the sum of T_i and U_i is not necessarily being 1. Our model of reputation also relies on the agents within any agent-group who honestly rate the others without cheating.

¹ Any reputation in the form of proposition can be expressed according to the contexts as follows: "A buyer has an intention and capability to pay," "The network system could be safe from any intrusions," "A car could be reliable for ten years," and so on.

The cumulative positive feedbacks in social networks result from cooperativeness, i.e., trusty interactions, and establish trustworthiness of an agent in p , while the possible number of negative feedbacks from the society affects untrustworthiness of the agent. Thus, the trustworthiness of an agent represents the positive expectations of third party agents about its future behaviors. The trustworthiness and untrustworthiness together constitute a reputation function as a quantitative concept. The reputation of an agent varies with time and size of society, and clearly influences its trust. Given a set of reputations, which is collected at different times and from various interactions made by other agent-groups, the trust as a representative reputation will be derived.

3.2 Calculating Trust Using Aggregation Rules

We define trust as a consensus from an aggregation of reputations. The trust² ω^p for an agent in a proposition p is defined as

$$\omega^p = \omega_i^p \otimes \omega_j^p = \{T, U\} \quad (2)$$

where

- ω_i^p and ω_j^p represent reputations accumulated from an agent-group i and an agent-group j , respectively;
- T is the trustworthiness of the agent in a proposition p and $0 \leq T \leq 1$;
- U is the untrustworthiness of the agent in a proposition p and $0 \leq U \leq 1$.

The trust, as described in (2), consists of trustworthiness and untrustworthiness. These two components are determined by a set of reputations, as previously defined in (1). To formulate the agent's trust from reputations, expressed in degrees of trustworthiness and untrustworthiness which may or may not have the mathematical properties of probabilities, therefore, we propose a set of aggregation rules [9]. Given reputations of ω_i^p and ω_j^p , the aggregation operators, $\otimes = \{\Psi_1, \dots, \Psi_n\}$, in the paper, are as follows:

1. Minimum (Ψ_1): $T = \min(T_i, T_j)$, $U = \min(U_i, U_j)$;
2. Maximum (Ψ_2): $T = \max(T_i, T_j)$, $U = \max(U_i, U_j)$;
3. Mean (Ψ_3): $T = (T_i + T_j) / 2$, $U = (U_i + U_j) / 2$;
4. Product (Ψ_4): $T = T_i T_j$, $U = U_i U_j$;
5. Dempster-Shafer theory [5, 15, 16] (Ψ_5):

$$T = \frac{T_i T_j}{1 - (T_i U_j + T_j U_i)}, \quad U = \frac{U_i U_j}{1 - (T_i U_j + T_j U_i)}.$$

² For the sake of simplicity, we explain our trust model in a much simpler case of two agent-groups i and j . Our model of trust can be simply extended in more complicated settings involving multiple agent-groups without loss of generality.

The trust representing the degrees of belief on agent's truthfulness can be obtained by applying aggregation rules to a set of reputations. The goal of aggregation is to combine reputations when each of them estimates the probability of trustworthiness and untrustworthiness for an agent, and to produce a single probability distribution that summarizes the various reputations.

The minimum and maximum aggregation rules provide a single minimum and maximum value for T and U , respectively. The mean aggregation operator simply extends a statistic summary and provides an average of T_k 's and U_k 's coming from different agent-groups. The product rule summarizes the probabilities that coincide in T and U , respectively, given a set of reputations. Dempster's rule³ for combining degrees of belief produces a new belief distribution that represents the consensus of the original opinions [16]. Using Dempster's rule, the resulting values of T and U indicate the degrees of agreement on trustworthiness and untrustworthiness of original reputations, respectively, but completely exclude the degrees of disagreement or conflict. The advantage of using the Dempster's rule in the context of trust is that no priors and conditionals are needed.

Among the possible outputs of trust, we denote the trust as the consensus output using a specific aggregator, which is defined as

$$\hat{\Psi}(t, u) = \Psi(\Psi_1(t, u), \dots, \Psi_n(t, u)) \quad (3)$$

where

- Ψ is a function determining a specific aggregation rule;
- $\hat{\Psi}(t, u)$ is the aggregation rule selected with the inputs of $t \in T_k$ and $u \in U_k$.

Example 1. Let $\omega_1^p = \{0.80, 0.10\}$, $\omega_2^p = \{0.70, 0.20\}$. This is interpreted that there are two agent-groups evaluating p and, in each group, the resulting number of positive feedbacks is much greater than that of negative feedbacks, respectively. Given reputations, aggregation rules can be applied to get trust, as defined in (2), denoting a consensus out of agent-groups' opinions. The possible outputs of trust using the aggregation rules are summarized in Table 1.

Table 1. The example computation of trust using five aggregation rules

Aggregation rules	$\omega_1^p = \{0.80, 0.10\}, \omega_2^p = \{0.70, 0.20\}$
	Trust ω^p
Minimum (Ψ_1)	$\{0.70, 0.10\}$
Maximum (Ψ_2)	$\{0.80, 0.20\}$
Mean (Ψ_3)	$\{0.75, 0.15\}$
Product (Ψ_4)	$\{0.56, 0.02\}$
Dempster-Shafer theory (Ψ_5)	$\{0.73, 0.03\}$

³ In this paper, a set of original reputations embedded in social networks are assumed to be consistent in measuring them. This assumption avoids the counterintuitive results obtained using Dempster's rule in the presence of significantly conflicting evidence, which was originally pointed out by Lotfi Zadeh [17].

For example, when we use Ψ_5 as an aggregation rule, the trust given reputations is calculated as follows:

$$T = \frac{(0.8)(0.7)}{1 - [(0.8)(0.2) + (0.7)(0.1)]} = 0.73;$$
$$U = \frac{(0.1)(0.2)}{1 - [(0.8)(0.2) + (0.7)(0.1)]} = 0.03.$$

Among possible outputs of trust, the trust can be denoted as $\omega^p = \{0.70, 0.10\}$, when $\hat{\Psi}(t, u) = \Psi_l$. When minimum, maximum, and mean aggregators are used, the resulting distribution of the trust similarly reflects the distributions of the reputation. In cases of product and Dempster-Shafer theory, however, the T 's (0.56 and 0.73) of the trusts are much bigger than their U values (0.02 and 0.03), compared with the original distributions of the reputation. The resulting T value in Ψ_5 is interpreted that there is a 0.73 chance that the agent in p has the trustworthiness, while the resulting U value indicates that there is only a 0.03 chance that the agent is negatively estimated. As we mentioned above, thus, normalizing the original values of trustworthiness and untrustworthiness, which is corresponding to the denominator in the above equation, makes the opinions associated with conflict being away from the trust as a consensus.

To show how the aggregation rules could be adapted to various distributions of reputation, we consider additional set of reputations. The possible outputs of trust with two different set of reputations are displayed in the second and the third column of Table 2, respectively.

Table 2. The possible outputs of trust with two different set of reputations

Aggregation rules	$\omega_1^p = \{0.20, 0.80\},$ $\omega_2^p = \{0.30, 0.70\}$	$\omega_1^p = \{0.30, 0.30\},$ $\omega_2^p = \{0.50, 0.50\}$
	Trust ω^p	Trust ω^p
Minimum (Ψ_1)	$\{0.20, 0.70\}$	$\{0.30, 0.30\}$
Maximum (Ψ_2)	$\{0.30, 0.80\}$	$\{0.50, 0.50\}$
Mean (Ψ_3)	$\{0.25, 0.75\}$	$\{0.40, 0.40\}$
Product (Ψ_4)	$\{0.06, 0.56\}$	$\{0.15, 0.15\}$
Dempster-Shafer theory (Ψ_5)	$\{0.10, 0.90\}$	$\{0.21, 0.21\}$

The example of second column shows the case that the number of positive feedbacks is much less than that of negative feedbacks, and the third column is an example that the number of both feedbacks is identical. Note that the resulting distributions of trustworthiness and untrustworthiness, as displayed in Table 2, mirror their distributions in the original set of reputations. □

Since the available feedbacks from multiple agent-groups in social networks are classified into positive, negative, and neutral ones, the positive and negative feedbacks

among them are adopted for the components of our trust model. However, these two values contradicting each other are still not enough to represent the trust itself as degrees of belief on agent's truthfulness. From pragmatic perspective, the trust is required to be a precise value as a metric.

3.3 Average Trust

We define average trust as the center of gravity of the distribution of beliefs, i.e., the degrees of trustworthiness and untrustworthiness for an agent. The average trust $\hat{\omega}^p$ is given as

$$\hat{\omega}^p = \frac{T}{T + U} \quad (4)$$

taking into account both trustworthiness and untrustworthiness of an agent. The average trust, thus, represents the overall beliefs on agent's truthfulness or cooperativeness, and translates the agent's trust into a specific value where $0 \leq \hat{\omega}^p \leq 1$. In the notion of average trust, the higher the average trust level for the agent, the more the expectation that the agent will be truthful or cooperative in future interactions. The calculation of average trust using equation (4) gives social insight on the agent's trust.

Example 1 (cont'd). Given a set of reputations in the three agent-groups above, the average trusts are shown in Table 3.

Table 3. The average trust values in three example sets of reputation

Aggregation rules	$\omega_1^p = \{0.80, 0.10\},$	$\omega_1^p = \{0.20, 0.80\},$	$\omega_1^p = \{0.30, 0.30\},$
	$\omega_2^p = \{0.70, 0.20\}$	$\omega_2^p = \{0.30, 0.70\}$	$\omega_2^p = \{0.50, 0.50\}$
	average trust $\hat{\omega}^p$		
Minimum (Ψ_1)	0.88	0.22	0.50
Maximum (Ψ_2)	0.80	0.27	0.50
Mean (Ψ_3)	0.83	0.25	0.50
Product (Ψ_4)	0.97	0.10	0.50
Dempster-Shafer theory (Ψ_5)	0.96	0.10	0.50

This example illustrates that the average trust provides a metric for the agent's overall truthfulness, which consists of trustworthiness and untrustworthiness. The simple aggregation rules, i.e., minimum, maximum, mean, and product, give a pretty representative trust value considering both trustworthiness and untrustworthiness, even though it is not clear which one is good for a particular setting. This may be the reason that these simple but surprisingly well applicable rules keep on being popular in any contexts [9]. The product rule and Dempster-Shafer theory highly rate the

agent's average trust than the other simple rules. We attribute this sharp contrast between trustworthiness (refer to 0.97 and 0.96 in Table 3) and untrustworthiness (0.10 and 0.10, respectively, in Table 3) to their purely conjunctive operation with completely ignoring the degrees of disagreement or conflict. $\square$

4 Applying Trust Model to Online Internet Transactions

We apply our trust mechanisms to online Internet transactions. Given the actual feedbacks of agent-groups in online multi-agent settings, we can convert the feedbacks into the agent's reputation, denote its trust as an aggregation of reputations, and compute the average trust for a measurable belief on the agent's truthfulness. In this section, we pursue how the trust mechanisms are involved in a rational decision-making of buyers and sellers.

Suppose that there are sellers and buyers in online Internet settings. Let R be a contract price, s be the quantitative size of the contract, $V(s)$ be the buyer's benefit (or value) function, which reflects his/her satisfaction acquired by purchasing a number of commodities, and $C(s)$ be the seller's cost function, which indicates the cost to produce the amount of the commodities. Given the average trust of the buyer $\hat{\omega}^M$, the expected utility of the buyer is given by⁴

$$EU_M(s) = \hat{\omega}^M V(s) - R. \quad (5)$$

Given the average trust of the seller $\hat{\omega}^N$, and also, the expected utility of the seller is defined as

$$EU_N(s) = \hat{\omega}^N R - C(s). \quad (6)$$

In equations (5) and (6), the average trust is interpreted as the overall beliefs on the buyer's and the seller's truthfulness or cooperativeness, respectively. When the average trusts of the seller and the buyer get higher, further, their expected utilities also increase. The Nash equilibrium [2, 12] in online transactions, then, provides a solution concept when the buyer and the seller have no incentives in case of choosing other alternatives. The Nash bargaining solution is

$$\arg \max_R (\hat{\omega}^M V(s) - R)(\hat{\omega}^N R - C(s)) \quad (7)$$

so that the buyer and the seller are beneficial to each other if they agree on their bargaining behavior. Note that equation (7) has a unique Nash equilibrium, since an R can be determined given average trusts of the buyer and the seller, $V(s)$, and $C(s)$.

Example 2. To derive R given the Nash bargaining solution, as defined in (7), let us take the first derivative of equation (7) as follows:

⁴ Our notation follows [2].

$$\begin{aligned}
\frac{d}{dR}(\hat{\omega}^M V(s) - R)(\hat{\omega}^N R - C(s)) &= 0; \\
\frac{d}{dR}(-\hat{\omega}^N R^2 + (\hat{\omega}^M \hat{\omega}^N V(s) + C(s))R - \hat{\omega}^M V(s)C(s)) &= 0; \\
\therefore R &= \frac{\hat{\omega}^M \hat{\omega}^N V(s) + C(s)}{2\hat{\omega}^N}.
\end{aligned}$$

Thus, the contract price R that they agree on can be determined in a Nash equilibrium. Substituting the above into (5) and rearranging terms, we get

$$\begin{aligned}
EU_M(s) &= \hat{\omega}^M V(s) - R \\
&= \hat{\omega}^M V(s) - \left(\frac{\hat{\omega}^M \hat{\omega}^N V(s) + C(s)}{2\hat{\omega}^N} \right) \\
&= \left(\frac{\hat{\omega}^M \hat{\omega}^N V(s) - C(s)}{2\hat{\omega}^N} \right)
\end{aligned} \tag{8}$$

In similar way, the expected utility of the seller is

$$\begin{aligned}
EU_N(s) &= \hat{\omega}^N R - C(s) \\
&= \hat{\omega}^N \left(\frac{\hat{\omega}^M \hat{\omega}^N V(s) + C(s)}{2\hat{\omega}^N} \right) - C(s) \\
&= \left(\frac{\hat{\omega}^M \hat{\omega}^N V(s) - C(s)}{2} \right)
\end{aligned} \tag{9}$$

Substituting (8) and (9) into the formula of (7), given the fact that the numerator of (8) and (9), i.e., $(\hat{\omega}^M \hat{\omega}^N V(s) - C(s))$, is identical, we observe that both of the buyer and the seller make their maximum gains, when the numerator gets maximized.

Suppose that the buyer's benefit function $V(s)$ is $48\ln(2s)$ and the seller's cost function $C(s)$ is $s^2 - 2s + 3$ as usual.⁵ When $\hat{\omega}^M = \hat{\omega}^N = 0.8$, the quantitative size of the contract s can be determined by

⁵ We assume that the buyer's benefit does not necessarily increase in proportion to the quantitative size of commodities while the seller's cost proportionally increases to produce a certain amount of commodities.

$$\frac{d}{ds} \left(\hat{\omega}^M \hat{\omega}^N V(s) - C(s) \right) = 0;$$

$$\frac{d}{ds} \left(0.8 \times 0.8 \times 48 \ln(2s) - (s^2 - 2s + 3) \right) = \left(-2s + 2 + 0.8 \times 0.8 \times 48 \times \frac{1}{s} \right) = 0;$$

$$\therefore s = 4.45$$

That is, they both maximize their expected utilities and, once the buyer's benefit function and the seller's cost function are decided, the quantitative size of the contract is computed as the above. Thus, $s=4.45$. The expected utilities of the buyer and the seller also can be calculated, and, in this case, we get $EU_M(s)=33.28$ and $EU_N(s)=26.63$ from equations (8) and (9). Consider now that the seller's average trust is low, say, $\hat{\omega}^N = 0.2$. Then, $s=2.52$, and their expected utilities are $EU_M(s)=20.28$ and $EU_N(s)=4.06$. Calculated above, both the overall quantitative size of contract and the expected utilities of the buyer and the seller are larger, when the average trust values of the agents are higher. $\square$

5 Conclusion

The model of trust in social networks has been continuously studied for safe and successful interactions. Our work contributes to a computational model of trust as an aggregation of consensus associated with multiple agent-groups. We formulated reputation based on available feedbacks resulting from social interactions, calculated trust among a set of reputations using aggregation rules, and represented average trust as a metric for the agent's truthfulness or cooperativeness. We have shown how our trust model can be calculated in a detailed example. To show how the trust mechanisms are involved in a rational decision-making of interactive agents, our trust model has been applied to electronic societies. We believe the computational trust model and mechanisms should be applicable to real societies of multi-agent environments.

As part of our ongoing work, we are applying our trust model to online Internet e-markets. To this end, we are designing and developing a practical test-bed to evaluate various models of trust including our framework. Given the actual feedbacks of customers in online multi-agent settings, we will convert the feedbacks into the agent's reputation, denote its trust as a numerical aggregation of reputations, and pursue how trust affects a rational decision-making of buyers and sellers. We will benchmark the amount of interactions between the buyers and the sellers, when they have higher trust values and/or they have lower trust values. The experiments that we are performing will also measure the global profits in a set of agent-groups employed with different trust values.

References

1. Axelrod, R.: The Evolution of Cooperation. Basic Books, New York (1984)
2. Braynov, S., Sandholm, T.: Contracting with Uncertain Level of Trust. Computational Intelligence 18(4), 501–514 (2002)

3. Coleman, J.: *Foundations of Social Theory*. Harvard University Press, Cambridge (1990)
4. Daskalopulu, A., Dimitrakos, T., Maibaum, T.: Evidence-Based Electronic Contract Performance Monitoring. *INFORMS Journal of Group Decision and Negotiation* 11, 469–485 (2002)
5. Dempster, A.P.: A Generalization of Bayesian Inference. *Journal of the Royal Statistical Society, Series B* 30, 205–247 (1968)
6. Golbeck, J.: Generating Predictive Movie Recommendations from Trust in Social Networks. In: *Proceedings of the Fourth International Conference on Trust Management*, Pisa, Italy (2006)
7. Josang, A., Knapkog, S.J.: A Metric for Trusted Systems. In: *Proceedings of the 21st National Information Systems Security Conference*, Virginia, USA (1998)
8. Kinader, M., Rothermel, K.: Architecture and Algorithms for a Distributed Reputation System. In: Nixon, P., Terzis, S. (eds.) *iTrust 2003*. LNCS, vol. 2692, pp. 1–16. Springer, Heidelberg (2003)
9. Kuncheva, L.I., Bezdek, J.C., Duin, R.: Decision Templates for Multiple Classifier Fusion: An Experimental Comparison. *Pattern Recognition* 34, 299–314 (2001)
10. Marsh, S.: *Formalizing Trust as a Computational Concept*, Ph.D. thesis, University of Stirling, UK (1994)
11. Mui, L., Mohtashemi, M., Halberstadt, A.: A Computational Model of Trust and Reputation. In: *Proceedings of the 35th Hawaii International Conference on System Sciences* (2002)
12. Nash, J.: The Bargaining Problem. *Econometrica* 28, 155–162 (1950)
13. Resnick, P., Zeckhauser, R.: Trust Among Strangers in Internet Transactions: Empirical Analysis of eBay's Reputation System. In: *The Economics of the Internet and E-Commerce*. vol. 11. *Advances in Applied Microeconomics*, Elsevier, North-Holland (2002)
14. Sabater, J., Sierra, C.: Review on Computational Trust and Reputation Models. *Artificial Intelligence Review* 24(1), 33–60 (2005)
15. Shafer, G.: Perspectives on the Theory and Practice of Belief Functions. *International Journal of Approximate Reasoning* 3, 1–40 (1990)
16. Shafer, G., Pearl, J. (eds.): *Readings in Uncertain Reasoning*, Chapter 3 Decision Making and Chapter 7 Belief Functions. Morgan Kaufmann Publishers, Seattle (1990)
17. Zadeh, L.A.: Review of Books: A Mathematical Theory of Evidence. *AI Magazine* 5(3), 81–83 (1984)

Enhancing Grid Security Using Trusted Virtualization^{*}

Hans Löhr¹, HariGovind V. Ramasamy², Ahmad-Reza Sadeghi¹, Stefan Schulz³,
Matthias Schunter², and Christian Stübke¹

¹ Horst-Görtz-Institute for IT-Security
Ruhr-University Bochum, Germany

{loehr,sadeghi,stueble}@crypto.rub.de

² IBM Zurich Research Laboratory Rüschlikon, Switzerland
{hvr,mts}@zurich.ibm.com

³ Max-Planck Institut für Eisenforschung, Germany
schulz.stefan@gmail.com

Abstract. Grid applications increasingly have sophisticated functional and security requirements. Current techniques mostly protect the grid resource provider from attacks by the grid user, while leaving the user comparatively dependent on the well-behavior of the provider. We present the key components for a trustworthy grid architecture and address this trust asymmetry by using a combination of trusted computing and virtualization technologies. We propose a scalable offline attestation protocol, which allows the selection of trustworthy partners in the grid with low overhead. By providing multilateral security, i.e., security for both the grid user and the grid provider, our protocol increases the confidence that can be placed on the correctness of a grid computation and on the protection of user-provided assets.

1 Introduction

Grid computing has been very successful in enabling massive computing efforts, but has hitherto been dominated by ‘big science.’ These projects are usually in the academic domain (such as SETI@HOME or distributed.net) and, although important, they usually have less stringent security requirements than commercial IT systems. Currently, security is built into grid toolkits (e.g. the Globus toolkit [1]) used at the *provider* sites (parties that offer resources for use in the grid). Secure channels, authentication, unsupervised login, delegation, and resource usage [2] are all handled by the toolkit. These mechanisms usually do not protect the grid *user* (the person or entity wishing to utilize resources).

The user is forced to trust¹ the provider, often without the possibility of verifying whether that trust is justified. However, in much of the current literature on grid

^{*} A preliminary version of this work was presented (without publication) at the *2nd Workshop on Advances in Trusted Computing 2006* and at the *1st Benelux Workshop on Information and System Security 2006*.

¹ In this paper, we consider “trust” to be the opposite of enforcement. Thus, a trusted component is a component whose well-behavior cannot be enforced by another component and, therefore, has the capability to violate a security policy. This view of trust contrasts with the notion put forward in other grid-related works, such as [3], which view trust as a positive, reputation-based property.

security (e.g., [4]), the user is not regarded as trustworthy. This trust asymmetry could potentially lead to a situation in which the grid provider causes large damage to the user with little risk of detection or penalty. An attacker might publish confidential data or sabotage the entire computation by providing false results. These problems are most evident in computational grids, especially in mobile code [5] scenarios. Other grids, such as storage or sensor grids, may also suffer from the negative consequences of this trust asymmetry. Because of this problem, companies are reluctant to utilize available grid resources for critical tasks.

Given this state of affairs, Mao et al. [6] have advocated the use of the emerging *Trusted Computing* (TC) technology for the grid. In a similar vein, Smith et al. [7] more closely examine scenarios that could benefit from TC techniques. TC can be used to enforce *multilateral security* i.e., the security objectives of all parties involved.

A trustworthy grid environment that enforces multilateral security would offer a number of benefits. Even sensitive computations could be performed on untrusted hosts. Most personal computers used today possess computing abilities in excess of what is required for casual or office use. These resources could be leveraged to run grid jobs in parallel to the users' normal workflow and provide the computational power necessary for next-generation modeling and simulation jobs, without costly investments into new infrastructure. Enterprises could utilize the already-present office machines more fully, resulting in an earlier return on their investment.

A large percentage of the platforms in large-scale grids are built using general-purpose hardware and software. However, it is easy and cheap for existing platforms to incorporate a *Trusted Platform Module* (TPM), based on specifications of the *Trusted Computing Group* (TCG). The module provides a trusted component, usually in the form of a dedicated hardware chip. The chip is already incorporated into many newly-shipped general-purpose computers. The TPM chip is tamper-evident (and ideally, tamper-resistant) hardware that provides cryptographic primitives, measurement facilities, and a globally unique identity. For verification purposes, a remote party can query the TPM's measurement of the *Trusted Computing Base* (TCB) by means of *attestation*. This mechanism, proposed by the TCG, enables (remote) verification of the status of a platform's TCB.

One approach to securing computing systems that process potentially malicious code (such as in many number-crunching grid applications) is to provide a virtualized environment. This technique is widely used for providing "V-Servers," i.e., servers running several virtual machines that may be rented to one or several users. Although users have full control over the virtual environment, they cannot cause damage outside that environment, except possibly through attempts at resource monopolization, for example, by "fork bombing." Although virtualization offers abstraction from physical hardware and some control over process interaction, there still are problems to be solved. For example, in the x86 architecture, direct memory access (DMA) devices can access arbitrary physical memory locations. However, hardware innovations such as Intel's Trusted Execution Technology [8] (formerly known as LaGrande) and AMD's Virtualization Technology [9] (formerly code-named Pacifica) aim to address these problems and could eventually lead to secure isolation among virtual machines. Virtualization technology can be leveraged for building a trustworthy grid environment, especially because

several works, such as [10], have already begun to consider architectures that feature policy enforcement in the virtualization framework.

Our Contribution. To address the trust asymmetry in grid computing explained above, we propose a realistic security architecture that uses TC functionality and enforces multilateral security in a grid scenario. Leveraging a combination of the isolation (between virtual machines) provided by virtualization and a trusted base system, our design is able to protect confidentiality and integrity in a multilateral fashion. We feel our compartmented security design offers a stronger level of protection than many current techniques can provide.

Using our security architecture, we propose a grid job submission protocol that is based on offline attestation. The protocol allows a user to verify that a previously selected provider is in a trusted state prior to accessing a submitted grid job, with little overhead and improved resistance to attack. Our protocol also guarantees *transitive* trust relations if the provider in turn performs further delegations to other providers.

2 Preliminaries

2.1 System Model and Notation

We consider the following abstract model of the grid. A grid user U can attempt to access any grid provider P . Each participant in the grid is considered to be a partner-and-adversary that potentially intends to harm other participants but also provides services. A participant can be depended upon to execute a given task correctly only if it can prove its inability to cause damage (break a partner's security policy).

A machine m is a single physical host. It can host one or more logical participants of either role. We consider delegation to be modeled as one participant being both a provider and a user. Every participant has its own, distinct policy. Each component of m is an independent actor offering some interface(s) to other components, and usually utilizing interfaces offered by other components. The set of providers and users need not be static, but can grow and shrink dynamically as new resources are being added to the grid virtual organization (VO), and some participants leave the VO. However, joining and leaving are not the focus of this paper.

For our purposes, a job image is a tuple $J = (\text{data}, \mathcal{C}, SP_U)$, where data may be an invocation to some predefined interface or carry executable code. For security purposes, both input data and executable code have the same requirements and can be protected using the same techniques. Therefore, we do not distinguish between “code” and “data,” and refer to both as data. $\mathcal{C}$ represents the credentials of the user U , which may be needed to gain access to the provider P . The user also passes a policy SP_U as part of its invocation, which specifies constraints to be upheld for that particular job. The job, once scheduled, can communicate directly with U (subject to the policy SP_U).

A machine m always has exactly one state σ describing the status of the TCB rather than a particular VM. This state comprises all code running as part of the TCB. TCB components are critical to the correct functioning of the system and need to be trusted. Adding, removing, or modifying such a component changes σ . However, σ will not change because of “user actions,” such as installing application software, browsing the

web, or executing a grid job. Furthermore, the system will not allow any party (not even system administrators) to alter the TCB without changing σ . σ' is the reported state of the platform, possibly different from σ . We assume that σ and σ' can be encoded as a configuration (or metrics) conf , a short representation of the state (e.g., a hash value) as determined by a *measurement* facility (e.g., the TPM) of the machine. A specific aspect of the user's security policy SP_U is the *good* set, which contains the conf values of all states σ considered to be trustworthy by that policy.

K denotes an asymmetric cryptographic key, with private part s_K and public part p_K . $\text{enc}_{p_K}(X)$ denotes a piece of data X encrypted with a public key p_K . $\text{sign}_{s_K}(X)$ denotes a data item X that has been digitally signed by a private key s_K .

2.2 Usage Scenario

We consider the following scenario: When a node joins the grid, it generates and publishes an attestation token τ , which can be used by potential partners to obtain assurance about the node's trustworthiness. Grid users retrieve attestation tokens from different grid nodes and select a token indicating a configuration they are willing to trust. The selection decision is made offline, and incurs negligible overhead on the part of the user. Once an acceptable provider has been found, users can submit jobs that can only be read by the selected node in the configuration they consider as trustworthy. If the node has changed to another configuration, communication will fail.

The main advantage of this approach is that the creation of the attestation tokens is decoupled from the process of job submission, while still providing freshness. In addition, these tokens are transferable and their correct creation can be verified without interacting with their creators.

2.3 Requirements

In this paper, we focus on security requirements, namely *integrity* and *confidentiality*. Providing integrity means protection against unauthorized modifications. For instance, user U should not be able to alter aspects of provider P to elevate its privilege level. Similarly, P should be prevented from modifying U 's job. Both the user and provider may require confidentiality, i.e., they may require their sensitive data be guarded against unauthorized disclosure. U may utilize confidential data as part of J , and demand that this data not be disclosed to any party other than J 's execution environment. Similarly, P may want to ensure that a malicious grid job cannot collect secrets stored on P 's platform (such as signature keys) and forward them to U .

3 A Trusted Grid Architecture

Figure 1 shows the abstract building blocks of our Trusted Grid Architecture (TGA). The hardware platform provides a TPM and untrusted storage. The Trusted Software Layer (TSL) consists of the attestation, grid management, compartment management, and storage management components. The TSL provides both security functionalities and virtualization of the hardware. The TCB consists of the TSL and the trusted hardware components. Security policies have to be enforced by the TCB, but a detailed

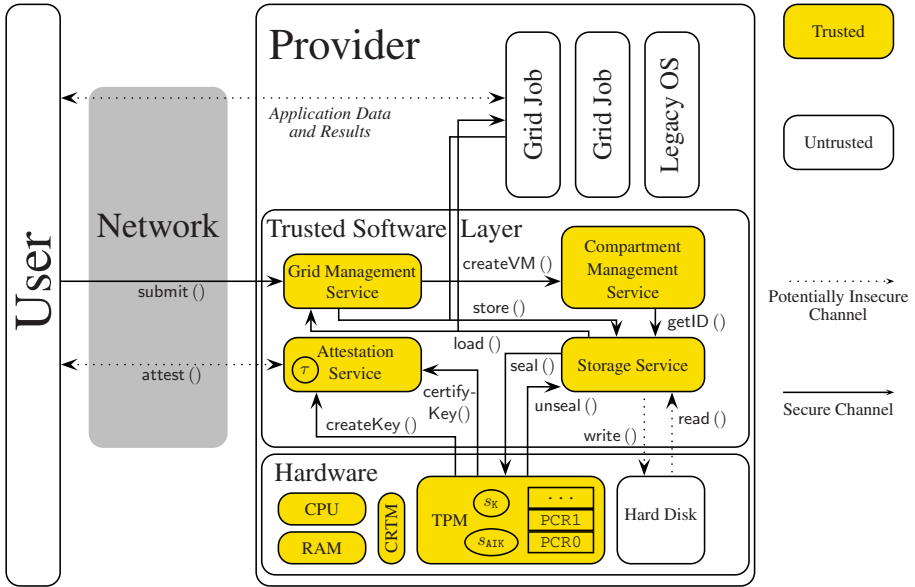


Fig. 1. Components of the Trusted Grid Architecture

treatment of policy enforcement is outside the scope of this paper. Other works, such as [10] and [11], have examined some necessary properties of policy engines. Proper design of a *minimum* set of trusted services can help to achieve a TCB with the highest possible resistance to attacks. Additional guarantees about runtime behavior and state (e.g., [12]) may be provided by a dedicated service or as an extension to our attestation service.

We now provide an overview of the TGA components; more details can be found in [13].

Hardware: The core hardware component is a TPM as specified by the TCG, providing cryptographic functions such as encryption and signing. Each TPM possesses a number of platform configuration registers (PCRs), at least 16 as of version 1.2 of the specification [14]. During system boot, the main software components (BIOS, bootloader, OS kernel, etc.) are *measured*. The measurement procedure involves computing a configuration *conf* or the cryptographic hash of the software components and security storing the hash in the TPM. For the TGA, we use four TPM operations: secure key generation, measurement, certification, and sealing. The TPM features a hardware random-number generator and implements generation of RSA key pairs $K = (p_K, s_K)$. For these key pairs, usage limitations can be defined, in particular *sealing*, which marks the private key as not being migratable and usable only when a specified subset of the PCRs contain the same values as were present during key generation. It is possible to obtain a certificate stating which usage conditions apply to a key pair (as represented by its public key p_K) from the TPM, signed by one of its *Attestation Identity Keys* (AIKs; generated by the TPM). The private key of an AIK cannot be extracted from the TPM, i.e., it is

non-migratable, and it cannot be used to certify migratable keys. AIKs can be certified by a Certification Authority (CA), or they can be proved to be valid AIKs anonymously by means of Direct Anonymous Attestation (DAA) [15]. Such a certificate or proof is denoted as $\text{cert}_{\text{CA}}(p_{\text{AIK}})$.

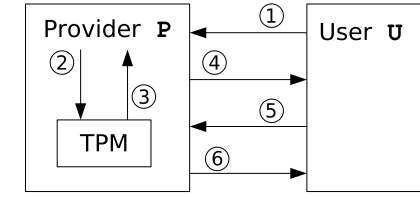
The TPM can report the platform configuration to other parties by signing the values of the PCRs with an AIK, which guarantees that the TPM generated the signed structure because an AIK cannot be used to sign arbitrary data. For our purposes, we use signed *KeyInfo* structures that are considered as certificates. A *KeyInfo* structure of a sealed key includes the selection of PCRs that were used for sealing, their values at the time of key generation, the values of the selected PCRs needed to use the sealed key (i.e., the *conf* of reported state σ'), and an indication whether a key is migratable. We use an AIK to sign such a structure with the *certifyKey* operation of the TPM and denote the resulting certificate by $\text{cert}_{\text{AIK}}(p_K)$. These restricted keys enable *data sealing*. Data sealed to a certain configuration of the system is encrypted with a public key whose corresponding private key is accessible only to a certain state and platform. If the data is successfully decrypted, this indicates that the state the key was sealed to is the actual state of that machine.

Attestation Service (AS): The AS provides metrics about the state σ to remote parties by means of an *attestation token* $\tau := (p_{\text{AIK}}, p_K, \text{cert}_{\text{CA}}(p_{\text{AIK}}), \text{cert}_{\text{AIK}}(p_K))$. From *conf* (contained in $\text{cert}_{\text{AIK}}(p_K)$), the user U is able to distinguish a trusted state σ' from an untrusted one because the values uniquely identify a set of programs that have been loaded since booting the platform, and possibly also the state of certain critical configuration files. The certificate $\text{cert}_{\text{AIK}}(p_K)$ identifies the key K as being sealed to *conf* and gives the assurance that the private key s_K can be used only in the reported state σ' . The user U can make its trust decision “offline” by examining the *conf* contained in τ . If *conf* is indicative of a trusted state σ' , s_K will be accessible to the provider P only if P still is in the same configuration. As the token does not change over time, it can be distributed to other parties. If the state σ of P ever changed, τ would automatically become invalid, although an explicit revocation might still be beneficial. Further details of this attestation mechanism and its security will be discussed in Section 4.

Compartment Management Service (CMS): This component creates *virtual machines* (VMs; also called *compartments*), which run on top of the TCB, and keeps track of the identity of compartments by assigning a unique identifier (ID) to each of them. The VMs are isolated from each other and can only communicate over well-defined interfaces. The CMS only manages VMs locally and does not address migration or delegation in the grid.

Storage Service (SS): The storage component provides trustworthy and non-volatile storage based on an untrusted hard disk. In particular, data stored by one compartment in one configuration is retrievable only by that compartment in the same configuration – even if the machine has entered an untrusted state in the meantime. To achieve this property, all data is encrypted and MAC-authenticated by a sealed key.

Grid Management Service (GMS): The GMS handles the actual grid job submission. It is responsible for receiving jobs, checking their access, and instantiating them. It will use the CMS to create a private compartment for each job. The GMS does any



Common input: attestation token

$\tau = (p_{\text{AIK}}, p_K, \text{cert}_{\text{CA}}(p_{\text{AIK}}), \text{cert}_{\text{AIK}}(p_K))$

U's input: job J and the accept set good_U

P's input: accept set good_P

TPM's input: (s_K, σ'_{s_K}) , current state σ

1. U verifies $\text{cert}_{\text{CA}}(p_{\text{AIK}})$, $\text{cert}_{\text{AIK}}(p_K)$, and $\text{conf} \in \text{good}_U$.

Upon verification, U randomly chooses nonces N and N', and a session key κ .

U sends $\text{enc}_{p_K}(\kappa)$ and $\text{enc}_{s_K}(N)$ to P.

2. P forwards $\text{enc}_{p_K}(\kappa)$ to TPM.
3. TPM decrypts κ if $\sigma = \sigma'_{s_K}$ and returns κ to P.
4. P decrypts N and sends $\text{enc}_{\kappa}(N, \text{good}_P)$ to U.
5. U verifies N and whether $\text{good}_P \subseteq \text{good}_U$; upon verification, U sends $\text{enc}_{\kappa}(N', J)$ to P.
6. P decrypts N' and J, and sends N' to U. U verifies N'.

Fig. 2. Submission Protocol submit ()

special pre-processing that the job needs before it is ready for execution. Once such pre-processing has been done, a VM image has been created from J, which can then be booted by the CMS. Furthermore, the GMS takes the policy of the user and notifies an enforcement component (not shown in Figure 1) of the restrictions and rights declared therein. It also handles the freshness verification of the attestation token τ when a job is submitted (described in Section 4).

4 A Protocol for Scalable Offline Attestation

Attestation is the process of securely reporting the configuration of a party to a remote challenger. The most commonly discussed type of attestation requires a remote challenger to provide a random nonce N, which is then signed (together with a hash over a subset of the current PCR values) by the TPM using an AIK. As freshness is achieved by means of a random nonce, each interaction necessitates a new attestation (and thus, a new TPM-generated signature). However, TPM signature generation is slow, and TPM commands generally cannot be parallelized. In addition, without appropriate counter-measures, this technology could potentially be vulnerable to a race between a successful attestation and a change of state prior to further interactions depending on the trusted state. If the state of the system changes after attestation has concluded, but before any further interactions take place, this change would not be noticed by the remote party. Also, without connecting attestation to a PKI identity, an attestation challenge could be relayed to a trusted platform by an attacker (by forwarding the trusted platform's reply to the verifier).

Scalable offline attestation is intended to enhance some aspects of current attestation systems. Having an *attestation token* that can be distributed freely within the VO as an informational item is advantageous, because this token states the current configuration of a provider P, without requiring the prospective user to interact with that provider right away. The user can collect such tokens over time, and select the most appropriate configuration offline. As such a token cannot guarantee freshness, some verification has to occur when the user contacts the provider of his choice. We propose a *sealed*

key approach, in which the provider's TPM allows usage of the private key only if the provider is in the same state as the key was stored in. The approach partitions the verification of P 's state into two phases: token creation and freshness verification.

A provider P creates an attestation token together with its TPM. The attestation service instructs the TPM to create a non-migratable key sealed to a collection of PCRs. Then, the attestation service uses the TPM's `certifyKey` operation to create a certificate $\text{cert}_{\text{AIK}}(p_K)$ with an AIK. The attestation service then constructs the attestation token τ from the public key p_K , the certificate of this key, $\text{cert}_{\text{AIK}}(p_K)$, the public part of the AIK, p_{AIK} , and a certificate of the AIK, $\text{cert}_{\text{CA}}(p_{\text{AIK}})$. The private key s_K is accessible only in the provider's state at the time of token generation, σ' , because the certification is done using the TPM-internal AIK, which cannot be misused, even by the platform owner. The attestation service then publishes the token. Publication of the attestation token τ in effect becomes an advertisement stating that a certain state σ' will be maintained at P .

The protocol shown in Figure 2 includes the actual submission of the job and addresses freshness verification. If the `conf` contained in the token is considered good by the user U , then U generates a symmetric session key κ and encrypts the key using p_K . The session key can be decrypted by the provider's TPM only if its state still matches the state at the time of τ 's creation, i.e., P 's reported state σ' . Verification of P 's ability to access s_K is sufficient to ensure that P is actually in the state that was advertised by `conf`. The rationale for including the session key is twofold. First, asymmetric cryptography is by orders of magnitude slower than symmetric methods. Second, the key's inclusion reduces the necessary TPM operations from the signature generation (in traditional schemes) to a single asymmetric decryption.

The submission protocol further guarantees *transitive* trust. As the job gets delegated from one provider to other providers, it is assured that each party that is entrusted with the job's data will satisfy the original submitter's requirements. This is done by ensuring that each platform X that gains control of the user U 's job J must satisfy the condition, $\text{good}_X \subseteq \text{good}_U$.

Extensions. In contrast to protocols like DAA [15], our proposed protocol does not feature any privacy guarantees. As the platform has to reveal its actual configuration, it is in effect exposing potentially sensitive information to another party. Integrating privacy guarantees into our proposal could be an interesting aspect for future research. To address some privacy issues and other well-known limitations of binary attestation, property-based attestation and sealing schemes (e.g., [16]) could be integrated into our TGA.

5 Security Analysis

Security of Offline Attestation. The offline attestation mechanism proposed in Section 4 is secure against man-in-the-middle attacks. If a user U seals a job to a trustworthy attestation token τ , only the platform in possession of the private part of key K can unseal the job, and only if it is in the state indicated by τ . An adversary cannot decrypt the job, even if it is running on the platform with the TPM that holds the private key, if `conf` (corresponding to the platform's current state σ) does not match `conf'` contained

in τ (corresponding to the platform's reported state σ'). Conventional techniques need to include additional verification (such as tying an AIK to a PKI identity) to achieve the same assurance as ours.

Delegation with transitive trust ensures that every provider P that gets a job J can only access J if the provider is in a state σ that is trusted by the original submitter U , i.e., $\text{conf} \in \text{good}_U$ (where conf corresponds to σ). Transitive trust is achieved during delegation without communication with the submitter because the provider that wishes to transfer a job attests other providers offline prior to transmitting the job. The delegating provider P_1 acts as user of the new provider P_2 and verifies that $\text{good}_{P_2} \subseteq \text{good}_{P_1}$, which immediately implies that $\text{good}_{P_2} \subseteq \text{good}_U$. Hence, the policy of the new provider P_2 is also acceptable to the original user. Moreover, offline attestation is secure against replay attacks, under the assumption that state changes can only occur between protocol runs. Replaying of old, trustworthy attestation tokens does not help an adversary: the TPM will not allow decryption if the current PCR values do not match the values the key was sealed against.

Our protocol has the following drawbacks. Like conventional attestation, our protocol is vulnerable to TPM compromises. A compromised TPM can expose the secret key to an adversary, which enables the adversary to attest to arbitrary states. Revocation of AIKs is necessary to limit the potential damage such attacks may cause. As with conventional attestation, another risk of offline attestation is corruption of the running TCB. If an adversary can corrupt the TCB while the system is running, it could change the system's state σ without changing the PCRs. Thus, σ would deviate from σ' , but the TPM would still allow the sealed key to be used.

Integrity Protection. Because we can establish a secure (confidential and integrity-protected) channel from user U to provider P using standard tools such as TLS, we need not consider in-transit modifications. Thus, for the purpose of this analysis, P receives an unaltered job J . We need to consider two kinds of integrity requirements for that image: before being instantiated and while executing. As results are reported directly, their integrity can again be achieved by established solutions. If job execution is delayed by the GMS, the job image and policy are stored in trusted storage. The key of the storage service is stored sealed, which guarantees that access to it is granted only to the same job in the same system state. In an untrusted state, no access is granted. Therefore, if a piece of data X in the storage service is altered, the signature of that data item cannot be updated, and the modification is detected the next time the data is retrieved from the storage service. While job J is executing, the isolation properties of our system guarantee that no untrusted application can gain access to the memory regions assigned to J , and hence, integrity is guaranteed. Circumventing such barriers would require breaching the TCB, which would contradict our assumption. As the TCB is based on a virtualization layer, even attack scenarios like “blue pill” [17] are ineffective, because such rootkits can only virtualize conventional systems that do not use virtualization techniques themselves. However, even if such a system were able to virtualize a virtualization layer, it would either need to compromise the TCB, or it would have to be loaded before the TGA (and thus, be measured in the boot process),

Confidentiality Protection. The two mechanisms employed for protecting the integrity of stored data and in-memory data also protect confidentiality. The CMS enforces

isolation between the VMs and foils *in-memory eavesdropping*, i.e., one process accessing data inside the virtual memory of another process. Sealing prevents untrusted configurations from decrypting data stored in non-volatile storage. Violating confidentiality implies breaching the TCB for the in-memory scenario, as the TCB enforces virtualization and therefore, limits each application to its own VM, whereas decrypting stored data outside of a trusted state would necessitate breaking the encryption scheme used, which we likewise consider infeasible.

6 Discussion

Integration of Legacy Systems. To maintain interoperability with legacy systems, we aim to provide the means to continue using applications designed for existing grid toolkits (such as Globus [1]), without giving up the advantages our architecture offers. One possible way for such an integration would be to provide an executable image for each toolkit supported. Whenever an invocation for a service using that toolkit is received, it is instantiated, and the request forwarded to that instance. However, the grid toolkit must be part of the TCB. After all, a malicious provider might use a good base configuration, and put all its attack code into a modified toolkit image. The attestation token τ should contain measurements of all execution environments available as “default installations” on the platform. Thus, the benefits of our proposal become applicable without forcing the user to significantly change its use of the grid. Alternatively, a grid job may consist of a full, bootable VM. While this is a radically different approach from traditional grid methods, it does not imply further trusted code, which is desirable to keep the TCB small and of low complexity.

Implementation. We have started implementing the core components of the TGA architecture in the PERSEUS framework [18], which is based on a micro-kernel with paravirtualized Linux. The framework’s design allows its porting to other systems (such as Xen), and features a strong separation of responsibilities even among the TCB (by running services as separate compartments), which significantly simplifies verification. Prototypes of the core TGA components have already been demonstrated in the context of the ongoing OpenTC [19] and European Multilaterally Secure Computing Base [20] projects.

Related Work. Several authors have suggested methods to increase the reliability of grid computation without TC technology. For instance, task replication or the introduction of quiz tasks [21] to detect misbehaving providers aimed at protecting the integrity of the results of grid computations. However, these techniques are wasteful in terms of resources and often not resistant to multiple colluding adversaries. Using virtualization to improve grid security has been proposed in numerous works (e.g., [22]).

Sailer et al. [10,23] investigated the possible enforcement of MAC policies at the level of the virtualization layer. Sailer et al. [24] also proposed an integrity measurement architecture for Linux. Such an architecture could be useful for the measurement and reporting of VM states in our TGA. Similarly, although the proposed system of Jaeger et al. [25] focuses on improving the integrity checking of SELinux, its underlying principles could be used for verifying the correctness of the Trusted Software Layer of our TGA.

The Daonity (e.g., see [26]) project that aims to strengthen the grid security infrastructure by integrating TC technology into the Globus toolkit. However, as Mao et al. [26] remark, the current version of Daonity does not take the operating system into account. For instance, an administrator could bypass the TC-based security mechanisms. To prevent such attacks, a system architecture with virtualization on top of a security kernel, as we propose in this paper, could be used.

Recently, Cooper et al. [27] proposed a security architecture for delegation on the grid based on TC and virtualization technologies. They describe a delegation service for enforcing local and global delegation policies. Offline attestation techniques, such as the one we propose, may be useful for their delegation service, whereas our solution in turn could benefit from their idea of enforcing hierarchical policies.

Dinda [28] proposed a novel scheme to protect the assets of the grid user against a malicious provider in order to address trust asymmetry. Similar to that proposal, encrypted computation (see, e.g., [29]) offers interesting results for some problems. By performing computations on encrypted data without decrypting it, some tasks can be completed without ever revealing plain text. However, these techniques have limited use outside the domain of some algebraic problems, and their widespread adoption seems unlikely.

7 Conclusion

In this paper, we proposed a protocol for scalable offline attestation based on a grid security architecture that uses virtualization and Trusted Computing technology. Our approach allows the grid user to choose a provider with a trustworthy configuration without interaction, by just selecting an attestation token. The attestation token is published by the provider once and does not have to be generated individually for every potential user. The job submission protocol then ensures that the provider can access the job only in the state considered trustworthy by the user. Current and future work include the implementation of job migration, the support for nodes joining and leaving the grid, and the integration of existing grid infrastructure into the TGA.

References

1. Foster, I., Kesselman, C., Tuecke, S.: The anatomy of the grid: Enabling scalable virtual organizations. *International Journal of Supercomputer Applications* 15, 200–222 (2001)
2. Foster, I., Kesselman, C., Tsudik, G., Tuecke, S.: A security architecture for computational grids. In: *Proc. 5th ACM Conference on Computer and Communications Security*, pp. 83–92 (1998)
3. Azzedin, F., Maheswaran, M.: Towards trust-aware resource management in grid computing systems. In: *Proc. 2nd IEEE International Symposium on Cluster Computing and the Grid*, pp. 452–457 (2002)
4. Hwang, K., Kwok, Y.K., Song, S., Chen, M.C.Y., Chen, Y., Zhou, R., Lou, X.: GridSec: Trusted grid computing with security bindings and self-defense against network worms and DDoS attacks. In: Sunderam, V.S., van Albada, G.D., Sloot, P.M.A., Dongarra, J.J. (eds.) *ICCS 2005*. LNCS, vol. 3516, pp. 187–195. Springer, Heidelberg (2005)

5. Fuggetta, A., Picco, G.P., Vigna, G.: Understanding code mobility. *IEEE Transactions on Software Engineering* 24, 342–361 (1998)
6. Mao, W., Jin, H., Martin, A.: Innovations for grid security from trusted computing (2005) Available online at <http://www.hpl.hp.com/personal/WenboMao/research/tcgridsec.pdf>
7. Smith, M., Friese, T., Engel, M., Freisleben, B.: Countering security threats in service-oriented on-demand grid computing using sandboxing and trusted computing techniques. *Journal of Parallel and Distributed Computing* 66, 1189–1204 (2006)
8. Intel Trusted Execution Technology Website: Intel trusted execution technology (2006) <http://www.intel.com/technology/security>
9. AMD Virtualization Website: Introducing AMD virtualization (2006) <http://www.amd.com/virtualization>
10. Sailer, R., Jaeger, T., Valdez, E., Cáceres, R., Perez, R., Berger, S., Griffin, J.L., van Doorn, L.: Building a MAC-based security architecture for the Xen open-source hypervisor. In: *Proc. 21st Annual Computer Security Applications Conference*, IEEE Computer Society, pp. 276–285. IEEE Computer Society Press, Los Alamitos (2005)
11. Nabhen, R., Jamhour, E., Maziero, C.: A policy based framework for access control. In: *Proc. 5th International Conference on Information and Communications Security*, pp. 47–59 (2003)
12. Garfinkel, T., Pfaff, B., Chow, J., Rosenblum, M., Boneh, D.: Terra: A virtual machine-based platform for trusted computing. In: *Proc. 19th ACM Symposium on Operating Systems Principles*, pp. 193–206 (2003)
13. Löhr, H., Ramasamy, H.V., Sadeghi, A.R., Schulz, S., Schunter, M., Stübke, C.: Enhancing grid security using trusted virtualization (extended version) (2007) <http://www.prosec.rub.de/publications.html>
14. TCG Website: TPM Specification version 1.2. (2006) Available online at <https://www.trustedcomputinggroup.org/specs/TPM>
15. Brickell, E., Camenisch, J., Chen, L.: Direct anonymous attestation. In: *Proc. ACM Conference on Computer and Communications Security*, pp. 132–145 (2004)
16. Sadeghi, A.R., Stübke, C.: Property-based attestation for computing platforms: caring about properties, not mechanisms. In: *Proc. 2004 New Security Paradigms Workshop*, pp. 67–77 (2004)
17. Rutkowska, J.: Blue pill. Presented at Syscan '06 (2006) <http://theinvisiblethings.blogspot.com/>
18. Pfitzmann, B., Riordan, J., Stübke, C., Waidner, M., Weber, A.: The PERSEUS system architecture. Technical Report RZ 3335 (#93381), IBM Research (2001)
19. OpenTC Website: The OpenTC project (2006) <http://www.opentc.net>
20. EMSCB Website: The EMSCB project (2006) <http://www.emscb.org>
21. Zhao, S., Lo, V., Gauthier-Dickey, C.: Result verification and trust-based scheduling in peer-to-peer grids. In: *Proc. 5th IEEE International Conference on P2P Computing*, pp. 31–38 (2005)
22. Cavalcanti, E., Assis, L., Gaudêncio, M., Cirne, W., Brasileiro, F., Novaes, R.: Sandboxing for a free-to-join grid with support for secure site-wide storage area. In: *Proc. 1st International Workshop on Virtualization Technology in Distributed Computing* (2006)
23. McCune, J.M., Jaeger, T., Berger, S., Cáceres, R., Sailer, R.: Shamon: A system for distributed mandatory access control. In: *Proc. 22nd Annual Computer Security Applications Conference*, pp. 23–32 (2006)
24. Sailer, R., Zhang, X., Jaeger, T., van Doorn, L.: Design and implementation of a TCG-based integrity measurement architecture. In: *Proc. Annual USENIX Security Symposium, USENIX*, pp. 223–238 (2004)

25. Jaeger, T., Sailer, R., Shankar, U.: PRIMA: policy-reduced integrity measurement architecture. In: Proc. 11th ACM Symposium on Access Control Models and Technologies, pp. 19–28 (2006)
26. Mao, W., Yan, F., Chen, C.: Daonity—grid security with behaviour conformity from trusted computing. In: Proc. 1st ACM Workshop on Scalable Trusted Computing (2006)
27. Cooper, A., Martin, A.: Trusted delegation for grid computing. In: Presented at: 2nd Workshop on Advances in Trusted Computing (2006)
28. Dinda, P.A.: Addressing the trust asymmetry problem in grid computing with encrypted computation. In: Proc. 7th Workshop on Languages, Compilers, and Run-Time Support for Scalable Systems, pp. 1–7 (2004)
29. Algesheimer, J., Cachin, C., Camenisch, J., Karjoth, G.: Cryptographic security for mobile code. Technical Report RZ 3302 (# 93348), IBM Research (2000)

A Wearable System for Outdoor Running Workout State Recognition and Course Provision

Katsuhiro Takata¹, Masataka Tanaka¹, Jianhua Ma¹, Runhe Huang¹,
Bernady O. Apduhan², and Norio Shiratori³

¹ Hosei University, Tokyo 184-8584, Japan
{i04t9002, n03k1120}@cis.k.hosei.ac.jp,
{jianhua, rhuang}@hosei.ac.jp

² Kyushu Sangyo University, Fukuoka 813-8503, Japan
bob@is.kyusan-u.ac.jp

³ Tohoku University, Sendai 980-5877, Japan
norio@shiratori.riec.tohoku.ac.jp

Abstract. The objective of this research is to develop a wearable prototype system to assist people doing outdoor running workout safely and effectively. One essential research issue is to correctly recognize a runner's state during a running workout process by analyzing contextual data obtained from sensors and GPS positioning device carried on by the runner. The running workout process is represented as a state transition diagram using the Space-Oriented Model. The state recognition is based on the state correlations with the runner's heart-beat rate and running speed. Our test results show that by utilizing the runner's state correlation, is more precise to recognize a runner's state as compared to the state judgment which is only based on detecting whether a sensed value exceeds some medical threshold value. Another function offered by the system is to provide the user an outdoor running course and may adjust the course according to the runner's speed, temperature, and course distance so as to advise the runner how to effectively burn the target amount of calories and safely achieve the exercise goal.

1 Introduction

Due to the fast progress of various sensors and corresponding technologies in processing sensed data, context-aware computing systems that can recognize and respond to users' conditions as well as its surrounding situations have been attracting more attention and research interests which offer many novel services in various fields [1-4]. It is a common sense though, that conducting regular physical exercises or workouts are very helpful in improving one's health and maintaining well-being. Thus, this research is focused on developing a ubiquitous system that uses wearable devices to assists people doing workouts, especially outdoor running workout.

At present, there are different kinds of sports machines that can facilitate users to do running workout. However, these existing sports machines can only record medical data of heartbeat rate, blood pressure and so on, and present these data and changes to the users. Actually, to make the workout more effective, a system should

be aware of the runner's state and accordingly adjust the workout state. Although some systems divide a running workout into several states, the state changes and their timing are fixed once a course is set by the user. In other words, with the existing systems the timing of the state change is decided according to the pre-defined course schedule without considering the actual user's body condition and real state. Therefore, if a runner wants to be in the same state, but the existing system may switch to another state following the pre-defined course, and so contradiction exists between the runner's real state and system working state. To solve this problem, it is necessary for the system to automatically recognize the user's actual state and correspondingly adjust its working state to adapt to the user's actual state.

This paper presents a workout state model represented by a state transition diagram, a state recognition using the state correlations of the current state data and the history workout data, and the running course generation to achieve a defined physical exercise goal. Hereon, the terms *runner* and *user*, and *heartbeat rate* and *heart rate*, are used interchangeably. In what follows, we first define the state transition diagram and analyze the changes of state, then explain the whole system for the state recognition and the course generation, and finally show our experiment results and conclusion.

2 The State Chart of Running Workout

To recognize a runner's situation, it is necessary to have a model to describe the running workout process. This model is represented with the runner's state transition diagram as shown in Fig. 1. It includes four main different states, namely:

- *Warming-Up*: A state to gradually ease the body into more intensive workout.
- *Main-Workout*: A state to get the body into some target workout intensity.
- *Cool-Down*: A state to slow down at the end of a workout to allow the body temperature and heart rate to decrease gradually.
- *Over-Training*: A state to get the body into some excessive intensity. If the intensity reaches a higher level over a specified value for a certain number of seconds, it is considered as overtraining. In this case, the system may advise the runner to slow down.

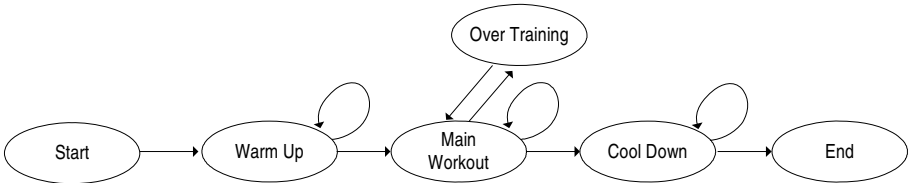


Fig. 1. The runner's state transition diagram with the four main states

In the running workout, it is assumed that the first three states exist inevitably and the change of state always happens in the same way, i.e., from *Warm-Up* to *Main-Workout*, and from *Main-Workout* to *Cool-Down* during the whole workout process. Based on the assumption that the present state is generally related to the runner's

heartbeat rate or other medical values, the runner's state can be recognized by comparing the closeness between the state-related set of template values and the set of measured values. In this paper, such closeness is based on the Space-Oriented Model [5] to calculate the *correlation* between the state template and measured time series data. That is to say, a state is a point in the state space and the closeness is referred to as the distance between the two state points in the state space. The correlation is further assumed that habitual things happened more than once before as shown in Fig. 2. The typical template values of the user's states and their orders are shown on the left side of the figure. Different people may have relatively different values depending on their ages, health conditions, habits, and other aspects. In other words, a set of template values varies upon individuals. However, it can be extracted from personal history running workouts.

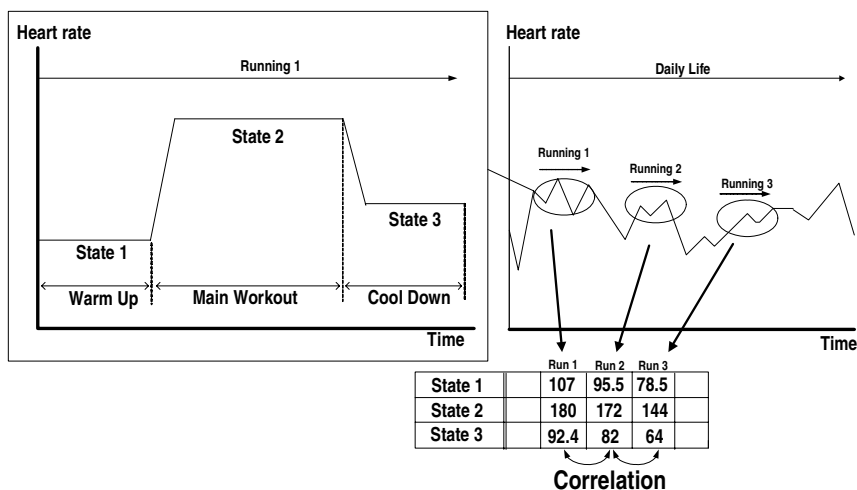


Fig. 2. Based on the state transition assumption, past running workouts in time series data are extracted by superimposing the extracted data

Therefore, the processing with this state chart is based on the correlation of the user's average records of training habits in daily life. The sensed data acquisitions and analysis are explained in detail in the next section.

3 Running Workout State Recognition

To recognize the four main states (*Warming-Up*, *Main-Workout*, *Cool-Down*, and *Over-Training*) in a running workout, we adopt the DP (dynamic programming) matching method [6], because this method is one of the most fundamental techniques for various time series analysis in pattern recognition, image processing, etc. To get the running characteristic pattern data, a set of wearable devices, i.e., heartbeats-sensor-watch, is used to acquire the user's heartbeat rate and running speed during workout. To detect which state a user is in, the sensed data will be compared with the

typical state pattern of the user's own time chart, which is gotten based on his/her past exercise records. To record the running activities, the user, aside from carrying with him the medical and motion sensor devices, he has with him a GPS receiver for getting location data and a PDA or cellular phone for data processing and communications. The whole system is shown in Fig. 3.

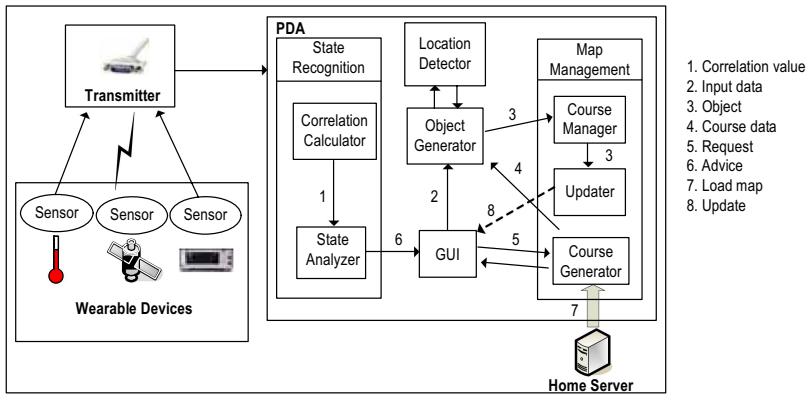


Fig. 3. Some sensors carried by the runner to sense his/her state contextual data. Before a workout, the system loads XML data from a home server which provide the relay points along the running course, and the course map is displayed on the PDA/cell phone screen.

The types of sensors used in the current system and their purpose are listed below.

- *Heart rate sensor:* To sense the runner's heart rate for state judgment. The training heart rate is also used to calculate the predictive maximum heart rate using Karvonen formula [7-9] below.

$$\text{predicted\_max\_heart\_rate} = 208 - 0.7 * \text{age} \tag{1}$$

Moreover, the heart rate plays a vital role in judging whether the user's condition is safe or not. The heart rate monitoring set used in our experiment, as shown in Fig. 4, consists of a sensor to sense the heart rate, and a watch (also equipped with receiving and transmitting functions) to receive and send data to a PDA.



Fig. 4. The heart rate monitor. It can sense the heart rate via a small chest sensor worn around the chest and send the data using wireless technology to the watch. The watch terminal can record continuous heart rate during the workout.

- *Speed sensor*: To sense running speed for the runner's state judgment. The speed is used to judge a runner's state when he/she is not walking nor running, such as taking a rest or standing due to a red light traffic signal in road crossing. Such a sensor used in our system is shown on the left side of Fig. 5.
- *GPS receiver*: To sense the runner's location information and also judge when a runner is getting close to a relay point. It is carried by a user as shown in the middle of Fig. 5.
- *Motion sensor*: To detect if the runner is doing workout normally. In doing so, the user's safety is monitored. For example, when a user suddenly faints, abrupt changes on the value can be detected and then some emergency action can be taken. Furthermore, a lot of researches on biomechanics for scientific-based physical exercise have been studied [6], so motion sensors can be used in this case and in many other ways [10-12]. It is mounted on the head as shown on the right side of Fig. 5.
- *Thermometer sensor*: To sense the ambient/body temperature which is used to judge if the temperature is suitable enough for the runner to do running workout or not. For example, if the environment temperature or runner's body temperature is too high, the system warns the runner to stop doing the workout [7, 13].



Fig. 5. The left image is a speed sensor which measures the accurate running speed/pace and distance. This sensor sends the data via wireless communication to the watch terminal as shown in Fig. 4. The center image is a GPS sensor device. This device can sense geographical location coordinate values sent to the PDA. The right image is a 3D motion sensor sensing motion data (x, y and z directions) that are also sent to the PDA.

In addition to the sensors listed above, a lot of other wearable sensors that can sense human's biological information have been developed in recent years, for example, wearable oxygen sensor [7]. Combining this with other different sensors will enable the system to get more precise biological information about the user during workouts. Before running, a user is asked to enter some values to the system, such as a certain workout level defined in the system, his/her age, weight (kg), the desired amount of calories (kcal) to be burned during the workout, and the distance (m) that the user can or could run at his/her best within twelve minutes. At each level, the corresponding workout intensity (%) is defined based on data proven in sports medical science. By inputting the workout level and age, the increased heart rate is also decided by the data relationship between the workout intensity which is equivalent to the maximum expected oxygen intake (called % VO_{2_max}) and heart rate during the

workout, according to Table 1 [13]. After that, the user's expected running pace (m/min) is decided by the data relationship between % VO2 and the result of 12 minutes running test (as a calculation reference), as shown in Table 2 [13].

Table 1. Relationship between workout intensity and heart rate

	Workout intensity (maximum oxygen uptake (%))	Age					Level
		20 29	30 39	40 49	50 59	Over 60	
Heart beat (beats/minutes)	100	190	185	175	165	155	Level 7
	90	175	170	165	155	145	Level 6
	80	165	165	150	145	135	Level 5
	70	150	145	140	135	125	Level 4
	60	135	135	130	125	120	Level 3
	50	125	120	115	110	110	Level 2
	40	110	110	105	100	100	Level 1

Table 2. Paces associated with intensity (%) and 12 minutes running test

12 minutes test running (m) \ maximum oxygen uptake (%)	40	50	60	70	80	90	100
1200	50	60	70	85	95	110	120
1300	50	65	80	90	105	115	130
1400	55	70	85	95	110	125	140
1500	60	75	90	100	115	130	145
1600	60	80	95	110	125	140	155
1700	65	85	100	115	130	150	165
1800	70	90	105	125	140	160	175
1900	75	95	110	130	150	165	185
2000	75	95	115	135	155	170	190
2100	80	100	120	140	160	180	200
2200	85	105	125	145	170	190	210
2300	90	110	130	155	175	200	220
2400	90	115	140	160	185	205	230
2500	95	120	140	165	190	210	235
2600	100	125	145	170	195	220	245
2700	100	130	155	180	205	230	255
2800	105	135	160	185	210	240	265
2900	110	140	165	195	220	250	275
3000	110	140	170	200	225	250	280
3100	115	145	175	205	230	260	290
3200	120	150	180	210	240	270	300
3300	125	155	185	215	250	280	310
3400	130	160	190	225	255	290	320
3500	130	165	195	230	260	295	325
3600	135	170	200	235	270	300	335

The running_pace (m/min) can be converted into oxygen intake that is called VO2 (ml/(kg, min)) using the calculation formula [13] below.

$$VO2 = \text{running\_pace} * 0.2 + 3.5 \tag{2}$$

VO2 can also be converted to METs (metabolic equivalents), that is, the ratio of the metabolic rate of the average person while seated and resting, to the metabolic rate of a particular person while performing some tasks, using the formula [14] below.

$$MET_s = VO2 / 3.5 \tag{3}$$

Finally, the distance that the user has to run to burn the target amount of calories is calculated by using expected running_pace (m/min), weight (kg), calories (kcal) and METs as the following calculation formula [7] shows.

$$\text{running\_dist} = (\text{calories} * \text{running\_pace}) / (MET_s * \text{weight}) \tag{4}$$

4 Running Course Provision and Experiment

To help users effectively conduct running workout activities, our system can provide the running courses according to his/her requirements and conditions. The course related map data is stored in a home server. The available relay points are preset, and the surrounding locations such as longitude, latitude and relay point numbers are kept in an XML file, as shown in Fig. 6.

```
<?xml version="1.0" encoding="euc-jp" standalone="yes"?>
<map>
  <point latitude="35.709611"
        longitude="139.521703">start/goal</point>
  <point latitude="35.711306"
        longitude="139.521619">relay1</point>
  <point latitude="35.704953"
        longitude="139.521915">relay2</point>
</map>
```

Fig. 6. The location information of relay points using the XML format for course generation

An initial running course is determined with the following steps. First, the above XML based file and a map around the runner are loaded from a map pool stored in a server. Then, the system chooses some relay points randomly so that the total distance from the starting point will be equivalent to the distance calculated by the above formulas in advance of a running workout, and connects them from the starting point to the other relay points to present a running course. The initial course is a rough estimation before the workout to let the runner burn his/her target amount of calories, and may be adjusted later on according to the user's running states or as the situation changes. While the runner is running, the system compares the accumulated burned calories with the target amount of calories to be burned that is calculated and reset as the runner gets to each relay point. If the accumulated burned calories is much less than the target amount of calories, the system calculates its difference, which can be converted to the distance based on an average running pace at the time. The converted distance will be added to the original/previous running course. That is, a new relay point will be searched and added. To change the relay point, however, the system has to get the information about which point a runner is at and whether the runner is approaching the goal point or not. Depending on the runner's burned calories, the running course can be dynamically adjusted by adding some additional relay points, as shown in Fig. 7.

The computers and devices used in our system and experiments are listed below.

- *Server:* Windows XP, Pentium (R) 4 3.19GHz, 1.49 GB (RAM), Java™ 2 JRE
- *PDA:* Zaurus, SL-C3100, Sharp
- *Heart Rate Sensor:* Polar S625X
- *Speed Sensor:* Polar S1, Running Speed-Distance Sensor
- *3D Motion Sensor:* NEC Tokin MDP-A3U9S6
- *GPS Receiver:* EMTAC Technology Corp. BT-GPS
- *Temperature Sensor:* Digi WatchPort USB

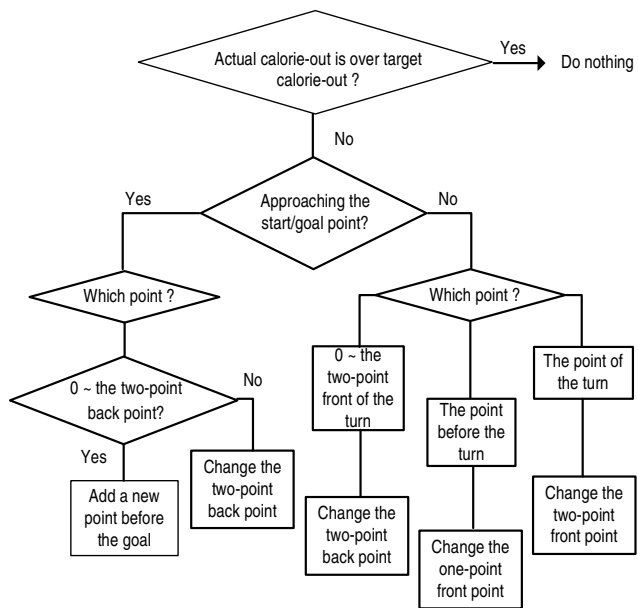


Fig. 7. The relay point change flow-diagram. This chart provides the clear objective for a user to be able to definitely burn his/her desired amount of calories.

Our system requires some personal information to be inputted by the user as shown in Fig. 8, e.g., the training level, the heart rate when taking a rest, the distance covered during the 12 minutes running test, age, weight, and the target amount of calories to be burned. With the inputted information, the system calculates the necessary calories to be burned, which can be converted into some running distance.

The screenshot shows a software window titled "Request" with a blue title bar. Inside, there are several input fields and buttons. At the top left, there is a "Level" dropdown menu with "Level4" selected. To its right is a "Resting Heart Rate" text box containing the number "58". Further right is a "12 Minutes Running Test" dropdown menu with "2800m" selected. Below these, there are three more input fields: "Age" with "21", "Weight(kg)" with "58", and "Target Burn Calories (kcal)" with "450". A "Request" button is located below these fields. At the bottom of the window, there is a large empty text box.

Fig. 8. The user condition and biological values are inputted via this window

The example course in our experiment was tested based on the relay points that were preset around the Koganei Campus of Hosei University. Fig. 9 shows the preset points and their connections on the map made by the Geographical Survey Institute of Japan. On the right side of the map, a runner's sign is shown and it moves as the user walks or runs. The user's current position is obtained via the processed sensed GPS data. This makes the user aware of where he/she is running.

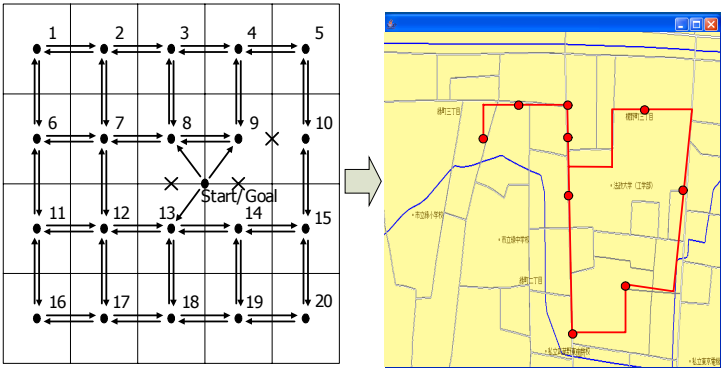


Fig. 9. This window shows the running course based on the user’s requirement

We performed some tests whether the system can recognize the runner’s state using actual workout data acquired from the above sensors for three persons who do physical exercises often, sometimes, and seldom, respectively. Due to space limitation, we only show the state recognition results of one person who has done exercises often in Fig. 10. The changes in state were correctly detected by finding abrupt changes of correlation values. A user may stop running, e.g., at crossroads due to a red light traffic signal during the main workout period. In such case, no state change will be judged until the runner resumes running or after a certain number of seconds had passed.

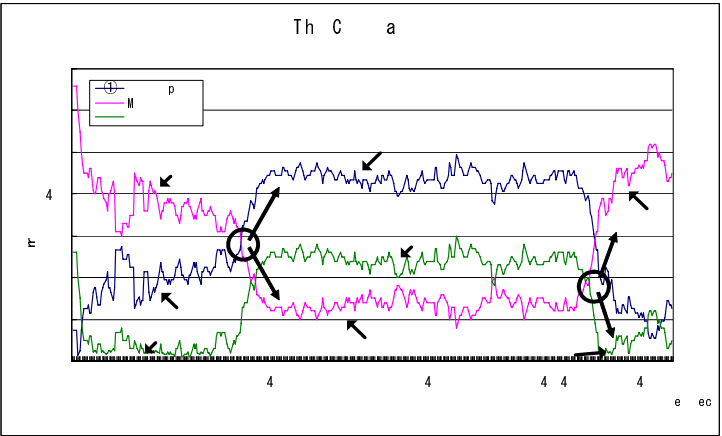


Fig. 10. The correlation values and changes in different workout states

5 Conclusion and Future Work

In this paper, we present a wearable system for outdoor running workout state recognition and running course provision. The system is able to dynamically analyze the

running states by processing the sensed information from the wearable sensors. Our processing method is based on the running state diagram during the workout process, which are divided into three main states according to the heart rate values. This makes it possible to judge a state by comparing which state the runner's present condition is closest to. From our test results, it showed that the system can recognize the runner's states and the state transition to some extent, using both the correlation of heart rate and actual running speed at that time.

One issue to further improve the system is the map service. If there's a map with geographical data, a course can be made along roads and the distance can be calculated at the same time as well. To do so, one possible approach is to use other available research results, e.g., combining the state recognition and map navigation, for the system to provide more effective running workouts.

References

1. Jain, R.: Digital Experience. *Communications of ACM* 44(3), 38–40 (2001)
2. Croft, W.B.: Information Retrieval and Computer Science: An Evolving Relationship. In: *Proceedings of International Conference of SIGIR*, pp. 2–3 (2003)
3. Gray, J.: What Next? A Dozen Information Technology Research Goal. *Journal of the ACM* 50(1), 41–57 (2003)
4. Zhang, Y., Zhou, Y.: Transparent Computing: A New Paradigm for Pervasive Computing. In: Ma, J., Jin, H., Yang, L.T., Tsai, J.J.-P. (eds.) *UIC 2006*. LNCS, vol. 4159, pp. 1–11. Springer, Heidelberg (2006)
5. Takata, K., Ma, J.: A Situation Analysis Architecture for Ubiquitous Outdoor Safety Awareness Using a Space-oriented Model. In: *Proc. of the SICE Annual Conference 2005*, pp. 1605–1610 (August 2005)
6. Milios, E., Petrakis, E.G.M.: Shape Retrieval Based on Dynamic Programming. *IEEE Trans. on Image Processing* 9(1), 141–147 (2000)
7. Hirakiba, K.: Long-distance Runner's Physiological Sciences (in Japanese), Kyorin Shoin Publication (2004) ISBN: 4764410702
8. Nielen, H., Boisset, V., Masquelier, E.: Fitness and Perceived Exertion in Patients with Fibromyalgia Syndrome. *Clin. J. Pain.* 16, 209–213 (2000)
9. Strath, S.J., Swartz, A.M., Bassett Jr., D.R., O'Brien, W.L., King, G.A., Ainsworth, B.E.: Evaluation of Heart Rate as a Method for Assessing Moderate Intensity Physical Activity. *Medicine and Science in Sports and Exercise* 32(9), 465–470 (2000)
10. Tenmoku, R., Kanbara, M., Yokoya, N.: A Wearable Augmented Reality System for Navigation using Positioning Infrastructures and a Pedometer. In: *Proc. 2nd IEEE/ACM Int. Sympo. on Mixed and Augmented Reality*, pp. 344–345 (2003)
11. Lee, S.W., Mase, K., Kogure, K.: Detection of Spatio-Temporal Gait Parameters by Using Wearable Motion Sensors. In: *Proceedings of The IEEE Engineering in Medicine and Biology 27th Annual Conference*, Shanghai, China, pp. 6836–6839 (September 2005)
12. Kondo, T., Kanosue, K.: Functional Association between Body Temperature and Exercise (in Japanese). *Physical Science* 54(1), 19–20 (2005)
13. Onuki, Y.: *Sports Medical Sciences* (in Japanese), Chuo Hohki. Publication, pp. 143–145 (1999) ISBN: 4805818166
14. Guezennec, C.Y., Vallier, J.M., Bigard, A.X., Durey, A.: Increase in Energy Cost of Running at the End of a Triathlon. *European Journal of Applied Physiology and Occupational Physiology* 5(73), 440–444 (1996)

Malicious Participants in Group Key Exchange: Key Control and Contributiveness in the Shadow of Trust

(Extended Abstract)

Emmanuel Bresson¹ and Mark Manulis^{2,*}

¹ DCSSI Crypto Lab Paris
emmanuel@bresson.org

² Horst Görtz Institute, Ruhr-University of Bochum, Germany
mark.manulis@nds.rub.de

Abstract. Group key exchange protocols allow their participants to compute a secret key which can be used to ensure security and privacy for various multi-party applications. The resulting group key should be computed through cooperation of all protocol participants such that none of them is trusted to have any advantage concerning the protocol's output. This trust relationship states the main difference between group key exchange and group key transport protocols. Obviously, misbehaving participants in group key exchange protocols may try to influence the resulting group key, thereby disrupting this trust relationship, and also causing further security threats. This paper analyzes the currently known security models for group key exchange protocols with respect to this kind of attacks by malicious participants and proposes an extended model to remove the identified limitations. Additionally, it proposes an efficient and provably secure generic solution, a compiler, to guarantee these additional security goals for group keys exchanged in the presence of malicious participants.

1 Introduction

The establishment of group keys is fundamental for a variety of security mechanisms in group applications. For example, group keys can be utilized by symmetric encryption schemes for the purpose of confidentiality which is one of the most frequent security requirements in group applications. Two different classes of protocols can be identified: (*group*) *key transport* (GKT), in which the key is chosen by a single party and transmitted to the other parties via secure channels, and (*group*) *key exchange* (GKE), in which all parties interact in order to compute the key. In GKE protocols, no secure channels are needed and, more important, no party is allowed to choose the key on behalf of the group: in other words, group members do not trust each other. This provides the background and motivation for considering malicious participants in such protocols and for defining in a formal way what security means in that case. Such formalization is one of the main goals of this paper.

* The corresponding author was supported by the European Commission through IST-2002-507932 ECRYPT.

In the paradigm of *provable security*, security analysis must hold in some formal security model. The first such model for GKE protocols (which we refer to as the BCPQ model) was introduced by Bresson *et al.* in [8], based on earlier work by Bellare and Rogaway [2,3], and with subsequent variants and refinements [7,18,19]; we refer to [22] for a survey. These models mainly focused on the following two notions: *authenticated key exchange* (AKE) security which requires the indistinguishability of computed group keys from random keys, and *mutual authentication* (MA) which means that any two parties authenticate bilaterally and actually compute the same key.

A number of papers [1,12,18,25] point out that the consideration of dishonest participants (either curious or malicious) is of prime importance in the group setting, because they can have catastrophic effects on the protocol security; e.g., Choo *et al.* [12] noticed that some protocols proven secure in the BCPQ-like models are vulnerable to *unknown key-share attacks*, in which the attacker is *believed* (from some participant's view) to be a group member.

Michel *et al.* in [25] first mentioned the issue of *key control* by which a misbehaving participant can influence the value of the key. A related notion called *contributiveness* was proposed by Ateniese *et al.* [1] requiring that all protocol participants equally contribute to the computation of the group key. These requirements implicitly state a difference between GKT and GKE protocols. The main reason is that key control and contributiveness assume that none of the protocol participants is trusted to choose the group key on behalf of other participants. However, the way towards formal definitions of these requirements is not obvious. A weaker model (as in [6]) would consider honest participants that have biased pseudo-random generators and a curious adversary obtaining some extra information about the key. In this paper we consider a stronger setting (in spirit of [4]), where malicious participants try to influence honest participants computing some special value as a group key (including the so-called *key replication attacks* [21]).

In addition to usual corruptions where the adversary obtains full control over the parties we also consider *strong corruptions* [7,26,27], that is, capabilities of the adversary to reveal internal memory of participants. We also consider strong corruptions in the context of a curious adversary that reveals (but not modifies) ephemeral secrets of honest participants. Currently, security against strong corruptions is considered in a rather restrictive way, as part of the *strong forward secrecy* requirement in the context of AKE-security [7]. In order to talk about security of GKE protocols against strong corruptions in general we expand these considerations for other requirements within our security model.

Contributions and Organization. This paper provides an extended treatment of security of GKE protocols in the presence of malicious participants and strong corruptions. In other words, we formally define what a “secure group key” means in such scenario. As a starting motivation, in Sections 2 and 2.1 we first discuss why currently known security models for GKE protocols are not mature enough to deal with malicious participants and strong corruptions. Then, in Section 3 we extend the notions of AKE- and MA-security and propose a new definition of contributiveness.

In Section 4 we describe the relationship between our formal definitions of MA-security and contributiveness through some informally stated requirements from the

previous literature. To prove the soundness and feasibility of our extensions, in Section 5 we propose a generic solution (compiler) which turns any AKE-secure GKE protocol into an enhanced protocol, which provably satisfies our advanced security requirements under standard cryptographic assumptions.

2 Related Work

General Security Notions for GKE Protocols. AKE-security as defined in [7, 8, 19] subsumes some informal security goals defined in the literature: *key secrecy* [14] or *implicit key authentication* [24] which ensures that no party except for legitimate participants learns the established group key, *security against impersonation attacks* [11] or a related notion of *entity authentication* [2] requiring that an adversary must not be able to replace an honest participant in the execution of the protocol; *resistance against known-key attacks* [10, 28] meaning that an adversary knowing group keys of previous sessions cannot compute subsequent session keys, *key independence* [20] meaning that an adversary knowing a proper subset of group keys must not be able to discover any other group keys. Also it subsumes (*perfect*) *forward secrecy* [14, 17, 24] which requires that the disclosure of long-lived keys must not compromise the secrecy of the previously established group keys. The latter can be strengthened by the requirement of *strong forward secrecy* [7] in which the adversary in addition to the long-lived keys reveals the internal data of participants such as ephemeral secrets used during the protocol execution.

The currently available formal definition of MA-security in [8] has been designed to cover the informal definitions of *key confirmation* [24, § 12.2] which combined with *mutual authentication* [2] ensures that all identified protocol participants have actually computed the same group key (this is also known as *explicit key authentication* [24, § 12.2]).

According to [12], however, these definitions do not consider security against *unknown key-share attacks* [5, 14], in which a corrupted participant can make an honest participant believe that the key is shared with one party though in fact it is shared with another party.

Informal Security Treatment of Key Control and Contributiveness. There have been only few attempts to handle malicious participants in GKE protocols. Misbehavior of protocol participants was mentioned first in [25]: the authors described the issue of *key control*. Independently, Ateniese *et al.* [1] introduced a more general notion of *unpredictability* (which intuitively implies security against key control). Further, they proposed a related notion called *contributory* group key agreement: the property by which each participant equally contributes to the resulting group key and guarantees its freshness. Moreover, they defined *verifiable contributory* GKE protocols where each participant must be assured of every other participant's contribution. Some subsequent security models have tried to formalize this approach.

The KS Model. Katz and Shin [18] proposed security definitions against malicious participants in a BCPQ-like model: briefly speaking any user U_i may have many instance oracles Π_i^s , $s \in \mathbb{N}$. Each oracle represents U_i in one of many possible concurrent

protocol executions. All participants of the same protocol execution are considered as *partners*. First, the KS model says that $\mathcal{A}$ impersonates (uncorrupted) U_j to (accepting) Π_i^s if U_j belongs to the (expected) partners of Π_i^s , but in fact no oracle Π_j^t is partnered with Π_i^s . In other words, the instance Π_i^s computes the session key and U_i believes that U_j does so, but in fact an adversary has participated in the protocol on behalf of U_j . Then, the authors call a protocol secure against *insider impersonation attacks* if for any party U_j and any instance Π_i^s , the adversary cannot impersonate U_j to Π_i^s , under the (stronger) condition that neither U_j nor U_i is corrupted at the time Π_i^s accepts.

The BVS Model. Bohli *et al.* [4] proposed another extension (which we refer to as the BVS model) towards security goals in the presence of malicious participants. The process dealing with key control and contributiveness, at an informal level, runs as follows. In a first stage, the adversary $\mathcal{A}$ interacts with the users and may corrupt some of them; $\mathcal{A}$ then specifies an unused instance oracle Π_i^s and a subset K in the session key space $\mathcal{K}$. In the second stage, the adversary tries to make Π_i^s accept a session key $k \in K$ but is not allowed to corrupt U_i . The BVS model defines a GKE protocol as being *t-contributory* if the adversary succeeds with only negligible probability, with the total number of corruptions remains (strictly) less than t . A *n-contributory* protocol between n participants is called a *key agreement*.

2.1 Discussion on the KS and BVS Models

Missing Key Control and Contributiveness in the KS Model. Katz and Shin proposed a compiler to turn any AKE-secure protocol (in the sense of BCPQ) into a protocol secure in their extended model. In the following we illustrate how malicious participants can predict the resulting value of the group key, that is, the KS model does not provide key control and contributiveness.

The KS compiler uses a pseudo-random function f_k with $k \in_R \{0, 1\}^\kappa$ and runs (intuitively) as follows. Each player acknowledges the key k_i obtained in the input protocol by signing and sending a token $\text{ack}_i := f_{k_i}(v_0)$ where v_0 is a public value. If all verifications match, players terminate the compiled protocol with key $K_i := f_{k_i}(v_1)$ where $v_1 \neq v_0$ is another public value. We argue that the compiler does not ensure unpredictability and contributiveness since K_i may be predictable as soon as k_i is, and K_i is not composed of each participant's contribution if k_i is not.

Absence of Strong Corruptions in the BVS Model. A first drawback of the BVS model is that $\mathcal{A}$ is not adaptive in her choice of Π_i^s , because she is required to commit to it in the first stage (not in the second). The second, main drawback is that strong corruptions are not allowed: therefore, contributiveness does not capture attacks in which $\mathcal{A}$ tries to influence the session key using the (passive) knowledge of the internal states of the honest oracles (but without knowing their long-term secrets).

3 Our Extended Security Model

In the following we propose a security model for GKE protocols that includes extended security definitions concerning MA-security and contributiveness, while taking into

account strong corruptions. Similar to [4, 18, 19] our model assumes that the communication channel is fully controlled by the adversary which can simply refuse to deliver protocol messages (even those originated by honest participants). Therefore, our definitions do not deal with the denial-of-service attacks and fault-tolerance issues but rather aim to recognize that the actual protocol execution deviates from the original specification and prevent that an honest participant accepts a “biased” group key.

3.1 Protocol Participants, Variables

Users, Instance Oracles. Similar to [7] $\mathcal{U}$ is a set of N users. Each user $U_i \in \mathcal{U}$ holds a long-lived key LL_i . In order to handle participation of U_i in distinct concurrent protocol executions we consider that U_i has an unlimited number of instances called *oracles*; Π_i^s , with $s \in \mathbb{N}$, denotes the s -th instance oracle of U_i .

Internal States. Every Π_U^s maintains an *internal state information* state_U^s which is composed of all private, ephemeral information used during the protocol execution. The long-lived key LL_U is, in nature, excluded from it (moreover the long-lived key is specific to the user, not to the oracle).

Session Group Key, Session ID, Partner ID. In each session we consider a new group $\mathcal{G}$ of $n \in [1, N]$ participating oracles. Each oracle in $\mathcal{G}$ is called a *group member*. By $\mathcal{G}_i$ for $i \in [1, n]$ we denote the index of the user related to the i -th oracle involved in $\mathcal{G}$ (this i -th oracle is denoted $\Pi(\mathcal{G}, i)$). Thus, for every $i \in [1, n]$ there exists $\Pi(\mathcal{G}, i) = \Pi_{\mathcal{G}_i}^s \in \mathcal{G}$ for some $s \in \mathbb{N}$. Every participating oracle $\Pi_U^s \in \mathcal{G}$ computes the *session group key* $k_U^s \in \{0, 1\}^\kappa$. Every session is identified by a unique *session id* sid_U^s . This value is known to all oracles participating in the same session. Similarly, each oracle $\Pi_U^s \in \mathcal{G}$ gets a value pid_U^s that contains the identities of participating users (including U), or formally

$$\text{pid}_U^s := \{U_{\mathcal{G}_j} \mid \Pi(\mathcal{G}, j) \in \mathcal{G}, \forall j = 1, \dots, n\}.$$

We say that two oracles, $\Pi_i^{s_i}$ and $\Pi_j^{s_j}$, are *partnered* if $U_i \in \text{pid}_j^{s_j}$, $U_j \in \text{pid}_i^{s_i}$, and $\text{sid}_i^{s_i} = \text{sid}_j^{s_j}$.

Instance Oracle States. An oracle Π_U^s may be either *used* or *unused*. The oracle is considered as unused if it has never been initialized. Each unused oracle Π_U^s can be initialized with the long-lived key LL_U . The oracle is initialized as soon as it becomes part of some group $\mathcal{G}$. After the initialization the oracle is marked as used, and turns into the *stand-by* state where it waits for an invocation to execute a protocol operation. Upon receiving such invocation the oracle Π_U^s learns its partner id pid_U^s (and possibly sid_U^s) and turns into a *processing* state where it sends, receives and processes messages according to the description of the protocol. During the whole processing state the internal state information state_U^s is maintained by the oracle. The oracle Π_U^s remains in the processing state until it collects enough information to compute the session group key k_U^s . As soon as k_U^s is computed Π_U^s *accepts* and *terminates* the protocol execution meaning that it would not send or receive further messages. If the protocol execution fails (due to any adversarial actions) then Π_U^s terminates without having accepted, i.e., the session group key k_U^s is set to some undefined value.

3.2 Definition of a Group Key Exchange Protocol

Definition 1 (GKE Protocol). A group key exchange protocol $\mathcal{P}$ consists of the key generation algorithm KeyGen , and a protocol Setup defined as follows:

$\mathcal{P}.\text{KeyGen}(1^\kappa)$: On input a security parameter 1^κ each user in $\mathcal{U}$ is provided with a long-lived key LL_U .

$\mathcal{P}.\text{Setup}(\mathcal{S})$: On input a set $\mathcal{S}$ of n unused oracles a new group $\mathcal{G}$ is created and set to be $\mathcal{S}$, then a probabilistic interactive protocol is executed between oracles in $\mathcal{G}$.

We call $\mathcal{P}.\text{Setup}$ an *operation*. We say that a protocol is *correct* if all oracles in $\mathcal{G}$ accept with the same session group key k . We assume it is the case for all protocols in this paper.

3.3 Adversarial Model

Queries to the Instance Oracles. We now consider an adversary $\mathcal{A}$ which is a Probabilistic Polynomial-Time (PPT) algorithm having complete control over the network. $\mathcal{A}$ can invoke protocol execution and interact with protocol participants via queries to their oracles.

$\text{Execute}(\mathcal{S})$: This query models $\mathcal{A}$ eavesdropping the honest operation execution of $\mathcal{P}.\text{Setup}$. $\mathcal{P}.\text{Setup}(\mathcal{S})$ is executed and $\mathcal{A}$ is given the transcript of the execution.

$\text{Send}(\Pi_U^s, m)$: This query models $\mathcal{A}$ sending messages to the oracles. $\mathcal{A}$ receives the response which Π_U^s would have generated after having processed the message m according to the description of $\mathcal{P}$. The adversary can ask an oracle Π_U^s to invoke $\mathcal{P}.\text{Setup}$ with the oracles in $\mathcal{S}$ via the query of the form $\text{Send}(\Pi_U^s, \mathcal{S})$ which gives $\mathcal{A}$ the first message that Π_U^s would generate in this case. Thus, using Send queries the adversary can actively participate in $\mathcal{P}.\text{Setup}$.

$\text{RevealKey}(\Pi_U^s)$: $\mathcal{A}$ is given the session group key k_U^s . This query is answered only if Π_U^s has accepted.

$\text{RevealState}(\Pi_U^s)$: $\mathcal{A}$ is given the internal state information state_U^s .

$\text{Corrupt}(U)$: $\mathcal{A}$ is given the long-lived key LL_U .

$\text{Test}(\Pi_U^s)$: This query is used to model the AKE-security of a protocol. It can be asked by $\mathcal{A}$ as soon as Π_U^s accepts, but only once in $\mathcal{A}$'s execution. The query is answered as follows: The oracle generates a random bit b . If $b = 1$ then $\mathcal{A}$ is given k_U^s , and if $b = 0$ then $\mathcal{A}$ is given a random string.

We say that Π_U^s is a *malicious participant* if the adversary has previously asked the $\text{Corrupt}(U)$ query. In all other cases Π_U^s is *honest*. We say that the adversary is *curious* if it asks a $\text{RevealState}(\Pi_U^s)$ query for some honest Π_U^s . This is possible since long-lived keys are separated from the ephemeral secrets stored in state_U^s .

3.4 Security Goals

AKE-Security with Strong Forward Secrecy. As defined in [7] strong forward secrecy states that AKE-security of previously computed session keys is preserved if the adversary obtains long-lived keys of protocol participants and internal states of their oracles in later protocol sessions.

Definition 2 (Oracle Freshness). *In the execution of $\mathcal{P}$ the oracle Π_U^s is fresh if **all** of the following holds:*

1. *no $U_i \in \text{pid}_U^s$ is asked for a Corrupt query prior to a query of the form $\text{Send}(\Pi_j^{s_j}, m)$ such that $U_j \in \text{pid}_U^s$ before Π_U^s and all its partners accept,*
2. *neither Π_U^s nor its partners are asked for a RevealState query before Π_U^s and all its partners accept,*
3. *neither Π_U^s nor any of its partners is asked for a RevealKey query after having accepted.*

We say that a session is fresh if all participating oracles are fresh.

Note that in our model each Π_U^s is bound to one particular protocol execution (session). Thus, Π_U^s remains fresh if RevealState and RevealKey queries have been asked to other oracles owned by U , that is to oracles participating in other sessions. Hence, in contrast to [7] (and [19]) our definition allows the adversary to obtain knowledge of internal states from earlier sessions too.

Definition 3 (AKE-Security). *Let $\mathcal{P}$ be a GKE protocol from Definition 1 and b a uniformly chosen bit. Consider an active adversary $\mathcal{A}$ participating in game $\text{Game}_{\text{sfs}, \mathcal{P}}^{\text{ake}-b}(\kappa)$ defined as follows:*

- *after initialization, $\mathcal{A}$ interacts with instance oracles using queries;*
- *at some point $\mathcal{A}$ asks a Test query to a **fresh** oracle Π_U^s which has accepted, and receives either $k_1 := k_U^s$ (if $b = 1$) or $k_0 \in_R \{0, 1\}^\kappa$ (if $b = 0$);*
- *$\mathcal{A}$ continues interacting with instance oracles;*
- *when $\mathcal{A}$ terminates, it outputs a bit trying to guess b .*

The output of $\mathcal{A}$ is the output of the game. The advantage function (over all adversaries running within time κ) in winning this game is defined as:

$$\text{Adv}_{\text{sfs}, \mathcal{P}}^{\text{ake}}(\kappa) := \left| 2 \Pr[\text{Game}_{\text{sfs}, \mathcal{P}}^{\text{ake}-b}(\kappa) = b] - 1 \right|$$

A GKE protocol $\mathcal{P}$ is AKE-secure with strong forward secrecy (AGKE-sfs) if this advantage is negligible.

MA-Security. Our definition of mutual authentication security differs from the one in [7, 8] which does not consider malicious participants and curious adversaries and is vulnerable to unknown key-share attacks.

Definition 4 (MA-Security). *Let $\mathcal{P}$ be a correct GKE protocol and $\text{Game}_{\mathcal{P}}^{\text{ma}}(\kappa)$ the interaction between the instance oracles and an active adversary $\mathcal{A}$ who is allowed to query Send, Execute, RevealKey, RevealState, and Corrupt. We say that $\mathcal{A}$ wins if at some point during the interaction there exist an uncorrupted user U_i whose instance oracle $\Pi_i^{s_i}$ has accepted with $k_i^{s_i}$ and another user U_j with $U_j \in \text{pid}_i^{s_i}$ that is uncorrupted at the time $\Pi_i^{s_i}$ accepts, such that*

1. it exists **no** instance oracle $\Pi_j^{s_j}$ with $(\text{pid}_j^{s_j}, \text{sid}_j^{s_j}) = (\text{pid}_i^{s_i}, \text{sid}_i^{s_i})$, **or**
2. it exists **an** instance oracle $\Pi_j^{s_j}$ with $(\text{pid}_j^{s_j}, \text{sid}_j^{s_j}) = (\text{pid}_i^{s_i}, \text{sid}_i^{s_i})$ that accepted with $k_j^{s_j} \neq k_i^{s_i}$.

The maximum probability of this event (over all adversaries running within time κ) is denoted $\text{Succ}_P^{\text{ma}}(\kappa)$. We say that a GKE protocol P is MA-secure (MAGKE) if this probability is a negligible function of κ .

Note that U_i and U_j must be uncorrupted, however, $\mathcal{A}$ is allowed to reveal internal states of their oracles.

Contributiveness. In the following we propose a definition which deals with the issues of key control, contributiveness and unpredictability of session group keys in case of strong corruptions; this, again, is important for the security of GKE protocols in the assumed trust relationship. Informally, we consider an active PPT adversary which is allowed to corrupt up to $n - 1$ group members and reveal internal states of all n oracles during the execution of P aiming to achieve that there exists at least one uncorrupted group member whose oracle accepts the session group key chosen by the adversary. Note also that our definition prevents so-called *key replication* attacks [21].

Definition 5 (Contributiveness). Let P be a correct GKE protocol and $\mathcal{A}$ an adversary running in two stages, **prepare** and **attack**, that interacts with the instance oracles in the following game $\text{Game}_P^{\text{con}}(\kappa)$:

- $\mathcal{A}(\text{prepare})$ is given access to the queries **Send**, **Execute**, **RevealKey**, **RevealState**, and **Corrupt**. At the end of the stage, it outputs $\tilde{k} \in \{0, 1\}^\kappa$, and some state information St . After $\mathcal{A}$ makes its output and all previously asked queries are processed the following sets are built: $\mathcal{G}_{\text{us}}$ consisting of all honest used oracles, $\mathcal{G}_{\text{std}}$ consisting of all honest oracles that are in the stand-by state ($\mathcal{G}_{\text{std}} \subseteq \mathcal{G}_{\text{us}}$), and Ψ consisting of session ids $\text{sid}_i^{s_i}$ for every $\Pi_i^{s_i} \in \mathcal{G}_{\text{us}}$. Then $\mathcal{A}$ is invoked for the **attack** stage.
- $\mathcal{A}(\text{attack}, St)$ is given access to the queries **Send**, **Execute**, **RevealKey**, **RevealState**, and **Corrupt**. At the end of the stage $\mathcal{A}$ outputs (s, U) .

The adversary $\mathcal{A}$ wins in $\text{Game}_{\mathcal{A}, P}^{\text{con}}(\kappa)$ if **all** of the following holds:

1. Π_U^s is terminated, has accepted with $\tilde{k}$, no $\text{Corrupt}(U)$ has been asked, $\Pi_U^s \notin \mathcal{G}_{\text{us}} \setminus \mathcal{G}_{\text{std}}$ and $\text{sid}_U^s \notin \Psi$.
2. There are at most $n - 1$ corrupted users U_i having oracles $\Pi_i^{s_i}$ partnered with Π_U^s .

The maximal probability (over all adversaries running within time κ) in winning the game is defined as

$$\text{Succ}_P^{\text{con}}(\kappa) := \Pr[\mathcal{A} \text{ wins in } \text{Game}_P^{\text{con}}(\kappa)]$$

We say that a GKE protocol P is contributory (CGKE) if this probability is a negligible function of κ .

Comments. The first requirement ensures that Π_U^s belongs to an uncorrupted user. The condition $\Pi_U^s \notin \mathcal{G}_{\text{us}} \setminus \mathcal{G}_{\text{std}}$ prevents the case where $\mathcal{A}$ while being an operation participant outputs $\tilde{k}$ for the still running operation which is then accepted by Π_U^s that participates in the same operation (this is not an attack since participants do not compute group keys synchronously). Note that $\mathcal{G}_{\text{us}} \setminus \mathcal{G}_{\text{std}}$ consists of all oracles that at the end of the **prepare** stage have already terminated or remain in the processing state. Similarly, the condition $\text{sid}_U^s \notin \Psi$ prevents that $\mathcal{A}$ while being in the **attack** stage outputs (s, U) such that Π_U^s has accepted with $\tilde{k}$ already in the **prepare** stage; otherwise as soon as Π_U^s computes some k_U^s in the **prepare** stage $\mathcal{A}$ can trivially output $\tilde{k} = k_U^s$. The second requirement allows $\mathcal{A}$ to corrupt at most $n - 1$ (out of totally n) participants in the session where Π_U^s accepts with $\tilde{k}$.

Note also that U must be uncorrupted but curious $\mathcal{A}$ is allowed to reveal the internal state of Π_U^s during the execution of the **attack** stage (this is because our model separates LL_U from state_U^s). Also, due to the adaptiveness and strong corruptions the adversary in this game seems to be strictly stronger than in [4].

The following example highlights this idea. We consider the well-known two-party Diffie-Hellman (DH) key exchange¹ [13], and show that if a malicious participant is able to (passively) reveal internal states of the oracles (strong corruptions) then it has full control over the obtained key. Let U_1 and U_2 have their corresponding oracles $\Pi_1^{s_1}$ and $\Pi_2^{s_2}$. They choose ephemeral secret exponents x_1 and x_2 , then exchange the (authenticated) values g^{x_1} and g^{x_2} , respectively, and finally compute the key $k := g^{x_1 x_2}$. Now assume U_1 is malicious. She specifies $\tilde{k}$ as $g^{\tilde{x}}$ for some chosen $\tilde{x}$ before the execution of the protocol. Since the communication model is asymmetric (this is also the common case in praxis) U_1 waits to receive g^{x_2} sent by the honest U_2 , then queries $\text{RevealState}(\Pi_2^{s_2})$ to obtain x_2 as part of the internal state of $\Pi_2^{s_2}$, and finally computes $x_1 := \tilde{x}/x_2$ and sends g^{x_1} to U_2 . It is easy to see that U_2 accepts with $k := (g^{x_1})^{x_2} = g^{\tilde{x}} = \tilde{k}$.

4 Unifying Relationship of MA-Security and Contributiveness

In this section we present some claims to illustrate that given definitions of MA-security and contributiveness unify many related informal definitions proposed in the previous literature, particularly in [1, 5, 24]. Note that missing formalism in the informal definitions allows only argumentative proofs.

Claim 1. *If $\mathbb{P}$ is a MAGKE protocol then it provides key confirmation and mutual authentication (explicit key authentication) in the sense of [24, Def. 12.6-12.8], i.e., every legitimate protocol participant is assured of the participation of every other participant, and all participants that have accepted hold identical session group keys.*

Proof (informal). If $\mathbb{P}$ does not provide key confirmation and (implicit) key authentication then there exists at least one honest participant $U_i \in \mathcal{G}$ whose oracle $\Pi_i^{s_i}$ has

¹ As observed in [23] similar attacks can be found against many currently known group key exchange protocols.

accepted with a session group key $k_i^{s_i}$ and there exists at least one another honest participant $U_j \in \text{pid}_i^{s_i}$ whose oracle $\Pi_j^{s_j}$ has accepted with a different session group key $k_j^{s_j} \neq k_i^{s_i}$. According to Definition 4 this is a successful attack against the MA-security of $\mathcal{P}$. This, however, contradicts to the assumption that $\mathcal{P}$ is a MAGKE protocol.

Claim 2. *If $\mathcal{P}$ is a MAGKE protocol then it is resistant against unknown key-share attacks in the sense of [5, Sec. 5.1.2], i.e., the adversary $\mathcal{A}$ cannot make one protocol participant, say U_j , believe that the session group key k is shared with $\mathcal{A}$ when it is in fact shared with a different participant U_i .*

Proof (informal). With respect to our model we assume that oracles $\Pi_j^{s_j}$ and $\Pi_i^{s_i}$ participate in the protocol on behalf of U_j and U_i , respectively. If an unknown key-share attack occurs then $\Pi_j^{s_j}$ and $\Pi_i^{s_i}$ accepted with the identical session group k , but since $\Pi_j^{s_j}$ believes that the key is shared with $\mathcal{A}$ we conclude that $U_i \notin \text{pid}_j^{s_j}$ must hold (otherwise after having accepted U_j would believe that the key is shared with U_i) whereas $U_j \in \text{pid}_i^{s_i}$. This implies $(\text{pid}_j^{s_j}, \text{sid}_j^{s_j}) \neq (\text{pid}_i^{s_i}, \text{sid}_i^{s_i})$. On the other hand, $\mathcal{P}$ is by assumption MAGKE. Thus, according to Definition 4 for any $U_j \in \text{pid}_i^{s_i}$ there must exist a corresponding oracle $\Pi_j^{s_j}$ such that $(\text{pid}_j^{s_j}, \text{sid}_j^{s_j}) = (\text{pid}_i^{s_i}, \text{sid}_i^{s_i})$. This is a contradiction.

Claim 3. *If $\mathcal{P}$ is a CGKE protocol then its output is unpredictable by any subset of $n - 1$ participants.*

Proof (informal). If the output of $\mathcal{P}$ is predictable by a subset $\tilde{\mathcal{G}} \subset \mathcal{G}$ of $n - 1$ protocol participants then there exists $\tilde{k}$ which was predicted by $\tilde{\mathcal{G}}$ and accepted by some oracle Π_U^s of an uncorrupted user $U \in \mathcal{G} \setminus \tilde{\mathcal{G}}$. However, this implies that there exists an adversary $\mathcal{A}$ who corrupts up to $n - 1$ users whose oracles are partnered with Π_U^s and predicts the session group key accepted by Π_U^s . This is a contradiction to the assumption that $\mathcal{P}$ is a CGKE protocol.

Claim 4. *If $\mathcal{P}$ is a CGKE protocol then $\mathcal{P}$ is contributory in the sense of [1, Def. 3.2], i.e., each participant equally contributes to the resulting session group key and guarantees its freshness.*

Proof (informal). If $\mathcal{P}$ is not contributory then there exists an honest oracle Π_U^s who accepts a session group key without having contributed to its computation, i.e., the session group key accepted by Π_U^s is composed of at most $n - 1$ contributions. This, however, implies that there exists an adversary $\mathcal{A}$ who corrupts up to $n - 1$ users and influences Π_U^s to accept a session group key built from contributions of these corrupted users. This is a contradiction to the assumption that $\mathcal{P}$ is a CGKE protocol.

Claim 5. *If $\mathcal{P}$ is a CGKE and a MAGKE protocol then $\mathcal{P}$ provides complete group key authentication in the sense of [1, Def. 6.3], i.e., any two participants compute the same session group key only if all other participants have contributed to it.*

Proof (informal). Since $\mathcal{P}$ is a CGKE protocol then according to the previous claim $\mathcal{P}$ is contributory. Hence, none of the honest users accepts the key without having contributed to its computation. Since $\mathcal{P}$ is a MAGKE protocol all honest users accept the

same session group key. Hence, all honest users have contributed to the session group key. Therefore, there can be no pair of users who accept the same group key which is not contributed to by all other honest users. Thus, $\mathcal{P}$ provides complete group key authentication.

The notion of verifiable contributiveness [1] is relevant to MA-security, since this mechanism is designed for providing confirmation (and thus, verification) that the protocol actually fits the security requirements. In the case of contributory protocols, it is intuitively true that the MA-security guarantees that the contributiveness was satisfied (otherwise, some player would be able to check that his own contribution was not properly taken into account). Hence,

Claim 6. *If $\mathcal{P}$ is a CGKE and MAGKE protocol then $\mathcal{P}$ is verifiable contributory in the sense of [1, Def. 7.3], i.e., each participant is assured of every other participant's contribution to the group key.*

Proof (informal). Since $\mathcal{P}$ is a MAGKE protocol all honest users accept the same session group key. Since $\mathcal{P}$ is also a CGKE protocol and, therefore, contributory the accepted group key is contributed to by each honest user.

5 Our Compiler for MA-Security and Contributiveness

In this section we propose a compiler which can be used to turn any AKE-secure GKE protocol into a GKE protocol which is additionally MA-secure and provides contributiveness.

Our compiler denoted C-MACON can be seen as an extension of the compiler in [18] that according to our model satisfies the requirement of MA-security² but not of contributiveness. If $\mathcal{P}$ is a GKE protocol, by C-MACON _{$\mathcal{P}$} we denote the compiled protocol.

In the following, we assume that each message sent by Π_U^s can be parsed as $U|m$ consisting of the sender's identity U and a message m . Additionally, an authentication token σ , e.g., a digital signature on m , can be attached. Our compiler is formally described in Definition 6: it is based on a one-way permutation π , a collision-resistant pseudo-random function ensemble F , and an existentially unforgeable digital signature Σ (we provide more details on these well-known primitives in the full version of this paper [9]). The description is given from the perspective of one particular operation execution (session). Therefore, by $\Pi_i^s \in \mathcal{G}$ we consider the i -th oracle in $\mathcal{G}$ assuming that there exists an index $j \in [1, N]$ such that U_j owns Π_i^s . Similar, by sk'_i and pk'_i (resp., sk_i and pk_i) we denote the private and public keys of U_j used in the compiled protocol (resp., in the underlying protocol).

Main Ideas. After computing the session group key k in the underlying protocol $\mathcal{P}$ participants execute C-MACON. In its first communication round they exchange randomly chosen nonces r_i that are then concatenated into a session id sid (this is a classical way to define unique session ids). Then, each participant iteratively computes values

² The proof of this statement can be directly derived from the proof of MA-security of our compiler (Theorem 2).

$\rho_1, \dots, \rho_n$ by adequately using the pseudo-random function f , in such a way that every random nonce (contribution of each participant) is embedded into the computation of $K := \rho_n$. The intuition is that malicious participant cannot influence this computation. The second communication round of C-MACON is used to ensure key confirmation. For this purpose we apply the same technique as in [18], i.e., every participant computes a key confirmatory token $\mu_i = f_K(v_1)$ using a public input value v_1 , signs it and sends it to other participants. After verifying signatures each party accepts with the session group key $\mathbf{K} = f_K(v_2)$ with public input value $v_2 \neq v_1$. All intermediate values are then erased.

Definition 6 (Compiler C-MACON). *Let $\mathcal{P}$ be a GKE protocol from Definition 1, $\pi : \{0, 1\}^\kappa \rightarrow \{0, 1\}^\kappa$ a permutation, $F := \{f_k\}_{k \in \{0, 1\}^\kappa}$, $\kappa \in \mathbb{N}$ a function ensemble with domain and range $\{0, 1\}^\kappa$, and $\Sigma := (\text{Gen}, \text{Sign}, \text{Verify})$ a digital signature scheme. A compiler for MA-security and n -contributiveness, denoted $\text{C-MACON}_{\mathcal{P}}$, consists of the algorithm INIT and a two-round protocol MACON defined as follows:*

INIT: *In the initialization phase each $U_i \in \mathcal{U}$ generates own private/public key pair (sk'_i, pk'_i) using $\Sigma.\text{Gen}(1^\kappa)$. This is in addition to any key pair (sk_i, pk_i) used in $\mathcal{P}$. **MACON:** After an oracle Π_i^s computes k_i^s in the execution of $\mathcal{P}$ it proceeds as follows.*

Round 1: *It chooses a random MACON nonce $r_i \in_R \{0, 1\}^\kappa$ and sends $U_i|r_i$ to every oracle Π_j^s with $U_j \in \text{pid}_i^s$. After Π_i^s receives $U_j|r_j$ from Π_j^s with $U_j \in \text{pid}_i^s$ it checks whether $|r_j| \stackrel{?}{=} \kappa$. If this verification fails then Π_i^s terminates without accepting;*

Round 2: *Otherwise, after having received and verified these messages from all other partnered oracles it computes $\rho_1 := f_{k_i^s \oplus \pi(r_1)}(v_0)$ and each $\rho_l := f_{\rho_{l-1} \oplus \pi(r_l)}(v_0)$ for all $l \in \{2, \dots, n\}$ where v_0 is a public value. Then, it defines the intermediate key $K_i^s := \rho_n$ and $\text{sid}_i^s := r_1 \dots r_n$ and computes a MACON token $\mu_i := f_{K_i^s}(v_1)$ where v_1 is a public value, together with a signature $\sigma_i := \Sigma.\text{Sign}(sk'_i, \mu_i|\text{sid}_i^s|\text{pid}_i^s)$. Then, it sends $U_i|\sigma_i$ to every oracle Π_j^s with $U_j \in \text{pid}_i^s$ and every other private information from state_i^s (including k_i^s and each ρ_l , $l \in [1, n]$). After Π_i^s receives $U_j|\sigma_j$ from Π_j^s with $U_j \in \text{pid}_i^s$ it checks whether $\Sigma.\text{Verify}(pk'_j, \mu_j|\text{sid}_i^s|\text{pid}_i^s, \sigma_j) \stackrel{?}{=} 1$. If this verification fails then Π_i^s terminates without accepting; otherwise it accepts with the session group key $\mathbf{K}_i^s := f_{K_i^s}(v_2)$ where $v_2 \neq v_1$ is another public value, and erases every other private information from state_i^s (including K_i^s).*

Note that C-MACON can be considered as an add-on protocol that should be executed after the execution of $\mathcal{P}$. Moreover, with the MACON nonces we achieve not only the uniqueness of session ids but also the randomization and contributiveness (via successive evaluations of f) for the intermediate value K , for the key confirmatory MACON tokens (as in [18]) and for the derived resulting session group key $\mathbf{K}$.

5.1 Complexity of C-MACON

Obviously, C-MACON requires two communication rounds. This is similar to the KS compiler [18] in case that no session ids are predefined and have to be negotiated first. Each

participant must generate one digital signature and verify n signatures where n is the total number of session participants. This is also similar to the KS compiler. C-MACON achieves contributiveness at an additional cost of n executions of the one-way permutation π and n executions of the pseudo-random function f per participant.³

5.2 Security Analysis

Let $\mathcal{P}$ be a GKE protocol from Definition 1. For this analysis we require Σ to be *existentially unforgeable under chosen message attacks* (EUF-CMA) [16], π to be *one-way*, and F to be *collision-resistant pseudo-random* [18]. We provide corresponding definitions in the full version of this paper [9].

Recall that we assume ephemeral secret information being independent of the long-lived key; that is, state_U^s may contain ephemeral secrets used in $\mathcal{P}$, the session key k_U^s computed in $\mathcal{P}$, and $\rho_1, \dots, \rho_n$ together with some (implementation specific) temporary variables used to compute these values. Note that state_U^s is erased at the end of the protocol. By contrast, temporary data used by $\Sigma.\text{Sign}(sk_U^l, m)$ usually depends on the long-lived key and thus should be executed under the same protection mechanism as sk_U^l , e.g., in a smart card [7]⁴. Let q_s be the total number of executed protocol sessions during the attack.

The following theorem (whose proof appears in [9]) shows that C-MACON $_{\mathcal{P}}$ preserves the AKE-security with strong forward secrecy of the underlying protocol $\mathcal{P}$.

Theorem 1 (AKE-Security of C-MACON $_{\mathcal{P}}$). *For any AGKE-sfs protocol $\mathcal{P}$ if Σ is EUF-CMA and F is pseudo-random then C-MACON $_{\mathcal{P}}$ is also a AGKE-sfs protocol, and*

$$\text{Adv}_{\text{sfs, C-MACON}_{\mathcal{P}}}^{\text{ake}}(\kappa) \leq 2N\text{Succ}_{\Sigma}^{\text{euf-cma}}(\kappa) + \frac{Nq_s^2}{2^{\kappa-1}} + 2q_s\text{Adv}_{\text{sfs, } \mathcal{P}}^{\text{ake}}(\kappa) + 2(N+2)q_s\text{Adv}_F^{\text{prf}}(\kappa).$$

The following theorems (whose proofs appear in [9]) concern the MA-security and the contributiveness of C-MACON $_{\mathcal{P}}$ in the presence of malicious participants and strong corruptions.

Theorem 2 (MA-Security of C-MACON $_{\mathcal{P}}$). *For any GKE protocol $\mathcal{P}$ if Σ is EUF-CMA and F is collision-resistant then C-MACON $_{\mathcal{P}}$ is MAGKE, and*

$$\text{Succ}_{\text{C-MACON}_{\mathcal{P}}}^{\text{ma}}(\kappa) \leq N\text{Succ}_{\Sigma}^{\text{euf-cma}}(\kappa) + \frac{Nq_s^2}{2^{\kappa}} + q_s\text{Succ}_F^{\text{coll}}(\kappa).$$

Theorem 3 (Contributiveness of C-MACON $_{\mathcal{P}}$). *For any GKE protocol $\mathcal{P}$ if π is one-way and F is collision-resistant pseudo-random then C-MACON $_{\mathcal{P}}$ is CGKE, and*

$$\text{Succ}_{\text{C-MACON}_{\mathcal{P}}}^{\text{con}}(\kappa) \leq \frac{Nq_s^2 + Nq_s + 2q_s}{2^{\kappa}} + (N+2)q_s\text{Succ}_F^{\text{coll}}(\kappa) + q_s\text{Adv}_F^{\text{prf}}(\kappa) + Nq_s\text{Succ}_{\pi}^{\text{ow}}(\kappa).$$

³ Note that costs of XOR operations are usually omitted in the complexity analysis if public-key cryptography operations are present. Note also that pseudo-random functions can be realized using techniques of the symmetric cryptography massively reducing the required computational effort.

⁴ Smart cards have limited resources. However, in C-MACON each II_U^s has to generate only one signature.

Remark 1. Note that the contributiveness of C-MACON_P depends neither on AKE-security of $\mathcal{P}$ nor on the security of the digital signature scheme Σ . Hence our compiler can also be used for unauthenticated GKE protocols by omitting digital signatures of exchanged messages. However, in this case it would guarantee only contributiveness but not MA-security in the presence of malicious participants. The latter can be only guaranteed using digital signatures (as also noticed in [18] for their definition of security against insider attacks). Note also that C-MACON_P provides contributiveness in some even stronger sense than required in Definition 5, i.e., $\mathcal{A}$ may even be allowed to output $\tilde{\mathbf{K}}$ before the uncorrupted user's oracle Π_U^s (that is supposed to accept with $\tilde{\mathbf{K}}$ in $\text{Game}_{\text{C-MACON}_P}^{\text{con}}(\kappa)$) starts with the MACON protocol of the compiler, and not necessarily before the execution of the new C-MACON_P session.

6 Conclusion

In this paper we have addressed the main difference in the trust relationship between participants of group key exchange (GKE) and those of group key transport (GKT) protocols, namely, the question of key control and contributiveness. This has been done from the perspective of malicious participants and powerful adversaries who are able to reveal the internal memory of honest participants. The proposed security model based on the extension of the well-known notion of AKE-security with strong forward secrecy from [7] towards additional requirements of MA-security and contributiveness seems to be stronger than the previous models for group key exchange protocols that address similar issues. The described compiler C-MACON satisfies these additional security requirements and extends the list of currently known compilers for GKE protocols, i.e., the compiler for AKE-security by Katz and Yung [19] and the compiler for security against “insider attacks” by Katz and Shin [18] (that according to our model provides MA-security but not contributiveness). Finally, group key exchange protocols that satisfy our stronger interpretation of key control and contributiveness also provide resilience in the following (weaker) cases: (i) where participants do not have intentions to control the value of the group key, e.g., do not know that their source of randomness is biased (as in [6]), and (ii) where the adversary is given access only to the weak corruptions (as in [4]).

References

1. Ateniese, G., Steiner, M., Tsudik, G.: Authenticated Group Key Agreement and Friends. ACM CCS, 17–26 (1998)
2. Bellare, M., Rogaway, P.: Entity Authentication and Key Distribution. In: CRYPTO, pp. 232–249 (1993)
3. Bellare, M., Rogaway, P.: Provably Secure Session Key Distribution: The Three Party Case. STOC, 57–66 (1995)
4. Bohli, J.-M., Vasco, M.I.G., Steinwandt, R.: Secure Group Key Establishment Revisited. To appear in International Journal of Information Security.
<http://eprint.iacr.org/2005/395>
5. Boyd, C., Mathuria, A.: Protocols for Authentication and Key Establishment. Springer, Heidelberg (2003)

6. Bresson, E., Catalano, D.: Constant Round Authenticated Group Key Agreement via Distributed Computation. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 115–129. Springer, Heidelberg (2004)
7. Bresson, E., Chevassut, O., Pointcheval, D.: Dynamic Group Diffie-Hellman Key Exchange under Standard Assumptions. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 321–336. Springer, Heidelberg (2002)
8. Bresson, E., Chevassut, O., Pointcheval, D., Quisquater, J.J.: Provably Authenticated Group Diffie-Hellman Key Exchange. ACM CCS, 255–264 (2001)
9. Bresson, E., Manulis, M.: Full version of this paper. Available from the authors' homepages
10. Burmester, M.: On the Risk of Opening Distributed Keys. In: CRYPTO, pp. 308–317 (1994)
11. Burmester, M., Desmedt, Y.: A Secure and Efficient Conference Key Distribution System. In: EUROCRYPT, pp. 275–286 (1994)
12. Choo, K.K.R., Boyd, C., Hitchcock, Y.: Examining Indistinguishability-Based Proof Models for Key Establishment Protocols. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 585–604. Springer, Heidelberg (2005)
13. Diffie, W., Hellman, M.E.: New Directions in Cryptography. IEEE IT 22(6), 644–654 (1976)
14. Diffie, W., van Oorschot, P.C., Wiener, M.J.: Authentication and Authenticated Key Exchanges. DCC 2(2), 107–125 (1992)
15. Goldreich, O.: Foundations of Cryptography - Basic Tools, vol. 1. Cambridge University Press, Cambridge (2001)
16. Goldwasser, S., Micali, S., Rivest, R.L.: A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. SIAM Journal of Computing 17(2), 281–308 (1988)
17. Günther, C.G.: An Identity-Based Key-Exchange Protocol. In: EUROCRYPT, pp. 29–37 (1989)
18. Katz, J., Shin, J.S.: Modeling Insider Attacks on Group Key-Exchange Protocols. ACM CCS, 180–189 (2005)
19. Katz, J., Yung, M.: Scalable Protocols for Authenticated Group Key Exchange. In: CRYPTO, pp. 110–125 (2003)
20. Kim, Y., Perrig, A., Tsudik, G.: Simple and Fault-Tolerant Key Agreement for Dynamic Collaborative Groups. ACM CCS, 235–244 (2000)
21. Krawczyk, H.: HMQV: A High-Performance Secure Diffie-Hellman Protocol. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 546–566. Springer, Heidelberg (2005)
22. Manulis, M.: Survey on Security Requirements and Models for Group Key Exchange. Technical Report. <http://eprint.iacr.org/2006/388>
23. Manulis, M.: Security-Focused Survey on Group Key Exchange Protocols. Technical Report. <http://eprint.iacr.org/2006/395>
24. Menezes, A., van Oorschot, P.C., Vanstone, S.: Handbook of Applied Cryptography. CRC Press, Boca Raton (1996)
25. Mitchell, C.J., Ward, M., Wilson, P.: Key Control in Key Agreement Protocols. El. Letters 34(10), 980–981 (1998)
26. Shoup, V.: On Formal Models for Secure Key Exchange (Version 4). Technical Report. <http://shoup.net/>
27. Steiner, M.: Secure Group Key Agreement. PhD thesis (2002)
28. Yacobi, Y., Shmueli, Z.: On Key Distribution Systems. In: CRYPTO, pp. 344–355 (1989)

Efficient Implementation of the Keyed-Hash Message Authentication Code Based on SHA-1 Algorithm for Mobile Trusted Computing

Mooseop Kim¹, Youngse Kim¹, Jaecheol Ryou², and Sungik Jun¹

¹ Electronics and Telecommunications Research Institute (ETRI)
161 Gajeong-dong, Yuseong-gu, Daejeon, 305-700, South Korea
{gomskim,sijun}@etri.re.kr

² Division of Electrical and Computer Engineering, Chungnam National University
220 Gung-dong, Yuseong-gu, Daejeon, 305-764, South Korea
jcryou@home.cnu.ac.kr

Abstract. The Mobile Trusted Platform (MTP) is developed and promoted by the Trusted Computing Group (TCG), which is an industry standard body to enhance the security of the mobile computing environment. The dedicated SHA-1 and HMAC engine in Mobile Trusted Module (MTM) are one of the most important circuit blocks and contribute the performance of the whole platform because they are used as key primitives verifying platform code, integrity and command authentication. Unlike desktop computers, mobile devices have very stringent limitations with respect to available power, physical circuit area, and cost. Therefore special architecture and design methods for low power SHA-1 and HMAC circuit are required. In this paper, we present a compact and efficient hardware architecture of low power SHA-1 and HMAC design for MTM. Our SHA-1 hardware can compute 512-bit data block using about 8,200 gates and has a power consumption about 1.1 *mA* on a 0.25 μ m CMOS process. The implementation of HMAC using the SHA-1 circuit requires additional 8,100 gates and consumes about 2.58 *mA* on the same process.

1 Introduction

The Trusted Computing Group(TCG) is an organization that develops and produces open specifications, with regard to security-based solutions for various computing systems. They have released several documents and specifications that define secure procedures as they relate to the boot-up, configuration management, and application execution for personal computing platforms. The core component of the TCG proposal is the Trusted Platform Module(TPM) that acts as a key component of monitoring and reporting.

TPM is a separate trusted coprocessor, whose state cannot be compromised by potentially malicious host system software. This chip is capable of securely storing cryptographic keys and other cryptographic functions like asymmetric encryption, signature schemes, and hash functions. Using these functionalities

user can attest the initial configuration of a platform and seal or bind data to a specific platform configuration.

Without proper security, mobile phones may become a target for hackers and malicious software. The benefit of hardware-based security is that users can rely on their phone and that private data is protected. For these reasons, TCG is now extending the security realizations into mobile technology and other embedded systems. The Mobile Phone Work Group has extended the TCG specifications specifically to support mobile phone devices. In these specifications, a mobile trusted module (MTM) must support the unkeyed hash function SHA-1 and also must support the keyed-hash function HMAC to compute and verify the integrity measurement value of underlying platforms.

Integrating TCG's security features into a mobile phone could be a significant engineering challenge because most mobile devices have limited memory, available power, and processing resources. Among these factors, the limitation of available power is the major issue in the mobile phones because they have limited battery life. Therefore, design methodologies at different abstraction levels, such as systems, architectures, as well as logic design, must take into account to design the compact SHA-1 and HMAC circuit for mobile trusted platforms.

In this paper, we introduce compact and efficient hardware architecture of low power SHA-1 algorithm for mobile trusted platforms. Then we implement an efficient HMAC hardware engine using the SHA-1 circuit. As a result, a compact and energy efficient SHA-1 and HMAC hardware implementation which is capable of supporting the integrity check and command authentication of mobile trusted platforms was developed and evaluated.

2 Previous Works

The HMAC standard [2] defines a mechanism that guarantees message authentication for transmission through a non-secure communication channel. The main idea is the use of a cryptographic hash function such as SHA-1 [1] or MD5. The purpose of HMAC in a mobile trusted module is to verify and to authenticate both the command of a underlying platform and its integrity measurement.

Numerous FPGA and ASIC implementations of HMAC [9], [10] and SHA-1 algorithm [3-8] were previously proposed and evaluated. Most of these implementations feature high speeds and high costs suitable for high-performance usages such as WTLS, IPsec and so on.

Early HMAC and SHA-1 design were mostly straightforward implementations of various loop rolling architectures with limited number of architectural optimization. This technique allows small-sized implementations through reuse of the same configurable operation block. S.Dominikus [4] used loop rolling technique in order to reduce area requirement. He proposed an efficient SHA-1 architecture uses only four operation blocks, one for each round. Using a temporal register and a counter, each operation block is reused for 20 iterations. G.Selimis [9] applied the reuse technique of [4] to the non-linear function of SHA-1 algorithm

for HMAC design. He modified the operation block to include the four non-linear functions.

Another architecture for design of HMAC and SHA-1 is based on the use of four pipeline stages. If the critical design parameter is higher throughput with a more relaxed area constraint, this method can be applied. N.Sklavos [7] and M.K.Michail [10] used the characteristics of the SHA-1 algorithm that requires a different non-linear function for four discrete rounds. Applying a pipeline stage to every round, they could achieve at least four times higher throughput than previous methods.

In practice, several vendors already deploy laptop computers that are equipped with a TPM chip [11], [12], [13] placed on the main board of the underlying platform. Unfortunately, most of these commercial chips and previous works have been designed aiming only at large message and high performance usages, with no power consumption taken into considerations.

3 Low Power Hardware Architecture

For our HMAC implementation, we assume that one 512-bit data block of pre-processed by microprocessor is stored in memory and available to our HMAC circuit for reading and writing.

We began the design of our low power HMAC hardware architecture by analyzing the basic architecture of SHA-1 algorithm because the SHA-1 is the most important circuit block for the performance of the HMAC design.

3.1 Implementation of Compact SHA-1 Core

SHA-1 algorithm [1] sequentially processes 512-bit data block when computing message digest. For each 512-bit message block, the SHA-1 round operation is processed 80 times. Each round operation performs several predefined processing, which involves four additions, two circular right shift operations, and logical function f_t operating on three 32-bit values and produces a 32-bit data as output.

The first step for our low power SHA-1 core design was to find a minimal architecture. This part was done by hand. A set of key components thus obtained. Components of SHA-1 core then designed and applied several low power technologies to each component. Figure 1 shows main components and interactions of our SHA-1 core.

Data input block in figure 1 is responsible for receiving data applied to an input from HMAC circuit. It also performs padding operation about the transformed data to generate the padded 512-bit block required by the algorithm. We use 32-bit data bus for efficient design of our SHA-1 circuit. It is not a good idea to make the bus width smaller than 32-bit, because all operation of SHA-1 algorithm and variables need 32 bits of data at one time. Although a smaller bus may requires less registers, it uses more data selectors and resource sharing is hindered, resulting in an inefficient implementation.

The controller logic block is used to generate signal sequences to check an input signal sequence or to control datapath parts. The basic structure of controller

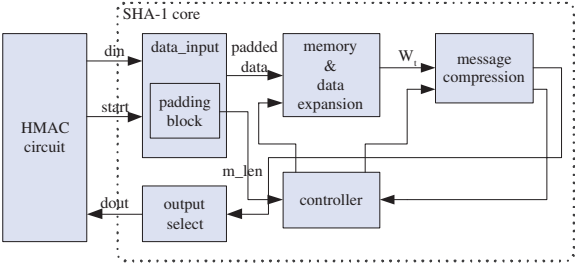


Fig. 1. Outline of SHA-1 circuit block

consists of state register and two logic blocks. The input logic block computes the next state as a function of current state and of the new sets of input signals. The output logic block generates the control signals for datapath using the function signals of the current states.

The efficiency of a low power SHA-1 hardware in terms of circuit area, power consumption and throughput is mainly determined by the structure of data expansion and message compression block. The message compression block performs actual hashing. In each step, it processes a new word generated by the message expansion block. The functional block diagram of message compression is presented in figure 2.

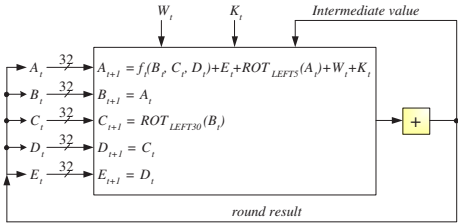


Fig. 2. Functional block diagram of data compression

Figure 2 shows that SHA-1 algorithm uses five 32-bit variables (A , B , C , D , and E) to store new values in each round operation. It can be easily seen from [1] that four out of the five values are shifted by one position down in each round and only determining the new value for A requires computation. Therefore, we use a five-stage 32-bit shift register for these variables.

The computation for A requires two circular right shifting and four operand addition modulo 2^{32} where the operands depend on all input values, the round constant K_t , and current message value W_t . For compact and low power SHA-1 core design, we use only one 32-bit adder to perform four additions and use register E to store temporary addition values. Therefore, four clock cycles are required to compute a round operation. Equation 1 shows the functional steps for this operation.

$$\begin{aligned}
 t_1 : E_{t_1} &= E_{t_0} + K_t \\
 t_2 : E_{t_2} &= E_{t_1} + ROT_{LEFT5}(A_t) \\
 t_3 : E_{t_3} &= E_{t_2} + W_t \\
 t_4 : A_t &= E_{t_3} + F(B, C, D)
 \end{aligned} \tag{1}$$

All aforementioned optimizations lead to the schematic of the compact architecture of the data compression. Dashed line in figure 3 shows the detailed structure of data compression block. At first, all registers are initialized and multiplexors choose path zero to load initialization constant $H_0 \sim H_4$ stored in KH. Five clock cycles are required to load initial vector to each register. For optimized power consumption, we applied gated clock to all registers in data compression. The F-function in figure 3 is a sequence of logical functions. For each round t , F-function operates on three 32-bit data (B, C, and D) and produces a 32-bit output word.

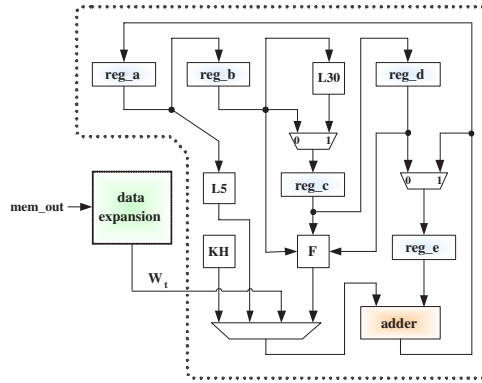


Fig. 3. Detailed architecture of data compression

During the final round operation, the values of the working variables have to be added to the digest of the previous message block, or specific initial values for the first message block. This can be done very efficiently with additional multiplexer and the reuse of five stage shift registers for working variables.

KH in figure 3 stores initial values H_i and constant values K_t . It also stores updated H_i values, which is used as the initial values for next 512-bit data block computing. It takes five clock cycles to compute the final hash value for one input message block.

Another important part of SHA-1 data path is data expansion. This block generates message dependant words, W_t , for each step of the data compression. Most implementations of data expansion in previous works use 16-stage 32-bit shift registers for 512-bit data block processing. This methods are inefficient to use in mobile platforms because they require a significant amount of circuit area and power consumptions.

We use only one 32-bit register to store temporary values during computation of the new W_t . Our message expansion block performs the function of the equa-

tion 2, where $\oplus$ means bitwise XOR and $M_t^{(i)}$ denotes the first sixteen 32-bit data of i -th data block.

$$W_t = \begin{cases} M_t^{(i)} & 0 \leq t \leq 15 \\ ROTL^1(W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}) & 16 \leq t \leq 79 \end{cases} \quad (2)$$

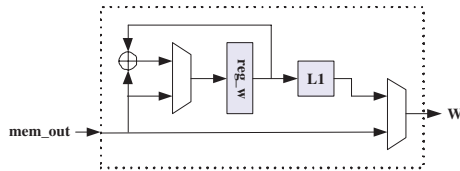


Fig. 4. Compact data expansion of SHA-1 core

Four values of memory data have to be read and the results have to be written back to memory in each round. This job takes four clock cycles, therefore, each round of SHA-1 takes four clock cycles. Dedicated hard wired logic is used for computation of necessary address. The detailed architecture of our data expansion module is shown in figure 4.

3.2 Implementation of HMAC Based SHA-1 Core

The HMAC value of data can be calculated by performing the equation 3, where *text* is the plain text of the message, *K* is the secret key, k_0 is *K* appended with zeros to form a block size of the hash function, i_{pad} and o_{pad} are predefined constants, and $\oplus$ is bitwise XOR operation. Figure 5 shows the architecture for whole HMAC implementation which incorporates the SHA-1 component.

$$HMAC(K, text) = H[k_0 \oplus opad || H[(k_0 \oplus ipad) || text]] \quad (3)$$

The microprocessor in figure 5 controls all of the internal operation on the mobile trusted module. It performs functions such as managing the interface to the mobile platform, controlling operation of the TMP crypto engines, processing TPM commands of TCG specifications such as SHA_INIT, SHA_UPDATE, SHA_FINISH, HMAC_INIT and so on received from the mobile system, and performing security check on the mobile platform.

The key-padding block extends the mandatory 20-byte input key and optional 49-byte and 64-byte input keys to a 64-byte key as required by the HMAC algorithm [2].

The padded key is then xored with the constant *opad* and *ipad* 64-byte constant values. The concatenate block serves to append the xored-key results, in the first instance with the input text and in the second instance with the resulting 160-bit hash, to form the input for the SHA-1 operation. Data padding block performs padding operation about concatenated data to be a multiple of 512-bit.

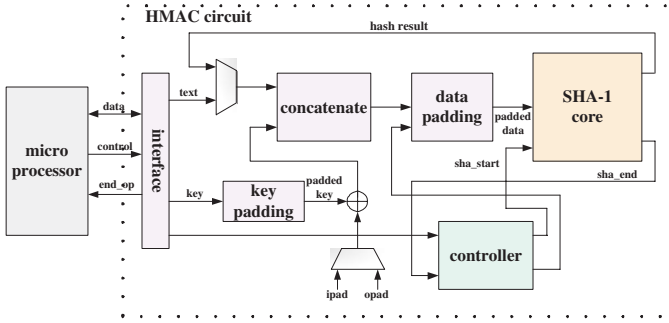


Fig. 5. Architecture of HMAC circuit

The HMAC algorithm utilizes two phase of the underlying hash function, where the inputs are the $(k_0 \oplus ipad)$ 512-bit block and the input data to achieve the first hash output. Actually, this step requires two times of SHA-1 calculation because the transformed data is larger than 512-bit. Including data input to SHA-1 core and data padding operation, 876 clock cycles are required to compute the first step of hash calculation.

The $(k_0 \oplus opad)$ value and the first hash output are the input data which achieve the final hash or HMAC output. This step also require two times of SHA-1 algorithm processing because the concatenated data with xored-key and the output of the first hash operation exceeds 512-bit length. This step requires additional 874 clock cycles to compute HMAC output.

In order to maintain a small area design, and since the first hash output forms a part of the second hash function input, we use only one SHA-1 component with a multiplexor to select input data. Before processing each SHA-1 processing, HMAC must be initialized with the TPM command of TCG specifications such as SHA_INIT or HMAC_INIT.

The HMAC controller manages the data flow through the circuit. Since it is necessary to wait until the hash function is performed, *sha_start* and *sha_end* signals are used in order to control overall input data for hash functions.

The memory used in our design is a register based and single port 512-bit memory using standard logic cells. In order to minimize the power consumption, the internal registers of memory are disabled when they are not being used, thus reducing the amount of unwanted switching activity.

4 Implementation Results and Comparison

All hardware architectures of our design were first described in VHDL, and their operation was verified through functional simulation using Active HDL, from Aldec Inc. The design was fully verified using a large set of test vectors.

In order to evaluate our compact SHA-1 core and HMAC design, we used Synopsys synthesize flows for the targeted technology. For the target technology,

we used 0.25 μ m CMOS standard cell library from Samsung Electronics. The applied voltage is 2.5V and the operating frequency is 25 MHz.

Although the maximum operating frequency obtained using timing analysis is 100 MHz, we use 25 MHz as the operating frequency for evaluating our circuit because the system clock of most mobile phones is about 20 MHz.

After synthesis, Synopsys PowerCompiler was used to calculate the overall power dissipation of our design. The activity of the netlist was estimated for various test messages so that the netlist activity could be considered as reasonable values. We would like to emphasize that our design is on the algorithmic and architectural level. Implementing our designs using an low power ASIC library or a full custom design will enable higher energy and power savings.

Table 1. Components and their complexity of SHA-1 core

Component	gates	percentage
Interface	568	6.9
memory	3,600	43.7
data expansion	378	4.6
controller	420	5.1
reg_a~e	1120	13.6
adder	360	4.4
data compression	1,784	21.7
Total	8,230	100%

Table 2. Components and their complexity of HMAC circuit

Component	gates	percentage
Interface	242	1.5
control reg.	304	1.9
controller	270	1.7
memory	6,800	41.6
out_sel	474	2.9
sha-1 core	8,230	50.4
Total	16,320	100%

Table 3. Power estimation and operating cycles for implemented design

Operation	power(mA)	clock cycles
SHA-1 core	1.1	430
HMAC	2.58	1,750

Table 1 and table 2 show the synthesis results of both SHA-1 core and HMAC design based on the logic blocks and circuit area. Table 3 summarize the results of power estimation and operating clock cycles for both SHA-1 core and HMAC circuit. Our SHA-1 core consumes an area of 8,230 gates and needs less than 430 clock cycles to compute the hash of 512 bits of data. The HMAC circuit requires additional 8,090 logic gates and it can computes 512-bit data block using at most 1,750 clock cycles. At this point, there are relatively few works available for comparison of consuming power because some of previous works did not provide information and others were synthesized and implemented using FPGA devices. Our design consumes the operating current about 1.1mA for SHA-1 calculating and 2.58 mA for HMAC computation at the 25 MHz operating frequency.

Table 4. Comparison with previous works of SHA-1 ASIC implementations

SHA-1 computation	Tech.(μm)	Freq.(MHz)	Circuit area
This work	0.25	100	8,230
Y.Ming-yan [3]	0.25	143	20,536
S.Dominikus [4]	0.6	59	10,900

In table 4, we present the comparison of our design with some previous works for SHA-1 ASIC designs. It can easily be seen from table 4 that our SHA-1 core uses less hardware resources than that of previous works by 27%-60%.

Table 5. Comparison with commercial TPM chips based on SHA-1 computations

	Operating freq.(MHz)	SHA-1 performance
This work	25	$<18 \mu s/64\text{-byte}$
AT97SC3203 [11]	33	$<50 \mu s/64\text{-byte}$
SSX35A [13]	33	$<258 \text{ ms}/1\text{M-bit}$

There exist several commercial TPM chips implementing SHA-1 algorithm [11], [12], [13]. In table 5, we present the comparison of our design with the most representative commercial TPM chips with the same functionality. Although the operating frequency of the proposed implementation is much lower than those of [11] and [13], the achieved throughput exceeds SHA-1 circuits of some commercial TPM chips mainly designed for high-speed desktop computers.

5 Conclusions

In this work, we proposed a compact yet high-speed architecture for both a low power SHA-1 core and HMAC circuit and evaluated through simulation and synthesis for ASIC implementation. The SHA-1 core has a chip area of 8,320 gates and has a current consumption of $1.1mA$ at a frequency of 25MHz. The HMAC circuit based on the compact SHA-1 core requires additional 8,090 gates and has a current consumption of $2.58mA$ at the same conditions. The SHA-1 and HMAC calculation of 512 bits data require 430 and 1,750 clock cycles respectively.

To our best knowledge, the proposed design is at least 270% faster than any commercial TPM chips supporting SHA-1 circuit, while using lower operating frequency and achieving a reduction of the required hardware. The results of power consumption, throughput, and functionality make our low power SHA-1 core and HMAC hardware suitable for trusted mobile computing and other low-end embedded systems that urge for high-performance and small-sized solutions. However, the major design advantage of our design is the low power dissipation that is required to calculate the hash and MAC value of any given messages.

References

1. NIST: Secure Hash Standard FIPS-Pub 180-1. National Institute of Standard and Technology (1995)
2. NIST: The Keyed-Hash Message Authentication Code FIPS-Pub 198. National Institute of Standard and Technology (2002)
3. Ming-yan, Y., Tong, Z., Jin-xiang, W., Yi-zheng, Y.: An Efficient ASIC Implementation of SHA-1 Engine for TPM. In: IEEE Asian-Pacific Conference on Circuits and Systems, pp. 873–876 (2004)
4. Dominikus, S.: A Hardware Implementation of MD4-Family Hash Algorithms. In: IEEE international Conference on Electronic Circuits and Systems. vol. 3, pp. 1143–1146 (2002)
5. Kang, Y.-K., et al.: An Efficient Implementation of Hash Function processor for IPSec. In: IEEE Asia-Pacific Conference on ASIC, pp. 93–96 (2002)
6. Michail, H.E., Kakarountas, A.P., Selimis, G.N., Goutis, C.E.: Optimizing SHA-1 Hash Function for High Throughput with a Partial Unrolling Study. In: Paliouras, V., Vounckx, J., Verkest, D. (eds.) PATMOS 2005. LNCS, vol. 3728, pp. 591–600. Springer, Heidelberg (2005)
7. Sklavos, N., Dimitroulakos, G., Koufopavlou, O.: An Ultra High Speed Architecture for VLSI Implementation of Hash Functions. In: 10th IEEE International Conference on Electronics, Circuits and Systems, pp. 990–993 (2003)
8. Huang, A.L., Penzhorn, W.T.: Cryptographic Hash Functions and Low-Power Techniques for Embedded Hardware. In: IEEE ISIE'2005, pp. 1789–1794 (2005)
9. Selimis, G., Sklavos, N., Koufopavlou, O.: VLSI: Implementation of the Keyed-HASH Message Authentication Code for the Wireless Application Protocol. In: 10th IEEE International Conference on Electronics, Circuits and Systems, pp. 24–27 (2003)
10. Michail, M.K., Kakarountas, A.P., Milidonis, A., Goutis, C.E.: Efficient Implementation of the Keyed-Hash Message Authentication Code (HMAC) using the SHA-1 Hash Function. In: 11th IEEE International Conference on Electronics, Circuits and Systems, pp. 567–570 (2004)
11. AT97SC3203: Atmel corp. (2005) available at <http://www.atmel.com/>
12. SLB 9635 TT1.2: Infineon (2005) available at: <http://www.infineon.com/>
13. SSX35A: Sinosun (2005) available at <https://www.trustedcomputinggroup.org/>

A Secure DRM Framework for User's Domain and Key Management*

Jinheung Lee¹, Sanggon Lee^{2,**}, and Sanguk Shin¹

¹ Interdisciplinary Program of Information Security, The Graduate School, Pukyong National University, 599-1 Deayeon 3-dong, Nam-Gu, Busan, Korea

² Division of Internet Engineering, Dongseo University, Busan 617-716, Korea
jinhung@hanmail.net, nok60@gdsu.dongseo.ac.kr, shinsu@pknu.ac.kr

Abstract. Digital rights management(DRM) systems are used to control the use and distribution of copyrighted content. OMA specification creates devices collection of domain, and the use of contents is free in this domain. Domains are created and managed directly by the RI(rights issuer) that issues rights to the domain. In this paper, we propose a new rights object acquisition protocol(ROAP) for DRM and a efficient key distribution protocol. The proposed ROAP provides billing functionality for purchasing rights via network operator which was not considered in OMA DRM 2.0 specifications.

1 Introduction

In the past year there has been an increasing interest in developing digital rights management(DRM) system. The main purpose of a DRM system[1,8] is providing digital data content in a way that protects the copyrights of content providers(CPs) and to enable options for new business models for content distribution. The DRM system of reference [1] and [2] enables CPs to distribute protected contents and rights issuers(RIs) to issue rights objects(ROs) for the protected content. For user consumption of the contents, users acquire permissions to protected contents by contacting RIs. RIs grants appropriate permissions for the protected contents to user devices. The content is cryptographically protected when distributed; hence, the protected content will not be usable without an associated RO issued for the user's device. The protected contents can be delivered to the device by any means. But the ROs are tightly controlled and distributed by the RI in a controlled manner.

Open Mobile Alliance(OMA) has released OMA DRM 2.0, a DRM standard, which improves the previous version. However, it does not specify a complete DRM infrastructure. For example, billing functionality for obtained rights is not provided and the mutual authentication between the CPs and the RIs is not covered.

* This research was supported by the Program for the Training of Graduate Students in Regional Innovation which was conducted by the Ministry of Commerce Industry and Energy of the Korean Government.

** This work was supported by University IT Research Center Project of MIC, Korea.

Recently, based on the OMA DRM 2.0 specification, reference [3] proposed the general system architecture for mobile DRM in which the mobile network operator is included for provisioning billing functionality. However, they mentioned only security requirements and a copy detection mechanism, but a key management protocol required for acquiring the copyright and a protocol for the billing process were not considered.

The work presented in this paper consists in the development of the empirical and DRM architecture with the following objectives:

A new ROAP providing a billing functionality

In this paper we propose a new ROAP provisioning a billing functionality via network operator(NO) which was not considered in the OMA DRM 2.0 standard. We use mobile phones to get digital rights via a NO. The NO can then be used to extract billing information and charge the end users together with the phone bill.

An efficient key distribution protocol of OMA DRM

In OMA DRM 2.0, the RSA is a default cryptographic primitive for key transfer and protocol message signing. We can archive a efficient key distribution protocol by replacing the existing public key encryption schemes used with a symmetric key encryption scheme in the key distribution protocol.

2 User's Domain and ROAP in OMA DRM 2.0

2.1 Domain

A Domain is a set of devices that process a common domain key provisioned by a RI. Devices in a domain may share domain RO and are able to consume and share any DRM contents format controlled by domain RO.

OMA DRM uses the concept of a domain to share content among a group of users. An RI defines the Domains, manages the domain keys, and controls which and how many devices are included and excluded from the domain. DRM agent of device may join or leave a domain by making a request to the RI that created the domain. RO intended for the domain are encrypted using an right encryption key(REK) itself encrypted with a domain key that is unique for that domain. Each domain key corresponds to a specific domain generation.

2.2 The ROAP Suite

The ROAP is the common name for a suite of DRM security protocols between a RI and a DRM agent in a device. The protocol suite contains a 4-pass protocol for registration of a device with an RI and two protocols by which the device requests and acquires RO. The 2-pass RO acquisition protocol encompasses request and delivery of an RO whereas the 1-pass RO acquisition protocol is only a delivery of an RO from an RI to a device. The ROAP suite also includes 2-pass protocols for devices joining and leaving a domain; the join domain protocol and the leave domain protocol.

The 4-pass Registration Protocol. The registration protocol is a complete security information exchange and handshake between the RI and the device and is generally only executed at first contact, but may also be executed when there is a need to update the exchanged security information. This protocol includes negotiation of protocol parameters and protocol version, cryptographic algorithms, device ID and RI ID for authentication, integrity protection of messages and optional device DRM time synchronization. Successful completion of the registration protocol results in the establishment of an RI context in the device containing RI-specific security information.

The 2-pass RO Acquisition Protocol. The 2-pass RO acquisition protocol is the protocol by which the device acquires RO. This protocol includes mutual authentication of device and RI, integrity-protected request and delivery of ROs, and the secure transfer of cryptographic keying material necessary to process the RO. The successful execution of this protocol assumes the device to have a pre-established RI context with the RI.

The 1-pass RO Acquisition Protocol. The 1-pass RO acquisition protocol is designed to meet the messaging/push use case. Its successful execution assumes the device to have an existing RI context with the sending RI. The 1-pass protocol is essentially the last message of the 2-pass variant.

The 2-pass Join Domain Protocol. The join domain protocol is the protocol by which a device joins a domain. The protocol assumes an existing RI context with the RI administering the domain. Successful completion of the protocol results in the establishment of a domain context in the device containing domain specific security related information including a domain key.

The 2-pass Leave Domain Protocol. The leave domain protocol is the protocol by which a device leaves a domain. The protocol assumes an existing RI context with the RI administering the domain.

3 A ROAP for Secure DRM System

3.1 Structure of Secure DRM System Architecture

Figure 1 shows the structure and processing procedure of the secure DRM system proposed in this paper. The proposed DRM system was built on the generalized secure DRM model of reference [3] and provides billing functionality via network operator. Refer to [3] for the details of each block in the Figure 1.

The processing steps of the proposed system are as follows. The contents provider registers in the RI (step 1 and 2). Publish the digital contents on the web server after encrypting it with contents encryption key(CEK) which is a symmetric key (step 3). Insert a watermark or a digital fingerprint, if needed, before the encryption. Register the CEK in the RI via a secret channel (step 4).

A device downloads some digital clips selected from the CP webpage by a user in step 5 and step 6. The DRM contents consuming agent in a device demands the CEK from the DRM secure device agent (step 7). Before getting some digital rights, device must registers in RI according through the 4-pass registration protocol. At this time, device shares the master key with the RI. This master key is used in the key distribution protocol of the proposed DRM system results in a light weighted protocol and used for protecting privacy of the device users such as the contents consuming information against the network operator. This master key is also used to protect a domain key distributed to a device during the join domain protocol.

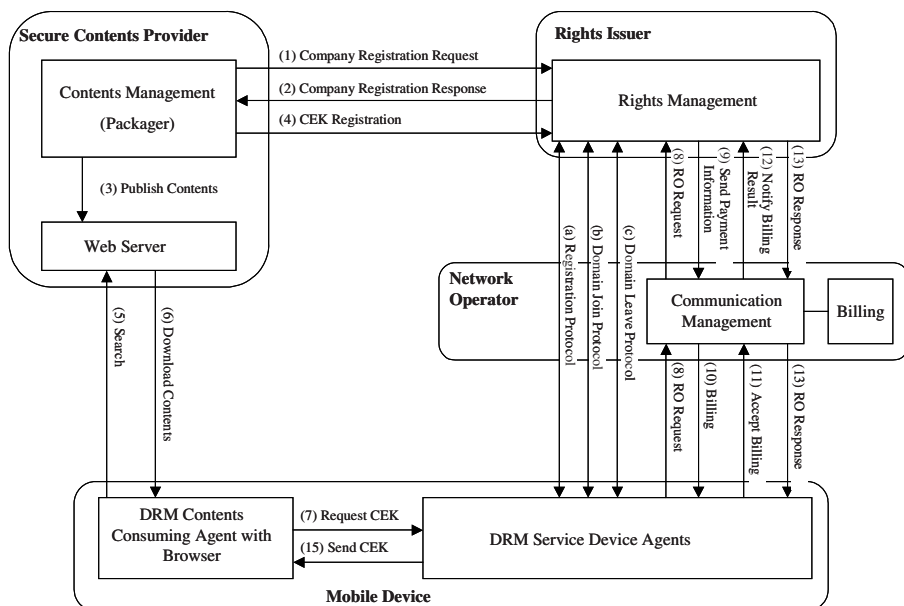


Fig. 1. Secure DRM system architecture

The DRM secure device agent in device sends a RO request message including RO identifier associated with the selected digital clips to the network operator. The network operator relays the request message to the RI via network operator RO request message. The RI decodes network operator RO request message and sends payment information message including the price of the requested RO to the network operator. The network operator sends billing messages to the device. The device sends accept billing message which confirms a payment for the RO purchase to the network operator and the network operator sends notify billing result message to the RI. The RI encrypts the REK of RO using the master key shared with device, and then sends RI, RO response message including the protected REK and CEK to the network operator. The network operator sends a network operator RO response message to the device. The message includes the results of the billing process which will charge the end users together with the phone bill. The DRM secure device agent checks the constraints details of RO and passes the CEK to DRM contents consuming agent if

the permission is still valid. The DRM contents consuming agent decodes the encrypted contents with the CEK and plays it. The detail of RO acquisition steps(step 8 to step 13) are described in the following section.

3.2 New ROAP Suits

In this section we describe a new ROAP suits suitable for DRM in figure 1. The proposed ROAP is based on the ROAP of the existing OMA DMR 2.0 specification and modified and extended to e suitable for the proposed DRM system architecture. The proposed ROAP suits consist of 4-pass registration protocol, 4-pass ROAP, 2-pass join domain protocol and 2-pass leave domain protocol. Figure 2 shows the proposed ROAP process. The (1) and (2) in figure 2 are the same as the existing one.

Table 1. ROAP message parameters

Parameter	Meaning
ID_{Dev}	Device ID
ID_{RI}	Rights Issuer ID
ID_{NO}	Network Operator ID
ID_{Domain}	Domain ID
Ver	Version
N_{Dev}	Device Nonce
N_{RI}	Rights Issuer Nonce
Sid	Session ID
T_{Req}	Request Time
$Sig_A()$	Signature of Entity A
Msg	Message excluding signature
$Status$	ROAP message handling status(Success or Fail)
$RIURL$	Rights Issuer URL
$MasterKey_{Protected}$	Master key encryption by public key crypto
$DomainInfo$	Domain information
$E_{Master}()$	Symmetric encryption by the key 'MASTER'
$ROinfo$	Rights Object Information
$Price$	Rights Object Price
$BillAns$	Answer to Billing (Accept or Reject)
$PayStatus$	Payment Status (Success or Fail)
$OCSP_{Res}$	OCSP Response

Table 1 explains the meanings of parameters used in the proposed ROAP message. As shown in figure 2-(1), the registration response message of the 4-pass registration protocol has an additional parameter $MasterKey_{Protected}$ which is not present in the existing protocol. This parameter is a symmetric key encrypted by public key system. This is a master key shared between a RI and a device, and used for key distribution. The IEEE 1363 standard DL/ECIES[4] combining Diffie-Hellman algorithm for the elliptic curve and AES is used to distribute the $MasterKey$. By sharing the symmetric key between RI and device through 4-pass registration protocol, we can provide symmetric key based key distribution for ROAP. Thus we can light weight the ROAP key management protocol than that of OMA DRM 2.0.

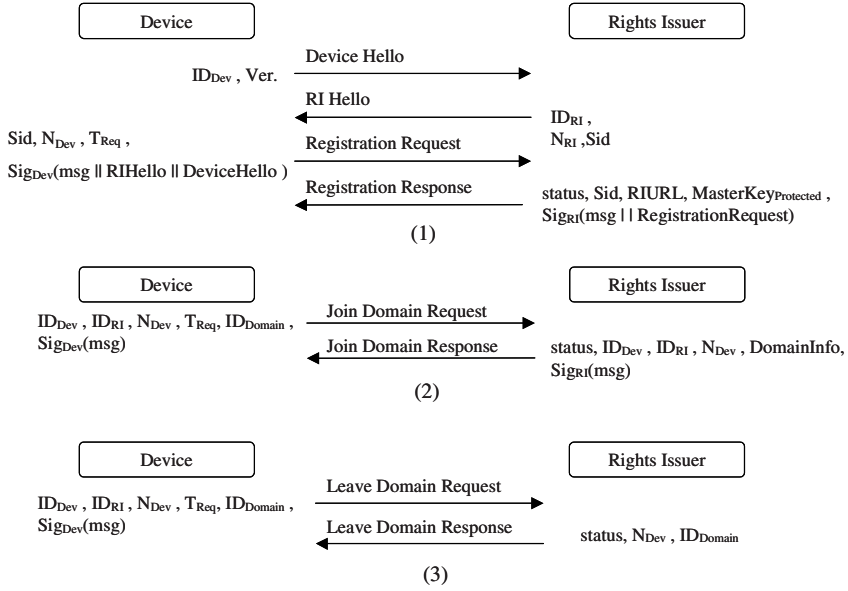


Fig. 2. Proposed ROAP suits except RO acquisition protocol; (1) 4-pass registration protocol, (2) 2-pass domain join protocol, (3) 2-pass domain leave protocol

In figure 2-(2), domain key in *DomainInfo* parameter in join domain response message is securely distributed by encrypting it with one time session key shared between device and RI. This session key is generated by the key derivation function defined in PKCS#1 with the inputs N_{Dev} and $MasterKey$.

Figure 3 shows the 4-pass ROAP. Based on the DRM general model proposed by reference [3], we designed a ROAP providing billing functionality. Every time 4-pass ROAP is executed, two session keys, *MeKey* and *KeKey*, are generated. *MeKey* is for protocol message protection and *KeKey* is for key protection. These two keys are generated as follow by key derivation function taking two inputs $MasterKey$ and N_{Dev} .

$$\{MeKey, KeKey\} = KDF(MasterKey, N_{Dev}) \quad (1)$$

where KDF stands for key derivation function. The *ROinfo* of RO request message must encrypt using *MeKey* because purchase information of contents is included.

The device checks the correctness of the price by comparing the price sent by the network operator with the one sent by the RI. The RI confirms the purchase intention of device and the corresponding payment result of network operator by checking $E_{MeKey}(ROinfo, BillAns, Price)$ and payment status parameter *PayStatus* in the notify billing result message.

The device confirms that the content is fully paid and the RO is acquired correctly since the RO response message received from network operator contains $E_{MeKey}(PayStatus)$ generated by RI in addition to its own *PayStatus*.

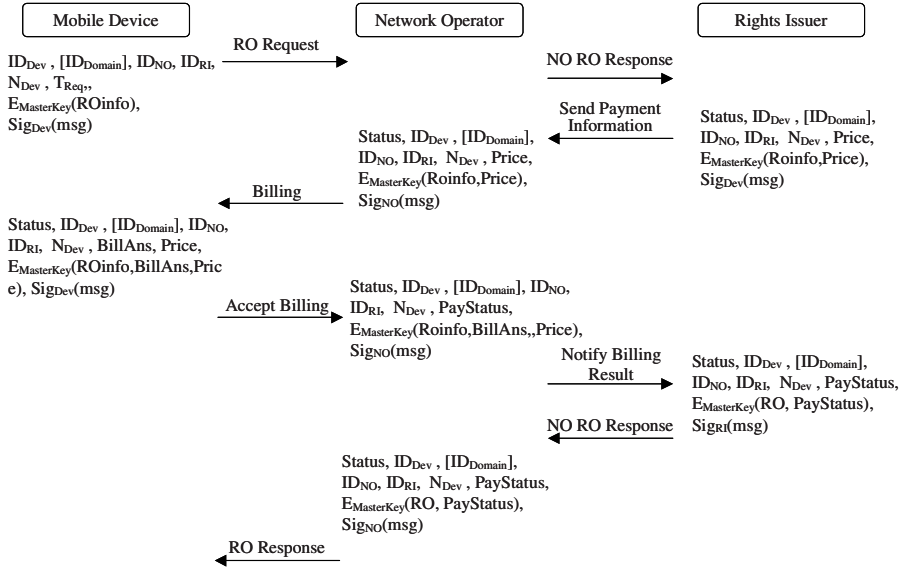


Fig. 3. 4-pass RO acquisition protocol

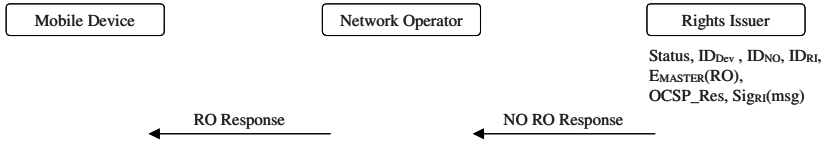


Fig. 4. 1-pass RO acquisition protocol

Figure 4 shows the 1-pass ROAP. This protocol can be applied to a push type delivery model. When RI sends a RO to device, it protects the rights encryption key or domain key in RO with session key $KeKey$. For 1-pass ROAP, key derivation function(KDF) takes N_{RI} instead of N_{Dev} as an input to compute $KeKey$. ECDSA is used for the signature of each message.

4 Lightweight Key Management Protocol

4.1 Sharing of Master Key Between RI and Devices

As explained in 4-pass registration protocol, RI sends MasterKey to device. This shared master key is used to generate session keys in a 4-pass RO acquisition, 1-pass RO acquisition and 2-pass join domain protocol. We use IEEE 1363 DL/ECIES to distribute this shared master key. This algorithm is composed of Diffie-Hellman algorithm over elliptic curve and AES.

The EL/ECIES encryption algorithm is

$$\{C, T, V\} = DL/ECIES-Enc(MasterKey, \text{Sender's Private Key}, \text{Receiver's Public Key}) \quad (2)$$

where C is ciphertext corresponding to $MasterKey$, T is key confirmation code, and V is sender's public key. The DL/ECIES decryption algorithm is

$$\{MasterKey, Result\} = DL/ECIES-Dec(C, V, T, \text{Receiver's Private Key}) \quad (3)$$

where $Result$ is the status of key verification result.

4.2 Distribution of Domain Key(K_D) and Rights Encryption Key(K_{REK})

In the OMA DRM 2.0, RSAES-KEM-KWS[5] defined in the X9.44 is used to send the K_{REK} included in the RO parameter of the RO response message and K_D included in the *DomainInfo* parameter in the join domain response message. This key distribution scheme is composed of RSA and AES. We use symmetric key encryption type key distribution scheme AES WRAP[6](the IETF standard RFC 3394) instead of public key encryption type key transfer scheme RSAES-KEM-KWS.

Figure 5 shows the mechanism of distributing K_{REK} or K_D to the device using AES WRAP scheme with the session key $KeKey$ generated by equation(1). $KeKey$ for join domain response message, and RO response and network operator RO response messages in 4-pass ROAP is derived from equation(1). But $KeKey$ for network operator RO response and RO response message in 1-pass RO acquisition is derived from equation(1) with input N_{RI} instead of N_{Dev} .

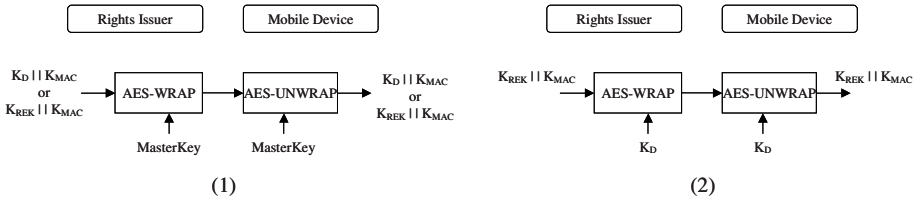


Fig. 5. (1) K_D or K_{REK} distribution under $MasterKey$, (2) K_{REK} and K_{MAC} transmission under $DomainKey$

In the OMA DRM 2.0, the symmetric key K_{MAC} is also sent together with K_{REK} or K_D as shown in figure 5-(1). K_{MAC} is used for key confirmation by message authentication code. Once the domain key is distributed, the K_{REK} for domain RO is distributed under the K_D as shown in the Figure 5-(2).

Table 2 shows the protocol messages employing the key distribution mechanism. Among the five ROAP protocols, the 4-pass registration and 2-pass leave domain protocols do not employ the key transfer mechanism.

The OMA DRM 2.0 specification uses a key transfer mechanism combining RSA and AES, and uses RSA as a default signature scheme. By sharing master key

between RI and device in the registration stage, we can provide symmetric key based key distribution mechanism. For we use elliptic curve crypto algorithm to share a master key in the registration stage, elliptic curve based signature scheme can be used efficiently without requiring another public key crypto engine such as RSA engine.

As shown in Table 2, all of the protocol messages use digital signature and the usage of the message carrying encrypted keys are high. So we archive high efficiency in key transfer and signature by introducing symmetric key management and elliptic curve signature scheme respectively.

Table 2. The protocol messages employing the key distribution mechanism

Protocol	Message	Use frequency	Digital signature
4-pass registration	Not applicable	Low	O
4-pass RO acquisition	RO response	High	O
1-pass RO acquisition	RO response	High	O
2-pass join domain	Join domain response	High	O
2-pass leave domain	Not applicable	Low	O

4.3 Security Analysis

A. Security of master key transfer.

Because we adapt the DL/ECIES defined in the X9.44 and IEEE 1363a standards as the master key transmission scheme, we can rely on the security of the standard.

B. Security of the key transmission protocol

Because we employ an IETF standard key wrapping algorithm AES WRAP in the key transmission protocol, we can count on its security. The key material to be transmitted with AES WRAP is not a finite element. So it is safe from a partition attack that is a kind of offline dictionary attack [7].



Fig. 6. Show screen that execute 4-pass registration protocol in RI



Fig. 7. Show screen that execute 2-pass RO acquisition protocol in RI

5 Implementation of ROAP

In this section, we implement ROAP protocol that described in section 3 and 4. The environment used for implementation of system is as following: program language is C and C++, OS is Windows XP professional and, DB used MySQL 4.0. All procedures of the implemented system must be performed as using secure agent of user's devices. That is, the user installs secure device agent and, registers device to RI by 4-pass registration protocol. After run this procedure, user can see information of own ROs.

Figure 6 shows RI server. If secure device agent of user's was registered, RI issues *MasterKey* and RO to secure device agent. Figure 7 shows screen that secure device agent acquires RO by 2-pass RO acquisition protocol.

6 Conclusions

We proposed a new ROAP for digital rights management provisioning billing functionality via network operator. The proposed ROAP can protect the consumer's purchase privacy information against network operator.

We also proposed the efficient key management protocol for OMA DRM 2.0. The key management protocol is light weighted by replacing the existing public key encryption method with the symmetric key encryption method for sending the right encryption key and domain key. Because the elliptic curve system is used for the master key transfer, the elliptic curve encryption method can be easily applied for signing protocol messages without introducing additional public crypto engine such as RSA engine. Therefore, it is expected to reduce the power consumption and processing time in device.

References

1. Open Mobile Alliance, DRM Architecture, Draft Version 2.0 (August 2004)
2. Open Mobile Alliance, DRM Specification, Candidate Version 2.0 (July 2005)
3. Soriano, M., Flake, S., Tacke, J., Bormann, F., Tomas, J.: Mobile digital Rights Management: Security Requirements and Copy Detection Mechanism. In: Database and Expert Systems Applications 2005, Proceedings Sixteenth Information Workshop, pp. 251–256 (2005)
4. IEEE 1363a, IEEE Standard Specification for Public-Key Cryptography – Amendment 1: Additional Techniques (2004)
5. Draft ANSI X9.44, Public Key Cryptography for the Financial Services Industry – Key Establishment Using Integer Factorization Cryptography, Draft 6 (2003)
6. Schaad, J., Housley, R.: Advanced Encryption Standards(AES) Key Wrap Algorithm, RFC 3394 (September 2002)
7. Patel, S.: Number Theoretic Attacks on Secure Password Schemes. In: Proceedings of the Symposium on Security and Privacy, IEEE, pp. 236–247 (1997)
8. Popescu, B.C., Kamperman, F.L.A.J., Crispo, B., Tanenbaum, A.S.: A DRM Security Architecture for Home Networks. In: Proceedings of the 4th ACM workshop on Digital Rights Management, pp. 1-10 (2004)

A Secret-Key Exponential Key Agreement Protocol with Smart Cards

Eun-Jun Yoon¹ and Kee-Young Yoo^{2,*}

¹ Faculty of Computer Information, Daegu Polytechnic College,
42 Jinri-2gil (Manchon 3dong San395), Suseong-Gu, Daegu 706-711, South Korea
ejyoon@tpic.ac.kr

² Department of Computer Engineering, Kyungpook National University,
1370 Sankyuk-Dong, Buk-Gu, Daegu 702-701, South Korea
Tel.: +82-53-950-5553; Fax: +82-53-957-4846
yook@knu.ac.kr

Abstract. The smart card based remote user authentication and key agreement protocol is a very practical solution to create a secure distributed computer environment. In this paper, we propose a smart card based secret-key exponential key agreement protocol called SEKA, which provides mutual authentication and key agreement over an insecure channel between user and server. The computational complexity that the client must perform is just one exponentiation and two hash functions during the runtime of the protocol.

1 Introduction

User authentication is a process that verifies a user's identity to ensure that the person requesting access to the private network is in fact, that person to whom entry is authorized. As such, a remote password authentication scheme authenticates the legitimacy of users over an insecure channel, where the password is often regarded as a secret shared between the remote system and the user. Based on knowledge of the password, the user can use it to create and send a valid login message to a remote system to gain the right to access. Meanwhile, the remote system also uses the shared password to check the validity of the login message and authenticate the user.

Following the fast growth of Internet, the smart card based remote user authentication and key agreement protocol is a very practical solution to create a secure distributed computer environment. Typically, smart cards provide a cryptographic token. Smart cards as an ubiquitous computing device provide several services to pervasive computing and ubiquitous services. They are a small, secure, temper-proof and cost effective medium to store personal data such as profiles or identification features. Their ability to store encryption keys and to encrypt

* Corresponding author.

or decrypt data as well as the possibility to load and run executable programs on current Java based cards make them ideally suited for being used as personalized computing resources and for a variety of tasks such as authentication, identification, and management of personal profiles.

In 1981, Lamport [1] proposed a remote password authentication scheme using a password table to achieve user authentication. However, one of the weaknesses of Lamport's scheme is that a verification table should be stored in the remote system in order to verify the legitimacy of a user. If an intruder can somehow break into the server, the contents of the verification table can be easily modified. Thus, recently, many password authentication schemes [2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17] have recognized this problem and proposed solutions using smart cards in which the verification table is no longer required in the remote system to improve security, functionality and efficiency. Due to their portability and the cryptographic capacity, smart cards have been widely used in many e-commerce applications. However, those all schemes do not provide the session key agreement mechanism. In 2004, Juang [17] first proposed an efficient password authenticated key agreement protocol using smart cards. After a user passes user authentication check of a server, the transmitted messages between the user and the server must be kept secret when the user uses a service of the server. They must agree on a session key to be used for protecting their subsequent communications.

In this paper, we propose a smart card based secret-key exponential key agreement protocol (SEKA) with more security and less computational cost than Juang's protocol. The proposed SEKA protocol is based directly on Simple Password Encrypted Key Exchange (SPEKE) [18], with modifications to fit the needs of this specific problem and to improve performance. SPEKE is similar to Encrypted Key Exchange (EKE) protocol [19], but instead of encrypting the Diffie-Hellman public numbers using hashed password $W = f(pw)$ by using a function $f(\cdot)$ to convert the secret values into a base for exponentiation. It uses a secret generator derived as a function $f(\cdot)$ of W instead of a fixed generator g . The generator is still a function of the user's password pw . Based on SPEKE protocol, SEKA protocol's main merits are as follows: (1) the protocol needs no verification table; (2) users can freely choose and securely change their own passwords; (3) the communication bandwidth and computational load are very low; (4) users and servers can authenticate each other and it generates a session key agreed by the user and the server; (5) Unlike many timestamp based authentication schemes [2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17], the protocol do not have a serious time-synchronization problem.

The remainder of this paper is organized as follows: Section 2, we describe security requirements of smart card based key agreement protocol. Section 3 briefly review Juang's key agreement protocol using smart card. The proposed SEKA protocol is presented in Section 4, while Section 5 and 6 discuss the security and efficiency of the proposed protocol, respectively. Some final conclusion are given in Section 7.

2 Protocol Requirements

The following security requirements are important for user authentication and key agreement protocol using smart card.

- No verification table: No password or verification table is required in a server.
- Freely chosen and change password: Users can freely choose and change their own passwords.
- Resistance to replay attack: It should be guaranteed that attackers can not impersonate or deceive another legitimate participant through the reuse of information obtained in a protocol.
- Explicit mutual authentication: Users and servers can authenticate each other.
- Session key agreement: The session key must be agreed between the user and the server so that it can support the dynamic key.
- Forward-Secrecy: Forward-Secrecy should be provided to ensure that attackers can not compute session keys from the sessions eavesdropped on previously, even when a long-term secret keying material of the entity participating in the protocol has been revealed.

The following efficiency requirements are also important for user authentication and key agreement protocol using smart card.

- Low computational load: The protocol requires a low computational load that can be borne by even low-power devices such as smart cards, and pre-computation to minimize online computation operations.
- Minimum number of message exchanges: In terms of network resource efficiency and network delay, it is advantageous to have as few communication rounds as possible. Therefore, the number of messages to be exchanged between the user and the server should be kept to a minimum.
- Minimum communication bandwidth use: The protocol message should be as short as possible.

3 Review of Juang's Password Authenticated Key Agreement Protocol Using Smart Cards

This section briefly reviews Juang's password authenticated key agreement protocol using smart cards. Juang's protocol is composed of two phases, which are the registration phase and the login and session key agreement phase.

Registration Phase: Assume that U_i submits his identity ID_i and password PW_i to the server S for registration. If the server S accepts this request, it will perform the following steps:

- Step 1. Compute U_i 's secret information $v_i = h(ID_i, x)$ and $w_i = v_i \oplus PW_i$, where x is a secret-key of server, $h(\cdot)$ is secure strong one-way hash function and $\oplus$ is bit-wise exclusive or operation.
- Step 2. Store ID_i and w_i to the memory of a smart card and issue this smart card to U_i .

Login and Session Key Agreement Phase: After getting the smart card from the server S , U_i can use it when he logs into the server. If U_i wants to log into S , he must attach his smart card to a card reader. He then inputs his identity ID_i and his password PW_i to this device. The following protocol is the j th login with respect to this smart card.

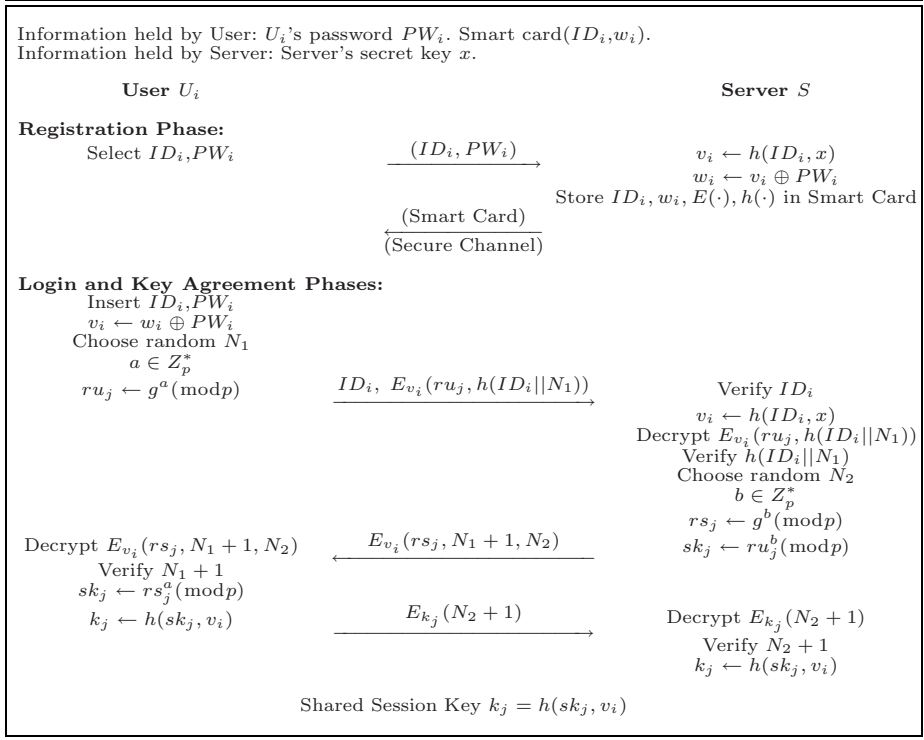


Fig. 1. Juang's DH-based password authenticated key agreement protocol

Step 1. $U_i \longrightarrow S : N_1, ID_i, E_{v_i}(ru_j, h(ID_i||N_1))$.

U_i 's smart card first computes $v_i = w_i \oplus PW_i$ and sends his ID_i , a nonce N_1 and the encrypted message $E_{v_i}(ru_j, h(ID_i||N_1))$ to S , where $E(\cdot)$ is a symmetric encryption function. The encrypted message includes the j th random value ru_j , which is used for generating the j th session key k_j , and the authentication tag $h(ID_i||N_1)$, which is for verifying the identification of U_i .

Step 2. $S \longrightarrow U_i : E_{v_i}(rs_j, N_1 + 1, N_2)$.

On receiving the message in Step 1, S first computes $v_i = h(ID_i, x)$ and then decrypts the message by computing $D_{v_i}(E_{v_i}(ru_j, h(ID_i||N_1)))$, where $D(\cdot)$ is a symmetric decryption function, and then checks to see if the message contains the authentication tag $h(ID_i||N_1)$ and if the nonce

N_1 is fresh. S rejects this login if the tag is not valid. If it is valid and the nonce N_1 is fresh, S sends the encrypted message $E_{v_i}(rs_j, N_1 + 1, N_2)$ back to U_i . The encrypted message includes the random value rs_j chosen by S , which is used for generating the j th session key k_j , and the nonce N_2 , which is for freshness checking.

Step 3. $U_i \rightarrow S : E_{k_j}(N_2 + 1)$.

On receiving the message in Step 2, U_i 's smart card decrypts the message by computing $D_{v_i}(E_{v_i}(rs_j, N_1 + 1, N_2))$. He then checks if the nonce $N_1 + 1$ is in it for freshness checking. If yes, U_i computes the j th session key $k_j = h(rs_j, ru_j, v_i)$ and sends the encrypted message $E_{k_j}(N_2 + 1)$ back to S .

Step 4. After receiving the message in Step 3, S decrypts the message by computing $D_{k_j}(E_{k_j}(N_2 + 1))$ and checks if the nonce $N_2 + 1$ is in it for freshness checking. Then U_i and S can use the session key k_j in secure communication soon.

In Juang's protocol, it can be used the Diffie-Hellman (DH) key agreement algorithm [20] for computing the session key and providing perfect forward secrecy. In this approach, we let $ru_j = g^a(\text{mod } p)$ and $rs_j = g^b(\text{mod } p)$, where p is a large prime number, g is a public, primitive element in $GF(p)$, a and b are random numbers chosen $\in Z_p^*$ by the user and the server separately, and shared session key $k_j = h(ru_j^y, v_i) = h(rs_j^x, v_i) = h(sk_j, v_i)$, where $sk_j = g^{ab}(\text{mod } p)$. Figure 1 illustrates the Juang's DH-based password authenticated key agreement protocol.

4 Proposed SEKA Protocol

In this section, we propose a smart card based secret-key exponential key agreement protocol(SEKA). The security of the proposed SEKA protocol is based on Diffie-Hellman (DH) key agreement [20] and one-way hash function [21,22], and consists of a registration, pre-computation, login and session key agreement phase. Instead of using a primitive element as the base for the exponential operation, SEKA uses a function $f(\cdot)$ to convert the secret values into a base for exponentiation. Figure 2 illustrates the proposed SEKA protocol.

Registration Phase: Let x be a secret-key maintained by the server. The user U_i submits their identifier ID_i and chosen password PW_i to the server S . These private data must be sent in person or over a secure channel. Upon receiving the registration request, the server S performs following steps:

Step 1. Compute $A_i = f(ID_i, x)$ and $B_i = A_i \oplus PW_i$, where $f(ID_i, x)$ is an element of large prime order in Z_p^* .

Step 2. Personalize the smart card to U_i with the secure information: $\{ID_i, A_i, B_i, h(\cdot), p\}$.

Pre-computation Phase: The user U_i 's smart card and card reader performs pre-computation in the idle time of the last running period. The pre-computation reduces time and computational load during the key agreement protocol execution. To be more specific, a random number a is selected from Z_p^* , and then $X_A = (A_i)^a \pmod{p}$ is calculated this pre-computation phase to key agreement protocol execution.

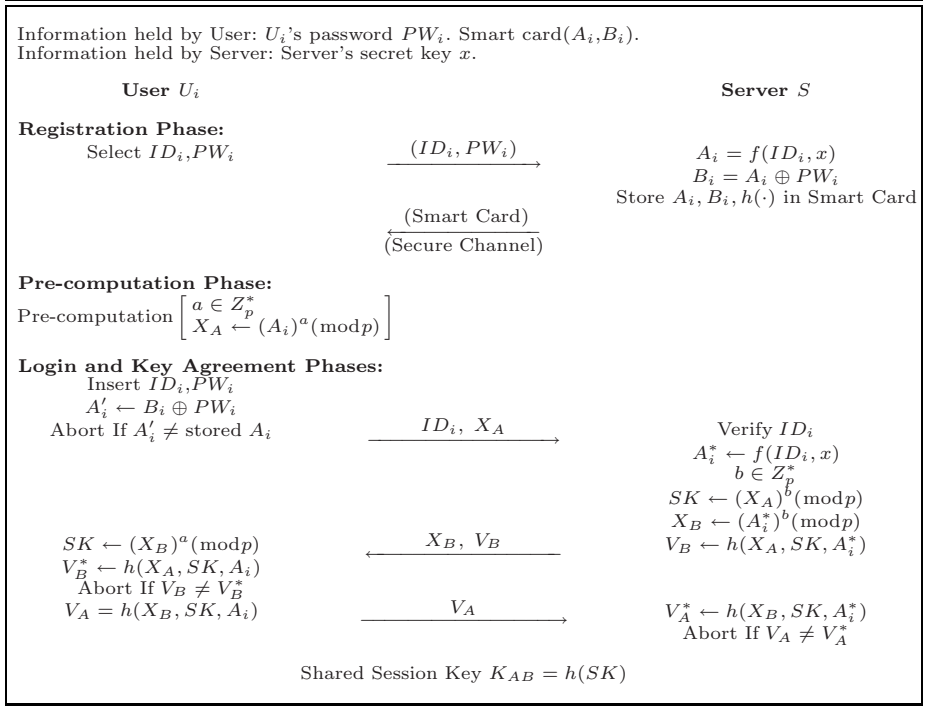


Fig. 2. Proposed SEKA protocol

Login and Session Key Agreement Phase: For mutual authentication and key agreement between the user U_i and the server S , S and smart card execute the following steps:

Step 1. $U_i \rightarrow S : ID_i, X_A$.

If U_i wants to login, they attach their smart card to the card reader and key in their identifier ID_i and password PW_i , then the smart card computes $A'_i = B_i \oplus PW_i$ and checks whether $A'_i \stackrel{?}{=} A_i$ holds or not. If it holds, the user sends a message ID_i and pre-computed value X_A to S .

Step 2. $S \rightarrow U_i : X_B, V_B$.

Upon receiving the authentication request message ID_i and X_A , S verifies the format of ID_i . If the format is correct, S computes $A_i^* =$

$f(ID_i, x)$. Then S selects a random number $b \in_R Z_q^*$ and computes the Diffie-Hellman key $SK = (X_A)^b = (A_i)^{ab}(\text{mod } p)$, $X_B = (A_i^*)^b(\text{mod } p)$ and $V_B = h(X_A, SK, A_i^*)$, where $h(\cdot)$ is a collision resistant one-way hash function. Finally, S sends X_B and V_B to the user.

Step 3. $U_i \longrightarrow S : V_A$.

Upon receiving the message X_B and V_B , U_i computes the Diffie-Hellman key $SK = (X_B)^a = (A_i^*)^{ab}(\text{mod } p)$ and $V_B^* = h(X_A, SK, A_i)$, and compares V_B and V_B^* . If they are equal, U_i believes that the responding part is the real server, and then sends V_A to S , where $V_A = h(X_B, SK, A_i)$. Otherwise U_i interrupts the connection.

Step 4. S computes $V_A^* = h(X_B, SK, A_i^*)$, and compares V_A and V_A^* . If they are equal, S then accepts the user U_i 's login request and the mutual authentication is complete, otherwise it rejects the login request. Finally, U_i and S compute the common session key $K_{AB} = h(SK) = h((A_i)^{ab}(\text{mod } p)) = h((A_i^*)^{ab}(\text{mod } p))$, respectively.

User U_i 's Password Change: If U_i wants to change their old password PW_i to a new password PW_i^* , U_i and smart card only need to perform the procedures below, without any help from the remote server.

Step 1. U_i inserts his smart card into the smart card reader of a terminal, and enters ID_i and PW_i .

Step 2. U_i 's smart card computes $A'_i = B_i \oplus PW_i$ and compares A'_i and stored A_i in smart card. If they are equal, then U_i selects new password PW_i^* .

Step 3. Finally, U_i 's the smart card compute $B'_i = A'_i \oplus PW_i^*$ and store B'_i in smart card in place of old B_i . Otherwise it reject the password change request.

5 Security Analysis

In this subsection, we analyzes the security of the proposed SEKA protocol. At first, we define the security terms needed for the analysis of the proposed SEKA protocol.

Definition 1. A weak secret (password) is a value of low entropy $W(k)$, which can be guessed in polynomial time.

Definition 2. A strong secret key is a value of high entropy $H(k)$, which cannot be guessed in polynomial time.

Definition 3. A secure one-way hash function $y = h(x)$ is one where given x it is easy to compute y and where given y it is hard to compute x .

Definition 4. The discrete logarithm problem (DLP) is explained by the following: Given a prime p , a generator g of Z_p^* , and an element $\beta \in Z_p^*$, find the integer α , $0 \leq \alpha \leq p-2$, such that $g^\alpha \equiv \beta(\text{mod } p)$.

Definition 5. *The Diffie-Hellman problem (DHP) is explained by the following: Given a prime p , a generator g of Z_p^* , and elements $g^c(\text{mod } p)$ and $g^s(\text{mod } p)$, find $g^{cs}(\text{mod } p)$.*

Here, five security properties: Guessing attack, replay attack, impersonation attack, mutual authentication, and perfect forward secrecy, would be considered for the proposed SEKA protocol. Under the above definitions, the following theorems are used to analyze the five security properties in the proposed protocol.

- (1) The proposed SEKA protocol can resist the server's secret key guessing attack. A guessing attack involves an adversary (randomly or systematically) trying long-term private keys (e.g. user password or server secret key), one at a time, in the hope of finding the correct private key. Ensuring long-term private keys chosen from a sufficiently large space can reduce exhaustive searches. Most users, however, select passwords from a small subset of the full password space. Such weak passwords with low entropy are easily guessed by using the so-called dictionary attack. Due to the fact that obtaining a and b from X_A and X_B is computationally infeasible, as it is a discrete logarithm problem by Definition 4, it is extremely hard for any attacker to derive the user's secret value A_i from X_A and X_B . Suppose that an attacker obtains the user's secret value $A_i = f(ID_i, x)$. Due to Definitions 2 and 3, however, it is also extremely hard for any attacker to derive the server's strong secret key x from $A_i = f(ID_i, x)$. Even if the smart card of U_i is picked up by an attacker, it is still difficult for the attacker to derive x .
- (2) The proposed SEKA protocol can resist the replay attack. A replay attack is an offensive action in which an adversary impersonates or deceives another legitimate participant through the reuse of information obtained in a protocol. For replay attacks, neither the replay of an old login message $\{ID_i, X_A\}$ in the login phase nor the replay of the server's response message $\{X_B, V_B\}$ in Step 2 and the user's response message V_A in Step 3 of the login and key agreement phase will work, as it will fail due to a and b are random values and the verification process V_B and V_A , respectively.
- (3) The proposed SEKA protocol can resist the impersonation attack. An attacker can attempt to modify a message X_A into X_A^* . However, such a modification will fail in Step 4 of the login and key agreement phase, because an attacker has no way of obtaining the value A_i to compute the valid parameter X_A . Although, an attacker try to modify a message X_B into X_B^* for masquerading S , such a modification will fail in Step 3 of the login and key agreement phase, because an attacker has no way of obtaining the value A_i to compute the valid parameter X_B .
- (4) The proposed SEKA protocol provides explicit mutual authentication. Mutual authentication means that both the client and server are authenticated to each other within the same protocol. Juang's protocol does not provide explicit mutual authentication since U_i and S does not confirm whether the shared session key k_j is true. Only S implicit confirms whether the k_j is true. However, the proposed protocol uses the Diffie-Hellman key exchange

algorithm to provide mutual authentication, then the key is explicitly authenticated by a mutual confirmation session key.

- (5) The proposed SEKA protocol provides perfect forward secrecy. Perfect forward secrecy means that if a long-term private key (e.g. user password or server private key) is compromised, this does not compromise any earlier session keys. A compromised long-term secret key x or A_i cannot be used to derive the session keys SK that were used before, since without knowing the used random values a and b , nobody can compute these used session keys SK .
- (6) The proposed SEKA protocol provides secure password change and can fast detect the wrong password. In Step 2 of the proposed password change scheme, U_i 's wrong input password PW_i can be easily detected by the smart card because the card verifies the computed A'_i is equal to the stored A_i in the smart card. Also, in Step 1 of the proposed login and key agreement phase, U_i 's wrong input password PW_i is fast detected by the smart card because the smart card also checks whether $A'_i \stackrel{?}{=} A_i$ holds or not.

6 Computational Costs

The computation costs of the proposed SEKA protocol in registration, pre-computation, login, and key agreement phases are summarized in Table 1.

- (1) Low computational load: The user is required to perform one exponentiation for pre-computation, and one exponentiation and two hash operations during the protocol. On the server side, the computational load is two exponentiation and three hash operations.
- (2) Minimum number of message exchanges: The protocol requires three passes to perform a mutual authentication and key agreement.
- (3) Minimum communication bandwidth use: Among five messages, two are exponentiation bits, two are hash output bits and one is the user's identifier.

The computational costs of Juang's DH-based key agreement protocol [17] and the proposed SEKA protocol in the login and key agreement phase (including pre-computation phase) are summarized in Table 2. In the login and key agreement phase, Juang's protocol requires a total of four exponent operations (to

Table 1. Computation costs of the SEKA protocol

	User	Server
Registration phase	No	1 Hash+1 Xor
Pre-computation phase	1 Exp	No
Login phase	1 Xor	No
Key agreement phase	1 Exp+2 Hash	2 Exp+3 Hash
Password change	2 Xor	No

Exp: Exponentiation operations; Hash: One-way Hash operations;

Xor : Bitwise XOR($\oplus$) operations.

providing the perfect forward secrecy), four symmetric encryption or decryption, three hashing and one exclusive-or operations, but the proposed SEKA protocol requires a total of four exponent operations, five hashing operations and three exclusive-or operations. Unlike Juang’s protocol, the SEKA protocol does not require the symmetric encryption or decryption. The hash functions are faster than the public key computations and symmetric key computations. On a typical workstation, the public key computations can be performed 2 times per second, symmetric key computations can be performed 2,000 times per second and hash function can be performed 20,000 times per second. The exclusive-or operations are very fast than these three operations. Additionally, the SEKA protocol provides explicit key agreement and freely password change. Obviously, the proposed protocol is more efficient and secure than Juang’s protocol.

Table 2. Comparisons of computational costs

	Juang’s protocol	Proposed protocol
Registration phase	1 Hash+1 Xor	1 Hash+1 Xor
Key agreement phase	4 Exp+4 Sym+3 Hash+1 Xor	4 Exp+5 Hash+3 Xor
Password change	No provide	2 Xor

Exp: Exponentiation operations; Sym: Symmetric encryption or decryption;
Hash: One-way Hash operations; Xor : Bitwise XOR($\oplus$) operations.

7 Conclusion

In this paper, we proposed a smart card based secret-key exponential key agreement protocol (SEKA) with more security and less computational cost. The proposed SEKA protocol has the following merits: (1) the protocol needs no verification table; (2) users can freely choose and securely change their own passwords; (3) the communication bandwidth and computational load are very low; (4) users and servers can authenticate each other and it generates a session key agreed by the user and the server; (5) Unlike many timestamp based authentication schemes, the protocol does not have a serious time-synchronization problem.

Acknowledgements

This research was supported by the MIC of Korea, under the ITRC support program supervised by the IITA (IITA-2006-C1090-0603-0026).

References

1. Lamport, L.: Password Authentication with Insecure Communication. Communications of the ACM 24(11), 770–772 (1981)
2. Chang, C.C., Wu, T.C.: Remote Password Authentication with Smart Cards. IEE Proceedings-E 138(3), 165–168 (1991)

3. Chang, C., Hwang, S.: Using Smart Cards to Authenticate Remote Passwords. *Comput. Math. Appl.* 26(7), 19–27 (1993)
4. Wang, S., Chang, T.: Smart Card based Secure Password Authentication Scheme. *Computers & Security* 15(3), 231–237 (1996)
5. Wu, T.C., Sung, H.S.: Authentication Passwords over an Insecure Channel. *Computer & Security* 15(5), 431–439 (1996)
6. Yang, W.H., Shieh, S.P.: Password Authentication Schemes with Smart Card. *Computer & Security* 18(8), 727–733 (1999)
7. Hwang, M.S., Li, L.H.: A New Remote User Authentication Scheme Using Smart Cards. *IEEE Trans. On Consumer Electronics* 46(1), 28–30 (2000)
8. Sun, H.M.: An Efficient Remote User Authentication Scheme Using Smart Cards. *IEEE Trans. on Consumer Electronics* 46(4), 958–961 (2000)
9. Chien, H.Y., Jan, J.K., Tseng, Y.M.: An Efficient and Practical Solution to Remote Authentication: Smart Card. *Computers & Security* 21(4), 372–375 (2002)
10. Fan, L., Li, J.H., Zhu, H.W.: An Enhancement of Timestamp-based Password Authentication Scheme. *Computers & Security* 21(7), 665–667 (2002)
11. Wu, S.T., Chieu, B.C.: A User Friendly Remote Authentication Scheme with Smart Cards. *Computers & Security* 22(6), 547–550 (2003)
12. Shen, J.J., Lin, C.W., Hwang, M.S.: Security Enhancement for the Timestamp-based Password Authentication Scheme Using Smart Cards. *Computers & Security* 22(7), 591–595 (2003)
13. Chen, K.F.: Attacks on the (Enhanced) Yang-Shieh Authentication. *Computers & Security* 22(8), 725–727 (2003)
14. Wu, S.T., Chieu, B.C.: A User Friendly Remote Authentication Scheme with Smart Cards. *Computers & Security* 22(6), 547–550 (2003)
15. Yoon, E.J., Ryu, E.K., Yoo, K.Y.: Security of Shen et al. 's Timestamp-based Password Authentication Scheme. In: Laganà, A., Gavrilova, M., Kumar, V., Mun, Y., Tan, C.J.K., Gervasi, O. (eds.) *ICCSA 2004. LNCS*, vol. 3046, pp. 665–671. Springer, Heidelberg (2004)
16. Yoon, E.J., Ryu, E.K., Yoo, K.Y.: Robust Remote User Authentication Scheme. In: Kahng, H.-K., Goto, S. (eds.) *ICOIN 2004. LNCS*, vol. 3090, pp. 935–942. Springer, Heidelberg (2004)
17. Juang, W.S.: Efficient Password Authenticated Key Agreement Using Smart Cards. *Computers & Security* 23(2), 167–173 (2004)
18. Jablon, D.: Strong Password-only Authenticated Key Exchange. *ACM Computer Communications Review* 26(5), 5–26 (1996)
19. Bellare, S., Merritt, M.: Encrypted Key Exchange: Password-based Protocols Secure Against Dictionary Attacks. In: *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pp. 72–84 (1992)
20. Diffie, W., Hellman, M.: New Directions in Cryptography. *IEEE Trans Inf Theory* IT-22(6), 644–654 (1976)
21. Rivest, R.: The MD5 Message-digest Algorithm. RFC 1321. Internet Activities Board. Internet Privacy Task Force (1992)
22. NIST FIPS PUB 180.: Secure Hash Standard. National Institute of Standards and Technology. U.S. Department of Commerce. DRAFT (1993)

Key Establishment Scheme for Sensor Networks with Low Communication Cost

Yong Ho Kim^{1,*}, Hwaseong Lee¹, Jong Hyuk Park², Laurence T. Yang³,
and Dong Hoon Lee¹

¹ Center for Information Security Technologies (CIST), Korea University, Seoul, Korea
{`optim,hwaseong,donghlee`}@korea.ac.kr

² Hanwha S&C Co., Ltd., Korea
`parkjonghyuk@gmail.com`

³ St Francis Xavier University, Canada
`ltyang@gmail.com`

Abstract. Recently, Huang et al. proposed an efficient authenticated key establishment scheme for self-organizing sensor networks. However, in their scheme, a sensor node and a security manager should exchange public-key certificates to authenticate each other. In this paper, we propose an efficient authenticated key establishment scheme which can reduce the communication cost of transmitting public-key certificates.

1 Introduction

The IEEE 802.15.4 Low-Rate Wireless Personal Area Network Standard specifies the physical layer and medium access control layer of a low data rate, ultra low power and low cost sensor network [10]. It also defines two physical device types, a Full-Functional Device (FFD) and a Reduced-Functional Device (RFD). An FFD takes the role of a security manager, while an RFD takes on the role of an end device, such as a low-power sensor.

To provide secure communication within self-organizing sensor networks, it is essential that a secret key should be securely established between a security manager and an individual sensor. The key may later be used to achieve some cryptographic goals such as confidentiality or data integrity. However, sensor nodes are often exposed to the risk of physical attacks. For instance, adversaries can capture sensor nodes to obtain secret information stored within the nodes' memory.

When we consider the design of key distribution schemes, the simplest method is to embed a single network-wide key in the memory of all nodes before they are deployed. In this case, however, if a single node is compromised, the entire network may be compromised. The opposite extreme is for each node to be

* “ This research was supported by the MIC(Ministry of Information and Communication), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute of Information Technology Advancement)” (IITA-2006-(C1090-0603-0025)).

preloaded with all potential keys for different security managers. This method has perfect resilience against node capture which means that even if a sensor is captured, non-captured sensors are secure. However, due to the high mobility and limited memory space of sensors, it may not be feasible to predict and preload all possible session keys required for communications with all possible security managers. Thus there is a need to identify a more realistic scheme for key establishment.

There has been a common perception that traditional public key infrastructure (PKI) is too complex, slow and power hungry to be used in sensor networks. For this reason, the most research is primarily based on symmetric key cryptography [2,4,5,6,11]. While symmetric mechanisms can achieve low computation overhead, they typically require significant communication overhead or a large amount of memory for each node. For these reasons, many researchers [7,8,14,17,18] have recently begun to challenge those old beliefs about PKI by showing that it is indeed viable in sensor networks.

Among the various flavors of authenticated key establishment (AKE), asymmetric techniques such as certificate-based systems and ID-based systems are commonly used to provide authentication. In a typical PKI deployed system, a user should obtain a certificate of a long-lived public key from the certifying authority and this certificate is given to participants to authenticate the user. Meanwhile in an ID-based system, it is sufficient for participants to know the public identity of the user such as an e-mail address. Thus, unlike certificate-based PKI systems, ID-based authenticated systems do not require the transmission of public-key certificates. In wireless sensor networks, this difference is significant because wireless transmission of a single bit consumes several orders of magnitude more power than a single 32-bit computation [1]. Experimentally, it has also been shown that communication costs are about 97% of energy consumption while computation costs account for just less than 3% [15]. For this reason, a few ID-based AKE schemes have been proposed for sensor networks [19,20,21]. However, the most ID-based AKE systems require low-power sensors to perform expensive computations such as Weil/Tate pairing and Map-To-Point operation. This is unfortunate because in sensor networks, only a small fraction of computations should be performed by low-power sensors. Moreover, despite some attempts to reduce the complexity of pairing, a pairing operation is still several times more costly than a scalar multiplication.

RELATED WORKS. Eschenauer and Gligor presented a random key pre-distribution scheme for pair-wise key establishment [6] in which a key pool is randomly selected from the key space and a key ring, a randomly selected subset from the key pool, is stored in each node before deployment. A common key in two key rings of a pair of neighbor nodes is used as their pair-wise key. This scheme has been subsequently improved by Chan et al. [2], Liu and Ning [11], and Du et al. [4,5]. However, these schemes require a significant pre-communication phase to discover the common key between two neighboring nodes.

Huang et al. [9] proposed two efficient key establishment schemes where a sensor node (RFD) and a security manager (FFD) can achieve key exchange and mutual authentication. These schemes are based on elliptic curve cryptography where each device can authenticate other devices through its certificate [12]. Compared with other public key based schemes, these schemes reduce high cost public-key operations on the sensor side. However, a sensor node and a security manager still must exchange each device's public-key certificates to authenticate each other.

Recently, some ID-based schemes [19,20,21] for sensor networks have been proposed where a sensor need not transmit its implicit certificate. This schemes offer low communication overhead, low memory requirement and perfect resilience against node capture. However, a sensor should still perform expensive computation such as Weil/Tate pairing and Map-To-Point operation.

CONTRIBUTIONS. In this paper, we propose an efficient ID-based scheme for key establishment in self-organizing sensor networks. The proposed scheme was devised after comparing the advantages and disadvantages of certificate-based and ID-based systems. When compared with Huang et al.'s schemes [9], the proposed scheme eliminates the communication overhead required to transmit public-key certificates. Also, a sensor need not perform the Weil/Tate pairing and Map-To-Point operations required in most ID-based schemes [19,20,21].

ORGANIZATION. After reviewing some preliminaries in Section 2, we propose an efficient ID-based scheme for key establishment in Section 3. In Section 4, we analyze the security and the performance of the proposed scheme, and conclude with Section 5.

2 Preliminaries

2.1 Bilinear Map

In this subsection, we review bilinear maps and some assumptions related to the proposed scheme. Let $\mathbb{G}_1$ be a cyclic additive group of prime order q and $\mathbb{G}_2$ be a cyclic multiplicative group of same order q . We assume that the discrete logarithm problems (DLP) in both $\mathbb{G}_1$ and $\mathbb{G}_2$ are intractable. We call $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ an *admissible bilinear* map if it satisfies the following properties:

1. Bilinearity: $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_q^*$.
2. Non-degeneracy: There exists $P \in \mathbb{G}_1$ such that $e(P, P) \neq 1$.
3. Computability: There exists an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in \mathbb{G}_1$.

The modified Weil and Tate pairings in elliptic curve are examples of the admissible bilinear maps.

2.2 Some Problems

Computational Diffie-Hellman (CDH) problem: The CDH problem is to compute abP when given P , aP and bP for some $a, b \in \mathbb{Z}_q^*$.

Modified Inverse Computational Diffie-Hellman (mICDH) problem: The mICDH problem is to compute $(a+b)^{-1}P$ when given b , P , aP and $(a+b)P$ for some $a, b \in \mathbb{Z}_q^*$.

Bilinear Diffie-Hellman (BDH) problem: The BDH problem is to compute $e(P, P)^{abc}$ when given P , aP , bP and cP for some $a, b, c \in \mathbb{Z}_q^*$.

Modified Bilinear Inverse Diffie-Hellman (mBIDH) problem: The mBIDH problem is to compute $e(P, P)^{\frac{1}{a+b}c}$ when given b , P , aP and cP for some $a, b, c \in \mathbb{Z}_q^*$.

The CDH and mICDH problems are polynomial time equivalent, and also the BDH and mBIDH problems are polynomial time equivalent [3].

We assume that above four problems are intractable. That is, there is no polynomial time algorithm solving these problems with non-negligible probability.

3 Proposed Scheme

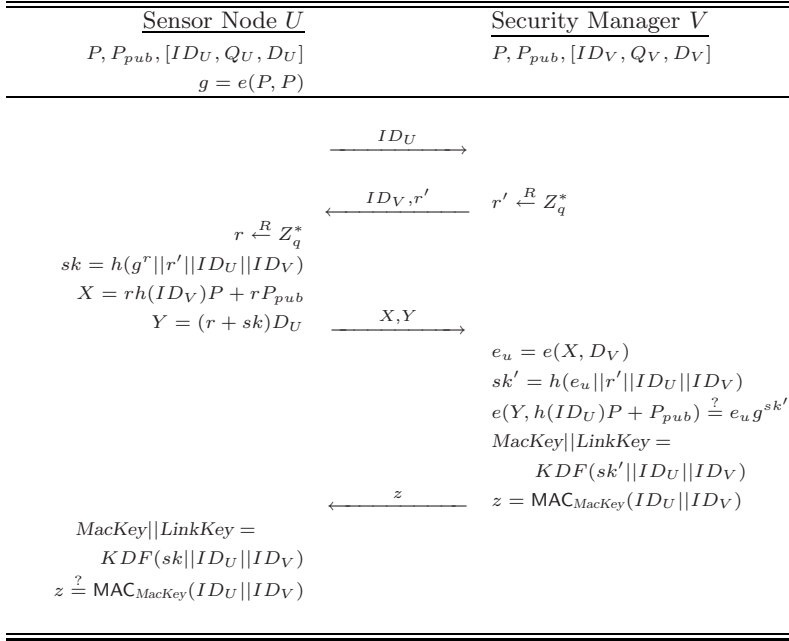
In this section, we propose an identity-based authenticated key establishment scheme for self-organizing sensor networks. Before network deployment, a trusted authority (TA) performs the following operations.

1. TA constructs two groups $\mathbb{G}_1, \mathbb{G}_2$, and a map e as described above.
2. TA chooses a cryptographic hash functions $h : \{0, 1\}^* \rightarrow \mathbb{Z}_q^*$.
3. TA computes $g = e(P, P)$, where P is a random generator of $\mathbb{G}_1$.
4. TA picks a random integer $\kappa \in \mathbb{Z}_q^*$ as the network master secret and sets $P_{pub} = \kappa P$.
5. For each device A (U or V) with identification information ID_A , TA calculates $Q_A = h(ID_A)P + P_{pub}$ and $D_A = (h(ID_A) + \kappa)^{-1}P$.

Next, each device A is preloaded with the public system parameters $(p, q, \mathbb{G}_1, \mathbb{G}_2, e, h, P, P_{pub}, g)$, its identification information ID_A , and its key pair (Q_A, D_A) .

1. After V obtains ID_U , V chooses a random number r' in $\mathbb{Z}_q^*$ and sends ID_V, r' to U .
2. After U obtains ID_V and r' , U chooses a random number r in $\mathbb{Z}_q^*$ and computes $sk = h(g^r || r' || ID_U || ID_V)$. Next, U sends X and Y to V where $X = rh(ID_V)P + rP_{pub}$ and $Y = (r + sk)D_U$.
3. V calculates $e_u = e(X, D_V)$ and $sk' = h(e_u || r' || ID_U || ID_V)$. Once deriving e_u and sk' , it verifies that the following equation holds:

$$e(Y, h(ID_U)P + P_{pub}) = e_u g^{sk'}.$$

**Fig. 1.** Proposed Scheme

The verification works since

$$\begin{aligned}
 e_u &= e(X, D_V) = e(rh(ID_V)P + rP_{pub}, (h(ID_V) + \kappa)^{-1}P) \\
 &= e(rh(ID_V)P + P_{pub}, (h(ID_V) + \kappa)^{-1}P) \\
 &= e(r(h(ID_V) + \kappa)P, (h(ID_V) + \kappa)^{-1}P) \\
 &= e(rP, P) = e(P, P)^r = g^r, \\
 sk' &= h(e_u || r' || ID_U || ID_V) = h(g^r || r' || ID_U || ID_V) = sk
 \end{aligned}$$

and

$$\begin{aligned}
 e(Y, h(ID_U)P + P_{pub}) &= e((r + sk)D_U, (h(ID_U) + \kappa)P) \\
 &= e((r + sk)(h(ID_U) + \kappa)^{-1}P, (h(ID_U) + \kappa)P) \\
 &= e((r + sk)P, P) = e(P, P)^{r+sk} \\
 &= g^{r+sk} = g^r g^{sk} = e_u g^{sk} = e_u g^{sk'}.
 \end{aligned}$$

If the equality holds, the security manager V believes that the sensor node U has the knowledge of its private key, $D_U = (h(ID_U) + \kappa)^{-1}P$. Also, it computes $MacKey || LinkKey = KDF(sk || ID_U || ID_V)$ and sends $z = MAC_{MacKey}(ID_U || ID_V)$ to U , where KDF is the specified key derivation function.

4. After U computes $MacKey || LinkKey = KDF(sk || ID_U || ID_V)$, it verifies $z = MAC_{MacKey}(ID_U || ID_V)$, where MAC is a message authentication code

function. If the equality holds, the sensor node U believes that the security manager V has the knowledge of its private key, $D_V = (h(ID_V) + \kappa)^{-1}P$.

4 Analysis

In this section, we analyze the security and the efficiency of the proposed scheme.

4.1 Security Analysis

Key Confidentiality. In the proposed scheme, the security of secret keys is based on the intractability of the mBIDH problem [3]. After performing the key establishment, an adversary can obtain $h(ID_V)$, P , $P_{pub} = \kappa P$, and $r(h(ID_V) + \kappa)P$. However, she cannot compute $e_u = g^r = e(P, P)^{\frac{1}{\kappa + h(ID_V)} r(h(ID_V) + \kappa)}$ and $sk = sk'$ since there is no polynomial time algorithm solving mBIDH problem with non-negligible.

Key Confirmation. The proposed scheme suitable for wireless sensor networks is a simplified adaptation of the ID-based AKE [3]. The previous scheme provides implicit key authentication if a participant is assured that no other participants except its intended partner can possibly learn the value of a particular secret key. However, the proposed scheme provide explicit key authentication.

Authentication. In the scheme, if $e(Y, h(ID_U)P + P_{pub}) = e_u g^{sk'}$ holds, the security manager V has verified that the sensor node U has the knowledge of sk' and its private key D_V . Also, if $z = \text{MAC}_{MacKey}(ID_U || ID_V)$ holds, the sensor node U has verified that the security manager V has the knowledge of sk and its private key D_U .

4.2 Efficiency Analysis

Unlike ID-based schemes [19,20,21] for sensor networks, a sensor need not perform Map-To-Point operation and Weil/Tate pairing which is several times more

Table 1. Comparison of the proposed scheme and Huang et al.'s schemes

	EC-RP	EC-FP	EXP	CC
Hybrid [9]	1	2	0	1437 bits
MSR-Hybrid[9]	0	3	1	3682 bits
Proposed Scheme	2	2	1	672 bits

Hybrid : Huang et al.'s hybrid authenticated key establishment
MSR-Hybrid : Huang et al.'s MSR-combined Hybrid
EC-RP : elliptic curve scalar multiplication of a random point
EC-FP : elliptic curve scalar multiplication of a fixed point
EXP : small modular exponentiation
CC : communication complexity

costly than a scalar multiplication. For each sensor, we summarize the efficiency of the proposed scheme and Huang et al.'s schemes in Table 1.

When compared to Huang et al.'s schemes [9], the proposed scheme features remarkable communication efficiency since it does not require the transmission of public-key certificates. If we assume the device *ID* is 64 bits, the certificate expiration time and the random number *k* are also 64 bits each, and the modulus for ECC and Rabin cryptosystem are 160 bits and 1024 bits respectively, the total communication cost is 1437 bits (Hybrid) or 3682 bits (MSR-Hybrid) [9]. However, in these assumptions, that of the proposed scheme is only 672 bits.

5 Conclusion

In this paper, we proposed an efficient authenticated key establishment scheme, in which a sensor need not transmit public-key certificates and perform expensive computation such as Weil/Tate pairing and Map-To-Point operation. In this way, the proposed scheme eliminates the major disadvantages of certificate-based schemes [7,9,14] and ID-based schemes [19,20,21].

References

1. Barr, K., Asanovic, K.: Energy aware lossless data compression. In: 1st Int. Conf. Mobile Syst. Applicat. Services, pp. 231–244 (2003)
2. Chan, H., Perrig, A., Song, D.: Random key predistribution schemes for sensor networks. In: IEEE Symposium on Security and Privacy, pp. 197–213 (2003)
3. Choi, K.Y., Hwang, J.Y., Lee, D.H.: ID-based Authenticated Key Agreement for Low-Power Mobile Devices. In: Boyd, C., González Nieto, J.M. (eds.) ACISP 2005. LNCS, vol. 3574, pp. 494–505. Springer, Heidelberg (2005)
4. Du, W., Deng, J., Han, Y. S., Chen, S., Varshney, P.K.: A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge. In: IEEE INFOCOM 04, pp. 586–597 (2004)
5. Du, W., Deng, J., Han, Y. S., Varshney, P.K., Katz, J., Khalili, A.: A Pairwise Key Pre-distribution Scheme for Wireless Sensor Networks. ACM Transactions on Information and System Security, 228–258 (2005)
6. Eschenauer, L., Gligor, V.D.: A key-management scheme for distributed sensor networks. In: ACM CCS 02, pp. 41–47 (2002)
7. Gaubatz, G., Kaps, J., Sunar, B.: Public keys cryptography in sensor networks? revisited. In: Castelluccia, C., Hartenstein, H., Paar, C., Westhoff, D. (eds.) ESAS 2004. LNCS, vol. 3313, pp. 2–18. Springer, Heidelberg (2005)
8. Gura, N., Patel, A., Wander, A., Eberle, H., Shantz, S.C.: Comparing elliptic curve cryptography and RSA on 8-bit CPUS. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 119–132. Springer, Heidelberg (2004)
9. Huang, Q., Cukier, J., Kobayashi, H., Liu, B., Zhang, J.: Fast authenticated key establishment protocols for self-organizing sensor networks. In: ACM WSNA 03, pp. 141–150 (2003)

10. IEEE Std. 802.15.4-2003, IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (WPANS) (2003)
11. Liu, D., Ning, P., Li, R.: Establishing Pairwise Keys in Distributed Sensor Networks. *ACM Transactions on Information and System Security*, 41–77 (2005)
12. Menezes, A.: *Elliptic Curve Public Key Cryptosystems*. Kluwer Academic Publishers, Boston (1993)
13. Mitsunari, S., Sakai, R., Kasahara, M.: A new traitor tracing. *IEICE Trans. E85-A(2)*, 481–484 (2002)
14. Malan, D.J., Welsh, M., Smith, M.D.: A public-key infrastructure for key distribution in tinys based on elliptic curve cryptography. In: *IEEE SECON 04*, pp. 71–80 (2004)
15. Perrig, A., Szewczyk, R., Wen, V., Cullar, D., Tygar, J.D.: SPINS: Security protocols for sensor networks. In: *ACM/IEEE International Conference on Mobile Computing and Networking*, pp. 189–199 (2001)
16. Pointcheval, D., Stern, J.: Security arguments for digital signatures and blind signatures. *J. of Cryptology* 13, 361–396 (2000)
17. Wander, A., Gura, N., Eberle, H., Gupta, V., Chang, S.: Energy analysis for public-key cryptography for wireless sensor networks. In: *IEEE PERCOM 05* (2005)
18. Watro, R., Kong, D., Cuti, S., Gardiner, C., Lynn, C., Kruus, P.: TinyPk: Securing sensor networks with public key technology. In: *ACM SASN 04*, pp. 59–64 (2004)
19. Zhang, Y., Liu, W., Lou, W., Fang, Y.: Securing sensor networks with location-based keys. In: *IEEE WCNC 05*, pp. 1909–1914 (2005)
20. Zhang, Y., Liu, W., Lou, W., Fang, Y.: Location-based compromise-tolerant security mechanisms for wireless sensor networks. *IEEE JSAC, Special Issue on Security in Wireless Ad Hoc Networks* 24(2), 47–260 (2006)
21. Zhang, Y., Liu, W., Lou, W., Fang, Y., Wu, D.: Secure localization and authentication in ultra-wideband sensor networks. *IEEE JSAC, Special Issue on UWB Wireless Communications - Theory and Applications* 24(4), 829–835 (2006)

A Worm Containment Model Based on Neighbor-Alarm^{*}

Jianming Fu^{1,2}, Binglan Chen¹, and Huanguo Zhang¹

¹ School of Computer, Wuhan University, Wuhan 430072, P.R.China

² The State Key Lab of Software Engineering, Wuhan University,
Wuhan 430072, P.R.China

jmfu@whu.edu.cn, zoeeee@tom.com, liss@whu.edu.cn

Abstract. How to detect and contain worms is an open issue as worm becomes a major threat to network security nowadays. Based on the help between neighbors in social network, this paper presents a model to mitigate the rapid spread of worms, and describes its dynamic equation. Since the performance of our model depends on the trust between neighbors, a method to calculate the trust is given in this paper. TPM can protect the authenticity of trust between neighbors, and thus decrease the worm propagation. Experimental results demonstrate that this model can greatly suppress the propagation of worms.

1 Introduction

Nowadays, the threat of worms to the computer security and network security has gradually increased. The diverse ways of worm propagation results in frequent outbreaks, wide spread and heavy losses. For example, there are typical worms, such as Morris in 1988, CodeRed and Nimda in 2001, SQL Slammer and blaster in 2003, MyDoom in 2004, and so on [1]. Worms can spread in the Internet by exploiting the loopholes of systems, software and network protocols. In 2005, 5990 loopholes have been released in CERT. Now potential loopholes are still being dug, and their amount may increase. As a result, the worms exploiting these loopholes will increase as well. Moreover, the combination of worm technology with other technologies, such as computer virus, deformation, polymorphism, distributed collaboration and rootkit, makes it more difficult to detect worms.

The best way to prevent worm is to timely patch systems and applications. Some operation systems can automatically patch themselves, such as the Microsoft Windows. However, general applications need to be patched manually, or even updated to the latest version. If loopholes exist in these applications, there will be zero-day worms utilizing these loopholes in the future. In this kind of situation, early detection including local detection and network detection is

^{*} Supported by the National Natural Science Foundations of China under Grant No.60673071 and No.60633020, and by Hi-Tech Research and Development Foundations of China under Grant No.2006AA01Z442, and by Hubei Natural Science Foundations of China under Grant No.2005AA101C44.

often used to avoid the outbreaks of the worms [2]. Local detection is based on worm signatures collected from concrete worm samples, while network detection is based on abnormal network flows. Deficiencies of these detections are that they take much time and there exist false negatives and false positives.

We hope that the infection of worms can be known as early as possible. Immunization techniques can be used to mitigate the spread of worms. Based on the help between neighbors in social network, this paper presents a worm containment model using neighbor-alarm. Neighbors help and alarm each other in order to detect worms and make themselves immune early. In this way worms can be contained.

This paper is organized as follows. Section 2 states the detection and defense of worm as well as the immunization of worm in related research. Section 3 focuses on the worm containment model based on neighbor-alarm. Section 4 presents results of the simulation to demonstrate its performance. The final section shows our conclusions.

2 Related Work

On one hand, people treat worms as traditional computer virus, study worm signatures, and detect worms from the network traffic. Using generic signature to defend against worm is adopted [1], which is similar to techniques of anti-virus. Earlybird uses a content-sifting approach to detect content prevalence and scaled bitmaps to estimate address dispersion [3]. Autograph automatically generates signatures from worms propagating with TCP traffic, which uses application-level multicast to share port-scan reports among distributed monitors [4]. Nicholas Weaver and his colleagues develop a fast scan-detection and suppression algorithm based on the Threshold Random Walk online malicious-host-detection algorithm [5]. Cliff Zou and his colleagues proposed trend detection system to discover the presence of worm in its early stage by using a Kalman filter estimation algorithm [6]. Microsoft Research's Shield project installs vulnerability-specific and exploit-generic network filters in the end systems once the vulnerability is discovered and before a patch is applied [7]. Cai Min and his colleagues present collaborative Internet worm containment system based on DHT to monitor traffic and to generate signature-worm [8]. D. Whyte and his colleagues use DNS abnormality to detect worm scanning [9]. The authors in [10] state anti-worm to suppress the worm spread. The authors in [11] describe the danger posed by P2P worm, and some immune-groups are provided to suppress the worm propagation in scale-free P2P [12]. But the technology of worm signature can not confront with encrypted or deformed worm.

On the other hand, static immunization techniques [13] [14] are presented to suppress the propagation of worms: random immunization, proportional immunization, targeted immunization, and random acquaintance immunization. With a random immunization strategy, most of the population needs to be immunized before an epidemic occurs. Random acquaintance immunization and proportional immunization are better than random immunization due to probability

selection of immunized nodes. However, targeted immunization that makes the most highly connected nodes immune, gains good performance at the expense of acquiring global knowledge of the network topology.

In order to mitigate the spread of worm effectively, anti-worm is introduced against worm [10]. Anti-worm scans network nodes for loopholes. And upon discovering vulnerable node, it immediately patches the node. Moreover, if this node has been infected in the past, anti-worm will remove the worm from the node. The shortage of anti-worm is that its scanning will incur extra network traffic. And how to distinguish whether a worm is malicious or benign is an open issue. Anti-worm technology is essentially a dynamic immune technology. The model that is presented by this paper is also a dynamic immune technology, but it is a different one since the traffic scan is not incurred and the automatic propagation is avoided. Whether the neighbor will be immune or not depends on trust between neighbors.

3 Worm Containment Based on Neighbor-Alarm

3.1 Our Motivation

IATF (Information Assurance Technical Framework) considers that everyone should be responsible of network security. A user that has detected worm has responsibilities and obligations to help others that he is familiar with. In human society, people have the awareness of group. That is, people are closely linked to different groups in terms of interest, profession and entertainment. People will help each other in the same group. At the same time, there exist different groups among nodes in the Internet. At present, the topology of Internet has small-world and scale-free characteristics. In fact, a node only communicates and interacts with a few other nodes in the same group. Therefore, a small-world community will be built after their communications. When an infected node is detected and then cured, it may send an alarm to its neighbors in its group. If these neighbors believe this alarm, they can check whether they are infected or not, and make themselves immune. Let θ be a variable of trust between neighbors, the value of θ ranges from 0 to 1. If θ equals to 0, neighbors mistrust each other, and will discard any alarm; if θ equals to 1, neighbors absolutely trust each other and the alarm, and will be automatically immunized. Therefore, the possibility of accepting an alarm directly depends on the value of θ .

Our alarm is an active and secure response mechanism. It can be used in a LAN, between different LANs, or in the individual network. Of course, the characteristics of worm, the detection and removal methods, and other information can be released on credible websites, such as bulletin boards. Its purpose is to make neighbors immune as early as possible.

3.2 Status of Node

A node in the system has three status: susceptible S , infected I and immune R . Before a susceptible node is infected, it becomes immune through actively downloading the related patch. If an infected node is detected, it will download the

patch and becomes immune. Immediately after that, the immune node will send an alarm to its neighbors. The neighbors which have received the alarm will then make a decision according to the trust value θ whether to ignore or deal with the alarm. Assuming a neighbor decides to deal with this alarm, if it is infected and has not been detected yet, it may be detected and become immune; if the neighbor is susceptible, it may directly become immune; if the neighbor is already immune, it will ignore the alarm. Status changes are shown in Fig.1.

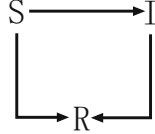


Fig. 1. States of a node. $S \rightarrow I$: infected by worm. $I \rightarrow R$: detected by user, or accepting an alarm. $S \rightarrow R$: downloading the patch, or accepting an alarm

3.3 Dynamic Equation

According to the analysis of worm containment above, the dynamic equations can be given:

$$\begin{cases} \frac{dI(t)}{dt} = \beta I(t)S(t) - \gamma I(t) - \theta \gamma I(t)I(t) \\ \frac{dS(t)}{dt} = -\beta I(t)S(t) - \theta \gamma I(t)S(t) \\ \frac{dR(t)}{dt} = \gamma I(t) + \theta \gamma I(t)S(t) + \theta \gamma I(t)I(t) \end{cases} \quad (1)$$

In the equations, $I(t) + S(t) + R(t) = 1$ at any time, $I(t)$ is the ratio of infected nodes to all the nodes at time t , $S(t)$ is the ratio of susceptible nodes at time t , and $R(t)$ is the ratio of immune nodes at time t . θ is related with the trust between neighbors, β is the infection rate, and γ is the recovery rate. At time t , $\gamma I(t)$ nodes become immune, and at the same time, $\theta I(t) \cdot \gamma I(t)$ nodes also become immune upon receiving alarms from $\gamma I(t)$. Moreover, these alarms will impact on $S(t)$, and $\theta S(t) \cdot \gamma I(t)$ nodes become immune. Thus, $\frac{dI(t)}{dt}$ and $\frac{dS(t)}{dt}$ have considered the effect of neighbor-alarm in (1).

Users can actively download related patches, and make susceptible nodes immune. This case could be considered into (1). But for simplicity, such a case is ignored. When θ equals to 0, the model will degrade to SIR model [1].

In a P2P system, a node only interacts with neighbors. An immune node can send an alarm to its neighbors. And these neighbors could then make themselves immune. Of course, these neighbors may ignore this alarm because of the possible deception in the network. This immune strategy is a neighborhood immunization, that is, the immunization is of one-depth. It effectively mitigates the spread of worm, and avoids the automatic propagation of anti-worm. Thus, it is better than the immunization method in literature [10].

Equation (1) is rewritten via a way of eliminating dt :

$$\frac{dI(t)}{dS(t)} = \frac{\beta I(t)S(t) - \gamma I(t) - \theta \gamma I(t)I(t)}{-\beta I(t)S(t) - \theta \gamma I(t)S(t)} = \frac{1}{\beta + \theta \gamma} \cdot \frac{\gamma + \theta \gamma I(t) - \beta S(t)}{S(t)} \quad (2)$$

If (2) equals to 0, there will exist a balance point in the phase space:

$$1 + \theta I(t) = \frac{\beta}{\gamma} S(t) \quad (3)$$

Given $\frac{\beta}{\gamma}$ and $S(t)$, if θ becomes bigger, $I(t)$ will become smaller. It shows that the prevalence of worm decreases with the growth of θ .

3.4 Trust Between Neighbors

How to assign the value of θ is an open issue, since θ affects the effect of worm containment. Without loss of generality, we just consider the trust T of neighbors as the value of θ . To be general, let T be 0.5 at the initial status. It means that a node will accept its neighbors' alarms with 50% possibility. T may be updated after neighbors' communications. T is updated according to the following equation:

$$T = T \cdot \alpha + (1 - \alpha) \cdot (1 - \delta^n) \quad (4)$$

In (4), α is the learning rate (a real number in the interval $[0,1]$), δ is the index factor, and n is the number of acceptable alarms in the current time window. The larger δ is, the faster the trust increases for a given n . Therefore, T belongs to the range of 0 to 1. When a node receives an alarm from its neighbor and regards this alarm benign, the trust of this neighbor will increase in the light of (4). However, if this alarm is malicious, the trust of this neighbor will become 0, and n will also become 0.

When a node receives an alarm in the time window, it directly checks the trust value of this alarm according to the neighbor's trust value. In other words, the trust of alarms equals to the trust of neighbors. If a node receives the same alarm from k neighbors in the time window, this node needs to make a decision. Assuming trusts of k neighbors are $T_1, T_2, \dots, T_k$ respectively, the trust of this alarm can be computed in the following way:

$$T = 1 - \prod_{i=1}^k (1 - T_i) \quad (5)$$

In order to guarantee the authenticity of an alarm, TPM may be deployed at each node in the network. Each node in the neighborhood can firstly verify each other via remote attestation [15], then the application's authenticity, and finally the alarm's authenticity. In this situation, an alarm is credible after remote attestation, and T is close to 1 for any alarm. Therefore, TPM can strengthen the suppression of worm propagation.

4 Simulation and Analysis

Network models can be divided into two categories: fully connected network and locally connected network. Worms can spread in these networks widely.

Fully connected network means that every two nodes in the network have a logical link. A node in the Internet is identified by its IP address. In the communication process, only the reachability of the nodes, not the specific route path, matters. In terms of the application layer, the Internet can be regarded as a fully connected network. A worm in fully connected network can infect any node in the whole network.

In the locally connected network, a node only communicates with its neighbors. In this network, a worm just infects any node in its neighborhood. Therefore, the worms' spread can be limited in the locally connected network.

In order to analyze the performance of worm containment discussed above, we make simulations on the following two networks. One is the fully connected network, the other one is the locally connected network including the small-world and scale-free network. The parameters of simulated networks are given in Table 1. We randomly select several nodes as the initial infected ones. The experiment will carry on until every node is infected or immune. p is the ratio of the number of infected nodes to all the nodes. The metric p indicates the performance of worm containment technology.

Table 1. Parameters of Network Models

Parameter	Value	Description
N	100000	the number of nodes in the whole network
β	0.005	the infection rate of each node in the network
γ	0.01	the recovery rate of each node in the network
I_0	3	the number of the initial infected nodes
k	3	the average degree of each node in scale-free network
d	0.01	the disconnection probability of each node in small-world network

4.1 General Trust

In the entire simulation, T keeps unchanged. The experimental results are shown in Fig.2, Fig.3, Fig.4 and Fig.5. We can obtain the following observations from these figures: x (1) In Fig.2, Fig.3 and Fig.4, regardless of the specific model, the greater trust value T is, the smaller the proportion of infection is. That is because when T is bigger, the alarms will be adopted more often, and the suppression effect will be better. In addition, the peak of p is exponential to T as shown in Fig.5.

(2) Given the same value of T , the fully connected network has the largest p , whereas the small-world network has the smallest p . That is because in the fully connected network, the average number of neighboring nodes is the largest; while in small-world network, the average number of neighboring nodes is the smallest.

(3) While T is different, the moments at which the infections peaks is almost the same in the fully connected network and scale-free network. But in the small-world network, when T is bigger, the peak moment comes earlier.

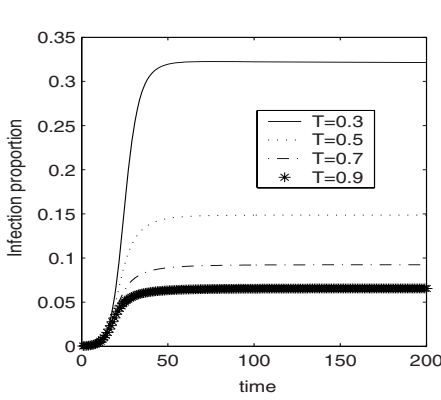


Fig. 2. Infection proportion in fully connected network

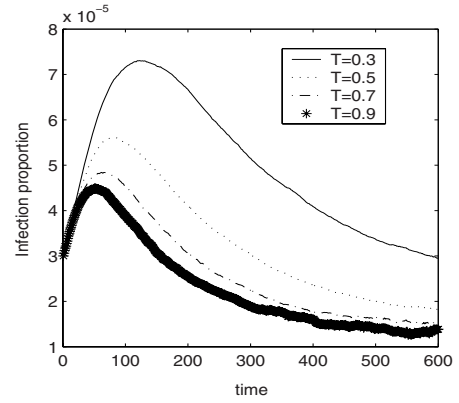


Fig. 3. Infection proportion in small-world network

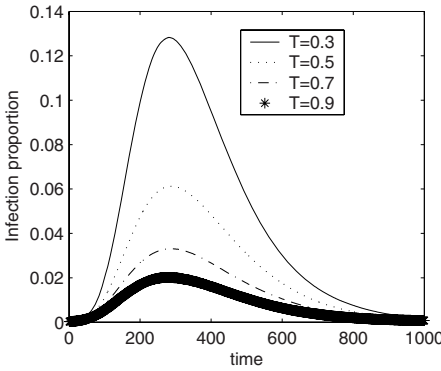


Fig. 4. Infection proportion in scale-free network

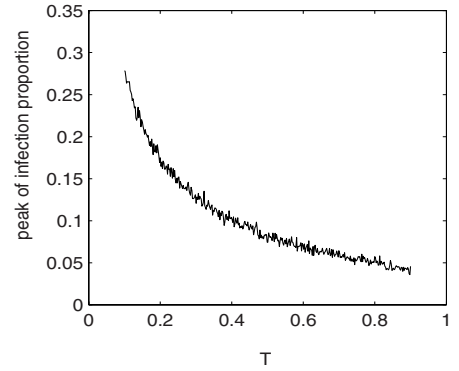


Fig. 5. Peak of Infection proportion in scale-free network

4.2 Normal Trust

In the actual network, the trust in the neighborhood may be changeful. In our simulation, we assume T obeys normal distribution. $T(0.5, 0.5)$ denotes that the mean of trust is 0.5 and the standard deviation is 0.5. The parameters of simulation in scale-free network are the same as those in Fig.4 except the value T .

Fig.6 shows infection proportion for the same standard deviation and different means, and Fig.7 gives infection proportion for the same mean and different

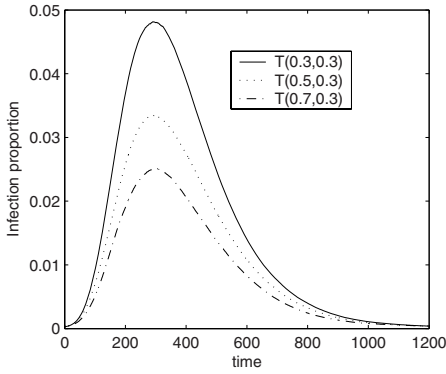


Fig. 6. Infection proportion with $T(0.3\sim 0.7,0.3)$

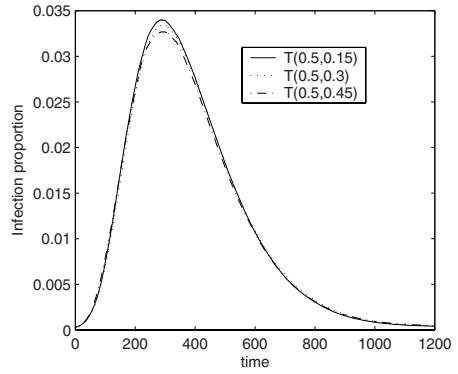


Fig. 7. Infection proportion with $T(0.5,0.15\sim 0.45)$

standard deviations. In Fig.6, different means obviously affect the infection proportion. From Fig.7, we can find that while the mean value of T is same, the proportion of infection does not change significantly. But while the standard deviation is smaller, the peak value of the proportion of infection is bigger.

5 Conclusion

Based on the help between neighbors in social network, we present a worm containment model based on neighbor-alarm. The performance of this model depends on the trust between neighbor nodes. We give some methods of getting and updating trust. These methods may conform to the actual situation. Experimental results demonstrate that the model can obviously suppress the propagation of worm.

The model requires that an immune node must send an alarm to its neighbors. If this node does not send alarms, the model will lose its suppression function of worm propagation. Therefore, we will provide some incentive mechanisms to encourage nodes to voluntarily send alarms in the future. In addition, other mechanisms will be presented to identify malicious nodes, since that these nodes may send false or malicious alarms, and that these alarms may produce DoS(Denial of Service) attacks. It is our future works to study these mechanisms.

References

1. Nachenberg, C.: From AntiVirus to AntiWorm: A New Strategy for A New Threat Landscape[R]. In: Proceedings of ACM Workshop on Rapid Malcode WORM 2004, USA (2004)
2. Zou, C.C., Gao, L., Gong, W., Towsley, D.: Monitoring and early warning for Internet worms. Technical Report, TR-CSE-03-01, Electrical and Computer Engineering Department, University of Massachusetts (2003)

3. Singh, S., et al.: Automated Worm Fingerprinting. In: Proceedings of Usenix Symp. Operating System Design and Implementation, Usenix Assoc. pp. 45–60 (2004)
4. Kim, H.A., Karp, B.: Autograph: Toward Automated Distributed Worm Signature Detection. In: Proceedings of Usenix Security Symp., Usenix Assoc. pp. 271–286 (2004)
5. Cai, M., Hwang, K., et al.: Fast Internet Worm Containment. IEEE Security and Privacy (2005)
6. Zou, C.C., et al.: Monitoring and Early Warning for Internet Worms. In: Proceedings of 10th ACM Conf. Computer and Comm. Security CCS 03, pp. 190–199. ACM Press, New York (2003)
7. Wang, H.J., et al.: Shield: Vulnerability-Driven Network Filters for Preventing Known Vulnerability Exploits. In: Proceedings of ACM SIGCOMM, ACM Press, New York (2004)
8. Sandhu, R., Xinwen, Z.: Peer-to-Peer Access Control Architecture Using Trusted Computing Technology. In: Proceedings of SACMAT05, Stockholm, Sweden (2005)
9. Whyte, D., Kranakis, E., van Oorschot, P.: DNS based detection of scanning worms in an enterprise network. In: Proceedings of the 12th Annual Network and Distributed System Security Symposium (2005)
10. Feng, Y., Haixin, D., Xing, L.: Modeling and analyzing interaction between worm and antiworm in network worm spread. SCIENCE IN CHINA SERIES E 34(8), 841–856 (2004)
11. Lidong, Z., Lintao, Z., Frank, M., Nicole, I., Manuel, C., Steve, C.: A first look at Peer-to-Peer Worms: Threats and Defense. In: Proceedings of the Peer-to-Peer Systems 4th International Workshop. Ithaca, NY, USA, pp. 24–35 (2005)
12. Jianming, F., Zhiyi, H., Binglan, C., Jingsong, C.: Containing Worm Based on Immune-group in Scale-free P2P. In: Proceedings of the First International Conference on Complex Systems and Applications, Huhhot, China, pp. 945–949 (2006)
13. Pastor Satorras, R., Vespignani, A.: Immunization of complex networks. Phys. Rev. E (2002)
14. Reuven, C., Shlomo, H., Danie, B.A.: Efficient Immunization Strategies for Computer Networks and Populations. Phys. Rev. Lett. (2003)
15. Weaver, N., Staniford, S., Paxson, V.: Very Fast Containment of Scanning Worms, In: Proceedings of 13th Usenix Security Symp., Usenix Assoc. pp. 29–44 (2004)

A Distributed Self-healing Data Store

Wolfgang Trumler, Jörg Ehrig, Andreas Pietzowski, Benjamin Satzger,
and Theo Ungerer

Institute of Computer Science
University of Augsburg
86195 Augsburg, Germany
{Trumler,Ungerer,Pietzowski,Satzger}@informatik.uni-augsburg.de

Abstract. Due to the huge amount of integrated devices and sensors in everyday objects ubiquitous systems are in vicinity and will be deployed in large scales in the near future. We expect these system to be unreliable as nodes may crash or vanish from time to time. Therefore a reliable data store is needed to offer application developers a secure place to store the data of the services. The data store itself is subject to the same unreliable infrastructure thus it must expose self-healing capabilities to overcome data loss due to node failures.

In this paper we propose a distributed self-healing data store for ubiquitous systems that guarantees the availability of the stored data even if there is a node failure every 36 seconds in a system consisting of 100 nodes. We also monitor the availability of the nodes to improve the way the data of the data store is distributed in the system.

1 Introduction

The rise of ubiquitous systems demands new approaches for the development of applications. They should no longer be monolithic software blocks, but a composition of services acting together on different devices of the networked nodes. Even current distributed systems are rather static in terms of service deployment. A common way is to distribute the components/services of an application once and to keep up this status-quo as long as possible.

Ubiquitous environments consist of a diverse congregation of devices with varying capabilities in terms of available resources. A huge amount of sensors will be available to monitor the environment and to assist more powerful devices up to PCs or Servers to facilitate the users' tasks. Not to build such systems from scratch middleware systems for ubiquitous environments like PCOM/Base [1], GaiaOS [2] and AMUN/OC μ [3,4] are needed to foster the development of ubiquitous applications. Especially OC μ incorporates the capability to relocate services from one node to another during runtime to implement self-configuration and self-optimization. The OC μ middleware for smart office environments is the target of the proposed data store.

Beside the architectural and structural changes, which are relevant for application designers, the nodes of ubiquitous systems are expected to be unreliable

and to appear and disappear suddenly. Keeping this assumption in mind services need to store their relevant information to a secure place to get it back in case of a crash or a vanishing node. To absolve the developers from building such capabilities for every application the middleware should offer a data store where services can easily store their information and get it back even in case of failures.

The proposed data store itself is subject to the same conditions as the application services, thus it must expose self-healing features to guarantee the availability of the stored data. To build a reliable data store exposing self-healing capabilities additional effort must be spent to secure the data. The data store must replicate the information to additional nodes to overcome node failures and disappearing nodes. Therefore additional memory must be used on other nodes and the data must be transferred to these nodes resulting in an additional consumption of free communication bandwidth. These two points will be considered in the design of the data store as well as in the evaluations.

The remainder of this paper is structured as follows. Section 2 describes the data store, its components, algorithms and metrics used to build the self-healing data store. Evaluations are given in section 3 and related work is presented in section 4. The paper closes with a conclusion and future work in section 5.

2 The Data Store

The main target of the data store is to offer a safe place for other application services to store their information. The effort to use the data store should be as low as possible and the distribution of the data must be transparent to the services. This means that a service that wants to read data from or write data to the data store must not worry about the physical location of the stored data. The data might either be stored locally or on a remote node resulting in longer access times. There are many possibilities to build the interface of the data store but there are only two appropriate patterns known from other systems where information can be stored.

The first one would be to use the paradigm of a file system. A service can open files for read or write operations and the information is sequentially read from or written into the file. The advantage as well as the disadvantage of a file system is its locality. If a file is stored on the server where the corresponding service is running on the access can be granted with maximum speed. On the other hand much effort is needed to create a reliable distributed file system suitable for ubiquitous systems.

The second approach is to use a database. Large databases are known to perfectly handle information stored on multiple servers. The way information is distributed on the servers depends on the relational structure of the databases. This structure allows an easy partitioning of the data on different physical nodes. The disadvantage of this approach is that the data must be separated to fit into the tables. The structure of the tables are fixed after creation and cannot be changed easily. Databases use transactions to guarantee a consistent view of the stored information. The access to the information must be expressed in an SQL statement, which is often hard to express especially for complex queries. Because

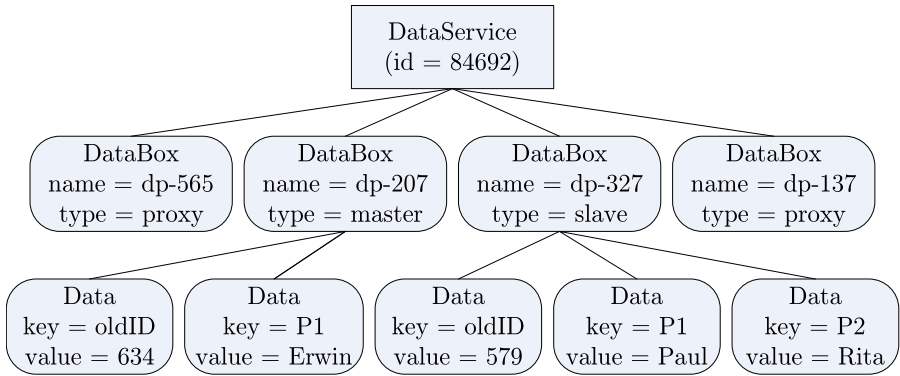


Fig. 1. Example of a data store structure

of the explicit *transaction begin* and *transaction end* statements a database knows when information must be stored persistently.

Concerning the interface of the data store a combination of the former two approaches would be the best. Using the simple access pattern of a file system in combination with the transactions of databases offers the possibility to access the information as easy as writing into a file and to guarantee a consistent view of the stored data. Furthermore the data store can handle the distribution of the information after a transaction is completed. The data store should also handle the access to remote information transparently. This means that the access to the stored information always seems to be local in terms of the application services.

2.1 Architecture

The structure of the data store is demonstrated in figure 1. The main instance of the data store is the *DataService*. An instance of a *DataService* is running on every node of the system, thus all other application services have local access to the stored information even if the information is not stored on the same node. The *DataService* is responsible to find the information within the network.

The *DataService* is the main instance that handles multiple *DataBox* objects. All information is stored in *DataBoxes*. *DataBoxes* can be considered like files in a file system. If a service wants to read or write information it has to give the name of a *DataBox* where the information will be read or written. Beside the name a *DataBox* has a type describing its status. The meaning of the states are described in further detail in the next section.

The information inside a *DataBox* is stored in a hash table. The advantage to the hash table based approach is that the information can be accessed by keys and not by positions, which is much easier to handle and less error prone. Furthermore the information can be extended without restructuring the whole *DataBox*.

2.2 Masters, Slaves and Proxies

A *DataBox* object can have one of following three types:

Master: A master is the main *DataBox* object where all read operations are coordinated and all write operations are performed. Therefore the name of the *DataBox* must be specified in a transaction. If a service from another node wants to read or write data, the *DataStore* of the remote node will use either a proxy or a slave *DataBox* object to contact the master.

Slave: Slave *DataBoxes* are used to perform read operations, to add redundancy to the data store, to overcome node failures, and to establish the self-healing features of the system. The amount of slaves per master is a measurement for the reliability of the data store. The information stored on the master is updated on the slaves after a transaction is finished. This process is described in more detail in section 2.3.

Proxy: If a service wants to read data from or write data into a *DataBox* on a node where neither the master nor a slave is present the *DataService* creates a proxy to handle the request. New *DataBox* objects are always created first as proxies. If no master or slave can be found, the *DataBox* changes its type to become a master. The same applies for the creation of new slaves.

As already mentioned a *DataBox* master object is the central point to perform all write operations. The slaves are used for read operations, to improve the reliability of the data store, and to implement the self-healing feature. Proxies are used to access information stored on remote nodes. Thus the state is important for the actions a *DataBox* must perform. Masters have to update the information of the slaves and to ensure that enough slaves are available at any time to guarantee the desired reliability. Slaves have to elect a new master if the master of a *DataBox* is vanished.

If a service wants to read or write any data it has to specify the name of the *DataBox*. The *DataService* first checks whether a *DataBox* is locally available with the given name. If no *DataBox* object is available the *DataService* has to find the corresponding *DataBox* object. Therefore it creates a proxy *DataBox*, which tries to find the master to perform the operation. If the master can be found the proxy will remain in its state. If the master is not found but a slave answers the request, the proxy can try to contact the master given in the response of the slave. A request for a master is always answered by the slaves of the master to reduce the impact of message loss in the network. If no answer is given to the proxy's request it assumes that no *DataBox* exists and switches to the master state.

If a slave *DataBox* object is available locally, the slave tries to connect its master to check if the local data are outdated. If the local information is up-to-date it can answer the request with the locally available data. Otherwise, if the data is outdated the master sends an update to the slave to renew the slave's data. Afterwards the slave can return the data in response to the former request. If the master can not be found, the slaves first have to elect a new master to

guarantee the actuality of the stored data. This is one part of the self-healing described in further detail in section 2.5.

The master can answer all requests without any further delay because it always holds the latest version of the stored data. A master is created on the local node if a *DataBox* is accessed for the first time and no other master or slave is on the network. A master does not degrade to a slave or a proxy any time.

2.3 Incremental Slave Update

A crucial point in the data store is the reliability of the stored information. To guarantee a specific level of reliability the amount of slaves per master can be defined. The information of the master is replicated to the slaves like a backup. The slaves are distributed over the network and the master ensures that two slaves of the same master are never created on the same node and that no slave is on the same node as the master itself. Both cases would degrade the reliability of the system.

The master is responsible for the replication of the stored information. If a transaction ended and some data were changed, the master has to update the data on its slaves. The updated information is transferred to the slaves via messages, which are also subject to message loss as described in the former section. On the other hand we would like to avoid the overhead of additional acknowledge messages if a master has sent an update message to its slaves.

Therefore every master and slave *DataBox* have a version number identifying the version of the stored information. A master increases its version number by one after every transaction containing a write operation. After a transaction with a write operation ended, the master sends an update message to the slaves with the changed data and the version number.

2.4 Metrics for Slave Distribution

The distribution of the slaves is another important criterion for the reliability of the data store. The less a master or a slave disappears the higher the reliability of the system and the less effort has to be spent on the self-healing. Thus a lot can be done in choosing the best nodes of the network to place the slaves on.

The slaves consume additional memory to store the data. So one crucial parameter is the amount of available memory on the nodes. To find the best node for a new slave the memory consumptions of the different nodes are collected and compared against each other. In the target middleware OC μ [3] the monitors can add additional information on the outgoing messages and the incoming monitors are able to extract this information for further processing. This mechanism can be used to distribute the information about the available memory of the nodes.

Simple Rating. To find the best node for a slave the nodes are rated according to their free memory. With the former described mechanism every node collects the information about the free memory of the other nodes. The free memory is used to calculate a rating for every node. The rating of a node is calculated from

the fraction of the available memory on the node divided by the maximum of the available memory.

Simulations showed the drawback of this rating i.e. all masters will place their slaves on nearly the same best nodes because all masters calculate the same ratings especially if some nodes have much more free memory than the other nodes.

Randomized Rating. To avoid the former problem of slave accumulation and to better distribute the slaves on the nodes of the network the rating should not only be based on the free memory of the nodes. Adding some randomness to the calculation of the ratings allows a better distribution of the slaves among the best nodes. To the value of the simple rating a random value is added such that the former value makes $2/3$ of the value and the random value $1/3$.

Depending on the size of the random value the randomized rating fosters or avoids the accumulation of slaves on a few nodes. The higher the value the more likely it is that slaves of different masters are created on the same nodes.

Proactive Rating. The rating can be further improved if some environmental information is taken into account. To gain a more reliable system a crucial point would be to know when a node is likely to vanish or to crash. A crash is hardly to predict but in environments like a smart office where the workstations are also used for the data store of the ubiquitous system the online periods of the workstations can be monitored and the remaining online time of a node can be predicted.

This additional information can be used to adopt the rating of the nodes depending on the remaining online time of a node (e.g. workstation, server etc.). The value of the predicted online time of a node is added to the part of the simple rating with an additional discount factor. Depending on the value of the discount factor the proactive rating favors either nodes with much free memory or node, which have a longer remaining online time. The same random value like in the randomized rating is used to avoid accumulations of slaves.

2.5 Self-healing

The self-healing part of the data store has to do with the already mentioned nature of ubiquitous systems that nodes of the system are unreliable, may crash or disappear from time to time. The data store uses the nodes of the network to replicate the data written to the master on other nodes. This adds redundancy to the data, which is the only useful way to overcome the mentioned problems without a persistent storage for the data.

The master is the central point for a *DataBox* to perform write operations. This puts the master in the position to be a single point of failure. If a node with the master disappears no further write operation can be performed. A missing master is detected when a service wants to write data to a *DataBox*. In this case the either the slave or the proxy initiates a master election sending his current version number of the data. The slave with the latest version number is elected to become the new master.

If a slave disappeared and the node was offline for just a short time, the a new slave will be restarted or updated with the next incoming update message of the master. If the node was just cut off from the network but the node itself is still running the *DataBox* might have missed some update messages, thus it request the latest information from the master.

3 Evaluation

To show that the proposed reliable self-healing data store is able to accomplish the promised features a simulator was implemented based on the assumption that the data store will be integrated into the $OC\mu$ middleware. We evaluated the improvement of the incremental slave update, the memory usage with the ratings and the error rate with different amount of slaves. The error rate of the data store is the most crucial one because the main target was to build a reliable data store.

The simulator uses special services (*LoadStoreService*), which read data from and write data to the data store through the *DataService*. The services check the results of a read operation against the last written data. If they are not equal a read error occurred. The services write small and large size data packets to the data store within different intervals.

3.1 Memory Consumption

The data store uses additional memory on the nodes of the network to store the information. Simulations with different setups of the nodes show how the memory is consumed on the nodes. Figure 2 shows the standard deviation of the memory consumption of the nodes with simple and with randomized rating. The lower the standard deviation the better is the distribution of the slaves over the network, which avoids the accumulation of slaves on a few nodes.

For simulation shown in figure 2 we assumed 50% of the nodes to have a 10 MBit network connection and 10 MB of memory. The simulation results show that especially in the second simulation the rating performs very well for a setup with varying devices in terms of resources.

3.2 Proactive Rating Versus Randomized Rating

In section 2.4 the randomized and the proactive ratings were introduced to improve the distribution of the slaves on the nodes. The randomized rating chooses a node depending on the free memory with some randomness. The proactive rating tries to further employ the remaining online time of a node. Figure 3 shows the standard deviation of the memory consumption for the randomized rating and for different values of r for the proactive rating. As expected the standard deviation of memory consumption is worse with the proactive rating because of the influence of the remaining online time of the nodes. It can be observed that for a value of 0.75 for r the memory consumptions converge for both ratings.

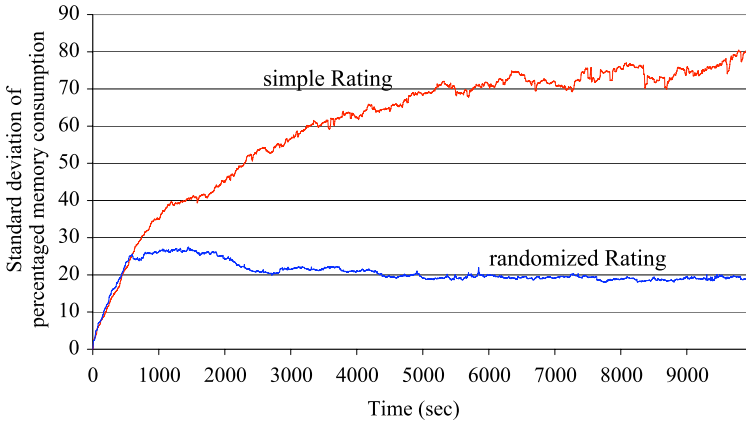


Fig. 2. Standard deviation of the memory consumption in a ubiquitous system

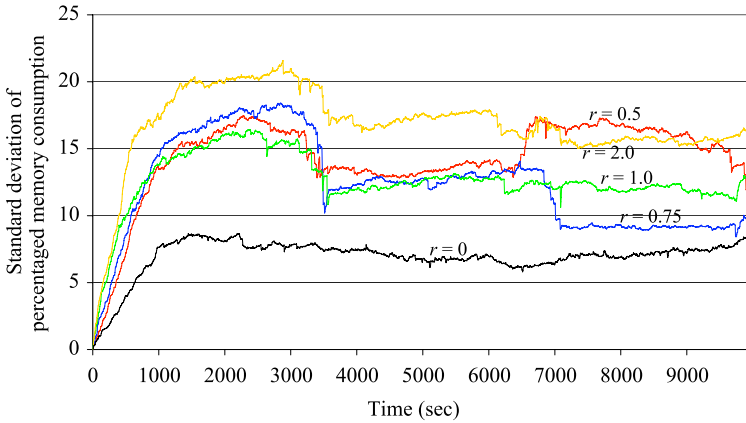


Fig. 3. Standard deviation of the memory for different values of r

3.3 Reliability

The most important point about the data store is the reliability and safety of the stored information. Therefore the error rate at read operations was measured for simulations with different amounts of slaves. An error occurs every time a service wants to read information from the data store and gets outdated information. This can happen if the master disappeared before at least one of the slaves receives the update message. Figure 4 shows the error rates with up to three slaves.

The simulation results show that with additional slaves it is possible to build a reliable data store even in an environment with high failure rates. Starting from two slaves no error occurs as long as the crash rate is higher than 25 seconds.

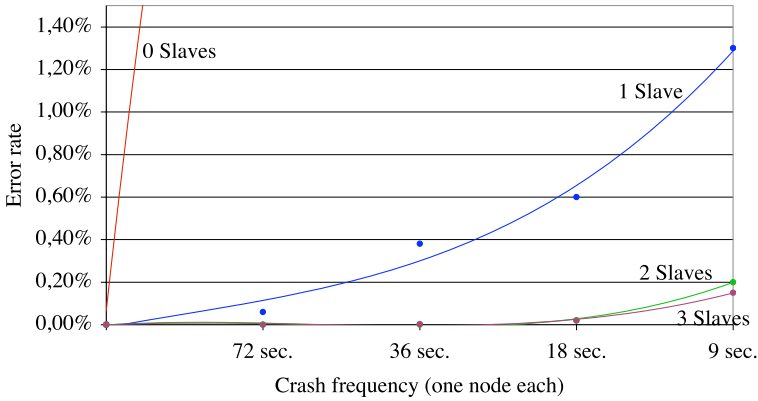


Fig. 4. Error rates with different amount of slaves

This means that every 25 seconds one nodes crashes or vanishes. The default for the simulation was 36 seconds, which means that every node of a network with 100 nodes crashes after an hour. Furthermore a network error of 0.5% is assumed, which means that one out of 200 messages is lost on the network. Assuming that the crash rate of the devices in a ubiquitous environment will be less than expected for the simulations we gain a reliable self-healing data store.

4 Related Work

Currently there are no distributed data stores for ubiquitous systems but there are other distributed systems where information can be stored. Most of them are derived from the idea of a shared memory. Distributed Hashtables like Chord [5] or CAN [6] are the successor of the former shared memory. They are able to store information from any node into the Hashtables and some of them also balance the memory consumption and add robustness [7]. But none of them guarantees reliability with the huge amount of node crashes assumed in the former sections neither do they offer a self-healing feature. Normally they provide a best effort approach suitable for internet-based applications and not for ubiquitous systems.

Databases and RAID systems are known to handle data recovery after crashes. Both are not suitable for ubiquitous systems. Databases are built to handle frequent crashes because the effort of the roll-forward mechanism with the stored log files is very high. RAID systems have the disadvantage that they normally use a few number of nodes compared to the size of a ubiquitous system where all write operations are performed and coordinated. If these nodes vanish, the data is no longer available.

5 Conclusions and Future Work

This paper presents a reliable and self-healing data store for middleware systems like OCμ targeting smart offices. The data store is distributed across the nodes of

the network and all application services can store information locally. The data store handles the storage of the information transparently even if the master for a specific *DataBox* is on a remote node.

The introduced ratings manage the creation of slaves such that the information is distributed to the nodes in terms of memory usage and available online time of the nodes. The self-healing feature guarantees that every application service gets the requested information even if a node crashes or vanishes every 36 seconds. Simulations showed that the additional communication overhead is acceptable related to the offered reliability.

Future work will be to evaluate the data store in a real setup and to further improve the ratings with additional parameters. To get better prediction results for the online times of the nodes we will employ a context prediction mechanism.

Furthermore masters are never relocated even with the proactive rating. Introducing a master relocation might further improve the reliability and reduce the network traffic avoiding the overhead if a master disappears.

References

1. Becker, C., Handte, M., Schiele, G., Rothermel, K.: PCOM - A Component System for Pervasive Computing. In: 2nd IEEE International Conference on Pervasive Computing and Communication (PerCom 04), Orlando, USA (2004)
2. Román, M., Hess, C.K., Ranganathan, A., Madhavarapu, P., Borthakur, B., Viswanathan, P., Cerqueira, R., Campbell, R.H., Mickunas, M.D.: GaiaOS: An Infrastructure for Active Spaces. Technical report, Department of Computer Science, University of Illinois at Urbana-Champaign (2001)
3. Trumler, W.: Organic Ubiquitous Middleware. PhD thesis, Universität Augsburg (2006)
4. Trumler, W., Bagci, F., Petzold, J., Ungerer, T.: AMUN - autonomic middleware for ubiquitous environments applied to the smart doorplate. *ELSEVIER Advanced Engineering Informatics* 19, 243–252 (2005)
5. Stoica, I., an David Kargerand, R.M., Kaashoek, M.F., Balakrishnan, H.: Chord: A scalable peer-to-peer lookup service for internet applications. In: *ACM SIGCOMM 2001*. San Diego, CA, pp. 149–160 (2001)
6. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Schenker, S.: A scalable content-addressable network. In: *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, October 2001, vol. 31, pp. 161–172. ACM Press, New York (2001)
7. Cates, J.: Robust and efficient data management for a distributed hash table. Master's thesis, Massachusetts Institute of Technology (2003)
8. Ehrig, J.: Selbstheilung in einem verteilten dienstbasierten Netzwerk. Master's thesis, Universität Augsburg (2006)

Malicious Codes Detection Based on Ensemble Learning

Boyun Zhang^{1,2}, Jianping Yin¹, Jingbo Hao¹, Dingxing Zhang¹, and Shulin Wang¹

¹ School of Computer Science, National University of Defense Technology,
Changsha 410073, P.R. China
hnjxzby@yahoo.com.cn

² Department of Computer Science, Hunan Public Security College,
Changsha 410138, P.R. China

Abstract. As malicious codes become more complex and sophisticated, the scanning detection method is no longer able to detect various forms of viruses effectively. In this paper, we explore solutions based on multiple classifiers fusion and not strictly dependent on certain malicious code. Motivated by the standard signature-based technique for detecting viruses, we explore the idea of automatically detecting malicious code using the n-gram analysis. After selecting features based on information gain, the probabilistic neural network is used in the process of building and testing the proposed multi-classifiers system. Each one of the individual classifiers is used to produce classification evidences. Then these evidences are combined by the Dempster-Shafer combination rules to form the final classification results for new malicious code. Experimental results produced by the proposed detection engine shows improvement compared to the classification results produced by the individual classifiers.

1 Introduction

Since the appearance of the first computer virus in 1986, a significant number of new viruses have appeared every year. This number is growing and it threatens to outpace the manual effort by anti-virus experts in designing solutions for detecting them and removing them from the system [1]. Though organizations have a wide variety of protection mechanisms (firewalls, antivirus tools, and intrusion detection systems) against cyber attacks, recent hybrid, and blended malware like Sasser, Blaster, Slammer, Nimda and CodeRed worked their way past the current security mechanisms. Since the number and intensity of malware attacks is on the rise, computer security companies, researchers, and users are hard-pressed to find new services to help thwart or defend against such assaults.

Excellent technology exists for detecting known viruses. Programs such as Norton and MacAfee's Antivirus are ubiquitous. These programs search executable code for known patterns. One drawback of this method is that we must obtain a copy of a malicious program before extracting the pattern necessary for its detection.

Then there have been few attempts to use data mining for the purpose of identifying new or unknown malicious code. In an early attempt, Lo et al. [2] conducted an analysis of several programs evidently by hand and identified tell-tale signs, which they

subsequently used to filter new programs. Researchers at IBM's T.J.Watson Research Center have investigated neural networks for virus's detection and have incorporated a similar approach for detecting boot-sector viruses into IBM's Anti-Virus software [3]. More recently, instead of focusing on boot-sector viruses, Kolter et al [4] used data mining methods, such as Naive Bayes, to detect malicious codes. There are other methods of guarding against malicious codes, such as object reconciliation, which involves comparing current files and directories to past copies. One can also compare cryptographic hashes. These approaches are not based on data mining.

In this paper, we propose a method to detect unknown malicious codes through multiple classifiers. The static code of malicious codes was analyzed by the detection engine to identify whether a program is malicious or benign. The classification system currently detects unknown malicious codes without removing any obfuscation.

In the following section, we first illustrate the procedure of the detection model. Then, the details of the method of extraction feature from Win32 PE format program are introduced. Following the describing of D-S Bagging algorithm, the multiple classifications combining method based on D-S theory is stated. Section 3 shows the experiment results. We state our conclusion and future work in Section 4.

2 Malicious Codes Detection Engine

2.1 Model Frame

An ensemble neural network is a learning paradigm where a collection of a finite number of neural networks is trained for the same task. It originates from Hansen and Salomon's work [5], which shows that the generalization ability of a neural network system can be significantly improved through an ensemble of neural networks. As an extension to the previous efforts, the objective of this study is to develop ensembles of ANNs for malicious codes detection.

In the present method, we apply probabilistic neural network (PNN) to construct individual classifier for detection. The detection engine uses the static feature of program, n-gram, to represent and recognize pattern. Finally the D-S theory of evidence is used to combine the contribution of each individual classifier to make a final decision. The detailed steps of detection engine are shown in Figure 1.

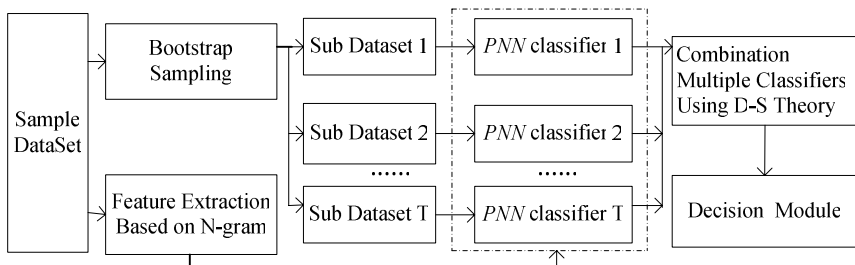


Fig. 1. Malicious Codes Detection Procedure

2.2 Feature Extraction

An n -gram [6] is a subsequence of n consecutive tokens in a stream of tokens. N-gram analysis has been applied in many tasks, and is well understood and efficient to implement. By converting a string of data to n -grams, it can be embedded in a vector space to efficiently compare two or more streams of data. Alternatively, we may compare the distributions of n -grams contained in a set of data to determine how consistent some new data may be with the set of data in question. An n -gram distribution is computed by sliding a fixed size window through the set of data and counting the number of occurrences of each “gram”.

Figure 2 displays an example of a 2-byte window sliding right one byte at a time to generate each 2-gram. Each 2-gram is displayed in the highlighted “window”. The choice of the window size depends on the application. In this work, we focus initially on 2-gram analysis of binary values of PE format file. After the preliminary processing, the frequency matrix of data set is obtained. An example of frequency matrix is shown in table 1.

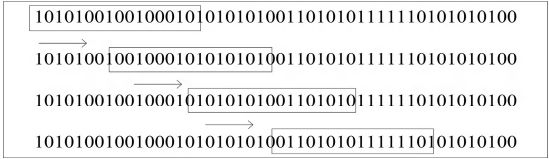


Fig. 2. Sliding window (window size=2 Byte)

Table 1. Feature Frequency Matrix ($n=2$)

Samples	Frequency of Features					
	0080	61C3	638D	9090	E020	...
Win32.Alcaul.a	2	4	21	8	79	
Win32.Alcaul.b	45	6	40	7	20	
Win32.Alcaul.c	11	11	18	8	14	
Win32.Alcaul.e	7	0	12	6	25	
Win32.Alcaul.f	9	20	15	5	27	...
Win32.Alcaul.g	11	0	17	3	20	
Win32.Alcaul.h	19	17	48	4	29	
...	...	...	...	...	...	

For feature selection in our approach we adopt correlation measures based on the information theoretical concept of entropy, a measure of the uncertainty of a random variable. The distinguishing power of each feature is derived by computing its *information gain* (IG) based on its frequencies of appearances in the malicious class and benign class. Features with negligible information gains can then be removed to reduce the number of features and speed the classification process.

The entropy of a variable X is defined as:

$$H(X) = -\sum_i P(x_i) \log_2(P(x_i)), \tag{1}$$

And the entropy of X after observing values of another variable Y is defined as:

$$H(X|Y) = -\sum_j P(y_j) \sum_i P(x_i | y_j) \log_2(P(x_i | y_j)), \quad (2)$$

where $P(x_i)$ are the prior probabilities for all values of X , and $P(x_i | y_j)$ is the posterior probabilities of x_i given the values of y_j . The amount by which the entropy of X decreases reflects additional information about X provided by Y is called *information gain*, given by

$$IG(X|Y) = H(X) - H(X|Y). \quad (3)$$

Information gain tends to favor variables with more values and can be normalized by their corresponding entropy.

Table 2. The Information Gain of Features ($n=2$)

Feature	Benign Sample Set		Malicious Sample Set		Information Gain
	$y_i=1$	$y_i=0$	$y_i=1$	$y_i=0$	
FF84	98	11	92	0	0.000954615
4508	106	3	85	7	0.000387123
FDFF	109	0	89	3	0.001103624
F33C	101	8	76	16	0.002371356
BF28	94	15	91	1	0.000767842
...	...	...	...	...	...

For our problem, the expected entropy calculated as:

$$H(X) = -[P(x \text{ is normal}) \cdot \log_2 P(x \text{ is normal}) + P(x \text{ is abnormal}) \cdot \log_2 P(x \text{ is abnormal})] \quad (4)$$

If the data set is further partitioned by feature y_i , the conditional entropy $H(X|y_i)$ is calculated as:

$$\begin{aligned} H(X|y_i) = & -P(y_i = 0) \cdot [P(x \text{ is normal} | y_i = 0) \cdot \log_2 P(x \text{ is normal} | y_i = 0) \\ & + P(x \text{ is abnormal} | y_i = 0) \cdot \log_2 P(x \text{ is abnormal} | y_i = 0)] \\ & - P(y_i = 1) \cdot [P(x \text{ is normal} | y_i = 1) \cdot \log_2 P(x \text{ is normal} | y_i = 1) \\ & + P(x \text{ is abnormal} | y_i = 1) \cdot \log_2 P(x \text{ is abnormal} | y_i = 1)] \end{aligned} \quad (5)$$

where $y_i = 0$ denotes that the feature y_i do not appear in the samples, $y_i = 1$ denotes that the feature y_i appears in the samples.

The information gain for each feature is detailed in table 2.

2.3 Ensemble Method

It is well known that a combination of many different predictors can improve predictions. In the neural networks community ensembles of neural networks has been investigated by several authors [7-8]. Combining the outputs of several neural networks into an aggregate output often gives improved accuracy over any individual output. The set of networks is known as an ensemble or committee. We construct individual classifier by using modified Bagging method, shown in Alg.1.

Bagging is a method for generating multiple versions of a classifier and using them to get an ensemble classifier [9]. The multiple versions formed by making bootstrap replicates of the training set and using these as new training sets. Given an original training set S of size N , a training set $S_t, t=1,2,...,T$ is sampled with replacement from the original dataset S . Then the individual probabilistic neural network (*PNN*) classifier C_t is trained for each training set S_t . To classify an unknown instance x each *PNN* classifier C_t , returns its class prediction. The classification evidences are combined by using D-S theory of evidence to make a final decision. The detailed procedure of fusion method is stated in the next section.

-
1. Given: a training set $S = (x_1, y_1), ..., (x_n, y_n)$ where $x_i \in X$ and $y_i \in Y = \{1, 2, ..., K\}$; T , the number of individual classifier.
 2. For $t = 1, 2, ..., T$:
 - (1) Generate a new training set $S' = \text{bootstrap sample from } S$ (i.i.d. sample with replacement).
 - (2) Apply the base *PNN* classifier on S' , $C_t : X \rightarrow Y$
 3. Output the aggregated classifier based on D-S theory of evidence:

$$E(x) = \arg \max_k Bel(\theta_k)$$
-

Algorithm 1. The D-S Bagging Algorithm

2.4 Combination of the Multiple Classifiers Based on D-S Theory

D-S theory of evidence has shown its power for modeling uncertainty. In order to take into account uncertainties in classification, the D-S theory of evidence was used in this paper to combine the classification results. In this section, we first briefly describe D-S theory [10], and then generalize the problem of multiple classifiers combination based on D-S theory.

2.4.1 Preliminary

Let Θ be a finite set of mutually exclusive and exhaustive propositions about some problem domain and 2^Θ be its power set. A basic probability assignment (*BPA*), $m : 2^\Theta \rightarrow [0, 1]$, is used to quantize the belief committed exactly to a subset of the frame of discernment Θ given a certain of evidence. Whenever $m(\emptyset) = 0$, $\sum_{A \subseteq \Theta} m(A) = 1$.

The summary of $m(B)$ for all subsets $B \subseteq A$ becomes the total belief in A , i.e.,

$$Bel(A) = \sum_{B \subseteq A} m(B). \quad (6)$$

$Bel(A)$ is a measure of the total belief committed to A .

Two independent evidences expressed as two BPAs m_1 and m_2 can be combined into a single joined basic assignment $m = m_1 \oplus m_2$ by Dempster's rule of combination:

$$m(A) = m_1 \oplus m_2 = \begin{cases} k \sum_{B \cap C = A} m_1(B)m_2(C) & A \neq \emptyset \\ 0 & A = \emptyset \end{cases} \quad (7)$$

where $k^{-1} = 1 - \sum_{B \cap C = \emptyset} m_1(B)m_2(C) = \sum_{B \cap C \neq \emptyset} m_1(B)m_2(C)$.

2.4.2 Combining the Result of Individual Classifiers

In D-S theory, *BPA* is the degree of the belief of truth induced by a certainty of evidence. In multi-classifier combination the beliefs of truth for outputs from each individual classifier can be evaluated by confidence values or the classifiers performance measure. Xu et al [11] had proposed a method to calculate the belief. In that approach, the recognition, substitution and rejection rates were used to measure the belief of each classifier. When tested experimentally, this method was found to be quite robust, and was shown to outperform majority voting.

However, the way the belief was measured in literature [11] is not optimal, as it does not take into consideration the accuracy with respect to each class label, and hence does not resolve conflicts between classifiers in an optimal way. This had an impact on the performance of this combination method. So we propose a method that the beliefs calculated from the different class performance of each individual classifier in this paper.

Let there be T classifiers $e^{(1)}, e^{(2)}, \dots, e^{(T)}$ used for a K class problem, such that each class is denoted as θ_k , $k = 1, 2, \dots, K$. For malicious code detection, there are only two type classes –Normal and Abnormal. Thus under Dempster-shafer framework we have a frame of discernment denoted as: $\Theta = \{\theta_1, \bar{\theta}_1, \theta_2, \bar{\theta}_2\} = \{N, \bar{N}, A, \bar{A}\}$, where N denotes normal, A denotes abnormal, $N \cap A = \emptyset$. The *BBA* is defined as

$$m: 2^{\{N, \bar{N}, A, \bar{A}\}} \rightarrow [0, 1], \quad m(\emptyset) = 0, \quad m(\{N, \bar{N}, A, \bar{A}\}) = 1 - m(N) - m(\bar{N}) - m(A) - m(\bar{A}).$$

Given a test sample x , the related BPAs for the evidence from a classifier $e^{(i)}$ are obtained from the global performance of $e^{(i)}$ as:

$$\begin{aligned} m_i(N) &= TP \text{ rate} / 2, \quad m_i(\bar{N}) = FP \text{ rate} / 2, \\ m_i(A) &= TN \text{ rate} / 2, \quad m_i(\bar{A}) = FN \text{ rate} / 2. \end{aligned}$$

where TP , FP , TN , FN are true positive, false positive, true negative and false negative of classifier $e^{(i)}$ respectively.

At the next stage, the final *BPA* is calculated as $m = m_{e^{(1)}} \oplus m_{e^{(2)}} \oplus \dots \oplus m_{e^{(T)}}$. As we know that the Dempster's rule of combination is P -complete [12]. However, following

Barnett's methods, we can arrive at a computing time cost $O(T)$ [13], where T is the number of total classes.

Now we can finally define the combined classifier E by the following rules:

$$E(x) = \theta_j \text{ if } bel(\theta_j) = \arg \max_{i \in K} Bel(\theta_i) \quad (8)$$

This rule is not take into consideration the beliefs $bel(\bar{\theta}_i)$, which also contain useful information for the final decision. The following rules are proposed in order to include this information

$$E(x) = \theta_j, \text{ if } Bel(\theta_j) = \arg \max_{i \in K} \{Bel(\theta_i) \mid \forall i, bel(\bar{\theta}_i) \leq \alpha\}. \quad (9)$$

where $0 < \alpha < 1$, the rule tries to pursue the highest recognition rate under the constraint of the abounded error rate.

3 Experiment Results

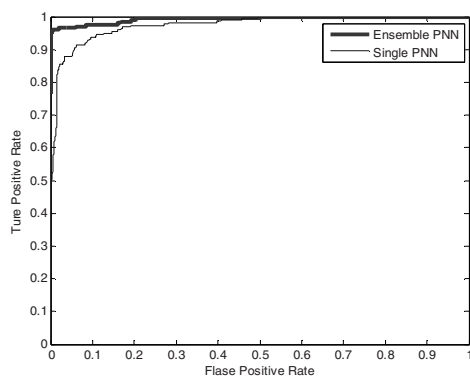
There is no benchmark data set available for the detection of malicious executables unlike intrusion detection. The malicious executables were downloaded from the website VX Heavens [14] were used by [4]. The data used here composed of 423 benign programs and 450 malicious executable codes. All the executables were in Windows PE format. They are divided into three datasets - Viruses, Trojan and Worm dataset- show in table 3. Each dataset contains 150 malicious samples. The clean programs were gathered from a freshly installed Windows 2000 server machine and labeled by a commercial virus scanner with the correct class label for our method.

Table 3. Dataset in Experiments

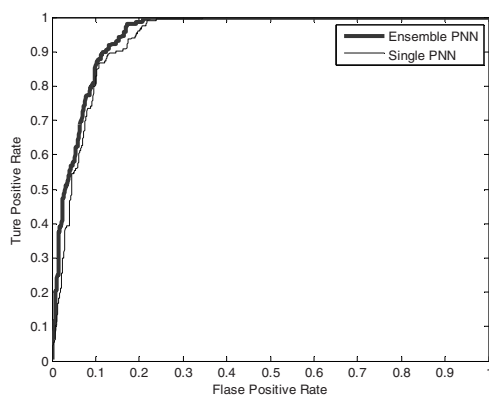
	Dataset	Benign	Malicious	classes
1	Virus dataset	423	150	2
2	Trojan dataset	423	150	2
3	Worm dataset	423	150	2

During the experiment, we use the software tools Ngrams[15] and MATLAB Neural Network Toolbox[16]. Ngrams tool is used in building byte n-gram profiles with parameters $8bit \leq n \leq 16bit$, and $20 \leq L \leq 4000$, where L is the number of n-gram. MATLAB tool is used to construct PNN classification.

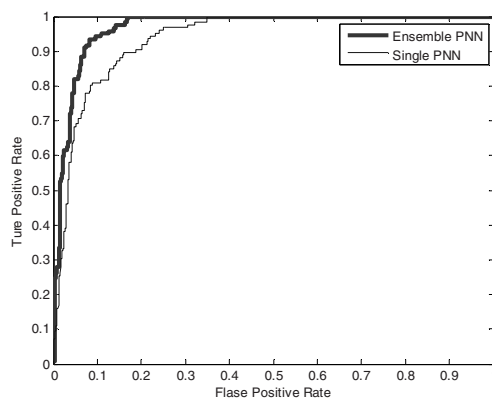
To obtain unbiased evaluation results, we performed 5-fold cross validation when check the single PNN classifier. The data is randomly partitioned into 5 disjoint datasets. Four of these datasets are used of training and the remaining dataset is used for testing. The process is repeated 5 times, each time using a different testing dataset. The folds are created in random, balanced way. Then extensive experiments have been carried out to test the performance of the approach proposed in Section 2.4. A number of experiments have been conducted using the decision rules given by (9) with different threshold values.



(a) Trojan



(b) Worm



(c) Viruses

Fig. 3. ROC graphs to compare our ensemble PNN approach with that of single PNN

Figure 3 shows the ROC curves of the proposed ensemble PNN method and the single PNN classifier. Taking area under ROC curve as performance criteria, from the visual inspection of the ROC curves it is clear that the ensemble PNN outperforms the single PNN in all cases. Based on the experimentation we infer the following. In the present ensemble PNN classifier, basic probability assignments based on individual classifier's class-wise performances results in more accurate calculation of beliefs, thus boosts the ensemble classifier's performance.

In literature [4], the authors obtained good results in the context of using n-grams for detecting malicious executables. When compare our work with that of [4], we can observe the same conclusion that ensemble method is do improve the performance of classification.

When it comes to polymorphic viruses, static analysis methods do not work. To tackle these malicious codes, static analysis methods should be combined with dynamic analysis methods for efficient detection. Here we assume that a polymorphic virus must decrypt before it can execute normally.

4 Conclusion

In this paper, we generalize the problem of multi-classifier combination based on Dempster-Shafer theory of evidence for detecting previously unknown malicious codes. The way to derive initial basic probability assignments takes into consideration each classifier's performance with respect to each class label. Our extensive experiments have shown that the combination approach improves the performance of the individual classifier significantly. It shows that the present method could effectively be used to discriminate normal and abnormal programs.

Future work involves extending our learning algorithms to better utilize n-gram. We are planning on testing this method over a larger set of malicious and benign executables for a fully evaluation of this method. In addition with a larger data set, we plan to evaluate this method on different types of malicious executables such as macros and visual basic scripts.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grant No.60373023 and the Scientific Research Fund of Hunan Provincial Education Department of China under Grant No.05B072.

References

1. Kephart, J., Arnold, W.: Automatic Extraction of Computer Virus Signatures. In: Proceedings of the 4th Virus Bulletin International Conference, Abingdon, pp. 178–184 (1994)
2. Lo, R., Levitt, K., Olsson, R.: MCF: A Malicious Code Filter. *Computers and Security* 14, 541–566 (1995)

3. Tesauro, G., Kephart, J., Sorkin, G.: Neural networks for computer virus recognition. *IEEE Expert*, 8, 5–6 (1996)
4. Kolter, J.Z., Maloof, M.A.: Learning to detect malicious executables in the wild. In: *Proceedings of the 10PthP ACM SIG KDD International Conference on Knowledge Discovery and Data Mining*, pp. 470–478. ACM Press, New York (2004)
5. Hansen, L.K., Salamon, P.: Neural network ensembles. *IEEE Trans Pattern Anal.* 12(10), 993–1001 (1990)
6. Jurafsky, D., James, H.: *Speech and Language Processing*. Prentice-Hall, New York (2000)
7. Zhou, Z.H., Wu, J.X., Tang, W.: Ensembling Neural Networks: Many Could be Better than All. *Artificial Intelligence* 137, 239–263 (2002)
8. Granitto, P.M., Verdes, P.F., Navone, H.D., Ceccatto, H.A.: Aggregation Algorithms for Neural Network Ensemble Construction. In: Werner, B. (ed.) *Proceedings of the VII Brazilian Symposium on Neural Networks*, pp. 178–183. IEEE Computer Society, Pernambuco (2002)
9. Breiman, L.: Bagging predictors. *Machine Learning* 24, 123–140 (1996)
10. Dempster, A.: Upper and lower probabilities induced by multi-valued mapping. *Annals of Mathematical Statistics* 2, 325–339 (1967)
11. Xu, L., Krzyzak, A., Suen, C.: Methods of combining multiple classifiers and their applications to handwritten recognition. *IEEE Transactions on Systems, Man and Cybernetics, SMC*. 22(3), 418–435 (1992)
12. Orponen, P.: Dempster's rule of combination is P-complete. *Artificial Intelligence* 1(2), 245–253 (1990)
13. Barnett, J.A.: Computational methods for a mathematical theory of evidence. In: *Proceedings of 7th Int. Joint Conf. Artificial Intelligence*. Vancouver, BC, pp. 868–875 (1981)
14. Vx heavens: <http://www.vx.netlux.org>
15. Perl package Text: Ngrams:
<http://search.cpan.org/author/vlado/text-ngrams-0.03/ngrams.pm>
16. Mathworks (ed.): *Neural Network Toolbox User's Guide (version 4)*. The Mathworks, Inc. Natick, Massachusetts (2001)

Generating Simplified Regular Expression Signatures for Polymorphic Worms^{*}

Yong Tang¹, Xicheng Lu¹, and Bin Xiao²

¹ College of Computer, National University of Defense Technology,
Changsha Hunan, 410073, P.R. China
{Ytang, Xclu}@nudt.edu.cn

² Department of Computing, Hong Kong Polytechnic University,
Hong Kong
csbxiao@comp.polyu.edu.hk

Abstract. It is crucial to automatically generate accurate and effective signatures to defense against polymorphic worms. Previous work using conjunctions of tokens or token subsequence could lose some important information, like ignoring 1 byte token and neglecting the distances in the sequential tokens. In this paper we propose the *Simplified Regular Expression (SRE)* signature, and present its signature generation method based on the multiple sequence alignment algorithm. The multiple sequence alignment algorithm is extended from the pairwise sequence alignment algorithm, which encourages the contiguous substring extraction and is able to support wildcard string alignment and to preserve the distance of invariant content segment in generated SRE signatures. Thus, the generated SRE signature can express distance information for invariant content in polymorphic worms, which in turn makes even 1 byte invariant content extracted from polymorphic worms become valuable. Experiments on several types of polymorphic worms show that, compared with signatures generated by current network-based signature generation systems (NSGs), the generated SRE signatures are more accurate and precise to match polymorphic worms.

1 Introduction

Signature-based intrusion detection system(IDS) is one of the most deployed and effective way for worm defense. Todate, the signatures used by these IDSs for detecting worms are manually generated by security experts, which is too slow (typically days after worm released) in contrast with the speed of worm propagation (usually outbreak in zero day). Motivated by increasing the rate of signature generation, a number of automatic signature generation systems or methods have been proposed in recent years, which could be classified into two

^{*} The work was partially supported by the National Basic Research Program of China (973) under Grant No. 2005CB321801, and the National Natural Science Foundation of China under Grant No. 90412011.

categories - network-based signature generation (NSG) [1,2,3,4,5,6] and host-based signature generation (HSG) [7,8,9,10]. NSG systems have the advantage that they have no influence on the protecting hosts or networks. They usually firstly collect suspicious flows that contain the samples of worms through flow classifier or honeypot, then output content-based signatures for the worms by analyzing these suspicious flows.

Accuracy of outputted signatures is the most important criteria to evaluate NSG systems. The earlier NSG systems [1,2,3] generate single contiguous byte string signatures, which have been proven to be not effective [4,5] for matching worms. Some up-to-date NSG systems [4,6] generate token-based signatures, where a token is a byte sequence that occurs in a significant number of suspicious flows. But as we shall show in later in the paper, these token-based signature could lose some important information, like ignoring 1 byte token and neglecting the distances in the sequential tokens.

Regular expression have significant advantages for intrusion detection, in terms of flexibility, accuracy, and efficiency [11,12]. In this paper we propose *Simplified Regular Expression (SRE)* signature, a more expressive and accurate signature type. Given the samples of a polymorphic worm, we propose to generate its SRE signature using the multiple sequence alignment, which encourages the contiguous substring extraction and is able to support wildcard string alignment and to preserve the distance of invariant content segment in generated SRE signatures. Thus, the generated SRE signature can express distance information for invariant content in polymorphic worms, which in turn makes even 1 byte invariant content extracted from polymorphic worms become valuable. Experiments on several types of polymorphic worms show that our approach outperforms previous works in terms of signature accuracy.

The rest of paper is organized as follows. We first introduce the anatomy of polymorphic worms and summarize the limitation of the signature types outputted by current NSG systems. In Section 3 we provide the formal definition of SRE signature. Next in Section 4 we describe how to generate SRE signature for single polymorphic worm using sequence alignment and provide the corresponding algorithms. We present the evaluation and limitation of our approach in Section 5 and Section 6.

2 Background: Polymorphic Worms and Limitation of Current Signature Types

Polymorphic worms employ polymorphism technique to change their byte sequence at every instance for evading detection. Within a polymorphic worm body, there are two classes of bytes. *Invariant content* are those bytes fixed in value and *must* be present in every worm sample to ensure infection successful. *Wildcard bytes* are those which will change value for each different worm sample. For example, a polymorphic Code Red II worm is presented in Fig. 1, which in turn contains seven invariant content: “GET”, “.ida?”, “XX”, “%u”, “%u7801”, “=”, and “HTTP/1.0\r\n”. Within these invariant content, “%u7801” is 4 bytes

after “%u”, “HTTP /1.0\r\n” is 7 bytes after “=”. We call these relations, like “possible start position of a substring” or “how many characters between two substrings”, *distance restrictions*. In practice, distance restrictions are critical for successful worm infections.

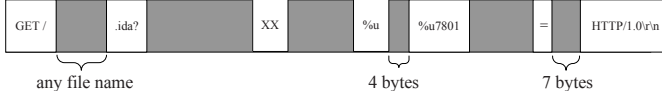


Fig. 1. Polymorphic Code Red II worm. Shaded content represents wildcard bytes, unshaded content represents invariant bytes.

Next we use this example to explain the limitations of the signature types outputted by current NSG systems. The earlier NSG systems [1,2,3] generate contiguous byte string signatures, like “HTTP /1.0\r\n” or “%u7801”, which are not accuracy enough to characterize this worm. Polygraph[4] and Hamsa[6] generate token (substring with a minimum length and a minimum coverage in the suspicious flows) based signatures. But we found that token-based signatures are still not accurate enough. First, tokens can not have the length of 1, like “=”; otherwise, every possible character (0-255 in value) will be extracted as a “token”. Second, Polygraph and Hamsa can not express the distance restrictions of invariant content, like “‘%u7801’ is 4 bytes after ‘%u’”. Y. Tang et al. [13] propose PADS signature. A PADS is a position-aware frequency distribution of characters in a fixed length region. But there is not an fixed length region that contains all of invariant content in this example.

3 SRE Signature

A regular expression describes a set of strings without enumerating them explicitly. It is widely believed that regular expression have significant advantages for intrusion detection, in terms of flexibility, accuracy, and efficiency [11,12]. Motivated by the insufficiency of the signature types of previous NSG systems, we propose a novel signature type—*Simplified Regular Expression (SRE) signature*.

A SRE signature is a simplified form of regular expression, in which there are only three repeating qualifiers: “*”, “[k_1, k_2]” and “[k]”. We replace the “.” in regular expression by “*” to represent an arbitrary string (including zero-length string), replace “. { k_1, k_2 }” by “[k_1, k_2]” to represent any string with a length from k_1 to k_2 , and replace “. { k }” by “[k]” to represent a string consisting of k number of arbitrary character. For example, “one’*two’[2]three’[3,5]” is a SRE signatures that is equal to the regular expression “one.*two.{2}three.{3,5}”.

Suppose $\Phi = \{*, [k], [k_1, k_2]\}$ is the set of the three repeating qualifiers we just introduced. Σ^+ is the set of not empty strings over a finite alphabet Σ . The formal definition of SRE signature is provided by Definition 1.

Definition 1 SRE Signature. A SRE signature $= (p_0)s_1p_1s_2p_2s_3 \dots p_{k-1}s_k(p_k)$, where $p_i \in \Phi$ is a *repeating qualifier*, $s_i \in \Sigma^+$ is a *substring* ($i \in [0, k]$), (p_0) and (p_k) means p_0 and p_k are optional.

Within a SRE signature, the *substrings* are used to express the invariant content in polymorphic worms, the *repeating qualifiers* are used to express the distance restrictions between invariant content. For the previous example, we can use the SRE signature “*GET /’*’.ida?’*’XX’*’%u’[4]’%u7801’*’=’[?]’HTTP / 1.0\’r\’n”* to precisely express the characteristic of Code Red II worm.

4 Generating SRE Signature for Polymorphic Worm Using Sequence Alignment

4.1 Overview

Sequence alignment is the procedure of comparing two (pairwise) or more (multiple) sequences by searching for a series of individual characters or character patterns that are in the same order in the sequences. Sequence alignment is widely used to quantify and visualize similarity between sequences, and it has been most prominently applied in bioinformatics [14,15]. A pairwise sequence alignment is a scheme of writing one sequence on top of another where the residues in one position are deemed to have a common character. Fig. 2 illustrates the pair-wise sequence alignment between “ONExxxTWOxxxTHREExxxx” and “dsfONEdsdTWOvvvTHREEb”. By inserting some gap(‘-’), the common characters are deemed to the same columns.

-	-	O	N	E	x	x	x	T	W	O	x	x	x	T	H	R	E	E	x	x	x	x		
d	s	f	O	N	E	d	s	d	T	W	O	-	v	v	v	T	H	R	E	E	-	-	b	
*	*	*	O	N	E	?	?	?	T	W	O	*	?	?	?	T	H	R	E	E	*	*	*	?

Fig. 2. Example of pair-wise sequence alignment

The alignment result can be described by a sequence with wildcards, in which the question wildcard ‘?’ presents one character, the asterisk wildcard ‘*’ presents one or zero character. The alignment result in Fig. 2 is expressed by the sequence “***‘ONE’???‘TWO’???*‘THREE’***?”. As we shall introduce later, such alignment result can be converted to a SRE signature according to its semantic. For instance, “***‘ONE’???‘TWO’???*‘THREE’***?” can be converted to SRE signature “[0,3]‘ONE’[3]‘TWO’[3,4]‘THREE’[1,3]”.

If we have captured a number of network flows that are the instances of a polymorphic worm, we generate the worm’s signature by the following steps.

1) Transform the flows to character sequences. In the rest of the paper, these character sequences are referred to *samples* of a worm.

2) Analyze these samples by multiple sequence alignment that arranges these samples in a scheme where positions believed to be invariant bytes are written in a common column. For example, suppose $A = \text{“oxnxexzxtwoxxw”}$, $B = \text{“ytwoyown”}$

yeyz”, $C = \text{“cvcvcvtwovcwc”}$ are three samples of a polymorphic worm, as Fig. 4 shows, aligning these three samples will get a result `“*****?‘two’??‘w’*****”`.

3) Transfer the alignment result to a SRE signature as the final signature for this worm. For previous example, the alignment result `“*****?‘two’??‘w’*****”` can be converted to the SRE signature `“[1,8]‘two’[2]‘w’[0,5]”`.

4.2 Problem of Current Sequence Alignment Algorithm

The Needleman-Wunsch algorithm [16] is a typical global alignment algorithm that computes the optimal alignment between two sequences by maximizing a similarity score function as Formula (1) gives, where k_m denotes the number of matches, W_m denote the score for character match, k_d denotes the number of mismatches, W_d denote the score for character mismatch, k_{gaps} denotes the number of gaps, δ denote the penalty score for a gap. If we set $W_m = 1$, $W_d = 0$, $\delta = -1$, for the example in Fig. 2, this algorithm outputs an alignment with similarity score 4 ($11 \times 1 + 7 \times 0 + 7 \times -1$).

$$SC(x, y) = k_m \times W_m + k_d \times W_d + k_g \times \delta \quad (1)$$

If a *piece* is a substring in alignment result with a length of only 1, we found that the Needleman-Wunsch algorithm is likely to output a large number of pieces in resulting alignment, instead of outputting the invariant content of polymorphic worms. Consider the simple example provided by Polygraph [4] that align two strings `“oxnxexzxtwox”` and `“ytwoyoyneyz”`. Fig. 3(a) shows the alignment result of the Needleman-Wunsch algorithm, which contains four pieces(‘o’, ‘n’, ‘e’, ‘z’). Creating too many trivial and useless pieces will prevent finding contiguous invariant content we are concerning about. Obviously Fig. 3(b) is a better alignment since the substring ‘two’ is semantical.

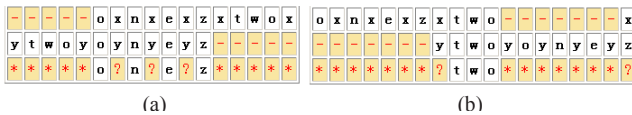


Fig. 3. Two Alignment results for different algorithm

Through a plenty of experiments, we found that no matter the parameters are adjusted, the Needleman-Wunsch algorithm always tend to product a large number of pieces and hence lost some invariant content in polymorphic worms.

4.3 Pairwise Sequence Alignment Algorithm

As Algorithm 1 shows, we propose a new pairwise sequence alignment algorithm for our approach—*CSR* (*contiguous substrings rewarded*) algorithm, which is extended from the Needleman-Wunsch algorithm by the following three means:

1.Rewarding contiguous substrings: Motivating by reducing pieces, we modify the similarity score function of the Needleman-Wunsch algorithm from Formula (1) to Formula (2) by introducing a score function $enc()$ to reward contiguous substrings. For the example given in Fig. 3, if we define $enc(x) = 3(x - 1)$ and set $W_m = 0.5$, $W_d = 0$, $\delta = -1$, the similarity score of Fig. 3(a) is -8 ($0.5 \times 4 + 0 \times 3 + -1 \times 10$), the similarity score of Fig. 3(b) is -6.5 ($0.5 \times 3 + 0 \times 2 + -1 \times 14 + 3 \times (3 - 1)$). Hence our CSR algorithm will output the better alignment Fig. 3(b).

$$SC(x, y) = k_m \times W_m + k_d \times W_d + k_g \times \delta + \sum_{\substack{s \text{ is substring} \\ \text{in alignment result}}} Enc(|s|) \quad (2)$$

2.Supporting wildcards: The CSR algorithm allows the input sequences contain two previously introduced wildcard-‘?’ and ‘*’. We provide a set of character comparison rules, as Table 1 depicts, where ‘ α ’ and ‘ β ’ denote two different characters, ‘-’ is a gap in alignment. By calling the function LookupCharCompTab(.) (line 27), the CSR algorithm lookup this table to determine the value of a position in result sequence.

Table 1. Character comparison rules for CSR algorithm. x and y are the characters in input sequences in the same column, r is the character will appear in the same column of alignment result sequence.

x	α	α	α	?	α	?	α	-	*	α	?
y	α	β	?	?	?	*	*	*	*	-	-
r	α	?	?	?	?	*	*	*	*	*	*

3.Preserving distance restriction: In order to preserve the distance restrictions during the alignments of polymorphic worm samples, we assign every character in sequences a length area $[min, max]$ during the alignment process, where min is the low-bound, max is the up-bound. As the line 4, 7 shows, we first initialize the length range of every character in input sequences to $[1, 1]$. During the alignment, we set the length range of a inserted gap to $[0, 1]$. As the algorithm line 28, 29 shows, finally the length range of a character in alignment result is calculated by minimizing the low-bound and maximizing the up-bound of the characters with the same column.

Convert alignment result to SRE signature. The alignment result of the CSR algorithm is a sequence with wildcards. Notice that each wildcard carries a length range, thus we can easily convert alignment result to a SRE signature by merging the wildcards between two substrings and accumulating their length range. For example, consider the alignment result in Fig. 3(b), the length range of every ‘*’ is $[0, 1]$, the length range of every ‘?’ is $[1, 1]$, we merge the eight wildcards before substring “two” to one repeating qualifier, and calculate its low-bound of length range to $0 \times 7 + 1 = 1$, up-bound to $1 \times 7 + 1 = 8$. Hence, the repeating qualifier before the substring ‘two’ is ‘ $[1, 8]$ ’. Repeat this process, we can get the converted SRE signature “ $[1, 8]$ ‘two’ $[1, 8]$ ” finally.

```

input      : sequence  $X, Y$ ,
output    : alignment result sequence  $R$ , similarity score  $SC$ 
Parameters:  $W_m$ : match score;  $W_d$ : mismatch score;  $\delta$ : gap penalty ; enc (): contiguous
                substring rewarding function ; LookupCharCompTab (.): lookup the character
                comparison table to determine the character placed in alignment result.

1  Initialization
2   $N \leftarrow$  the length of  $X$ ;  $M \leftarrow$  the length of  $Y$ ;
3  foreach  $i$  such that  $1 \leq i \leq N$  do
4     $F_{i,0} \leftarrow i\delta$ ;  $T_{i,0} \leftarrow 0$ ;  $PTR_{i,0} \leftarrow \text{Up}$ ;  $X_i.min \leftarrow 1$ ;  $X_i.max \leftarrow 1$ ;
5  end
6  foreach  $j$  such that  $1 \leq j \leq M$  do
7     $F_{0,j} \leftarrow j\delta$ ;  $T_{0,j} \leftarrow 0$ ;  $PTR_{0,j} \leftarrow \text{Left}$ ;  $Y_j.min \leftarrow 1$ ;  $Y_j.max \leftarrow 1$ ;
8  end
9   $F_{0,0} \leftarrow 0$ ;  $T_{0,0} \leftarrow 0$ ;  $PTR_{0,0} \leftarrow \text{TraceEnd}$ ;
10 Iteration
11 foreach  $i$  such that  $0 \leq i \leq N$  do
12   foreach  $j$  such that  $0 \leq j \leq M$  do
13     if  $X_i, Y_j$  are not wildcard and  $X_i = Y_j$  then
14        $S_{X_i, Y_j} = W_m$  ;  $T_{i,j} = T_{i-1, j-1} + 1$  ;
15     else  $S_{X_i, Y_j} = W_d$  ;  $T_{i,j} = 0$  ;
16        $F_{i,j} \leftarrow \max \begin{cases} F_{i-1, j-1} + S_{X_i, Y_j} + \text{enc}(T_{i,j}) & [case1] \\ F_{i-1, j} + \delta & [case2] \\ F_{i, j-1} + \delta & [case3] \end{cases}$ 
17        $PTR_{i,j} \leftarrow \begin{cases} \text{Dial} & \text{if } [case1] \\ \text{Left} & \text{if } [case2] \\ \text{Up} & \text{if } [case3] \end{cases}$ 
18   end
19 end
20 Traceback
21  $t \leftarrow PTR_{N, M}$  ;  $i \leftarrow N$ ;  $j \leftarrow M$ ;
22 Allocate a empty sequence  $R$ ;
23 while  $t \neq \text{TraceEnd}$  do
24   Allocate a new character  $r$  and add it to the head of sequence  $R$ ;
25   switch  $t$  do
26     case Dial
27        $r \leftarrow \text{LookupCharCompTab}(X_i, Y_i)$ ;
28        $r.min \leftarrow \min(X_i.min, Y_j.min)$ ;  $r.max \leftarrow \max(X_i.max, Y_j.max)$ ;
29        $i \leftarrow i - 1$ ;  $j \leftarrow j - 1$ ;
30     case Up
31        $r \leftarrow *$ ;  $r.min \leftarrow 0$ ;  $r.max \leftarrow \max(X_i.max, 1)$  ;  $i \leftarrow i - 1$ ;
32     case Left
33        $r \leftarrow *$ ;  $r.min \leftarrow 0$ ;  $r.max \leftarrow \max(Y_j.max, 1)$ ;  $j \leftarrow j - 1$ ;
34   end
35    $t \leftarrow PTR_{i,j}$ ;
36 end
37 Return
38  $SC \leftarrow F_{N, M}$ ; return  $SC, R$ 

```

Algorithm 1. CSR algorithm

4.4 Multiple Sequence Alignment Algorithm

Given the samples of a polymorphic worm, we aim to generate its SRE signature using multiple sequence alignment. Because the CSR algorithm supports wildcard characters and can preserve distance restrictions, we simply design our multiple sequence alignment algorithm by progressively employing CSR algorithm, as Algorithm 2 gives.


```

input : sequence set  $S$ ,
output: alignment result sequence  $R$ 
1 repeat
2   randomly select two sequences  $X$  and  $Y$  from  $S$ ;  $S \leftarrow S \setminus \{X, Y\}$ ; employ the CSR
   algorithm to align  $X$  and  $Y$ , the result sequence is  $A_{X,Y}$ ;  $S \leftarrow S \cup \{A_{X,Y}\}$ 
3 until  $|S| = 1$ ;
4 return  $A_{X,Y}$ 

```

Algorithm 2. Multiple sequence alignment algorithm

For example, if the sequences $A = \text{"oxnxexzxtwoxxw"}$, $B = \text{"ytwoyownyeyz"}$, and $C = \text{"cvcvcvtwovcw"}$ are three samples of a polymorphic worm, as Fig.4 shows, we first align A and B , and then use the result (denoted as $\text{MALIGN}(A, B)$) to align with C , finally get the alignment result $\text{"*****?two'??'w'*****"}$, which can be converted to the SRE signature $\text{"[1,8]'two'[2]'w'[0,5]"}$ at last. We can see that the distance restrictions, such as "w" is 2 bytes after "two" , are correctly preserved in this SRE signature.



Fig. 4. Example of multiple sequence alignment algorithm

5 Evaluation

Experiment Settings. Similar to Polygraph [4] and Hamsa [6], we use three synthetically generated polymorphic worms (ATPhttpd exploit, Code Red II exploit, and BIND-TSIG exploit) to evaluate our approach. In order to test the false positive rate of generated signatures, we use the first week's network TCPdump data of the 1999 DARPA Intrusion Detection Evaluation Data Sets [17] (1.8GB) as the normal traffic data, and set $\text{enc}(x) = 3(x - 1)$, $W_m = 0.5$, $W_d = 0$, $\delta = -1$ as the parameters of CSR algorithm.

Signature Quality. For each worm, we generate its signature by analyzing its eight samples using our multiple sequence alignment algorithm. Table. 2 gives the generated signatures of the three worms. Table 5 gives the signature comparison for Code Red II worm between our approach and Polygraph. we can see that our SRE signature is more precise than Polygraph's *conjunction signature*, because our signature preserves the distance restrictions for invariant content through the repeating qualifiers "[4]" and "[7]" ; in addition, our signature contains "=" , which is an invariant content with a length of 1, however Polygraph does not.

Performance Study. Suppose all flows are l bytes long, although the CSR algorithm added some sentences (line 13-15) each with a runtime of $O(1)$ in the main iteration of Needleman-Wunsch algorithm, the time and space overhead

Table 2. Generating signatures for three worms

Worm	Signature	False positive	False negative	Speed (secs)	Memory usage (MB)
BIND-TSIG	'*\xFF\xBF'[200]*\x00\x00\xFA'[2]	0	0	2.1	4.0
ATPhttpd	'GET /* *\xFF\xBF'*HTTP/1.1\r\n'	0	0	9.2	4.3
Code Red II	'GET /* *.ida?*'XX*'%u'[4] '%u7801'*= '[7]HTTP/1.0\r\n'	0	0	1.1	3.9

Table 3. Signature type comparison for Polymorphic Code Red II worm

Signature	Generated signature
Conjunction signature (Polygraph)	GET /*.ida?.*XX.*%u.*%u7801.*HTTP/1.0\r\n
SRE signature	'GET /* *.ida?*'*XX*'%u'[4] '%u7801'*= '[7]HTTP/1.0\r\n'

of CSR algorithm is still $O(l^2)$. Aligning θ flows using our multiple sequence algorithm takes $O(\theta l^2)$ time and $O(l^2)$ space. Table 2 shows the time and memory consumption of generating signatures for the three worms. All experiments were executed on a PC with single 3.0GHz Intel Pentium IV processor.

6 Limitation and Future Work

In this work we focus on signature generation, how to capture the samples of polymorphic worms is beyond the scope of this paper. And we only consider generating single signature given the samples of a polymorphic worm, which is the base of the fully general case that generating signatures for mixed several different worms.

In our future work, we plan to design a system that can automatic generate signatures for multiple polymorphic worms in a live environment. Given the suspicious flows that contain the samples of several polymorphic worms, we first need to cluster them into some clusters. Fortunately there are already some researches on worm samples clustering [4,18]. After clustering, we can generate a signature for each cluster using the method presented in this paper.

7 Conclusion

In this paper, we propose a new signature type—SRE signature. A SRE signature is a simplified form of regular expression, which is more effective for characterizing polymorphic worms. We present a multiple sequence alignment based method to generate SRE signature for single polymorphic worm. In order to overcome the problem that the typical Needleman-Wunsch algorithm will product a large number of useless pieces, we propose a novel pairwise sequence alignment algorithm—CSR algorithm. Experiment results indicate that our approach is effective for automatic signature generation of polymorphic worms.

References

1. Kreibich, C., Crowcroft, J.: Honeycomb - creating intrusion detection signatures using honeypots. In: Proceedings of the Second Workshop on Hot Topics in Networks (Hotnets II), Boston (November 2003)
2. Kim, H.A., Karp, B.: Autograph: Toward automated, distributed worm signature detection. In: USENIX Security Symposium, pp. 271–286 (2004)
3. Singh, S., Estan, C., Varghese, G., Savage, S.: Automated worm fingerprinting. In: Proc. 6th USENIX OSDI, San Francisco, CA (December 2004)
4. Newsome, J., Karp, B., Song, D.: Polygraph: Automatically generating signatures for polymorphic worms. In: Proceedings of the 2005 IEEE Symposium on Security and Privacy, pp. 226–241. IEEE Computer Society Press, Washington (2005)
5. Crandall, J.R., Su, Z., Wu, S.F., Chong, F.T.: On deriving unknown vulnerabilities from zero-day polymorphic and metamorphic worm exploits. In: Proceedings of the 12th ACM conference on Computer and communications security, pp. 235–248. ACM Press, New York (2005)
6. Li, Z., Sanghi, M., Chen, Y., Kao, M.-Y., Chavez, B.: Hamsa: Fast signature generation for zero-day polymorphic worms with provable attack resilience. In: Proceedings of the 2006 IEEE Symposium on Security and Privacy (S&P'06), IEEE Computer Society Press, Washington (2006)
7. Newsome, J., Song, D.: Dynamic taint analysis for automatic detection, analysis, and signaturegeneration of exploits on commodity software. In: NDSS (2005)
8. Liang, Z., Sekar, R.: Fast and automated generation of attack signatures: a basis for building self-protecting servers. In: CCS '05: Proceedings of the 12th ACM conference on Computer and communications security, pp. 213–222. ACM Press, New York (2005)
9. Xu, J., Ning, P., Kil, C., Zhai, Y., Bookholt, C.: Automatic diagnosis and response to memory corruption vulnerabilities. In: CCS '05: Proceedings of the 12th ACM conference on Computer and communications security, pp. 223–234. ACM Press, New York (2005)
10. Wang, X., Li, Z., Xu, J., Reiter, M.K., Kil, C., Choi, J.Y.: Packet vaccine: black-box exploit detection and signature generation. In: CCS '06: Proceedings of the 13th ACM conference on Computer and communications security, pp. 37–46. ACM Press, New York (2006)
11. Sommer, R., Paxson, V.: Enhancing byte-level network intrusion detection signatures with context. In: CCS '03: Proceedings of the 10th ACM conference on Computer and communications security, pp. 262–271. ACM Press, New York (2003)
12. Kumar, S., Dharmapurikar, S., Yu, F., Crowley, P., Turner, J.: Algorithms to accelerate multiple regular expressions matching for deep packet inspection. In: Proceedings of ACM SIGCOMM'06, vol. 36, pp. 339–350. ACM Press, New York (2006)
13. Tang, Y., Chen, S.: Defending against internet worms: A signature-based approach. In: Proceedings of the 24th Annual Conference IEEE INFOCOM 2005 (March 2005)
14. Gelfand, M.S., Mironov, A., Pevzner, P.: Gene recognition via splices sequence alignment. In: Proc. Natl.Acad. Sci. USA, pp. 9061–9066 (1996)
15. Goad, W.B., Kanehisa, M.I.: Pattern recognition in nucleic acid sequences: a general method for finding local homologies and symmetries. *Nucleic Acids Research* 10, 247–263 (1982)

16. Needleman, S.B., Wunsch, C.D.: A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.* 48, 443–453 (1970)
17. Lippmann, R., Haines, J.W., Fried, D.J., Korba, J., Das, K.: The 1999 darpa off-line intrusion detection evaluation. *Comput. Networks* 34(4), 579–595 (2000)
18. Yegneswaran, V., Giffin, J.T., Barford, P., Jha, S.: An Architecture for Generating Semantics-Aware Signatures. In: *Proceedings of the 14th USENIX Security Symposium*, Baltimore, MD, USA, pp. 97–112 (August 2005)

AAA for Spontaneous Roaming Agreements in Heterogeneous Wireless Networks

Zhi (Judy) Fu¹, Minh Shin², John C. Strassner¹, Nitin Jain¹, Vishnu Ram¹,
and William A. Arbaugh²

¹ Motorola Inc., 1301 E Algonquin Rd., Schaumburg, IL 60196

{judy.fu, john.strassner, nitin, vishnu}@motorola.com

² Department of Computer Science, University of Maryland, College Park, MD, 20742

{mhshin, waa}@cs.umd.edu

Abstract. A current challenge in heterogeneous wireless networks is to enable them to work together in a spontaneous fashion, without having pre-established roaming agreements. Currently, formal roaming agreements are manually set up, which is a costly and time-consuming process. It is highly desirable for network cooperation to be established on the fly. However, establishing spontaneous roaming agreement is a very challenging research issue. This paper presents a novel AAA (Authentication, Authorization and Accounting) architecture to support policy-based negotiation for establishing spontaneous roaming agreements. The new architecture integrates policy-based negotiation into the normal user association and authentication process for spontaneous and dynamic roaming agreements and interworking. This integration minimizes changes to existing AAA architecture for enabling the new paradigm of automated provider interworking and cooperation.

1 Introduction

Providers are using heterogeneous wired and wireless systems to offer consumers increased network connectivity. Since it is unlikely that one wireless provider can provide ubiquitous coverage, high bandwidth access, and all possible services, the best way for consumers to get the most coverage for their desired services is for heterogeneous providers to cooperate and provide a single “composite” service in a seamless manner. However, the heterogeneous technologies and different administrative policies create significant challenges when various wireless networks are converged.

Currently, different wireless providers work together through formal roaming agreements that are statically defined. Setting up a roaming agreement today between two providers is a manual process. Typically, business people from the two operators meet and agree on the commercial terms and sign the necessary paperwork defining the agreement, and then technicians from each operator exchange technical information and configure elements within their own network. Even with industry standards, the roaming agreement setup is a costly and time-consuming process. It is therefore appropriate for long-term partnerships with large sessions but not suitable for spontaneous collaborations with short sessions. On the other hand, there will be numerous

providers with different service offerings, technologies, size and locations. It is not feasible to set up formal roaming agreements with every possible provider. However, since a consumer's access and services are limited by established roaming agreements, if a roaming agreement does not exist, users will either be disconnected or need to buy access at a prohibitively high cost.

It is thus highly desirable to enable spontaneous inter-working without pre-established roaming agreements between heterogeneous wireless providers. Not only would consumers get more services and coverage with only one subscription, providers would be able to generate more revenue with flexible partnerships and lower cost in providing more services to their customers. This is also beneficial for start-up providers to quickly offer their differentiated values versus established providers.

To address this need, brokered roaming agreement models have been deployed [1,2]. With the brokered model, operators establish roaming agreements with a broker and the broker then acts as a proxy to handle all roaming related signaling and traffic on behalf of the operators. With this model, operators benefit from not having to establish individual roaming agreements with other operators. However, there are also serious drawbacks to this model. First, the signaling, AAA and roaming traffic will have to go through the broker, incurring unnecessarily long latency. Second, operators have limited control and flexibility over establishing roaming terms with another operator. Third, operators have to pay brokers for any traffic going through the broker, and thus the profit margin becomes lower.

To overcome the limitations of the brokered roaming model, the Ambient networks project [3] proposed mechanisms for automatically establishing bilateral roaming agreements directly between operators. They proposed automatic negotiation between two servers to replace manual negotiation. The negotiations are conducted offline with triggers such as a new member of an industry association or deployment of a new access networks. With this automation, bilateral roaming agreements can be established efficiently at a lower cost. However, the agreements are still pre-established but not spontaneous roaming agreements that can be established on the fly. The main limitation of this approach is that random roaming activities at different locations cannot always be predicted; thus, pre-determined roaming agreements cannot cover all possible networks that users may roam to.

Therefore, the ideal case is to enable spontaneous roaming agreements to fulfill the vision of seamless and ubiquitous roaming for users. This also gives operators the highest flexibility and efficiency with the lowest attendant cost. However, there are significant challenges in enabling spontaneous roaming agreements of heterogeneous networks. First, there are significant challenges in establishing roaming agreements that fulfill all of the terms of current paper roaming agreements in an automatic yet efficient manner in (near) real time. Second, access to local resources still need to respect each organization's access, billing, administration, and other policies.

We propose to address these issues by adding a new module in existing AAA (Authentication, Authorization and Accounting) architectures to handle policy based negotiation for spontaneous roaming agreement establishment. Our paper makes the following contributions.

- We propose a novel AAA architecture with a Partnership Management Module to enable policy based negotiation for defining spontaneous and dynamic roaming agreements.
- We propose methods and models for basic trust establishment between providers for spontaneous inter-working.
- We design a new user entry and authentication process (i.e., a modified EAP_AAA process) at a unknown foreign network for spontaneous interworking with the home network with minimized changes to existing AAA processes.
- We specify policies and policy based negotiation processes for negotiating specific QoS, security, pricing, and other per-session parameters.
- We design a new Diameter application, called PMA (Partnership Management Application), for supporting spontaneous roaming agreements.
- We propose mechanisms to optimize the performance of establishing spontaneous roaming agreements.

The remainder of the paper is structured as follows. Section 2 describes the overall framework of our proposed AAA architecture. Section 3 presents the detailed Diameter PMA application design. Section 4 defines the policies for inter-working with unknown networks and presents a detailed policy based negotiation process. Section 5 discusses related work, and finally in section 6, we conclude the paper and outline our future work.

2 Architectural Design of AAA for Spontaneous Roaming Agreement

2.1 AAA for Spontaneous Roaming Agreement Architecture

Nowadays heterogeneous wireless networks are converging to provide IP services. The standard AAA architecture for cellular networks interworking with WLAN/WiMax is EAP with backend RADIUS or DIAMETER AAA server. To enable spontaneous roaming agreements, the new AAA architecture is illustrated in the following figure.

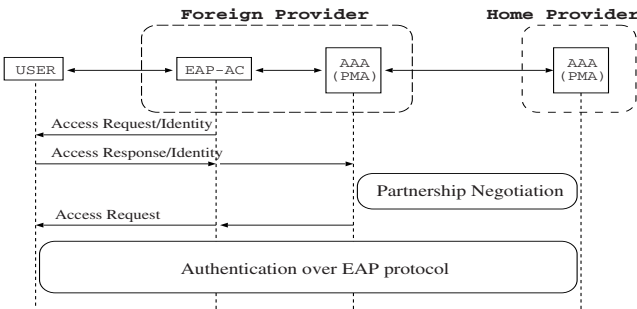


Fig. 1. New AAA Architecture with Partnership Negotiation

In this figure, the user is a subscriber of the home provider and the foreign provider is unknown to the home provider. EAP-AC¹ (Extensible Authentication Protocol [4] Access Controller) is used to process EAP messages. In the above AAA framework, there is a new module called the Partnership Management Application (PMA) that enables two providers to conduct policy-based negotiation. As illustrated above, the EAP-AC passes the identity response to AAA of the foreign provider. When it is determined that the user belongs to an unknown home network, the local AAA starts the PMA module for policy based negotiation. When dealing with unknown providers, the partnership negotiation must first succeed before the normal authentication process for inter-working can start.

2.2 Trust Establishment Between Providers

Without prior agreements, establishing trust among providers is the driving factor for inter-working. Without trust, there is no guarantee that services will be honored and paid for. In addition, the performance can be a huge issue for starting negotiation from scratch. In today's world, the problem with negotiating everything on paper roaming agreements (even through on-line version, such as through secure web services) is that this process takes too long to be usable for users that want immediate, on-demand services, and in any case two providers have to have a basic trust to start with for negotiation. We propose the following possible trust models for two providers to establish a basic level of trust as a starting point for further focused and speedy negotiation.

- **Consortium model:** Providers join a consortium in which they all agree upon a basic set of agreements on common offerings, such as liability, customer care, basic security, minimum and maximum charging, and basic QoS. Members of the consortium are issued certificates, so that providers carrying a consortium-issued certificate are trusted by other consortium members that their subscribers will pay for the service as in the basic agreement.
- **Third Party Certification Model:** An independent trusted third party evaluates different providers, giving them a certificate with a relative score. A provider can check another provider's score on the fly through the third party to determine the trustworthiness of an unknown provider.
- **Transitive Trust Model:** Participating providers build a set of established trusts between them; using transitive trust, providers can derive additional trust relationships as needed.

In all three models, some sort of certificate verification will suffice to establish a basic trust between different providers.

The above models present alternative trust models that providers can match their own specific AAA and security requirements to, enabling them to establish a basic level of trust as a starting point. Among the three models, the first model is considered the most practical, and thus we will focus on the first model in this paper.

¹ EAP-AC is either the native layer-2 EAP entity like Access Point (AP) or a special entity for processing EAP over IP PANA authentication traffic. See section 2.6 use case for further explanation.

In the consortium model, different providers have joined one consortium, say consortium X. Consortium X has a Master roaming agreement that includes a dispute settlement procedure, limitation of liability, billing procedure and responsibilities, customer care responsibilities, fraud tools and processes, agreement suspension and termination, minimum maximum charge of airtime or wholesale rate, and other required features. Members of X agreed on the above basic requirements and X issues a certificate to its members. This enables all members of X to identify each other, and hence establish trust.

If a member of X encounters an unknown provider, the two providers need to establish a basic level of trust (such as that provided by X to its members) to ensure that the new partner will fulfill its responsibilities and liabilities. The member of X can either request the unknown provider to join consortium X (which will then enable trust to be provided through certificate verification) or the member of X can use policy to decide if trust negotiation should be initiated or not. If trust is not established, then no inter-working will be possible; otherwise, if trust is established, on-line negotiation can be performed to define specific per-session requirements (such as QoS, security, and pricing) to finalize their partnership agreement.

The consortium model can also be extended to a multiple consortium case with cross-certification. For example, a group of GSM providers is one consortium, and a group of WiMax providers is another consortium. If two different consortiums have issued cross-certifications, then members of two consortiums will be able to verify each other and establish a basic trust between them. This model has the advantage that providers can keep their existing membership without having to join new consortiums.

2.3 Inter-provider Policy-Based Negotiation for Spontaneous Roaming Agreement

2.3.1 Policies

Before conducting the negotiation, each provider prepares and specifies two different sets of policies – one for working with known providers, and another for working with unknown providers. The policies for working with unknown providers include at least the following functions:

- Foreign Provider's policy in providing service to non-subscribers
 - Home Provider trust policy: the certification and qualification of the non-subscriber's Home Provider that the Foreign Provider can trust
 - Non-subscriber's identification, authentication, and authorization policies
 - Other policies governing per-session features for the non-subscriber, such as QoS, security, and billing settings
- Home Provider's policy for subscribers accessing unknown Foreign Providers
 - Foreign Provider trust and qualification policies
 - Subscriber's identification, authentication, and authorization policies
 - Other policies governing per-session features for the non-subscriber, such as QoS, security, and billing

2.3.2 Inter-provider Negotiation Overview

To enable spontaneous inter-working, the Foreign Provider and the Home Provider negotiate to achieve the following:

- **Establish Secure Channel:** Two providers will first establish a secure channel to protect their negotiation. For example, they use IPSec tunnel with consortium-issued certificates for mutual authentication.
- **Establish Business Trust:** Two providers exchange qualification related info to establish business trust. With the mutual trust, the Foreign Provider ensures that the service will get paid and the Home Provider ensures that the Foreign Provider is a legitimate and trusted partner...
- **Agree on Session Profile:** The two providers negotiate and agree on per-session features, such as what type of QoS is provided for which services.
- **Agree on Session Security:** The two providers negotiate and agree on methods for identification, authentication, and authorization, as well as for mechanisms for protecting user traffic.
- **Agree on Billing:** The two providers negotiate and agree on pricing and other billing related features.

To achieve the above goals, two providers will first exchange consortium identities and find a common consortium. Then, the two providers will use the consortium certificate to authenticate each other and establish an IPSec [5,6] tunnel to protect their further negotiation traffic. Once the basic level of trust and the secure tunnel for negotiation are established, they will focus on specific features, such as QoS, security and pricing in the negotiation. The negotiation can be done using a simple request/response protocol in the new PMA application.

2.4 Performance Optimization for Spontaneous Roaming Agreement Negotiation

The following performance optimization techniques are adopted.

- Once the basic trust and security tunnel for negotiation are established, the negotiation on QoS, security and other functions can be done in parallel.
- Subscribers of a Home Provider can be categorized into groups (e.g., Gold vs. Silver vs. Bronze classes), and one negotiation result can be reused many times in other sessions for the same user class.
- Similarly, a past negotiation result can be either suggested to the user or group or automatically reused, if desired
- Latency at handoff between providers can be critical. However, negotiation can be done at the pre-authentication phase while still connecting to the current network for seamless handoff.

3 DIAMETER AAA Framework for Spontaneous Roaming Agreement Establishment

In this section we describe the enhancements to existing AAA frameworks for establishing spontaneous and dynamic roaming agreements. To facilitate the formation of

dynamic roaming agreements, existing AAA frameworks need to be upgraded with the new PMA module (Partnership Management Application). This requires appropriate interfaces to be added to the PMA application, so that it can be integrated with the existing AAA and L2 authentication protocols (e.g., EAP). A summary of the new additions to existing AAA frameworks is listed below.

- New PMA module in AAA servers
- Related impacts to the AAA messaging
- Changes to EAP messaging
- Changes to User device

One of our design goals is to minimize the changes to existing AAA infrastructure, although some changes are inevitable. In the following subsections, we will discuss these changes individually.

3.1 PMA (Partnership Management Application)

To facilitate the formation of spontaneous roaming agreements, existing AAA frameworks will have to be enhanced with the PMA module. The PMA is an AAA application that performs the negotiation portion of the roaming agreement. This application provides a framework for the negotiation and also specifies the roaming agreement parameters to be negotiated between the two operators. The PMA module can be implemented either as a DIAMETER[15] application or as middleware interfacing with the AAA server. Since the PMA is a policy defining entity for the access network, it is a good design option to integrate it with the AAA framework by building the PMA module as a Diameter application for Diameter server. For an AAA server using the RADIUS protocol, the middleware is the only option. While we focus on DIAMETER application design in this paper, the design of the middleware for RADIUS will be similar.

We define four main messages: CRR (Credential Request), CRA (Credential Answer), NIR (Negotiation Information Request) and NIA (Negotiation Information Answer). The new AVPs we define for PMA application include type of negotiation, trusted CA IDs, proposed price, data rate, security algorithm etc. More attributes can be easily added to support negotiation of other features. One of the advantages that our new framework has is to support providers to negotiate only issues that they care about the most and skip other issues that have been specified in the Master roaming agreement, which enables both dynamics and fast performance in roaming agreement establishment. We omit detail here for respect of page limit.

3.2 Changes to Standard AAA Server

The new AAA server will be different from the traditional AAA server, exhibiting the following new behaviors:

- The AAA server has a new PMA application or module, and communicates with a policy system to learn the appropriate policies to use for a given situation.
- The foreign AAA server will start the PMA upon a request from a non-subscriber.

- Upon completion of the PMA negotiation process, the foreign AAA will send the negotiation result (e.g. authentication method) to EAP-AC, which then relays the result to the MS (Mobile Station/Device).
- If the negotiation is a success, normal AAA process to the home AAA will start. Otherwise, a “negotiation failure” error message will be communicated to the user and the user is disconnected.

3.3 Other Changes

- **Changes to EAP Messages:** Similarly, the EAP message needs to be extended to communicate the negotiation result back to the MS. The negotiation result may contain 1) identification, authentication and authorization methods, 2) other per-session features, such as QoS and billing. The Diameter EAP messages can be found in RFC4072.
- **Changes to User Device:** The entire negotiation process is almost transparent for the users. However, the user device is required to be equipped with EAP client capability if it has not already done so. Other than that, the only change to the user device is the added capability to process the EAP negotiation result message and to start the authentication process after a successful negotiation.

4 Policy Engine for Spontaneous Roaming Agreements

Foreign and Home Providers use policies to govern the negotiation on what Providers they will establish roaming agreements with as well as the per-session features that each Provider will support. In this section, we explore various policies required for Providers to allow spontaneous access.

The Foreign Provider needs a policy that defines the requirements for a non-subscriber's access and its restrictions (called a *non-subscriber policy*). In contrast, the Home Provider needs a policy that defines the requirements for a subscriber's access to outside services (called a *foreign-access policy*). In this section, we discuss each type of policy and present some examples.

We use [14] as a source for the following formal definitions. **Policy** is a set of rules that are used to manage and control the changing and/or maintaining of the state of one or more managed objects. A **Policy Rule** is an intelligent container. It contains data that define how the Policy Rule is used in a managed environment as well as a specification of behavior that dictates how the managed entities that it applies to will interact. The contained data is of four types: (1) data and metadata that define the semantics and behavior of the policy rule and the behavior that it imposes on the rest of the system, (2) a set of events that can be used to trigger the evaluation of the condition clause of a policy rule, (3) an aggregated set of policy conditions, and (4) an aggregated set of policy actions. For flexibility, the DEN-ng model defines three clauses (a Policy Event clause, a Policy Condition clause, and a Policy Action clause) that aggregate individual and groups of **Policy Events**, **Policy Conditions**, and **Policy Actions**. Each of these three clauses are treated as atomic objects that are in turn aggregated by a Policy Rule. A Policy Event defines the necessary occurrence or combination of occurrences that are used to trigger the evaluation of the Policy Condition

clause. A Policy Condition defines the necessary state and/or prerequisites that define whether the actions aggregated by that same Policy Rule should be performed. This is signified when the Policy Condition clause associated with a Policy Rule evaluates to TRUE. (Note that in the DEN-ng policy language, an alternative set of Policy Actions can be defined that are executed when the Policy Condition clause evaluates to FALSE.) A Policy Action defines the necessary actions that should be performed if the Policy Condition clause evaluates to TRUE.

Most importantly, the effect of the Policy Action clause is to apply a set of actions to a set of managed objects, and have the effect of either **maintaining an existing state**, or **transitioning to a new state**, of that set of managed objects.

We have designed our policy system as a set of reusable components and built a prototype policy implementation. We have to omit details here due to page limit.

5 Related Work

We have talked about major related work in the introduction section. For respect of page limit, we briefly discuss related work here. First, brokered roaming agreement model [1, 2] is being deployed to reduce burdens of bilateral agreements. Comparing with them, our proposed system offers a more efficient, low cost, and dynamic solution to roaming. To overcome the limitations of the brokered roaming model, Ambient networks project [3] proposed mechanisms for automatically establishing bilateral roaming agreements directly between operators. However, this type of pre-determined roaming agreements are relatively fixed and can't be dynamically adapted in different conditions. Current inter-working related AAA work assumes the use of pre-established roaming agreements. [7, 8, 9]. Research on spontaneous access is, to date, mostly devoted to access control models [12, 13] without authentication architecture.

6 Conclusion

We presented a novel AAA architecture for heterogeneous providers to work together spontaneously and securely without pre-established formal roaming agreement. Spontaneous and dynamic roaming agreements are established through policy based negotiation. Building upon basic agreements established at a consortium(s), policy based negotiation for spontaneous roaming agreement is conducted upon user request and is seamlessly integrated into user association and authentication process. The online negotiation focuses on the issues that providers care about the most and can be done quickly with performance optimization techniques. Furthermore, we designed a new Diameter application to handle the negotiation for spontaneous roaming agreement, and we also designed policy language and policy based negotiation process. Work is currently in progress to prototype the system, and refine the proposed model.

References

1. Weroam service. <http://www.weroam.com>
2. Comfone service. http://www.comfone.com/_main_pages/services/broker/key2roam.htm

3. Ambient Networks Security Architecture document at http://www.ambientnetworks.org/phase1web/publications/D7_2_Ambient_Network_Security_Architecture_PU.pdf
4. Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., Levkowetz, H.: Extensible Authentication Protocol (EAP). RFC 3748 (2004)
5. IPsec information may be found here, <http://www.ietf.org/html.charters/OLD/ipsec-charter.html>
6. IETF, Internet Key Exchange (IKEv2) Protocol, RFC4306
7. Kim, H., Ben-Ameur, W., Afifi, H.: Toward Efficient Mobile Authentication in Wireless Inter-domain. In: Proceedings of IEEE ASWN (Applications and Services in Wireless Networks), Berne, Switzerland (2003)
8. Meyer, U., Cordasco, J., Wetzel, S.: An Approach to Enhance Inter-Provider Roaming Through Secret Sharing and Its Application to WLANs. WMASH 2003, Germany (2003)
9. Salkintzis, K., et al.: WLAN-GPRS Integration for Next-Generation Mobile Data Networks. IEEE Wireless Communications (2002)
10. 3GPP TS 23.234, v2.4.0. 3GPP system to Wireless Local Area Network (WLAN) Interworking. System Description (Release 6) (2004)
11. 3GPP TS 33.234, v1.0.0, Wireless Local Area Network (WLAN) Interworking Security (Release 6) (2003)
12. Liscano, R., Wang, K.: A SIP-based Architecture model for Contextual Coalition Access Control for Ubiquitous Computing. Mobiquitous 2005 (2005)
13. Cohen, E., Thomas, R.K., Winsborough, W., Shands, D.: Models for Coalition-based Access Control (CBAC). SACMAT 2002 (2002)
14. Strassner, J.: Policy Based Network Management. Morgan Kaufman Publishers, Seattle (2003)
15. Calhoun, J., et al.: RFC 3588 - Diameter Base Protocol. <http://www.faqs.org/rfcs/rfc3588.html>

A Prediction-Based Fair Replication Algorithm in Structured P2P Systems

Xianshu Zhu¹, Dafang Zhang¹, Wenjia Li², and Kun Huang¹

¹ College of Computer & Communication, Hunan University, Changsha,
Hunan 410082 China

`zhu_xianshu@hotmail.com, {dfzhang, huangkun}@hnu.cn`

² Department of Computer Science & Electrical Engineering, University of Maryland
Baltimore County, Baltimore MD 21250 USA
`wenjia1@cs.umbc.edu`

Abstract. Highly skewed query distribution in structured Peer-to-Peer system may cause huge amount of dropped queries and consequently lead to poor system performance. This paper describes a Prediction-based Fair Replication Algorithm (PFR), which aims to maintain excellent system performance when the query is highly skewed. For the purpose of fairly distributing load onto each node, nodes that host hot items always shed load onto light-loaded nodes by creating replicas along the query path. Through the use of a simple prediction method, we can foresee traffic surge and replicate beforehand. Consequently, the number of dropped queries will decrease. Further more, each node can fairly decide the load redistribution speed for itself merely based on local information. The experimental evaluation demonstrates the effectiveness of our approach, which can simultaneously reduce the number of dropped queries as well as the number of created replicas without introducing unaffordable overhead.

1 Introduction

P2P networks, such as Gnutella [1] Freenet [2] Chord [3] CAN [4] and Pastry [5], have been widely used in recent years. The implementation of structured P2P network assumes that all data items are of the same popularity. However, the distribution of queries for real data items has shown to be highly skewed, with several popular objects being requested most of the time. This type of traffic may overwhelm not only the source nodes that host the frequently-accessed data items but also the nodes along the busy query path. When flash crowd [11] happens, the amount of requests for the popular objects can increase dramatically to tens or hundreds of times as compared with the original amount, which is far beyond the node capacity. Such nodes may suffer from severe performance failures, and almost all the services they provide will be unavailable. Therefore, a poor system performance can be expected if the solution can't be found.

The common method of balancing load is to distribute replicas of the popular data items to various nodes, by which we can help the overloaded source nodes shed load. And thus the system performance is enhanced. In this paper, we propose a Prediction-based Fair Replication Algorithm (PFR), which can mitigate flash crowd symptoms

with low overhead. To distribute load fairly onto each node, the PFR algorithm creates replicas at proper time, with proper speed, and at proper location:

(1) Appropriate replication time. By employing a simple prediction method, we can recognize traffic surge ahead of time. Thus, replicas can be scattered before flash crowd happens.

(2) Fairly-decided replication speed. In this paper, Replication Speed (RS) can be measured by the ratio of the number of nodes chosen to hold replicas to the number of all nodes that have encountered along the query path. The higher the RS, the faster the node's load will be redistributed. We fairly set optimized RS for each node according to its predicted load.

(3) Fairly-determined replica location. Replicas should always be put on the light-loaded nodes, which can avoid heavy-loaded node taking the responsibility of shedding load and becoming overloaded consequently.

Furthermore, in our PFR algorithm we employ the replica location dissemination method proposed by LAR [6]. Soft states that contain replica locations are disseminated by piggybacking on existing messages. When combined with load prediction method, our PFR algorithm can help replicas be efficiently utilized in shedding load. In a word, PFR algorithm is a lightweight algorithm for its optimized replication strategies and high replica utilization rate.

The paper is organized as follows. In Section 2 the related replication methods in P2P networks will be summarized. Then, we will describe our algorithm in details in Section 3. In Section 4 we present our simulation results. Finally, the conclusion and future work will be discussed in Section 5.

2 Related Work

A great deal of the previous work used replication and caching techniques to balance load and dissipated flash crowd. According to the classification from [13], all the current replication methods in P2P systems can be divided into three different categories. They will be discussed respectively in more details below.

1. Path replication strategy: Objects are replicated on every node along the query path. [7] proposes DHash, which caching data items on all nodes on the query path. Hot data items are quickly replicated throughout the network. In this way, DHash can respond quickly to the unexpected changes in data popularity. However, it performs poorly under moderate load because of their high overhead. Furthermore, without considering the actual load on each node, this blind replication strategy obviously violates the fairness goal of the replication method.

2. Owner replication: Only the node that originates the request keeps the copy. It always keeps its RS as 1. V. Gopalakrishnan et al. [6] proposes LAR algorithm, in which replication process is triggered whenever a predefined threshold is reached. The LAR algorithm cannot well achieve the goal of fairness for its two drawbacks: On one hand, the node which originates the request can easily become overloaded because all nodes along the query path will intend to shed load onto it. On the other hand, for the low replication speed that only one replica can be created for one query process, LAR algorithm cannot achieve satisfactory performance when flash crowd happens. Our

method gets a trade-off between LAR and DHash. We adaptively adjust the number of replicas created for each query process according to the node's load status.

3. Random replication: We create replicas on randomly chosen nodes along the query path, which is more flexible than the other two strategies that have been discussed above. Our method can be characterized as a kind of load-biased random replication, which creates replicas on light-loaded nodes with higher probability.

Some proactive web caching methods [9, 10] are proposed to address flash crowd phenomena. By predicting the arrival of flash crowd, they can take preventive actions proactively. These kind of proactive methods work well if sufficient replicas are created before hotspot really happens. Our method can anticipate traffic surge and create replicas in advance as well.

3 Prediction-Based Fair Replication Algorithm

The basic idea of the PFR algorithm can be summarized as follows: We create replicas for the node whose predicted load or current load reaches certain predefined threshold and adaptively adjust the RS for each replication process. The higher the threshold level, the faster the load should be redistributed, which means higher RS. For the purpose of distributing load fairly onto each node, replicas are always created on light-loaded nodes. There are three specific issues that the PFR algorithm must be addressed:

1. Load prediction. In order to keep the system performance at a high level when flash crowd happens, preventive actions should be taken in advance. Therefore, we make use of prediction method to do proactive replication.
2. Replica creation. We need to specify three aspects of the replica creation strategy: the time, the number and the location.
3. Replica location dissemination. We use the same replica location dissemination policy as LAR [6]. Lightweight hints effectively direct queries towards new replicas. The more nodes know the location of replicas, the better replicas can be utilized to shed load.

In the rest of this section, we will describe the first two issues in detail. We assume that the system has the following two characteristics:

1. Stability: no node joins or leaves the system.
2. Homogeneity: same characteristics for all nodes (CPU, storage size).

3.1 Prediction Algorithm

Given a time period T (such as 1 seconds), we use the Period Exponential Weight Prediction Algorithm (PEWP) to predict the query Average Inter-Arrival Time (AIAT) at each node in the next time interval ($AIAT_{n+1}$). PEWP is specifically described as follows:

We setup a query list for each node, which records the number of queries received in the past N (such as $N=5$) time periods respectively. $AIAT_n$ denotes the AIAT in the n th time interval. Then, the predicted $AIAT_{n+1}$ can be described in Equation (1):

$$\text{Pr } eAIAT_{n+1} = AIAT_n + PI(n+1) \quad (1)$$

In Equation (1), $PI(n+1)$ denotes the predicted AIAT difference between the $(n+1)$ th time interval and the n th time interval. We compute the predicted $PI(n+1)$ by using an iterative exponential weighted method, which is similar to [9]:

$$PI(n+1) = (1 - \alpha) * PI(n) + \alpha * (AIAT_n - AIAT_{n-1}) \quad (2)$$

The constant α is a smoothing factor, and the value of α is set to be 0.125. By continuous iterating, $PI(n+1)$ can be expressed as the linear combination of the previous n PIs, in which the weight corresponding to $PI(x)$ will be gradually higher if x gradually approaches to $(n+1)$. PEWP Algorithm only incurs low computation overhead, so it is applicable to online prediction.

In this paper, we define node's capacity as the number of queries a node can route or handle per second (C). We use queue length to specify the number of queries that can be buffered until additional capacity is available. If the average query arrival rate is higher than the node's capacity, excessive queries are queued in the node's input buffer (Queue). When the sum of the queued queries and $1/\text{Pr } eAIAT_{n+1}$ is larger than node's capacity, the node would become overloaded in the next time interval. As a result, node's predicted load fraction can be computed by Equation (3):

$$\text{Pr } eLoad = \frac{1/\text{Pr } eAIAT_{n+1} + \text{Queue}}{C} \quad (3)$$

3.2 Fair Replication Strategy

Our replication strategy can be described as a load-biased random path replication method that aims to achieve goal of fairness during the load rebalance process. In the case of flash crowd, a better solution is to redistribute load for the hotspot node at the highest speed that can be reached. However, when the load is in a moderate level, replication is necessary but not of urgent need. In this case, it will be unwise to create replicas at high speed because of high replication overhead. Consequently, for each node we define five thresholds based on the load fractions of node's capacity to achieve different RS, as described in Fig. 1.

As Fig. 1 shows, threshold L^{hi} indicates that a node is approaching its capacity and it is in an emergency state to shed load in order to prevent itself from overloading and consequent dropping queries. When the node's load fraction is between the L^{hi} and L^{low} thresholds, replication is necessary only when the difference between the load of two nodes is greater than the L^{low} . With the decrease of threshold, its corresponding emergency level is decreasing accordingly. Replication process should not be triggered only when the node's load is below the L^{low} threshold.

Through the use of multiple thresholds, we can dynamically adjust the value of RS, which may consequently achieve high replication efficiency. This method is much better than the LAR algorithm, in which load can be merely shed to one node along

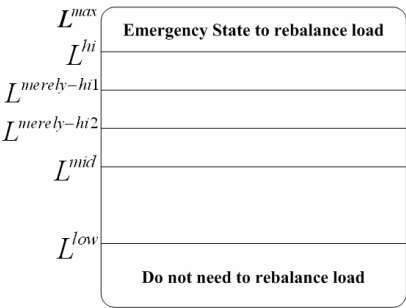


Fig. 1. Thresholds according to node’s load fraction

one query path without discriminating the heavy-loaded node and moderate-loaded node. PFR replication strategy is described in more details below.

Set the length of the query path be N . When query packet is routed through node S_i , we compute and piggyback its predicted load on the query packet. (According to the highly efficient routing protocols in structured P2P networks, the maximum hops to reach the destination node should be $\log n$, where n is the total number of nodes in structured P2P networks. Thus, the storage overhead for the piggybacking load information can be affordable.) At the same time, node S_i checks the value of the predicted load $Pr eLoad$ to determine whether load rebalancing is necessary or not. With respect to nodes' predicted load fraction, we set 5 different replication levels (show in Table 1), which specify the RS for each query. Whenever stored replicas reach the node’s maximum storage size, new replicas will replace the old replicas using a Least-Recently-Used (LRU) algorithm.

Table 1. Replication Level

Level	Load Range	RS
1	$Pr eLoad \geq L^{hi}$	N
2	$L^{merely-hi1} \leq Pr eLoad < L^{hi}$	3N/4
3	$L^{merely-hi2} \leq Pr eLoad < L^{merely-hi1}$	N/2
4	$L^{mid} \leq Pr eLoad < L^{merely-hi2}$	N/4
5	$L^{low} \leq Pr eLoad < L^{mid}$	1

In table 1, the column “RS” describes the replication speed. For instance, 3N/4 means that load should be redistributed to 3/4 amount of the total number of nodes along the query path. There are two types of nodes along a query path: query’s destination node and the nodes just forwarding queries. The replication strategy for the first type of node is strictly followed Table 1. However, for the second type of node, replication is necessary only when its predicted load has reached the L^{hi} threshold. Simulation shows a better result if we discriminate these two types of nodes. When the replication is triggered, we will attempt to create replicas of the n heaviest-loaded

items on each node selected, such that the sum of the local loads caused by these n items will be greater than or equal to the difference in loads between the two nodes.

Given the possible inaccuracy of the prediction, our replication strategy is not only determined by node's forecasted load fraction but also determined by node's current load fraction. When replication is not necessary according to the prediction value, we can further check whether the replication is necessary or not based on the node's current load fraction: high current load fraction may indicate that the load will be high in the near future with high possibility and the load redistribution is necessary. In this case, the corresponding replication level should be decreased by 1 accordingly. We will explain it in the example showed in Fig. 2.

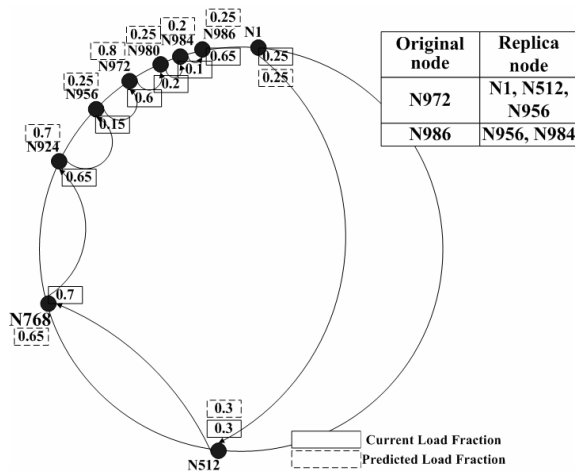


Fig. 2. The Fair Replication Strategy in Chord

There is an important point in replication strategy: how to choose replica location along the query path. Here are two rules that should be complied with when we choose the replica location: first of all, replicas should be put on light-loaded nodes and specifically these nodes' loads should be lower than the originated node by a certain fixed value, since a light-loaded node is less likely to become overloaded as a result of shedding load on it. Secondly, it is obviously unfair for the node with the lightest load if all the nodes along the query path are shedding load on it. We adopt the load-biased random chosen method in our algorithm: the lighter the current load on the node, the higher the probability of creating replicas on it.

As it is shown in Fig. 2, in a Chord ring with 1000 nodes, Node N1 originates a request to find file with id 987, which is stored on N986. According to Chord routing protocol, the query path is {N1, N512, N768, N924, N956, N972, N980, N984, N986}. We set the five thresholds as: $L^{hi} = 0.8$, $L^{merely-hi1} = 0.7$, $L^{merely-hi2} = 0.6$, $L^{mid} = 0.5$, $L^{low} = 0.3$. When the query is routed through N972, its predicted load fraction is 0.8, which has reached the L^{hi} threshold. There are five previous nodes that have encountered along the query path. We choose N1, N512, and N956 and create replicas on them, since their load is all lower than N972's load by some fixed value. When

routed through N986, its predicted load fraction is 0.25. So there is no need to create replicas. However, replication is necessary according to the current load fraction 0.65, which is within the load range in replication level 3. As we have discussed above, the replication level should be decreased by 1 accordingly. Then, the ultimate replication level for N986 should be level 4, which indicates that load should be redistributed to two nodes along the query path. N1, N512, N956, N980 and N984 can be the possible candidates to help N986 shed load. We randomly choose N984 and N956 from them to create replicas on them.

4 Performance Evaluation

In this paper, we compare the performance of PFR with LAR in networks where query distribution is extremely skewed. Our performance results are based on a significantly modified simulator used in Chord project [12]. When we adapt PFR to Chord, finger list is the default item of replication. We replicate the data item only if the load on the node caused by the actual data item is more than that caused by the finger list. The default system parameters are shown in Table 2.

Table 2. Default simulation parameters

System size	1000 nodes	Average system load	25%
Number of data	32767	Node's capacity	10 per sec
Number of queries	250K	Node's queue length	25

4.1 Performance on Different Query Distributions

From Fig. 3 to Fig. 7, we compare the two replication algorithms under three different query distributions. In the first two of the three experiments, the first 100 seconds of input are uniform, and then the query distribution changes suddenly, which is 90% of the input queries are directed to 1 item, or to 0.1% (32) items respectively (and the rest 10% queries uniformly distributed over all items). For the third experiment, queries follow Zipf-like distribution with parameter $\alpha = 1$ [8].

In Fig. 3, we plot the number of dropped queries for every 10 seconds. We can find that, no matter what the query distribution is, PFR drops much less queries than LAR at the beginning when query distribution becomes highly skewed. From the Fig. 4, compared with LAR, the total number of queries dropped for PFR is decreased by 60%, 78%, and 46% when queries follow 90% $\rightarrow$ 0.1%, 90% $\rightarrow$ 1 and Zipf distribution, respectively. Generally, when it comes to the dropped packets, PFR can achieve a much better performance.

Fig. 5 - Fig. 7 show the overhead comparison results between PFR and LAR. The number of documents replicated for PFR is decreased by 12%, 57%, and 46% when queries follow 90% $\rightarrow$ 0.1%, 90% $\rightarrow$ 1 and Zipf distribution, respectively. And, the number of finger list replicated for PFR is decreased by 37%, 71%, 52%, respectively. However, as for replica location hints, PFR creates more hints than LAR, which is 49% 65% 46% more than LAR respectively. Since the overhead for creating a

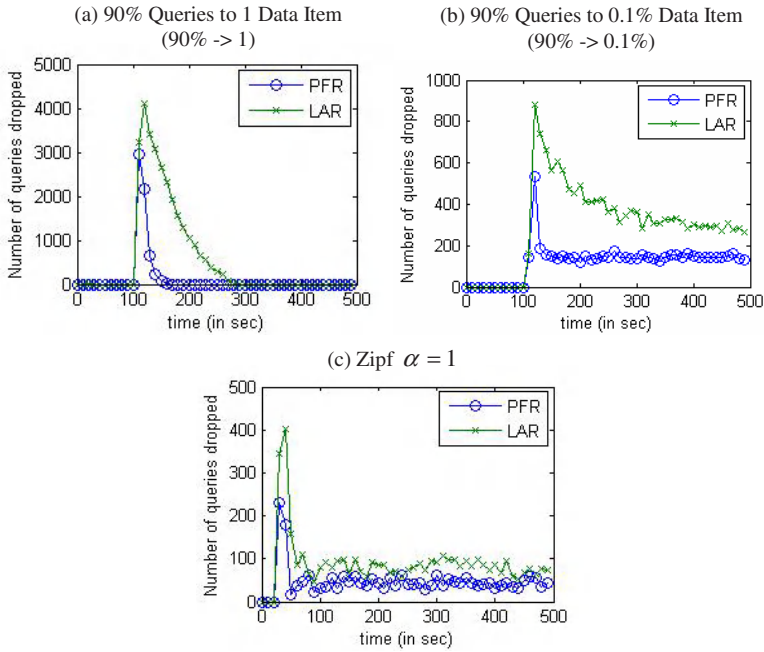


Fig. 3. Number of queries dropped over time for different query

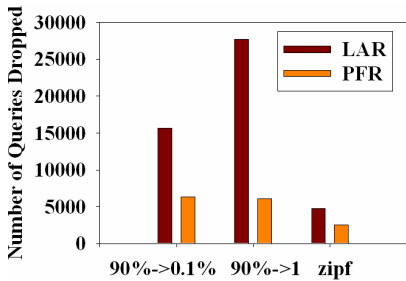


Fig. 4. Total number of dropped queries

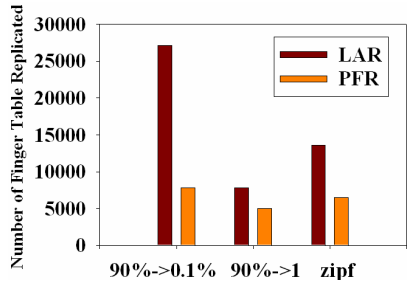


Fig. 5. Total number of Finger Tables replicated

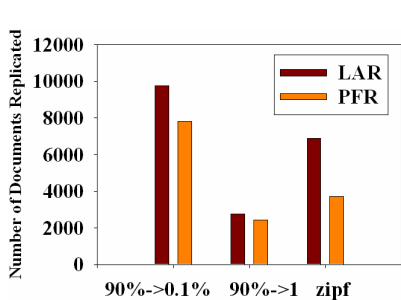


Fig. 6. Total number of documents Replicated

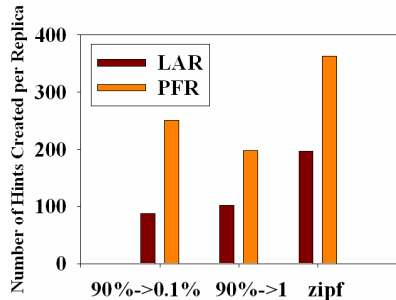


Fig. 7. Total number of routing hints created

replica is much larger than the overhead for creating a replica location hint. Therefore, the result shows that, PFR incurs much lower overhead than LAR. In a word, PFR can largely decrease the number of dropped queries with low overhead.

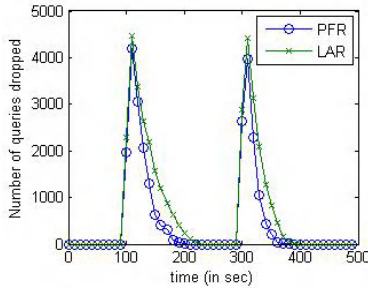


Fig. 8. Number of queries dropped over time when hotspot changed

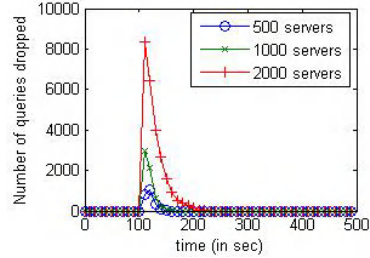


Fig. 9. Number of queries dropped overtime for various system sizes

4.2 Performance in Change of Hotspots

Fig. 8 shows how PFR reacts to changes in hotspots over 500 seconds. For this experiment, 90% of the queries have a single item as their common target. In the first 100 seconds the queries follow a uniform distribution, and in the following 400 seconds, there is a change to the hot item every 200 seconds. We can see that PFR can adjust quickly to these changes, and the replication process is adapted to the changes in hot data item. Thus we can conclude that PFR is robust against drastic changes in input distribution over very short time scales.

4.3 Scalability

For the experiments displayed in Fig. 9, 90% of the queries have a single item as their common target. We change the system size to 500, 1000 and 2000 nodes respectively. We adjust the query stream to keep the average load for systems with different size as 25%. We can find that more queries dropped at the beginning when the query distribution becomes skew for the larger system size. However, the system with larger size can quickly stop dropping queries at 200s. As a result, we can safely conclude that PFR has perfect scalability.

5 Conclusion and Future Work

This paper describes a prediction-based fair replication algorithm (PFR). The PFR algorithm can conduct fair replication through dynamically adjusting the replication speed for heavy-loaded and moderate-loaded nodes and randomly choosing light-loaded nodes to be replica nodes. We give the light-loaded nodes along the query path the priority to shed load. The employment of a simple prediction method PEWP, which can help work out the replication levels better for each node more precisely.

The simulation results have shown that our replication algorithm can achieve a satisfactory performance when query distribution is highly skewed: it can notably decrease the number of dropped queries with a relatively small overhead.

The main challenge to the DHT design is node heterogeneity. As a result, our future work includes taking node heterogeneity into consideration, analyzing the possible impact that has been brought by heterogeneity and further improving our replication algorithm.

References

1. Gnutella. <http://www.gnutella.com/>
2. Clarke, I., Sandberg, O., Wiley, B., et al.: Freenet: A distributed anonymous information storage and retrieval system. In: Federrath, H. (ed.) *Designing Privacy Enhancing Technologies*. LNCS, vol. 2009, pp. 46–66. Springer, Heidelberg (2001)
3. Stoica, I., Morris, R., Karger, D., et al.: Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications. In: *Proc. of ACM SIGCOMM'01*, pp. 149–160. ACM Press, New York (2001)
4. Ratnasamy, S., Francis, P., Handley, M., et al.: A Scalable Content-Addressable Network. In: *Proc. of ACM SIGCOMM'01*, pp. 161–172. ACM Press, New York (2001)
5. Rowstron, A., Druschel, P.: Pastry: Scalable, Distributed Object Location and Routing for Large-Scale Peer-to-Peer Systems. In: Guerraoui, R. (ed.) *Middleware 2001*. LNCS, vol. 2218, pp. 329–350. Springer, Heidelberg (2001)
6. Gopalakrishnan, V., Silaghi, B., Bhattacharjee, B.: Adaptive replication in peer-to-peer systems. In: *Proc. of 24th ICDCS*, pp. 360–369. IEEE Computer Society, Japan (2004)
7. Dabek, F., Kaashoek, M.F., Karger, D., et al.: Wide-area cooperative storage with CFS. In: *Proc. of the 18th SOSP*, pp. 202–215. ACM Press, Banff (2001)
8. Gupta, A., Dinda, P., Bustamante, F.E.: Distributed popularity indices. In: *Proc. of the annual conference of the Special Interest Group on Data Communication*, ACM Press, Philadelphia (2005)
9. Felber, P., Kaldewey, T., Weiss, S.: Proactive Hot Spot Avoidance for Web Server Dependability. In: *Proc. of 23rd IEEE Symposium on Reliable Distributed Systems*, pp. 309–318. IEEE Computer Society, Switzerland (2004)
10. Zhao, W., Schulzrinne, H.: DotSlash: Handling Web Hotspots at Dynamic Content Web Sites. In: *Proc. of 24th INFCOM*, vol. 4, pp. 2836–2840. IEEE Computer Society, Florida (2005)
11. Adler, S.: The Slashdot effect: An analysis of three Internet publications (2000) <http://ssadler.phy.bnl.gov/adler/SDE/SlashDotEffect.html>
12. <http://www.pdos.lcs.mit.edu/chord> (2001)
13. Lv, Q., Cao, P., Cohen, E., et al.: Search and Replication in Unstructured Peer-to-Peer Networks. In: *Proc. of 16th ACM International Conference on Supercomputing*, vol. 30, p. 258. ACM Press, New York (2002)

TransCom: A Virtual Disk Based Self-management System

Li Wei, Yaoxue Zhang, and Yuezhi Zhou

Tsinghua University, Beijing, China
knight99@mails.tsinghua.edu.cn

Abstract. With the rapid advances in hardware and networks, the current computing systems, including desktop or embedded system for end users and routers for professional operators, is becoming more and more complex and unmanageable. Autonomic computing [6] proposed by IBM is to design computing systems that can manage themselves to reduce managing complexity of the global systems. This paper introduces TransCom, a novel system to eliminate most of machine individual management. It decouples software and data from the underlying hardware by virtualizing and streaming the storage, centralizing software and data at servers while leveraging local machines' CPU and memory resources to accomplish computing tasks. We have implemented such a pilot system that can support Windows and Linux desktop platforms. In this paper, we focus on the scheme that is used to centralize the software and data. We also presented our early experience in e-learning classrooms to demonstrate that its feasibility, efficiency and usable.

1 Introduction

In the last two decades, with the rapid advances in hardware and network, computing systems have given us abundant computation, storage, and communication capacity. Various special-purpose devices such as sensor, hand-helds, wearable, and smart-phone have appeared and the amount of them has increased at an enormous rate. These heterogeneous devices are usually integrated into corporate-wide computing systems by various networks. As systems become more interconnected and diverse, architects are less able to anticipate and design interactions among components, leaving such issues to be dealt with at runtime. Soon systems will become too massive and complex for even the most skilled system integrators to install, configure, optimize, maintain, and merge. And there will be no way to make timely, decisive responses to the rapid stream of changing and conflicting demands. To solve this problem, IBM propose autonomic computing concept, which is inspired by the human autonomic nervous system that handles complexity and uncertainties, and aims at realizing computing systems and applications capable of managing themselves with minimum human intervention.[6]; The essence of the autonomic computing is *self-management*. [6] explains Self-management in four aspects as following: *Self-configuration, Self-optimization, Self-healing, Self-protection*.

We have developed a self-management computing system called TransCom, which aims to liberate users from the burden of administrating desktop systems. In TransCom, desktop systems can be configured, protected and recovery automatically. The core idea of our system is decoupling nearly all of software data and state from underlying hardware. The software and data are centralized on the server, while the computing tasks in our system are carried out with local machines resources. Thus, this model can combine merits of central management of software and data, while leveraging the local cheap and powerful local resources of individual machines. This can reduce system management sharply and can support a more hassle-free and service-oriented computing infrastructure. There are two main mechanisms in TransCom system. One is the OS-independent remote boot method that can boot the computing environment remotely from the central software and data repositories. In this paper, we will mainly present the other scheme based on virtual disk concept to manage and deliver the software and data centrally and show that it is feasible, efficient and usable. Our early experience with TransCom used in university e-learning classrooms confirms the viability for future computing.

The rest of the paper is organized as follows. Section 2 provide a overview of TransCom system. Section 3 describe the scheme used to centralize the software and data and how to deliver them to users. Section 4 shows our early experience. Section 5 and Section 6 are related works and the conclusion.

2 System Overview

TransCom centralizes OSs and applications and delivers them to users on demand. To use TransCom, users power on the machine and then a desktop is available just like a common personal computer. TransCom adopts the conventional client and server architecture, consisting of TransCom servers, delivery network and TransCom clients, as shown in Fig. 1.

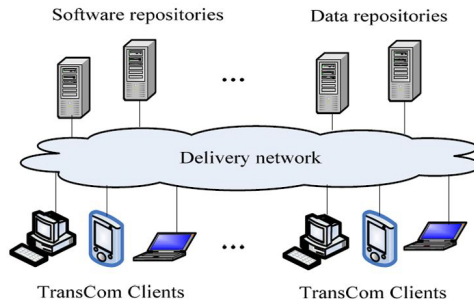


Fig. 1. The architecture of TransCom

TransCom clients are almost bare hardware like fix-function appliances without any local storage of software and data. They requests the software and data just like services from the server repositories. The clients boot remotely and fetch

software and data on demand to run with their local resource, including processor, memory and network. Thus, the clients are just alike to service transceivers. The software, including OSs and applications, and data are delivered to them like an audio or video streaming. The central mechanism of this system is the virtual disk-based management and delivery scheme, which can emulate disks for clients to boot up and run personal computing environment. The contents of virtual disks are stored in the server repositories and thus can be managed centrally.

TransCom servers act as software and data repositories. These repositories hold different images containing different OSs, applications and data, which will be delivered to TransCom clients on demand through the delivery network. The delivery network can be any LAN or high speed WAN environment in the future.[1]

The implemented TransCom pilot system can support both Windows 2000 professional and RedFlag 4.1 Desktop Linux [10] (with a kernel of 2.4.26-1). This system has been deployed in universities e-learning and e-education classrooms and used by students every day for 14 months. Our practical experience is that the system is running stably most of time. Most of system crashes are due to software accidental errors and can be recovery by rebooting, only with a few times due to the service crash down. Due to the central deployment, the system service down time because of deploying new software for education is getting far shorter than past, which has gotten applause from the users.

3 Design Issue

TransCom introduced a smart and flexible, but simple scheme based on the concept of virtual disk (VDisk). On one side, the VDisk emulates exactly the same interfaces as a local hard disk for users and user- end OSs. On the other side, the contents of the VDisk, including the software and data, are actually stored in the server repositories. This make it available to access remote software and data in the same way with accessing a local disk. Because the VDisk works in a block level, it can support the traditional OSs and applications with the least modification and obtain an approximate performance with a local disk. We also use a sharing and isolation mechanism to protect the VDisk from outside threats or attacks. In the following sections, we will describe the detailed design and implementations.

3.1 VDisk and VDisk Images

VDisks are virtual devices for clients. Every VDisk is mapped to a VDisk image in server repositories. The VDisk image holds the VDisk content. It is the basic management unit. This mapping can be illustrated in Fig. 2. The contents of VDisk images are organized in blocks corresponding to VDisk blocks seen by the client machine. While a real disk's parameter will be stored in the machine CMOS memory, the parameter of a virtual disk image is stored in the

image's added block #, which can be queried by the client. The block # of an image contains the following information about emulated disk: total size, block length, CHS(Cylinder/Head/Sectors) parameters. Note the mapping between VDisks and VDisk images is very flexible. This means one image can be shared among different users and one VDisk seen by user with a same drive letter can be mapped to a user-specific image transparently. We will discuss this in detail later.

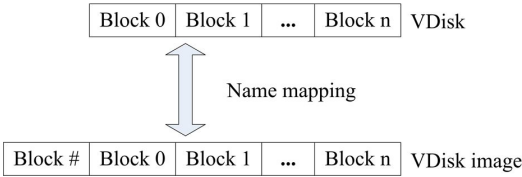


Fig. 2. Mapping between VDisk and image

A VDisk image is created by the administrator as an empty image or as a replication of an existing hard disk with software and data. As a management unit, each image has several management attributes, which are maintained by the server. These attributes are listed as follow: name,type,access mode and access control list. The name of a image is a unique ID that represent the certain image. Type(refer to section 3.2) describe the usage of a image. There are three access mode of a image: privilege, protect and user. The privilege mode is used just for maintaining task, in which a curtain image can be upgraded by the administrator. If an image is in protect mode, it can not be changed by users, and thus can be protected from security threat. The user mode will let a user have all control of the image contents.The access control list tells the system which users can access this image.

The access model of VDisk is shown in Fig.3. There are three components in client structure. The VDisk emulator will instantiate one or multiple disks accessing interface for the machine's BIOS module or the file system of an OS. It will receive the disk requests from them and forward to the service agent if possible. After receiving a request from the VDisk emulator, the service agent will first check its local cache for the requested contents. If yes, it will reply to the emulator instantly; otherwise, it will pack the disk request into a remote service request and redirect it to the remote service handlers in the server. With a request from the client, the service handler will also look up the image cache. If no, it will fetch the needed contents in the image repositories. The management database is consulted when the access control should be enforced. These two caches have different effects. The local cache can reduce the network communication, while the image cache is used to reduce the server's hard disk access times.

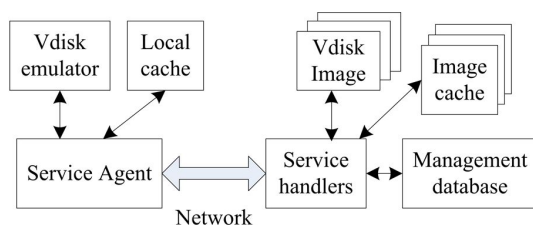


Fig. 3. The VDisk access model

There are two categories of message that exchanged between the service agent and the service handlers.

- Management messages: Management messages are used to query the disk related parameter, deal with the connection setup and user authentication, clear data, etc.
- Data messages: Data messages deliver the data through network. The data message contains the following information: operation type (read or write), data location and data length, error information, etc.

3.2 VDisk Image Types

As described earlier, centralizing software and data in thin clients can centralize the management tasks. However it can not reduce the management of multiple desktop. In order to reduce the system management in TransCom, we divide the VDisk into different categories and isolate them from each other. First, we separate software and data for the fact that most users will use the same software and thus can share them, but the data are often user-specific.

Moreover, we adopt a strategy that the VDisk image containing OS and applications is immutable by users. This will incur another problem, because some applications must write to the disk they reside to function properly. For example, some programs will create temporary files in their residing directory. To solve this problem, we adopt a copy-on-write (COW) mechanism which is illustrated in Fig. 4. There are two types of VDisk(System VDisk (S) and User VDisk (U)) which response to three types of VDisk images(System VDisk Image(SI), User VDisk Image(UI) and Shadow Image(SHI)).

System VDisk image is also called original system image which provides original OS and applications. It is one of the two components that compose System VDisk in client view(the other is shadow image). These images are created by the administrator. Further, in order to save the storage, the system image is shared among users. In addition, to protect the client system from the outside threats or attacks, the system image is immutable for end users.The shadow image is a user-specific COW copy of the system VDisk image. It is a solution the write problem of the immutable system VDisk image. It is a kind of special image that it can not present a VDisk directly. It stores blocks of the system

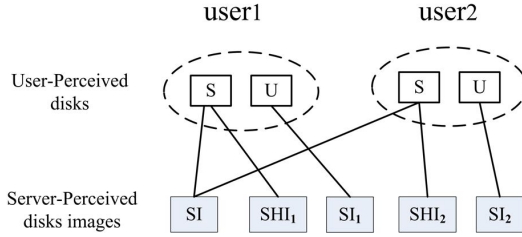


Fig. 4. The sharing and isolation state

VDisk image that is modified at runtime of the client systems. This means that when a client system tries to write a block on the system VDisk, a COW copy block will be created on the shadow image and the following reading or writing operation will be carried out with the COW block. This is a key mechanism of our method to make the system easy to be protected and recovered from disease. We will discuss about protection and healing in more details in the next section. User VDisk images are owned by users to store their private user files or data. They are user-specific images which map to User VDisks of corresponding users.

Note that in most cases, system VDisk images are working under protect mode. However, administrators can switch them to privilege mode, in which reading and writing operations will be done directly to the system images for an update. User VDisk images are always in privilege mode, so that users can read or write to them as needed.

If the client system is compromised by accidental hardware error, software bugs, user errors, or other attacks such as virus, worms and spyware, the shadow images will be discarded and the system will resume to the original status. We will discuss the healing mechanism of transCom in the next section.

3.3 Healing and Protection

As we mentioned above, the healing and protection mechanisms are based on the separation of the system VDisk images and shadow images. OSs and applications stored in system VDisk images are safe because they have been verified by administrators. Programs introduced by users are stored in the shadow image, which can be discarded whenever a fatal error occurs. By discarding the shadow image, the client system will roll back to the original system created by the administrator. The shadow image is a special image whose structure is different from the system and user VDisk images. The structure of the shadow image is shown in Fig. 5. The shadow image consists of three sections: timestamp, block map, and block area. The timestamp records the last modified time of the image. The block map keeps the mapping blocks between the shadow image and the system VDisk image. Blocks in the block area are COW copies of the system VDisk image. The size of the block area increases dynamically when clients try to write to a system VDisk block.

Time stamp	Block Map	Block1	Block2	Block3	...

Fig. 5. The architecture of shadow image

TransCom recovers system from disaster by discarding shadow image, therefore all of private configuration and applications installed by users will be lost. In case of that, we provide a snapshot mechanism to control the version of the system VDisk. The mechanism of the version control is shown in Fig. 6. The System VDisk in client view is comprised by system VDisk image linking with several shadow images. Only one shadow image will be modified when a COW operation occurs. We call it 'current shadow image'(CSHI). The other images are called as 'history shadow image'(HSHI) which keep the previous versions of the system VDisk.

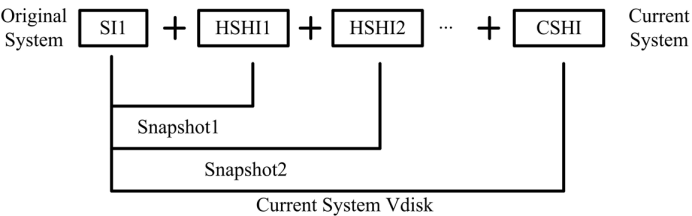


Fig. 6. Version control of system VDisk

There are three operations to manage shadow images as following:

- Create: This operation creates a snapshot of system VDisk. The current shadow image will become history shadow image and a new current shadow image will be created.
- Merge: Several history shadow images can be merged to one image in order to save storage space. During the merge, the latest block recorded by the history shadow images will be stored into the merged image.
- Discard: This operation will discard shadow images whose timestamp is after a certain time specified by users. Therefore desktop will roll back to a previous point when the snapshot is created.

The version control of shadow images makes the recovery of system more flexible and convenient for both users and administrators. Next section we will introduce the implementation of virtual Disk in intel X86. As current desktop OSs work in both real mode and protected mode, our implementation is divide into two part. We modify the BIOS int 13 in real mode and develop OS specified driver, e.g. a SCSI driver in protect mode.

3.4 The Implementation of VDisk

Remote boot by VDisk. In order to boot the system, after powered on TransCom clients will first use a remote boot mechanism to load the desired OS environment from the server. This is implemented by a boot agency that burned into the machine's BIOS and will be triggered after the BIOS initiated. As a first step, the boot agency sends a booting discovery message to the server and then obtains an IP address for subsequent network connection. A boot manager in the server maintains a mapping table from the clients MAC address to IP address pools. Given the boot discovery message, the boot manager will send back the corresponding IP address to the client. After setup an IP connection to the server, the TransCom client sends a request to query the available system images lists provided by the system. The list will be displayed at the screen and users can select one of them to boot up.

The boot agency then downloads from the boot manager a Universal OS loader. The Universal OS loader is to setup a BIOS-enabled service agent to communicate with the service handler. The VDisk emulator is implemented as to replace the BIOS hard disk access function with a customized one, specifically, by installing a new handler of INT 13. Once the Universal OS loader is up and running, the client will have the ability to access the system VDisk remotely. Note that all kinds of OS environments are loaded by the Universal OS loader, but with a different system image. The immediate next step is to read and execute the OS-specific loader provided by different OSes. Usually, the loader is the Master Block Record in the VDisk, which is the first step in a regular OS boot process. After this point, the OS take control and continue to boot up.

When loading other user-specific types of VDisk, the service agent will also first broadcasts a message to locate the server that hold the needed data image specified by OS types and users name (in our current implementation, each system image will have its own related data image for simplicity) . The service handler will then authenticate the user through a network authentication protocol, such as Kerberos [7]. If getting through, the service handler will accept this connection.

Usage Process. After the selected OS is loaded by the boot process, the BIOS-enabled Vdiks emulator and service agent will not function for its real mode memory accessing that are not taken by modern common OS due to performance reasons. Thus an OS-specific emulator and agent must be implemented. This VDisk emulator can be implemented as a specialized block disk device driver for different OSs, for example, SCSI device driver. The service agent can be an in-kernel module that will be loaded before the network device because it must need to use the network device to communicate with the remote server. Note that the service agent may use a memory cache to reduce the network communication overhead. In order to maintain the data consistency, we adopted a write through strategy with profile or user VDisk, while a write-back strategy for shadow VDisk.

4 Early Experience

We have deployed TransCom Systems as e-learn class room in many colleges and universities in China where users know little about backups, security patches and antivirus. Before TransCom System, they adopt PCs for e-class room and get in trouble with many managing works, such as struggle with virus, installing/uninstall to every PC and reinstall the OS whenever a system is down. TransCom System makes all such managing work become simple and easy. With the help of centralization of software and data, the administrator just need to maintain the server for the whole system. The sharing of System VDisk Image allows that the installation and updating of software can be deployed to every user's desktop as soon as the installation is completed. The administrator may configure the system to discard shadow image whenever clients reboot. This make virus and unnecessary software can not resident in the system for a long time.

5 Related Work

The tremendous increase in complexity of computing machinery across the globe and the resultant inability of administrators to deal with, has initiated activities in academia and industry to make system self-managing. A vision of autonomic computing as described in [6] is to design computing systems that can manage themselves given high-level objectives from administrators. Of the four aspects of self-management defined in the vision, we are focusing on self-protect and self-heal. We also introduce a way to configure to distribute systems more easily. Our architectural approach is based on centralized management which is similar to network-based file system [4,11] and thin client [2,5]. Network-based file systems were adopted widely in many institutes and corporations. In these systems, users' data storage is centralized on servers. The file or data administration and management tasks, like creation, backup, recovery, are left to dedicated system administrator and operators. However, this model can only deal with the data management problems. Those tasks regarding to software system, such as installation, upgrade, patch still burden on users. The most popular approaches to revive the central mainframe computing is the thin-client computing, which was deployed academically and commercially. Reminiscent of central computing, computation and storages are performed on centric servers. The thin clients use a remote display protocol to access their computing environments through a special terminal or general software. Unfortunately, this model has higher hardware cost. Today, the cost of a thin client is nearly as much as a stand-alone PCs that without a hard disk. A thin client needs a server to finish the computing tasks, thus multiple users will need a more powerful server, increasing cost. Also, the challenge of managing multiple desktops still remains, for example, threats from user mistakes, virus attacks, though it is centralized in a site.

6 Conclusion and Future Work

In this paper, we have presented the virtual disk based self-management scheme of software and data in TransCom system. It centralizes the storage and management of software and data and delivers them to users in a streaming way based on a block level virtualization principle. We described its design and implementation and The early experience showed that by using this central management scheme, TransCom clients reduce most of the managing burden from users and provide a easy way to manage the system for administrators.

In future, we will extend this scheme and architecture to more types of devices and platforms. For example, we are now working to extent the architecture to support Open Solaris [3]. In the Internet, the increasing number of routers and the diversity of data transportation requirement are making the management tasks a great challenge [8]. We argue that the central management scheme of software and data may provide an approach to investigate for routers software deployment and configuration to reduce the management of routers.

References

1. 100x100Projects. <http://100x100network.org/>
2. Baratto, R., Kim, L., Nieh, J.: THINC: A Virtual Display Architecture for Thin-Client Computing. In: Proc. of the Twentieth ACM Symposium on Operating Systems Principles (2005)
3. Greenberg, A., Hjalmtysson, G., David, A., et al.: A Clean Slate 4D Approach to Network Control and Management. ACM SIGCOMM Computer Communication Review 35(5) (2005)
4. Howard, J., Kazar, M., Menees, S., Nichols, D., Satyanarayanan, M., Sidebotham, R., West, M.: Scale and Performance in a Distributed File System. ACM Transactions on Computer Systems 6(1), 51–81 (1988)
5. Boca, I.: Research: Citrix ICA Technology Brief, Technical White Paper. Boca Raton (1999)
6. Kephart, J.O., Chess, D.M.: The Vision of Autonomic Computing. IEEE Computer 36(1), 41–50 (2003)
7. Neuman, B.C., Ts'o, T.: Kerberos: An Authentication Service for Computer Networks. IEEE Communications 32(9), 33–38 (1994)
8. Nieh, J., Yang, S.J., Novik, N.: Measuring Thin-Client Performance Using Slow-Motion Benchmarking. ACM Transactions on Computer Systems (TOCS) 21(1), 87–115 (2001)
9. Preboot Execution Environment (PXE) Specification (1999)
<ftp://download.intel.com/labs/manage/wfm/download/pxespec.pdf>
10. RedFlag Linux. <http://www.redflag-linux.com/eindex.html>
11. Sandberg, R., Goldberg, D., Kleiman, S., Walsh, D., Lyon, B.: Design and Implementation of the Sun Network Filesystem. In: Proceedings of the Summer USENIX Conference (June 1985)

Defending Against Jamming Attacks in Wireless Local Area Networks^{*}

Wei Chen, Danwei Chen, Guozi Sun, and Yingzhou Zhang

Computer College

Nanjing University of Posts and Telecommunications

Nanjing 210003, Jiangsu, China

{chenwei, chendw, sun, zhangyz}@njupt.edu.cn

Abstract. Wireless local area networks(WLANs) work upon a shared medium and broadcast data frames via radio waves. These features let legitimate users connect to Internet conveniently at WLAN access points but also let a malicious attacker misuse network flexibly. Attackers can launch jamming attacks by exploiting vulnerabilities in 802.11, which is a dominant protocol used by WLAN. We discuss two jamming attacks, RTS and CTS jamming, and address their attacking conditions and threats. To defend against jamming attacks, a CUSUM based detection method is proposed, which can accurately detect jamming attacks with little computation and storage cost. When continuous attacking packets are sent out, a change-point will appear in a traffic sequence. Our detection method is capable to observe such change-points and then launch alarms. The efficiency of defense methods is verified by simulation results.

1 Introduction

Rapid advancements in wireless networks have led to wide deployments of public wireless local area networks(WLANs). Because of its convenience and low cost, WLAN has become a standard feature on laptop computers. More laptop users prefer to access internet through WLAN in public areas, such as in hotel, cafe, airport and so on.

Increasing popularity of WLANs brings more security concerns[1,2]. Wireless networks work upon a shared medium and broadcast data frames via radio waves. These features make wireless network more vulnerable than wired one since an attacker can flexibly perform attacks without physical infrastructure constraints. Among widely known vulnerabilities of WLAN, jamming attack is one important class of security threats. A malicious attacker can prevent users from getting access to services by inserting bogus packets into the WLAN. Bellardo[3] introduced vulnerabilities in the 802.11 management and media access services,

^{*} This work is supported by the National Natural Science Foundation of China(Science Department Special Dean Foundation) under Grant No. 60642006 and by China NJUPT Climbing Project Foundation NY206077.

which can be utilized to perform Denial of Service(DoS) attacks. It has been reported from the “black hat” community that tools for performing the deauthentication/disassociation attack, a kind of WLAN DoS attack, has already been available for download.

In this paper, we discover a novel attack which exploits the vulnerability of RTS/CTS mechanism in 802.11 protocol. This jamming attack periodically emits spoofing CTS attacking packets and seriously degrades the throughput of WLAN. Different from RTS jamming attacks proposed by Bellardo[3], the proposed CTS jamming attack uses less attacking packets. It is more difficult to detect attacking behavior accurately.

To detect RTS and CTS jamming attacks, we propose a Cumulative-Sum-based(CUSUM-based)[4] detection method. The signature of RTS/CTS jamming attack is indicated by repeated CTS or RTS frames sent out from the same source in a short period. We apply a non-parameter CUSUM method to evaluate these CTS and RTS frames. During jamming attacks, repeated CTS and RTS frames will trigger suspicious scores and these scores are accumulated by CUSUM-based detection method. An alarm is launched when accumulated score reaches the predefined threshold.

There are three principle contributions in this paper. First, we expose the possibility of performing jamming attacks, which exploits the vulnerabilities of RTS/CTS mechanism in 802.11 protocol. The proposed CTS jamming attack can dramatically degrade the performance with low-rate attacking traffic, which is a more sophisticated attack than other jamming attacks. Second, a detection method based on CUSUM has been proposed to detect jamming attacks. When repeated attacking packets are sent out, a change-point will appear in the traffic sequence. CUSUM based detection method is capable to observe such change-points and then give alarms. A jamming attack can be accurately detected by this method with very small computation and storage cost. At last, we use NS2, a popular network simulator, to simulate the wireless jamming attack scenarios and evaluate the performance of detection method.

The paper will be organized as follows. Section 2 discusses vulnerabilities in RTS/CTS mechanism and possibilities to launch jamming attacks. Section 3 illustrates the CUSUM-based detection methods. RTS/CTS windows and detection algorithm are proposed in this section. In section 4 simulation results show wireless jamming attacks can seriously degrade network performance in WLAN. They also show that our detection method can accurately detect a jamming attack. Section 5 introduces some related works about WLAN security research. Section 6 offers our conclusion and future works.

2 Wireless Jamming Attacks

In this section we discuss two jamming attack methods, RTS and CTS jamming. We first introduce RTS/CTS mechanism, which is a standard for the 802.11 MAC layer to avoid hidden node problem in most WLANs. Then we propose two wireless jamming attacks, RTS jamming and CTS jamming. Since RTS/CTS

mechanism does not have authentication protection, attackers can easily forge RTS or CTS frames and use these spoofed RTS/CTS frames to launch wireless jamming attacks.

2.1 The RTS/CTS Mechanism in 802.11

There exists hidden node and exposed node problems in 802.11 since not all nodes can sense each other in wireless network. One problem is hidden node problem. For example, Figure 1 shows four nodes, A, B, C and D. A and C cannot be aware of each other but both of them can sense the existence of B. When A and C begin to send data frame to B at the same time, the data frames collide with each other at B. Neither A nor C can know this collision. A and C are hidden nodes to each other. The other problem is the exposed node problem. B and C are in each other's radio range. B intends to send data frame to A and C sends data frame to D. This does not bring any problem since C's transmission to D will not interfere with B sending data to A. But it is not allowed by media access control and this is called the exposed problem. RTS/CTS mechanism is proposed to solve these two problems. The purpose of using RTS/CTS mechanism in 802.11 is to reduce

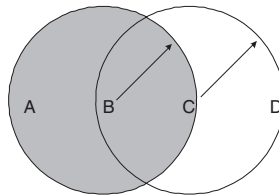


Fig. 1. Collision problems in a wireless network

collisions caused by hidden nodes and increase the performance of the network. In this mechanism, the sender and receiver use Request to Send(RTS)/Clear to Send(CTS) handshake before data transmission. When handshake begins, the sender informs all contiguous nodes that it will transmit data by sending a RTS frame. A field in RTS frame, Network Allocation Vector(NAV), indicates how long the sender wants to hold the medium to finish data transmission. When receiver gets this RTS, it replies with a CTS frame. Any node that hears CTS frame will keep silent in the consequent period indicated by RTS. On the other hand any node that can hear the RTS frame but not the CTS frame is close to sender but not close to receiver. It is free to transmit, which avoids exposed node problem.

RTS/CTS mechanism does not provide any authentication protection, which brings potential vulnerabilities. Attackers can easily spoof nodes' identification. For example an attacker can fill the source address of an attacking frame with access point's address, which makes this frame look like a frame sent from access point. This is an important reason why attackers can exploit RTS/CTS mechanism to perform attacks. The RTS/CTS handshake continues for each frame,

as long as the frame size exceeds the threshold set in the corresponding access point. The frequent RTS/CTS handshake implies jamming attacks exploiting RTS/CTS mechanism can be performed at any time during wireless communication.

2.2 RTS/CTS Jamming Attacks

We discuss two possible jamming attack methods, RTS jamming and CTS jamming, which misuse RTS/CTS frames.

RTS jamming has been first proposed by Bellardo in [3]. The attacker occupies channel by sending RTS frame with large enough NAV. As Figure 2(a) shows, the attacker continuously sends out RTS frames to access point(AP). The AP replies with CTS which can be heard by nearby nodes. These nodes will keep silent for a period of time indicated by NAV. After the previous NAV expires, another attacking RTS is sent out by the attacker. When attacking RTSs have been injected into wireless networks, the other nodes can hardly occupy channel to communication with AP.

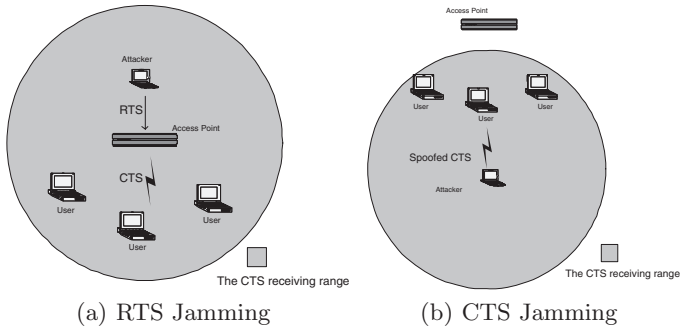


Fig. 2. RTS/CTS jamming methods which exploit CSMA/CA vulnerabilities

We propose a novel jamming attack called CTS jamming attack, which is more subtle than RTS jamming attack. During CTS jamming attack, the attacker sends CTS frames with spoofed ID which is as same as AP. But the attacker should keep AP unaware of its spoofing behavior. This can be achieved by using directional antenna or keeping far enough from AP but still close to other nodes(as shown in Figure 2(b)). Then legitimate nodes believe the AP is busy receiving data from a hidden node and will stop sending any data frame in the following period. The attacker repeatedly keeps the channel busy long enough to make legitimate nodes have no chance to occupy the channel. To avoid being detected, the attacker may periodically send attacking CTS frames instead constantly sending packets. The attacker can turn into sleep state for a while until next attacking time comes. Although this attacking strategy cannot totally prevent other nodes from communication, it can seriously degrade the network throughput. This attack has lower traffic rate than normal jamming attack and is more difficult to detect.

3 Defending Against RTS/CTS Jamming Attack

A Cumulative-Sum-based(CUSUM-based)[4] detection method is proposed in this section. The signature of RTS/CTS jamming attack is indicated by repeated CTS or RTS frames sent out from the same source in a short period. Two data windows, CTS and RTS data windows, are used to record recent CTS and RTS frames. When a new CTS frame comes, it will be evaluated by a function. Then a non-parameter CUSUM method, a sequential detection change-point method, is used to evaluate these scores.

3.1 RTS/CTS Data Windows

Two data windows, RTS window and CTS window, are used to record recent received RTS/CTS frames. All nodes and access point in a wireless domain will establish these two data windows. When a node senses a new RTS or CTS frame, the node will record source ID information of the frame in corresponding window. The size of the window is fixed and the new arriving frame will replace the oldest one when there is no enough space in the window. The newest record has the smallest index.

Figure 3 shows the RTS/CTS processing algorithm. When a CTS frame comes, a score is given to indicate whether it is suspicious. If the source ID of a CTS frame appears in CTS window, it means another CTS frame from the same source has been received recently. This CTS frame is more suspicious than the one which does not appear in the window. This frame will be checked further in the RTS window.

If the source ID of CTS also appears in the RTS window, it means this CTS does not forge its source ID and may be caused by a RTS jamming attack. If the source ID is not found in the RTS window, this CTS may be spoofed by a CTS jamming attacker or there exists a hidden node. Different scores will be given to these suspicious CTS frames using a evaluation function $f(i) = \log(\frac{\alpha}{i})$. i is the index of matched record in CTS window and α is a parameter. Usually α is set no less than the CTS window size. The smallest index, which means the CTS frame matches the latest CTS, gains the highest score. This CTS frame is more like to belong a series continuous CTS frames sent from a CTS jamming attacker. Otherwise if a CTS matches the oldest record, the minimum score will be given. A higher score is given for the CTS frame that does not appear in RTS window since the former CTS may belong to a CTS jamming attack, which is more dangerous than RTS jamming. Parameter $\beta(\beta > 1)$ is used to increase CTS scores for the CTS frames that are suspected of CTS jamming attack.

3.2 Detection Method Using CUSUM

After the score is given by RTS/CTS processing algorithm, a CUSUM method is used to accumulate these scores. Generally if more CTS frames with the same source ID appear contiguously in the CTS/RTS window, the higher CUSUM result will be generated.

```

RTS/CTS Processing(Input: a CTS or RTS frame)
if Receive a RTS frame  $f_{RTS}$  then
    Record it in RTS window. Replace the oldest RTS record with this one
    return
end if
if Receive a CTS frame  $f_{CTS}$  then
    Record it in CTS window. Replace the oldest CTS record with this one
     $score = 0$ 
    for  $i=1; i \leq$  Total count of records in CTS window;  $i++$  do
        if  $f_{CTS}$  matches  $record_i$  in CTS window then
            if  $f_{CTS}$  matches a record in RTS window then
                //It may be caused by a RTS jamming attack
                 $score = score + \log(\frac{\alpha}{i})$ 
            else
                //It may be caused by a CTS jamming attack
                 $score = score + \beta \times \log(\frac{\alpha}{i})$ 
            end if
        end if
    end for
    return  $score$ 
end if

```

Fig. 3. CTS and RTS frame processing algorithm. New CTS and RTS frames are recorded in windows. A score is given when a new CTS arrives.

Cumulative Sum (CUSUM) is a sequential detection change-point method which assumes that the mean value of some variable under surveillance will change from negative to positive value whenever a change occurs. We assume the channel is nearly fairly shared among nodes. Therefore the source ID distribution of CTS and RTS frames are uniform. If a node continuously keeps channel, this uniform distribution in this period will change. CUSUM is applied to detect such changes in CTS window. When a change point happens and is detected, the corresponding CTS frames are identified as suspicious.

We first introduce the essence of sequential change-point detection[4]. Suppose the observations of a random process X_t (with discrete or continuous time) are received sequentially. At a certain moment (random or not, but unknown), some probabilistic characteristics of this process change. An observer must make a decision as quickly as possible as to whether a change-point has happened or not, while keeping the false alarm rate to be as low as possible.

Suppose that a sequence $X_1, \dots, X_r$ of independent random variables is observed. For each $1 \leq v \leq r$, consider the hypothesis H_v that $x_1, \dots, x_{v-1}$ have the same density function $f_0(\cdot)$ and $x_v, \dots, x_r$ have another density function $f_1(\cdot)$. Denote by H_0 a hypothesis of stochastic homogeneity of the sample. Then the likelihood ratio statistic for testing the composite hypothesis H_v ($1 \leq v \leq r$) against H_0 is:

$$\max_{0 \leq k \leq r} (S_r - S_k) = S_r - \min_{0 \leq k \leq r} S_k,$$

where

$$S_0 = 0, S_k = \sum_{j=1}^k \log \frac{f_1(x_j)}{f_0(x_j)}.$$

The mathematical expectation of $y_r = \log(f_1(x_r)/f_0(x_r))$ is negative before and positive after the change-point. The stopping rule for change-point detection is:

$$\tau = \inf\{r \geq 1 : S_r - \min_{0 \leq j \leq r} S_j \geq b\},$$

where $b > 0$ is the alarm threshold. There is a nonparametric version of the CUSUM statistic:

$$y_r = (y_{r-1} + x_r)^+, y_0 = 0,$$

and the corresponding decision rule is

$$d_N(\cdot) = d(y_r) = I(y_r > N),$$

where $I(\cdot)$ is the indicator function and N is the threshold. d_N is the decision at time r , which gives a value of 1 to indicate an attack and 0 to indicate a normal condition.

When CUSUM is applied to jamming attack detection, C_t is defined for a series CTS scores received sequentially. At a certain moment when continuous CTS frames from the same source arrive, probabilistic characteristics of this sequence change and C_t becomes larger than normal value. In normal situation, $E(C_t) = c$. We choose a parameter a as the upper bound of c , i.e., $a \geq c$. The score above a means a suspicious value, otherwise it is legal. Then we define $c_t = C_t - a$ so that it has a negative value during normal operation. When an attack takes place, there are continuous CTS frames received by detector. The CTS scores will suddenly grow larger and the value $c_t = C_t - a$ will become positive. When the CUSUM value exceeds the threshold N , jamming attack alarm will be launched.

4 Simulation Results

The popular network simulator NS2 is used to evaluate the proposed CTS jamming attacks and detection method. We set a simple but typical WLAN topology in simulation. The WLAN includes an access point(AP), an attacker and four legitimate mobile nodes. The TCP sender is connected with AP using wired networks through a router. The link bandwidth between the router and the AP is set to 10Mbps and delay is set to 2ms. The legitimate mobile nodes receive TCP data from the sender through the AP. The sending rate of the AP is set to 11Mbps.

4.1 Results of Jamming Attacks

We first evaluate the network throughput performance under CTS jamming attacks. Jamming attacks periodically send out attacking CTS frames. The attacking period is set to 1 second and each jamming attack lasts 32 ms. The simulation runs for total 100 seconds.

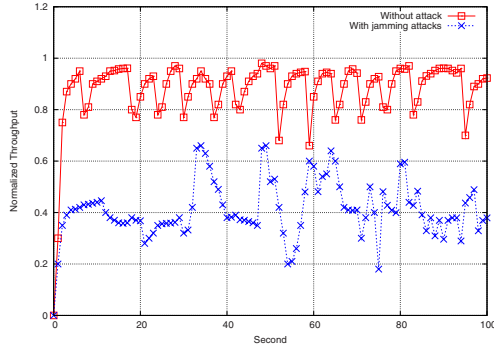


Fig. 4. TCP throughput under wireless jamming attacks

As shown in Figure 4, network throughput is seriously degraded with CTS jamming attack. We can see that TCP throughput only reaches 40% of normal situation level. Note that the attacking time occupies only about 10% of total time. Simulation results show CTS jamming attack can destroy network communication with a little overhead.

4.2 Detection Results

Simulations are designed to evaluate how well our detection method performs. We use same simulation topology as the previous one. Two scenarios are used to test CUSUM based method. One scenario is with jamming attacks and the other one is free of attacks. The attacking period is set to 1 second. The sizes of CTS and RTS windows are set to 6. α equals 7 and β equals 2. The simulations run total 100 seconds but we just give part of CUSUM results(10 seconds) in Figure 5 for clarity.

From Figure 5(b), we can see that when continuous CTS frames are observed, the CUSUM result grows large dramatically. When CUSUM results exceed the

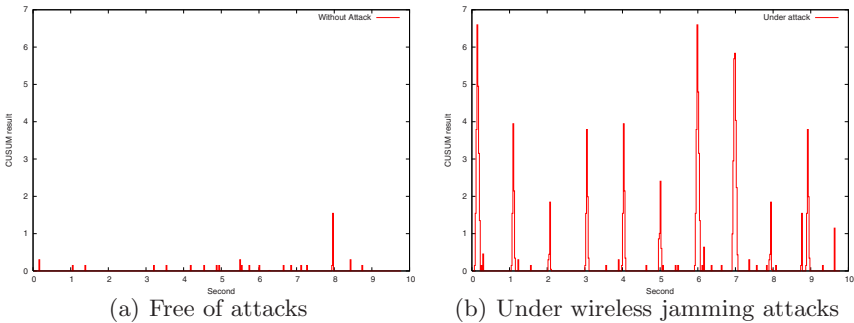


Fig. 5. CUSUM results for two scenarios: one is free of attacks and the other is under jamming attacks

threshold N , jamming attacks alarm will be sent out. Figure 5(a) shows the CUSUM results for a healthy wireless network are rather small when compared with those under attacks. Though some CUSUM results are higher than 0, which may be caused by continuous CTS frames sent by a legitimate user, CUSUM can adjust the results to 0 rapidly. This ensures low false positives during jamming attacks detection. The upper bound a is set to 1.8 and the threshold N to 5 according to experience. From Figure 5, we can see that it is not difficult to set these two parameters since the abnormal CTS frames are easily distinguished from the normal ones.

5 Related Work

Our work presented in this paper relates to Denial-of-Service(DoS) attacks and jamming attacks in wireless networks. We briefly introduce these related works as follows.

DoS attacks in wireless environment, including ad hoc and infrastructure networks, has been widely researched[5,3,6,7,8,9]. In [3], some vulnerabilities in 802.11 were outlined. Malicious DoS attacks can be performed targeting 802.11 management and media access protocols. Two important classes of DoS attacks were practically implemented and their practical effectiveness are investigated. Xu proposed two strategies, channel surfing and spatial retreats, to evade a MAC/PHY-layer jamming-style wireless DoS attack in [10]. The defense philosophy was quite different from others since they tried to evade impacts from attacks instead of counteracting attacks directly. In Xu's following works[7], they addressed several jamming attacks models and detection method using consistency checking. In [11], MAC-layer greedy behavior to gain bandwidth at the expense of other stations was studied. DOMINO was proposed to detect MAC misbehavior.

6 Conclusion and Future Works

This paper discusses two jamming attacks, RTS jamming and CTS jamming, which shed light on research of attack defense. We also present a detection method using CUSUM algorithm. CUSUM detection method can accurately detect RTS/CTS jamming attacks with little computation and storage cost, which is rather useful to handle numerous CTS frames in wireless environment. The simulation results show that it is possible to launch wireless jamming attacks, which may become a potential threat in real world. Fortunately proposed CUSUM detection method can accurately distinguish jamming attacker from normal users.

Our future work will focus on jamming attacking models and conditions with detailed parameters. Response and prevention method will also be researched. We have began to perform experiments in real wireless networks environment to evaluate our methods and experiment results will come soon.

References

1. Hubaux, J.P., Butty, L., Capkun, S.: The quest for security in mobile ad hoc networks. In: Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing(MobiHoc '01), pp. 146–155. ACM Press, New York (2001)
2. Butty, L., Hubaux, J.P.: Report on a working session on security in wireless ad hoc networks. SIGMOBILE Mobile Computing and Communications Review 7, 74–94 (2003)
3. Bellardo, J., Savage, S.: 802.11 denial-of-service attacks: Real vulnerabilities and practical solutions. In: Proceedings of the 12th USENIX Security Symposium, Washington, DC, pp. 15–28 (2003)
4. Brodsky, B.: Nonparametric Methods in Change-Point Problems. Kluwer Academic Publishers, Netherlands (1993)
5. Gupta, V., Krishnamurthy, S., Faloutsos, M.: Denial of service attacks at the MAC layer in wireless ad hoc networks. In: Proceedings of MILCOM 2002. vol. 2, pp. 1118–1123 (2002)
6. Housley, R., Arbaugh, W.: Security problems in 802.11-based networks. Communications of ACM 46, 31–34 (2003)
7. Xu, W., Trappe, W., Zhang, Y., Wood, T.: The feasibility of launching and detecting jamming attacks in wireless networks. In: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing(MobiHoc05), pp. 46–57. ACM Press, New York (2005)
8. Aad, I., Hubaux, J.P., Knightly, E.W.: Denial of service resilience in ad hoc networks. In: Proceedings of the 10th annual international conference on Mobile computing and networking(MobiCom '04), pp. 202–215. ACM Press, New York (2004)
9. McCune, J.M., Shi, E., Perrig, A., Reiter, M.K.: Detection of denial-of-message attacks on sensor network broadcasts. In: Proceedings of 2005 IEEE Symposium on Security and Privacy, pp. 64–78 (2005)
10. Xu, W., Wood, T., Trappe, W., Zhang, Y.: Channel surfing and spatial retreats: defenses against wireless denial of service. In: Proceeding of 2004 ACM workshop on Wireless security, pp. 80–89 (2004)
11. Raya, M., Hubaux, J.P., Aad, I.: DOMINO: a system to detect greedy behavior in IEEE 802.11 hotspots. In: Proceedings of the 2nd international conference on Mobile systems, applications, and services(MobiSys '04), pp. 84–97. ACM Press, New York (2004)

Schedulability Analysis of the Fault-Tolerant Hard Real-Time Tasks with Limited Priority Levels*

Jun Li¹, Fumin Yang¹, Gang Tu¹, Wanhua Cao², and Yansheng Lu¹

¹ Department of Computer Science, HuaZhong University of Science and Technology,
Wuhan, 430074, P.R. China
hustlee@126.com, yangfm@csmail.hust.edu.cn, tugang@tom.com,
lys@mail.hust.edu.cn

² Wuhan Digital Engineering Institute, Wuhan, Hubei, 43007 P.R. China
caowh@709.net.cn

Abstract. In this paper, we consider fixed priority scheduling of fault-tolerant hard real-time tasks in which the priority level of the system is insufficient. This paper extends necessary and sufficient conditions for the purposed of limited priority levels on fault-tolerant hard real-time systems which takes into account the effect of temporary faults. The major contribution of our approach is to consider the recovery of tasks running with higher system priorities for the case of limited priority levels. This characteristic is very useful since the available slack time of higher system priority tasks can be make use of for recovering the faulty tasks of lower system priorities. Due to its flexibility and simplicity, the proposed approach provides an effective schedulability analysis, where the schedulability utilization of the system can be improved.

1 Introduction

While operating system software may be capable of supporting essentially unlimited priority levels, the number of priorities supported by network or backplane hardware is usually quite small. The *natural* priority of a task is defined as the priority that would have been assigned to it on a system with unlimited priorities. A task set may require more natural priority levels than the system can support. In this case, more than one task must be grouped into the same *system* priority. In this paper we consider fixed priority scheduling of fault-tolerant hard real-time tasks with limited priority levels.

Several schedulability analysis for the purposed of limited priority levels can be found in the literature. Lehoczky and Sha [1] considered sufficient schedulability conditions and developed an expression for schedulability loss due to limited priorities. And the necessary and sufficient schedulability bounds has been provided by [2] for the case of limited number of priority levels. Katcher et al. [3] expanded on

* This work was supported by the National Natural Science Foundation of China under Grant No.60603032.

the work in [1] and [2] and developed both necessary and sufficient conditions for determining whether the time properties of a particular task set can be guaranteed when it is scheduled on a system with limited number of priority levels. Scheduling assignment algorithms with limited priority levels have been proposed in [4-7]. However, Most of these mechanisms have been used for the design of systems on the assumption that there is no error during system execution. In this paper, we expand on the systems' computation model, and take into account the fault-tolerant model.

The remainder of this paper is organized as follows: The next section presents the assumed our computational model. Section 3 describes briefly some concepts and theorems regarding the schedulability analysis for the case of limited priority levels. The main result of this paper is presented in section 4. Section 5 describes a simple evaluation of our solution based on simulation. Finally, section 6 offers our conclusion.

2 Computation Model

We assume that there is a set $\Gamma = \{\tau_1, \tau_2, \dots, \tau_n\}$ of n tasks, called *primary tasks* that must be scheduled by the system in the absence of faults. Any primary task τ_i in Γ has a period T_i , a deadline D_i and a worst-case computation time C_i . Tasks can be periodic or sporadic. For sporadic tasks the period refers to the minimum inter-arrival time. For simplicity, we only need to refer to $\bar{\tau}_i$, whose worst-case computation time $\bar{C}_i$ is the largest one among these worst-case computation time of all alternative tasks associated with τ_i , as the worst-case alternative task in case of faults in task τ_i . So the total response time of τ_i is the sum of the response time of τ_i and that of $\bar{\tau}_i$.

We denote p_i and $\bar{p}_i$ as the priority of each primary task τ_i and its alternative task $\bar{\tau}_i$, respectively. Here, the definition of p_i is given for two cases. When unlimited priority levels are supported, primary tasks are scheduled according to some fixed priority assignment algorithm (e.g. DMS [8]), which attributes a distinct priority to each primary task. However, it is different when the system supported limited priority levels. Primary tasks are scheduled according to some fixed priority assignment algorithm (e.g. FPA [5]), which makes some primary tasks to be assigned to one system priority level. In general, we consider m ($m \leq n$) different priority levels (1, 2, ..., m), where 1 and m represent the highest and the lowest priority level, respectively. When a primary task and an alternative task of other tasks are ready to execute at the same priority level, we assume that the alternative task is scheduled first.

We consider only the occurrence of temporary software faults in a uniprocessor system. When a fault hits τ_i during its execution, the system must schedule the alternative task $\bar{\tau}_i$. We also assume that there is no cost associated with any scheduling of primary or alternative tasks. Also, we assume that all faults are detected by the system and there is no fault propagation, which means that faults affect only the executing task. Meanwhile, we assume in the analysis that there is a minimum time between two consecutive error occurrences, T_E .

3 Background

Lehoczky, Sha, and Ding [2] developed necessary and sufficient schedulability conditions for a system with unlimited priority levels. Assuming that task τ_i arrives simultaneously with all higher priority tasks, the cumulative work that has arrived from priority levels 1 to i in the time interval $[0, t]$ is given by

$$W_i(t) = \sum_{j=1}^i C_j \lceil t/T_j \rceil. \quad (1)$$

If $W_i(t)/t \leq 1$ then the elapsed time is at least as great as the time required to complete the work arrived by time t . This forms the basis for the following theorem [2], which allows deadlines to be less than periods.

Theorem 1. *Let a periodic task set $\tau_1, \tau_2, \dots, \tau_n$ be given in priority order and scheduled by a fixed priority scheduling algorithm using those priorities. If $\forall i, D_i \leq T_i$, the task set will meet its deadlines if*

$$\max_{1 \leq i \leq n} \min_{0 < t \leq D_i} W_i(t)/t \leq 1. \quad (2)$$

When the system supports fewer priority levels less than the number of natural priorities of a task set, the case is different. Consider a set of task $\tau_1, \tau_2, \dots, \tau_n$ arranged in decreasing natural priority order. Assume the system to be scheduled supports m distinct priorities denoted by $g_1, g_2, \dots, g_m$, also arranged in decreasing priority order. When $m < n$, tasks at different priorities must be assigned the same system priority, called priority group. Each priority group is a set of tasks that execute at the same priority level. Katcher, Sathaye, and Strosnider [3] proved the necessary and sufficient schedulability condition with limited priorities. Let τ_i be a member of priority group g_k . For group g_k , assume that there are r tasks in priority groups with higher system priority than g_k . Note that r may be zero. If $r=0$, let the summation of $\sum_{j=1}^r = 0$.

Theorem 2. *Consider a periodic task set arranged in natural priority order. Let the task set $\tau_1, \tau_2, \dots, \tau_n$ be scheduled on a system with m priority groups, $g_1, \dots, g_m$, by a fixed priority scheduling algorithm. Let $\tau_i \in g_k$, and $r = \sum_{1 \leq u < k} |g_u|$, if $D_i \leq T_i$, then τ_i will meet all its deadlines if $\min_{0 < t \leq D_i} W_i(t)/t \leq 1$, where:*

$$W_i(t) = \sum_{j=1}^r C_j \lceil t/T_j \rceil + \sum_{\forall \tau_l \in g_k} C_l \quad (3)$$

In order to compare schedulability with limited priorities to that with unlimited priorities using the necessary and sufficient conditions, the *degree of schedulable saturation* was introduced by [3] as such a metric for both the unlimited priority case ($S_{\max}$) and the limited priority case ($S'_{\max}$). A particular scheduling situation is said to be *better* if it results in a smaller value of the degree of schedulable saturation. Here, we give the definitions of $S_{\max}$ and $S'_{\max}$ described as follows. S_i is the cumulative work due to equal or higher priority tasks in a busy period normalized by time.

$$S_{\max} = \max_{1 \leq i \leq n} S_i, \text{ where } S_i = \min_{0 < t \leq D_i} S_i. \quad (4)$$

If $S_{\max} \leq 1$, then the task set is schedulable. Similar to $S_{\max}$, if $S'_{\max}$ is greater than unity the system is unschedulable with limited priorities. We again assume that there r tasks in priority groups with higher priority than g_i , and k tasks in g_i , where g_i is the system priority group of task τ_i .

$$S'_{\max} = \max_{1 \leq i \leq n} S'_i, \text{ where } S'_i = \min_{0 < t \leq D_i} W_i(t)/t. \quad (5)$$

As can be seen from the theorem 1 and theorem 2, task τ_i will meet all its deadlines if $\min_{0 < t \leq D_i} W_i(t)/t \leq 1$ when the priorities are whether unlimited or limited, though the derivations of $W_i(t)$ are different. However, it is worth note that both of the two necessary and sufficient schedulability conditions above described are based on the assumption that there is no error during system execution. The fault-free assumption is, in face, not realistic. In the following sections, we will discuss the schedulability conditions for the fault-tolerant hard real-time task sets with limited priority levels.

4 Schedulability Analysis

In this section, we mainly discuss the schedulability condition with limited priorities when considering fault tolerance. According to the analysis described in the Section 3, the derivation of $W_i(t)$ is a key to solve the problem on the schedulability of task τ_i .

When faults are considered, $W_i(t)$ consists of two parts: the time needed to execute task τ_i and all tasks with each of whose priority is higher than p_i , say $I_i(t)$, and the time necessary to recover the fault tasks, say $F_i(t)$. According to the computation model described in the Section 2, it is very clear that the computation time of task τ_i is given when the task set Γ is given. So the problem to calculate $W_i(t)$ divides into two child problems: how to calculate $I_i(t)$ and how to calculate $F_i(t)$. Note that the computation of $I_i(t)$ and $F_i(t)$ is related to the current running task, which can be some primary tasks or some alternative tasks. The priority assignment algorithm of primary tasks or alternative tasks becomes a key to calculate $I_i(t)$ and $F_i(t)$.

4.1 Unlimited Priority Levels

When the priorities are unlimited, the primary tasks are scheduled according to some fixed priority assignment algorithm. For simplicity, we assume that the recovery can be carried out at the same priority level as its primary priority, say $\bar{p} = p$. Considering that some errors occur on any task in the time interval $[0, t]$, only any task τ_j that satisfies the condition $p_j \geq p_i$ can interrupt the execution of task τ_i . Let $hpe(i)$ denote such the subset as $hpe(i) = \{k | k=1, \dots, i\}$. In the worst-case scenario, there may be $\lceil t/T_E \rceil$ errors in the time interval $[0, t]$.

Theorem 3. Consider a set of fixed-priority scheduled set of primary tasks and their alternative tasks. For any value of $T_E > 0$, if $\forall i, D_i \leq T_i$, the task set will meet its deadlines if $\max_{1 \leq i \leq n} \min_{0 < t \leq D_i} W_i(t)/t \leq 1$, where:

$$W_i(t) = \sum_{j=1}^i C_j \lceil t/T_j \rceil + \lceil t/T_E \rceil \max_{k \in hpe(i)} \bar{C}_k \quad (6)$$

Proof. If $\bar{C}_i = 0$ for any task τ_i , the case is the same as that of the fault-free case. It is obvious that the cumulative work of task by (6) is equal to that by (1). That means that this theorem is equal to theorem 1. If $\bar{C}_i \neq 0$ for any task τ_i , this theorem follows according to the analysis described earlier when taken into account fault tolerance for the case of unlimited priority levels. $\square$

4.2 Limited Priority Levels

When the priority levels that a task set requires are more than the system can support, more than one task must be assigned the same priority. This causes the priority mapping problem. For the convenience, we assume that the system supports m priority levels to a task set $\tau_1, \tau_2, \dots, \tau_n$.

Consider the case when $m < n$. Let τ_i be a member of priority group g_k . If τ_i arrives at time 0, the total worst case cumulative work that must be completed before τ_i completes execution at some time t' consists of the primary work $I_i(t)$ and the recovery work $F_i(t)$. The primary work includes higher priority work that arrives in the time interval $[0, t]$ from priority groups 1 to $k-1$, higher natural priority work which has arrived by time 0 within priority group k , and lower natural priority that has arrived by time 0 within group k . In the following section, we will mainly discuss the problem on how to assign the system priority levels of alternative tasks.

1. $\bar{p} = p$

For this case, the priority level of any alternative task will be mapped into the same system priority group as the priority level of its primary task. Before and after some error on some task τ_i , task τ_i always executes at some system priority level. In the time interval $[0, t]$, the possible errors which can be carried out occur by time 0 in the execution of each task τ_u whose system priority is from priority groups 1 to k . Let f_k denote such the task subset as $f_k = \{\tau_u | \tau_u \in g_1 \cup \dots \cup g_k\}$.

Theorem 4. Consider a set of fixed-priority scheduled set of primary tasks and their alternative tasks arranged in natural priority order. Let the task set $\tau_1, \tau_2, \dots, \tau_n$ be scheduled on a system with m priority groups, $g_1, \dots, g_m$, by a fixed priority scheduling algorithm. For any value of $T_E > 0$, let $\tau_i \in g_k$, and $r = \sum_{1 \leq u < k} |g_u|$, if $D_i \leq T_i$ and $\bar{p} = p$, then τ_i will meet all its deadlines if $\min_{0 < t \leq D_i} W_i(t)/t \leq 1$, where:

$$W_i(t) = \sum_{j=1}^r C_j \lceil t/T_j \rceil + \sum_{\forall \tau_l \in g_k} C_l + \lceil t/T_E \rceil \max_{\tau_u \in f_k} \bar{C}_u, \quad (7)$$

Proof. If $\bar{C}_i = 0$ for any task τ_i , the case is the same as that of the fault-free case. It is obvious that the cumulative work of task by (7) is equal to that by (3). That means that this theorem is equal to theorem 2. If $\bar{C}_i \neq 0$ for any task τ_i , this theorem follows according to the analysis described earlier when taken into account fault tolerance for the case of limited priority levels. $\square$

In order to compare the schedulability with limited priorities to that with unlimited priorities, let us consider the task set in table 1 where $T_E = 10$. For each τ_i , $D_i = T_i$, $\bar{p}_i = p_i$ and $\bar{C}_i = C_i$. As shown as table 1, $S_i \leq 1$ for each task τ_i , which means that the task set

with unlimited priorities is schedulable. But it is different when the task set is split into two subset $g_1 = \{\tau_1, \tau_2, \tau_3\}$ and $g_2 = \{\tau_4, \tau_5\}$. It means that the natural priority levels of task τ_1, τ_2 and τ_3 are mapped into the highest system priority level, and the natural priority levels of task τ_4 and τ_5 are mapped into the lowest system priority group level. Meanwhile, the alternative tasks of each task are allowed to be executed at the same system priority level as its primary system priority level. In this case, $S'_{\max} > 1$ due to the fact that $S'_4 > 1$, which makes the task set unschedulable. Thus, the schedulability utilization of fixed priority scheduling will be reduced greatly if priority levels of the system are insufficient.

Table 1. The degree of schedulable saturation of the task set

$task$	T_i	C_i	S_i	S'_i
τ_1	10.53	1.04	0.21	0.52
τ_2	13.37	1.28	0.36	0.50
τ_3	15.56	1.46	0.52	0.43
τ_4	32.55	4.39	0.91	1.01
τ_5	50.63	2.88	1.00	0.91

Nevertheless, it is interesting to note that there is a considerable amount of slack time at the highest system priority level that could be used to execute τ_4 . In order to improve the schedulability in this case, we consider that alternative tasks are allowed to run at higher system priority levels. Thus, we can make use of the available slack time at higher system priority levels for carrying out the execution of the alternative tasks at the lower system priority levels. In the following sections, we discuss the solution of how to improve the schedulability for this case according to which we can deal with alternative tasks running with higher priority levels. We say our approach as $\bar{p} \geq p$.

2. $\bar{p} \geq p$

For this case, the priority level of any alternative task will be mapped into some higher system priority level. Let us assume that $\bar{\tau}_i \in g_u$, and $u \leq k$. In other words, the alternative tasks of task τ_i may be mapped into the priority group g_k or higher priority group. Consider that some error occurs on task τ_i in the interval $[0, t]$. With $\bar{p} = p$, some primary task τ_j within higher system priority g_h can interrupt the execution of task $\bar{\tau}_i$ in the time interval $[0, t]$. But it is different for the case of $\bar{p} \geq p$. Due to the fact that the priority level of task $\bar{\tau}_i$ may be mapped into some higher priority group g_u and $u \leq h$, task τ_j cannot interrupt the execution of task $\bar{\tau}_i$. As a result, $W_i(t)$ may be reduced. To some extent, the task set may be schedulable with $\bar{p} \geq p$ when it is unschedulable with $\bar{p} = p$. However, it is different when that some error occurs on task τ_l whose alternative priority level is mapped into priority group g_l and $l < k$. Due to this case that $\bar{\tau}_l \in g_l$, the interference of the alternative task of task τ_l should be taken

into account in the derivation of $W_i(t)$ but should be not with $\bar{p} = p$. As shown as the above analysis, $W_i(t)$ may be raised or reduced for the case of $\bar{p} \geq p$. Now, the derivation of $W_i(t)$ splits into two branches: the derivation of $W_i^{\text{int}}(t)$ due to the internal errors and the derivation of $W_i^{\text{ext}}(t)$ due to the external time. Thus, $W_i(t)$ for this case is given by (8) for the worst-case scenario.

$$W_i(t) = \max(W_i^{\text{ext}}(t), W_i^{\text{int}}(t)) \quad (8)$$

Consider the external errors. In other words, some errors occur in the execution of any task but task τ_i in the time $[0, t]$. For this case, task τ_i always run at the priority level p_i . Let us assume that some error of task τ_j occurs. When the system priority of the alternative tasks of task τ_j is no higher than the system priority of task τ_i , the recovery to this error on task τ_j cannot be carried out. Otherwise, the recovery of task τ_j can interrupt the execution of task τ_i . From the analysis described above, $W_i^{\text{ext}}(t)$ due to external errors is given by equation (9).

$$W_i^{\text{ext}}(t) = \sum_{j=1}^r C_j \lceil t/T_j \rceil + \sum_{\forall \tau_l \in g_k} C_l \lceil t/T_E \rceil \max_{\bar{\tau}_u \in f_k - \{\bar{\tau}_i\}} \bar{C}_u \quad (9)$$

When some error occurs in the execution of task τ_i in the time $[0, t]$, it is more complex. Before the error occurs, task τ_i is carried out within the priority group g_k . And, the alternative task of task τ_i will be executed within the priority group g_u after this error occurs. Then, the interference that τ_i and $\bar{\tau}_i$ suffer to the execution of other tasks should be different when $u < k$. Assume that the first error occurs at the time t' , and $0 < t' \leq t$. The objective of the derivation of $W_i^{\text{int}}(t)$ is to derive the cumulative work of task τ_i before and after the error, say $W_i^{\text{int}}(0, t')$ and $W_i^{\text{int}}(t', t)$, respectively.

In the interval $[0, t']$, the case is the same with the case for external errors. Before this error occurs, task τ_i is running at the system priority level supported by the priority group g_k . The recoveries to the possible errors may be released on such a task τ_j as its fault-tolerant system priority group should be among the subset f_k . Note that the alternative tasks of task τ_i should not be considered for the assumption that the first error hits at the time t' . As a result, $W_i^{\text{int}}(0, t')$ is computed by equation (10).

$$W_i^{\text{int}}(0, t') = \sum_{j=1}^r C_j \lceil t'/T_j \rceil + \sum_{\forall \tau_l \in g_k} C_l \lceil t'/T_E \rceil \max_{\bar{\tau}_u \in f_k - \{\bar{\tau}_i\}} \bar{C}_u \quad (10)$$

Now, let us consider $W_i^{\text{int}}(t', t)$. After the time t' , the priority group of task τ_i is raised from g_k to g_u . Then the preemptive interference from other primary tasks should take into account such any task τ_j within the priority group g_l that $l < u$. For group g_u , we assume that there are r' tasks in priority groups with higher system priority than g_u . It is very clear that $r' \leq r$. To avoid the computation of any task τ_j among these r' tasks more than once, we compute the worst-case preemptive time by subtract $\lceil t'/T_j \rceil C_j$ from $\lceil t/T_j \rceil C_j$ for τ_j . In the worst-case scenario, there may be $(\lceil t/T_E \rceil - \lceil t'/T_E \rceil)$ errors in the interval $[t', t]$. The first error accounts for the $\bar{C}_i$, while the others may cause the release of the recovery of any task whose fault-tolerant priority group is no lower than g_u , denote f_u . It is obvious that the subset f_u includes task $\bar{\tau}_i$ due to the assumption that these recoveries of the successive faults hit in the execution of task τ_i . Thus, $W_i^{\text{int}}(t', t)$ is given by equation (11).

$$W_i^{\text{int}}(t', t) = \sum_{j=1}^{r'} C_j (\lceil t/T_j \rceil - \lceil t'/T_j \rceil) + \bar{C}_i + (\lceil t/T_E \rceil - \lceil t'/T_E \rceil - 1) \max_{\bar{\tau}_m \in f_u} \bar{C}_m \quad (11)$$

Due to the fact that the first error of task τ_i occurs at random, the time t' cannot be known in advance. Moreover, $0 < t' \leq t$. For the worst-case scenario, the cumulative work of task τ_i due to internal errors can be given by $W_i^{\text{int}}(0, t')$ and $W_i^{\text{int}}(t', t)$.

$$W_i^{\text{int}}(t) = \max_{0 < t' \leq t} (W_i^{\text{int}}(0, t') + W_i^{\text{int}}(t', t)) \quad (12)$$

From the above schedulability analysis for the case of $\bar{p} \geq p$, we emphasize that the described analysis represent a generalization of the analysis by $\bar{p} = p$ when the system priority levels are insufficient. This is proven by lemma below.

Lemma 1. Consider a set of fixed-priority scheduled set of primary tasks and their alternative tasks arranged in natural priority order. Let the task set $\tau_1, \tau_2, \dots, \tau_n$ be scheduled on a system with m priority groups, $g_1, \dots, g_m$, by a fixed priority scheduling algorithm. For any value of $T_E > 0$, let $\tau_i \in g_k$, and $r = \sum_{1 \leq m < k} |g_m|$, if $D_i \leq T_i$, $W_i(t)$ given by (7) equals to the maximum of $W_i^{\text{int}}(t)$ and $W_i^{\text{ext}}(t)$, whenever the system priority level of the alternative task of each task is mapped to its corresponding primary system priority level.

Proof. When the system priority level of the alternative task of each task τ_i is mapped to its corresponding primary system priority level, $u=k$, $r=r'$. After some simple algebra, (9) and (12), respectively, can be rewritten as follows:

$$W_i^{\text{ext}}(t) = \sum_{j=1}^r C_j \lceil t/T_j \rceil + \sum_{\forall \tau_l \in g_k} C_l + \lceil t/T_E \rceil \max_{\bar{\tau}_u \in f_k - \{\bar{\tau}_i\}} \bar{C}_u, \text{ and}$$

$$W_i^{\text{int}}(t) = \sum_{j=1}^r C_j \lceil t/T_j \rceil + \sum_{\forall \tau_l \in g_k} C_l + \bar{C}_i + (\lceil t/T_E \rceil - 1) \max_{\bar{\tau}_u \in f_k} \bar{C}_u.$$

It is clear that if $\bar{C}_i = \max_{\bar{\tau}_u \in f_k} \bar{C}_u$, $W_i^{\text{ext}}(t) < W_i^{\text{int}}(t)$. Otherwise, $W_i^{\text{ext}}(t) > W_i^{\text{int}}(t)$.

The maximum of these two equations can then be rewritten as a simple equation, which yields (7). $\square$

4.3 An Illustrative Example

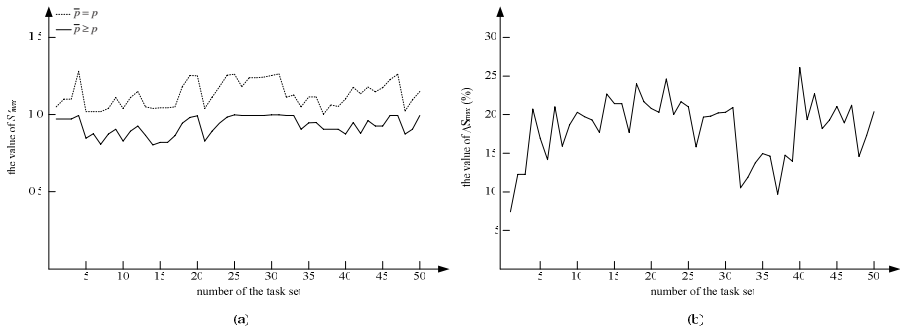
As mentioned earlier, when $m=2$, the task set presented in table 1 is unschedulable for the case that $\bar{p} = p$. Table 2 shows that, when the system priority level of the alternative tasks of task τ_4 is mapped within the priority group g_1 , the task set is schedulable according to the analysis described earlier. This is because the slack time available at higher system priority level is being used to execute $\bar{\tau}_4$. From this sense, the schedulability utilization of the system can be improved if the system priority levels of the alternative tasks are allowed to be mapped within higher priority group.

Table 2. $S'_{\max}$ for $\bar{p} \geq p$

$task$	T_i	C_i	$\bar{p}_i$	S'_i	
				ext	int
τ_1	10.53	1.04	2	0.52	0.46
τ_2	13.37	1.28	2	0.50	0.38
τ_3	15.56	1.46	1	0.41	0.33
τ_4	32.55	4.39	2	0.86	0.43
τ_5	50.63	2.88	1	0.91	0.54

5 Simulation Result

This section characterizes the effectiveness of the described approach by simulation, where 1000 task sets (5 tasks per task set) were generated when the system only supported two system priorities. The worst-case computation times of each task were generated according to a uniform distribution with minimum and maximum values set to 1 and 20. The periods and deadlines of tasks were assigned according to a uniform distribution with minimum and maximum values set to 10 and 100, respectively. Deadlines were allowed to be less than or equal to periods. Here, the system priorities of primary tasks were assigned by the FPA algorithm. In order to show the effectiveness of our proposed approach, we compared $\bar{p} \geq p$ with $\bar{p} = p$ (see Fig. 1). Here, we only give the result of 50 task sets among 1000 task sets.

**Fig. 1.** (a) The curve of $S'_{\max}$ (b) The curve of $\Delta S'_{\max}$

As can be seen from Fig. 1 (a), each of the value of $S'_{\max}$ of $\bar{p} = p$ from our simulation, say $(S'_{\max})_{\bar{p}=p}$, is more than 1. In other words, all of the simulated task sets are unschedulable. However, the case is different when alternative tasks are allowed to execute with higher priorities due to the fact that each of the value of $S'_{\max}$ of $\bar{p} \geq p$, say $(S'_{\max})_{\bar{p} \geq p}$, is less than 1. In order to show the gain in terms of the degree of schedulable saturation reduction that can be obtained from our proposed approach, we can quantitatively evaluate the relative schedulability of a system with

$\bar{p} = p$ compared to a system with $\bar{p} \geq p$ as $((S'_{\max})_{\bar{p}=p} - (S'_{\max})_{\bar{p} \geq p}) / (S'_{\max})_{\bar{p}=p}$, denoted as $\Delta S_{\max}$. As can be seen from Fig. 1 (b), the average obtained increment on $\Delta S_{\max}$ is about 18.3%. Therefore, in terms of the degree of schedulable saturation, the approach $\bar{p} \geq p$ is better than the approach $\bar{p} = p$ for the case of limited priority levels.

6 Conclusion

In this work we have addressed the problem of providing a schedulability analysis for fault-tolerant hard real-time systems regarding temporary faults with limited priority levels. We have proposed a suitable solution on the schedulability utilization of the system. One important characteristic of our solution is that it allows task recovery to be carried out at higher system priority levels. This, as we have seen, has flexibility of the analysis, where slack times of higher system priority tasks can be better explored. We have illustrated the advantages of using the described schedulability analyses. By analyzing the data collected from simulation we have been that significant reductions in terms of task recovery time may be obtained by applying the analysis.

Future works will consider heuristics for deciding which higher system priority levels to use for general task sets.

References

1. Lehoczky, J., Sha, L.: Performance of real-time bus scheduling algorithms. *ACM Performance Evaluation Review* 14(1), 44–53 (1986)
2. Lehoczky, J., Sha, L., Ding, Y.: The rate monotonic scheduling algorithm: Exact characterization and average case behavior. In: *Proceeding of the IEEE Real-Time Systems Symposium*, pp. 166–171 (1989)
3. Katcher, D., Sathaye, S., Strosnider, J.: Fixed priority scheduling with limited priority levels. *IEEE Transaction on Computers* 44(9), 1140–1144 (1995)
4. Orozco, J., Cayssials, R., Santos, J., Santos, R.: On the minimum number of priority levels required for the rate monotonic scheduling of real-time systems. In: *Proceeding of the 10th EUROMICRO Workshop on Real Time Systems*, Berlin, Germany (1998)
5. Bin, X.L., Yang, Y.H., Jin, S.Y.: Optimal fixed priority assignment with limited priority levels. In: Zhou, X., Xu, M., Jähnichen, S., Cao, J. (eds.) *APPT 2003*. LNCS, vol. 2834, pp. 194–203. Springer, Heidelberg (2003)
6. Wang, B.J., Li, M.S.: A priority mapping algorithm without affecting the schedulability of tasks set. *Journal of Computer Research and Development* 43(6), 1083–1089 (2006)
7. Wang, B.J., Li, M.S., Wang, Z.G.: Uniprocessor static priority scheduling with limited priority levels. *Journal of Software* 17(3), 602–610 (2006)
8. Audsley, N.C.: Deadline monotonic scheduling, Ph.D. dissertation, Dept. Computer Science, University of York (1990)

A Property-Based Technique for Tolerating Faults in Bloom Filters for Deep Packet Inspection

Yoon-Hwa Choi and Myeong-Hyeon Lee

Computer Engineering Department
Hongik University, Seoul, Korea
{yhchoi}@cs.hongik.ac.kr

Abstract. In network security applications, such as network intrusion detection, string matching is used to scan packets to detect malicious content. Bloom filters have drawn a great attention due to the fact that they can provide constant lookup times at the cost of small false positives. A fault in Bloom filters, however, cannot guarantee no-false-negatives. In this paper, we present a property-based technique for tolerating faults in Bloom filters for deep packet inspection. It employs a single spare hashing unit in each Bloom filter to detect and eliminate false negatives until the spare itself is faulty. The design is simple to be implemented in hardware. Moreover, the process for eliminating false negatives can be done without reducing the system throughput.

1 Introduction

String matching can be used to scan packets in network applications, including network intrusion detection. Predefined signatures have to be compared against the contents of any packet payload that passes through network ports. Since the location of such strings in the packet payload and their length is unknown, techniques for detecting strings of different lengths starting at arbitrary locations in the packet payload have been developed [2],[3],[6-8].

Bloom filters have drawn a great attention due to the fact that they can provide constant lookup times at the cost of small false positives [1]. Dharmapurikar et al. have presented a hardware-based technique using Bloom filters to achieve high-speed hashing and lookup operations at line speed [2]. They group signatures according to their length (in bytes) and store each group of signatures in a unique Bloom filter. An analyzer is employed to resolve false positives. Artan et al. [3] have proposed a space-efficient method to follow and detect signatures that are fragmented over multiple packets.

If there is a fault in Bloom filters, however, no false negatives cannot be guaranteed. A faulty hashing unit, for example, might generate an incorrect location at which 0 is stored instead of 1, resulting in a false negative. For a given fault, the probability that a false negative will occur due to the fault is extremely high unless some provisions are made to detect and eliminate them.

In this paper, we present an efficient technique for tolerating faults in Bloom filters for deep packet inspection. It is based on property checking of Bloom filters with a single spare hashing unit in each Bloom filter and immediate or delayed identification of faulty hashing units. Packets may proceed at line speed, even with the added circuits for fault tolerance.

2 Bloom Filters

A Bloom filter is a bit vector M of m bits, initially all set to 0, with k independent hash functions $h_1, h_2, \dots, h_k$ that map each element of a set $S = \{x_1, x_2, \dots, x_n\}$ to the set $\{1, 2, \dots, m\}$. The output of each hash function h_i is uniform over the set $\{1, 2, \dots, m\}$. Given a string x , the Bloom filter computes k hash functions on it, producing k hash values ranging from 1 to m . The filter then sets k bits in an m -bit vector M at the addresses corresponding to the k hash values. To check if a given string, y , belongs to S , the k independent hash functions are applied to y resulting in a set of locations. The filter then looks up the bits in the m -bit vector M at the locations corresponding to the k hash values to see if they are all 1's. If all these locations are set to 1, it accepts y with high probability as a member of S . On the other hand, if any of the mapped locations is zero, y is not a member of S . Thus a Bloom filter could result in false positives. However, it can never generate false negatives.

Fig. 1 shows a packet scanning system using a group of Bloom filters [2]. It consists of w hardware Bloom filters, a false positives resolver, and a hash table. Each Bloom filter contains signatures of a particular length. The system tests each string for membership in the Bloom filters. If it identifies a string to be a member of any Bloom filter, the system then declares the string to be suspicious. The string detected receives further probing by the false positives resolver, which determines if the string is indeed a member of the set, using the hash table.

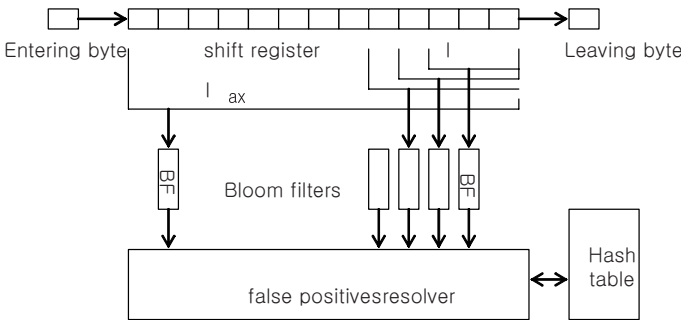


Fig. 1. A packet scanning system using hardware Bloom filters

The system reads as input a data stream that arrives at the rate of one byte per clock cycle. It then verifies the membership of each substring in a single

clock cycle, using the appropriate Bloom filter. All of the w strings are verified in parallel by the w Bloom filters. If none of the Bloom filters find a match, the data stream can advance by a byte. If they find a match, a hash table is queried to determine if an exact match has occurred. In the case of multiple matches at the same time in the Bloom filters, the false positives resolver probes the substrings, from longest to shortest. The search stops as soon as it first confirms the match of a substring [2].

A fault in Bloom filters in Fig. 1, may cause a false negative to occur. In the following section, we define our fault model for Bloom filters and present how to deal with faults to eliminate false negatives.

3 Fault-Tolerant Bloom Filter

In this paper, we assume that there is a single faulty hashing unit in a Bloom filter. A faulty hashing unit is assumed to generate a random hash value ranging from 1 to m . In other words, all the m possible hash values are equally-likely. Comparators to be employed are assumed to be fault-free. In addition, the bit vector M is self-checkable by employing an error checking code. Finally, the mean-time-between-failures (MTBF) is much longer than the worst case string checking time.

The fault model we use in this paper is functional and thus independent of the internal structure of the hashing units. Although a single fault is assumed, multiple hashing units may fail as long as the spare hashing unit to be addressed shortly is functional. Without loss of generality, we consider only a single Bloom filter among the w Bloom filters. Let S be the set of signatures for the Bloom filter and h_j be the j -th ($1 \leq j \leq k$) hash function. For a given string x the expected outputs of the k hash functions are denoted by $h_1(x), h_2(x), \dots, h_k(x)$, respectively. Suppose that there is a fault in the hashing unit for h_j . Then we need to consider the following two cases depending on the string x . In the description, $\tilde{h}$ denotes the function actually performed by the hashing unit for h , and is same as h if it is fault-free.

- (1) ($x \in S$): The hashing unit for h_j generates any number $\tilde{h}_j(x)$ between 1 and m . If $M[\tilde{h}_j(x)] = 0$, then a false negative occurs.
- (2) ($x \notin S$): The hashing unit for h_j , where $M[h_j(x)] = 0$, might generate an incorrect hash value $\tilde{h}_j(x)$ such that the bits at positions of the entire k hash values, including the incorrect one, in M are all 1's, resulting in a false positive. On the other hand, a false positive in a fault-free Bloom filter will not occur if $M[\tilde{h}_j(x)] = 0$.

Now we present our fault-tolerant Bloom filter shown in Fig. 2, where a single spare hashing unit h_s and zero-selector/fault-detector are added to the original Bloom filter. The primary role of the spare hashing unit is to detect false negatives (due to faults in a Bloom filter), not to identify faulty hashing units. Faulty units as long as they do not cause a false negative may remain active, until they are determined to be the cause of a false negative. Each Bloom filter now has

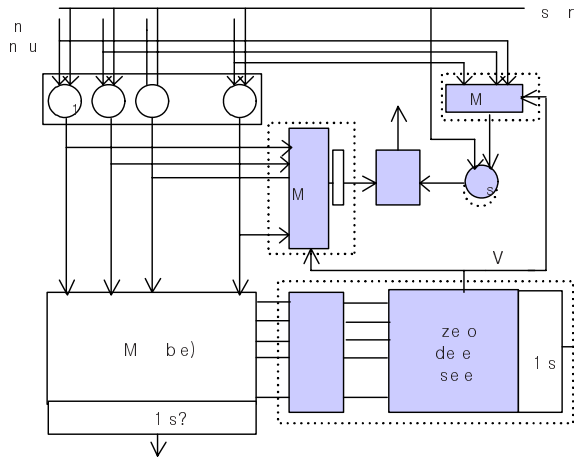


Fig. 2. A fault-tolerant Bloom filter

$k + 1$ hashing units. If some random numbers are involved in computing hash functions, like H_3 in [5], they can be made sharable by using simple parities for self checking.

If the string given is a member of S , the table lookups with k hash values will result in all 1's. Hence a necessary condition for a false negative to occur under the single fault assumption is a single zero in the table lookups. The zero selector is employed to check to see if the table lookups result in at least one zero, not necessarily a single zero. Upon detecting and selecting a zero, it activates the spare hashing unit, h_s , to perform the same hash function to see if there is a fault in the hashing unit selected. A single comparison is sufficient to detect a fault if it exists. Identifying the faulty hashing unit, although necessary, may be postponed as long as no false negatives can be guaranteed. If the faulty unit is identified based on the information at the time of error detection, it can be isolated from the rest of the system. Otherwise, the Bloom filter may continue scanning without interruption since it still behaves correctly.

One way of selecting a zero is shown in Fig. 2, where a priority selection logic is used to find the *id* (or *vector* V) of a hash function with a zero in the table(M) lookup. Disable logic is employed to selectively disable (set the output to 1) the results of table lookups, effectively isolating some of the hashing units. Suppose that for a given string x the zero selector finds that h_i is a hash function with $M[\tilde{h}_i(x)] = 0$. Then $\tilde{h}_s(x)$ will be compared with $\tilde{h}_i(x)$ (stored in R) to see if they match. If they do, they will be determined to be fault-free. Otherwise, either $\tilde{h}_i(x)$ or $\tilde{h}_s(x)$ is incorrect, but we do not know which one is erroneous. Removing the hashing unit h_i under suspicion without identifying the faulty unit might increase false positives, resulting in reduced throughput. To maximize the system performance while guaranteeing no-false-negatives we need to find a way

to either locate the faulty hashing unit or use the suspicious hashing units h_i and h_s without committing false-negatives.

To guarantee no false-negatives even with an unidentified faulty hashing unit being active, we take into account $M[\tilde{h}_s(x)]$ and $A[x]$, the decision that can be made by the false positives resolver on the membership of x . Here $A[x]=1$ denotes that x is a member of the set S . Table 1 shows the possible outcomes with some explanations.

From the first row, we cannot identify the faulty hashing unit. However, the system is still functional since x is not a member string. The second row can never occur under the single fault assumption, since for $A[x]=1$ (i.e., x is a member) both $M[\tilde{h}_i(x)]$ and $M[\tilde{h}_s(x)]$ cannot be zero at the same time, while $\tilde{h}_s(x) \neq \tilde{h}_i(x)$. In the third row, we can determine that the hashing unit for h_i is faulty due to the conflict between $M[\tilde{h}_i(x)]=0$ and $A[x]=1$. Under a single fault assumption we can assume that the resolver with a hash table is fault-free. In fact, the hash table is an external memory which can be made self-checkable by using a coding scheme. The last row does not provide any clue on diagnosis. The fact that x is not a member implies that both $M[\tilde{h}_i(x)]$ and $M[\tilde{h}_s(x)]$ may assume any value, either zero or one.

From Table 1 we can find that $M[\tilde{h}_i(x)]=0$ and $A[x]=1$ is sufficient to determine that the hashing unit for h_i is faulty and a false negative has occurred. In all other cases, the Bloom filter behaves correctly even if there is a faulty hashing unit unidentified.

Hence in our design the string x will be sent to the resolver when $M[\tilde{h}_i(x)]=0$ and $\tilde{h}_s(x) \neq \tilde{h}_i(x)$ to see if the hashing unit for h_i is faulty, without checking the value of $M[\tilde{h}_s(x)]$. If it is determined to be a non-member, the Bloom filter may continue its normal operation until the faulty hashing unit is identified based on the subsequent string scanning operations. Sending a string x to the resolver due to a fault in an unknown hashing unit does not cause any burden to the resolver since the mean-time-between-failures (MTBF) is expected to be much longer than the clock cycle time of the Bloom filters.

The time required for detecting an incorrect hash value by comparison is expected to be longer than the clock cycle time. Hence we split the required work for eliminating a false negative into two parts and do the work in a pipelined fashion. In the first part, a zero in the table lookups is detected and the corresponding hash function, h_i , will be identified. In the second part, the spare hashing unit is activated to compute the same hash function h_i in h_s and the

Table 1. Interpretation of $M[\tilde{h}_s(x)]$ and $A[x]$ when $M[\tilde{h}_i(x)]=0$ and $\tilde{h}_s(x) \neq \tilde{h}_i(x)$

$M[\tilde{h}_s(x)]$	$A[x]$	explanation
0	0	unidentified, but $x \notin S$
0	1	impossible
1	1	The hashing unit for h_i is faulty
1	0	unidentified, but no false negative

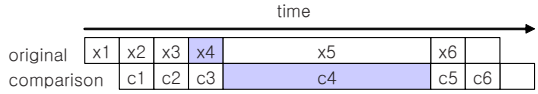


Fig. 3. A two-stage pipeline for eliminating false negatives

resulting hash value $\tilde{h}_s(x)$ is compared with $\tilde{h}_i(x)$. In the case of a mismatch, the second cycle is extended by preventing the shift register from advancing as illustrated in Fig. 3, where c_4 shows a mismatch in comparison, resulting in a cycle extension. Accordingly x_5 has to wait until the end of c_4 .

In the two-stage process, the spare hashing unit generates a hash value in the second cycle. Hence one byte leaving the Bloom filter should be either stored or made available. Although a (one-cycle) delayed comparison is performed, it does not cause any problem. If the Bloom filter is determined to be fault-free, this additional work already done is simply ignored since the filter has already advanced to the next round. If the Bloom filter is determined to be faulty, the faulty hashing unit needs to be removed to resume normal filter operation.

4 Delayed Fault Identification

The proposed technique can eliminate false-negatives even with a faulty hashing unit being active in the system. However, it cannot locate the faulty unit except for some special case (third row in Table 1). In this section, we present a technique for delayed identification of a faulty unit under the given fault model. In the description, $(x_1, x_2, \dots, x_n)$ represent the input strings to a Bloom filter at cycles 1, 2, ..., and n , respectively.

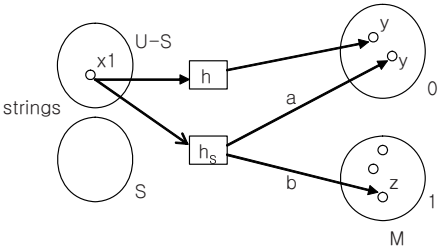


Fig. 4. Fault detection when x_1 is not a member of S

Suppose that a fault is detected when a Bloom filter is inspecting the string $x_1 \notin S$ but we cannot determine which hashing unit is faulty as shown in Fig. 4, where a and b correspond to the first and last rows in Table 1, respectively. Then we let the system continue its normal operation with the zero-selector input for h_i set to 1 by the disable logic. This will prevent any additional comparisons

between $\tilde{h}_i$ and $\tilde{h}_s$ while inspecting subsequent strings. The reason for momentarily disabling h_i is that additional comparisons between h_i and h_s will be useless unless the input string is a member of S .

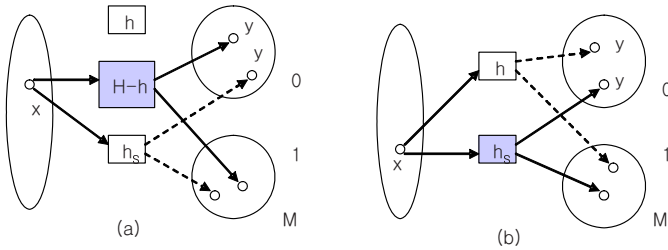


Fig. 5. Two additional steps for delayed fault identification

In the next cycle, if the string x_2 is not a member string and a false positive does not occur, h_j ($j \neq i$) will be selected by the zero selector and $h_j(x_2)$ will be compared with $\tilde{h}_s(x_2)$ as shown in Fig. 5(a), where the shaded block contains all the fault-free $k-1$ hash functions including h_j selected. If there is a mismatch, we can conclude that the spare hashing unit h_s is faulty. Otherwise, we have to consider the following three cases. (1) It is a transient fault in the hashing unit h_s . (2) It is a permanent fault in the block for h_s , but the hash value $\tilde{h}_s(x_2)$ is accidentally the same as $h_j(x_2)$. (3) The fault, either transient or permanent, has occurred in the hashing unit for h_i .

In cases (1)(3), we cannot make any progress in diagnosis even if we perform additional checking. Hence if (2) is not true, we need to move on to the block for h_i to see if it is faulty. To make sure that (2) is not true, in the next cycle with x_3 , $\tilde{h}_s(x_3)$ will be compared with $h_p(x_3)$, where $p \neq i$. If they match, we have more confidence to determine that the block for h_s does not have a permanent fault. Under the assumption that strings to be inspected are independent, the probability that at least q consecutive matches, excluding the cases for all 1's in table lookups, will occur is expected to approach zero rapidly as q increases, although it depends on the hashing circuits employed. In [5] a class of universal hash functions are introduced using AND and XOR logics. A fault in the circuit can be detected with a high probability using a relatively small number of random test inputs. A correct diagnosis is desirable, but not necessary each time a comparison mismatch occurs as long as the filter does not allow any false negatives.

Comparison cycles for delayed fault identification described above will follow their corresponding original cycles as shown in Fig. 6(a). If the signal for all 1's is 1, no comparison is performed as illustrated in Fig. 6(b), where c_3 and c_7 become *idle* cycles. These cycles may be used to compare $\tilde{h}_i$ and $\tilde{h}_s$ since the third row in Table 1 can now be applied. Since those cases are expected to be less frequent, they are ignored without loss of generality in the description of delayed fault identification.

If it is quite sure that the block for h_s does not have a permanent fault, we change the direction to the block for h_i to see if it has a permanent fault. In the following cycle, say x_v is the input string, $\tilde{h}_i(x_v)$ is compared with $h_s(x_v)$ as shown in Fig. 5(b), where the block for h_s is shaded to indicated that it is now treated as fault-free. If they show a mismatch, we can conclude that the unit for h_i has a permanent fault, and it thus needs to be removed. On the other hand, if q consecutive matches, excluding the idle cycles, occur, we can claim that the fault was not permanent. Although a transient fault cannot be located, no false negative has occurred and the system may go back to normal operation without reconfiguration.

5 Extended Fault Model

So far we have assumed that there are faults in hashing units in Bloom filters and the added circuits for eliminating false negatives are fault-free. In this section, we extend the fault model to cover faults in the added circuits and check to see if the proposed techniques can still be applied to guarantee no false negatives. Faults in the following three blocks, dotted in Fig. 2, are considered in this section: 1) Disable logic and zero selector; 2) Multiplexer along with a register for delayed comparison (MUX for short); 3) MUX₂. For each of three dotted blocks, we discuss how to deal with faults in it. For convenience, we use L , V , $match$ (in Fig 2), to denote the result of table lookup, the output of the zero-selector, and the output of the circuit for checking the condition of all 1's in L , respectively. If the zero selector is faulty, it might generate an erroneous vector (i.e., an incorrect hashing unit), leading to the comparison of two hash values of an incorrectly selected hash function. Four possible cases, including error-free, assuming that the checker checking for all 1's is fault-free and none of the k hashing units are disabled, are shown in Table 2.

Table 2 shows that either the faulty zero selector can be correctly identified or the Bloom filter still behaves correctly. Hence we can claim that faults in the zero-selector can be covered as long as the checker checking for all 1's is fault-free. If this assumption appears to be strong, we can employ an additional checker right after the zero selector in Fig. 2 to localize the fault. If the two

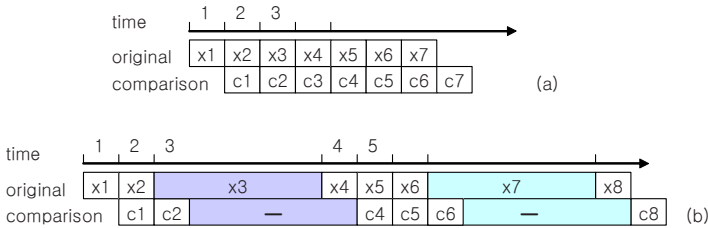


Fig. 6. A two-stage pipeline: (a) without all 1's, (b) with all 1's, in table lookup

Table 2. Interpretation of L , V , and $match$, when there is a fault in zero selector. In the table, $\bar{1}$ denotes all 1's (i.e., $\bar{1}$ ='111...1').

L	V	match	explanation
$\bar{1}$	$\bar{1}$	1	error-free
$\bar{1}$	not ($\bar{1}$)	1	faulty zero selector
not ($\bar{1}$)	$\bar{1}$	0	faulty zero selector
not ($\bar{1}$)	not ($\bar{1}$)	0	undetectable, but still safe

checkers do not show the same output, we can conclude that the hashing units are fault-free, under the single fault assumption. In the last row of Table 2, if the zero selector is faulty, it might generate a vector (i.e., an id) which is different from the correct one. As long as $M[h_{id}(x)] = 0$, we are safe since we still do not allow any false negatives. Eventually, faults in the zero selector will be identified when $M[h_{id}(x)] = 1$.

If the second block (MUX) is faulty, an incorrect hash value, either modified or incorrectly selected, might be compared against a correct hash value generated by the spare hashing unit, resulting in a mismatch. A mismatch in comparison, in this case, makes the fault diagnosis more complicated since the fault cannot be distinguished from a fault in a hashing unit since both will result in a mismatch. As we discussed in the previous section, however, this added fault does not cause a false negative. Hence we can still claim that the Bloom filter is functioning correctly even with the additional type of fault. If there is a fault in the third block (MUX₂), it can be treated as a fault in the spare hashing unit h_s . In any case, the Bloom filter runs free of false negatives.

6 Graceful Degradation

In the proposed Bloom filter, if a permanent fault is identified, we can reconfigure the system in one of two different ways: i) Maintain k hash functions with a slight modification like the original Bloom filter without error detection. ii) Reduce the number of hash functions to $k-1$, without modifying the table M , and use the spare to detect false negatives. In the second approach, the performance of the system will degrade.

Now we estimate the amount of increase in false positive probability for a Bloom filter operating in a degraded mode. For a given set S of n elements to support membership queries and k hash functions of range m , the probability that a particular bit of the bit vector M of length m is still 0 is given by $P_0 = (1 - \frac{1}{m})^{kn} \approx e^{-\frac{kn}{m}}$. Hence the probability of a false positive, P_{fp} , can be written as $P_{fp} = (1 - (1 - \frac{1}{m})^{kn})^k \approx (1 - e^{-\frac{kn}{m}})^k$. When a faulty hashing unit is removed, the probability of a false positive $P_{fp}^{deg} = (1 - (1 - \frac{1}{m})^{kn})^{k-1} \approx (1 - e^{-\frac{kn}{m}})^{k-1}$. The increase in the probability of a false positive is $\frac{P_{fp}^{deg}}{P_{fp}} \approx \frac{1}{1 - e^{-\frac{kn}{m}}}$. It is well known that P_{fp} can be minimized when k is equal to $\ln 2 \cdot (\frac{m}{n})$ [4]. If the condition is satisfied, the ratio becomes 2. That is, the number of false positives is doubled

for the proposed Bloom filter in the degraded mode. Each time an additional false positive occurs, some extra delay is necessary for the resolver to get rid of it. Consequently, some reduction in the system throughput is unavoidable.

7 Conclusions

In this paper, we have presented a property-based technique for tolerating faults in Bloom filters for deep packet inspection. It checks the invariant of Bloom filters to see if false negatives occur during normal operation by employing a single spare hashing unit in each Bloom filter. Hashing units with a permanent fault have been identified either right away or with some delay without reducing system throughput. Hashing units with a transient fault cannot always be located. However, any false negatives induced by faults in Bloom filters have been eliminated without any interruption. The design is simple enough to be implemented in hardware. Moreover, the proposed fault-tolerant Bloom filter can operate in a degraded mode in the event of a failure.

References

1. Bloom, B.: Space/time tradeoffs in hash coding with allowable errors. *Communications of the ACM* 13(7), 422–426 (1970)
2. Dharmapurikar, S., Krishnamurthy, P., Sproull, T.S., Lockwood, J.W.: Deep packet inspection using parallel Bloom filters. *IEEE Micro*, 52–61 (2004)
3. Artan, N.S., Chao, H.J.: Multi-packet signature detection using prefix Bloom filters. *IEEE GLOBECOM*, 1811–1816 (2005)
4. Broder, A., Mitzenmacher, M.: Network applications of Bloom filters: A survey. *Internet Mathematics*, 485–509 (2003)
5. Ramakrishna, M.V., Fu, E., Bahcekapilli, E.: Efficient hardware hashing functions for high performance computers. *IEEE Trans. Computers* 46(12), 1378–1381 (1997)
6. Tan, L., Sherwood, T.: A high throughput string matching architecture for intrusion detection and prevention. *IEEE Int. Symp. Comput. Arch.* 112–122 (2005)
7. Sourdis, I., Pnevmatikatos, D.N., Wong, S., Vassiliadis, S.: A reconfigurable perfect-hashing scheme for packet inspection. In: *IEEE Int. Conf. Field Programmable Logic and Applications*, pp. 644–647 (2005)
8. Tuck, N., Sherwood, T., Calder, B., Varghese, G.: Deterministic memory-efficient string matching algorithms for intrusion detection. *IEEE Infocom*, pp. 2628–2639 (2004)

A Fuzzy Logic Approach for Secure and Fault Tolerant Grid Job Scheduling

Congfeng Jiang, Cheng Wang, Xiaohu Liu, and Yinghui Zhao

Engineering Computing and Simulation Institute,
Huazhong University of Science and Technology,
430074 Wuhan, China
congfang.jiang@smail.hust.edu.cn,
{wangch, xhliu}@mail.hust.edu.cn,
zhaoyinghuihust@gmail.com

Abstract. Secure grid computing needs fault tolerant job scheduling with security assurance at grid sites. However, the uncertainties of grid sites security and user jobs are main hurdle to make the job scheduling secure, reliable and fault-tolerant. Job replication is usually used in grids to provide fault tolerance and high scheduling success rate. A Fuzzy-logic based Self-Adaptive job Replication Scheduling (FSARS) algorithm is proposed to handle the fuzziness or uncertainties of job replication number which is highly related to trust factors behind grid sites or user jobs. Remote Sensing Based Soil Moisture Extraction (RSBSME) experiments were run to evaluate the proposed approach and the results show that a higher scheduling success rate and less grid resource utilization can be achieved through FSARS. And FSARS is applicable for grids where security conditions fluctuate frequently.

1 Introduction

Computational Grids[1] are motivated by the desire to share resources among many virtual organizations to solve large-scale problems. In a large-scale grid, distributed resources belong to different administrative domains. Job executions are usually carried out between many virtual organizations in business applications or scientific applications for faster execution or remote interaction. However, grid security is a main hurdle to make the job scheduling secure, reliable and fault-tolerant. A lot of algorithms have been developed for scheduling jobs in grids [2,3,4,5]. Unfortunately, most of the existing proposed scheduling algorithms had ignored the security problem while scheduling jobs onto geographically distributed grid sites with a handful of exceptions. Thus the existing proposed heuristics are not applicable in a risky grid environment.

Job replications are commonly used to provide fault-tolerant scheduling in grids. However, existing job replication algorithms use a fixed number replication[4,5]. Thus an adaptive job replication is necessary for real grid job scheduling. Although a ladder-like adaptive number of job replication can partially solve this problem, the transformation process from grid security conditions to replication number of each job

may bring the sharp-border problem. Thus, we apply the concept of fuzzy set to the process from grid security conditions to replication number of each job so as to avoid the deviation caused by the sharp-border problem.

In this paper, we have tackled the secure grid scheduling problem and offered a Fuzzy-logic based Self-Adaptive job Replication Scheduling (FSARS) algorithm for use under failure-prone and risky conditions. We then compare the proposed algorithm with existing heuristics based on fixed number of job replications. Experiment results show that security assurance performance could be achieved if we use fuzzy logic to decide the number of job replications when scheduling.

The rest of the paper is organized as follows: Section 2 presents a brief review of related work. In Section 3, we present the fuzzy-logic based self-adaptive job replication model. In Section 4, we present experimental results and discuss the relative performance and scalability of the proposed self-adaptive job replication algorithm. Finally, we summarize the works in Section 5.

2 Related Works

Trust and security challenges within the Grid environment are driven by the need to support scalable, dynamic distributed Virtual Organization[1]. Azzedin and Maheswaran[6] suggested integrating the trust concept into grid resource management. In this paper, we focus on how to establish a fuzzy-logic based self-adaptive job replication model and how the job replication number affects the overall performance of the user jobs in grids.

Abawajy[5] presented a Distributed Fault-Tolerant Scheduling (DFTS) to provide fault tolerance for job execution in a grid environment. Their algorithm uses fixed number replications of jobs at multiple sites to guarantee successful job executions. In our paper, we use an adaptive job replication scheme such that the number of replications could change with the security level of the grid environments.

Song [7] developed a security-binding scheme through site reputation assessment and trust integration across grid sites. In this paper, the advantage of using fuzzy-logic to quantify replication number of each job in Grid is that fuzzy inference is capable of quantifying imprecise data or uncertainty when deciding the replication number of each job.

Our work is based on the related works on grid security, fuzzy theory, and fault-tolerant job scheduling. We use self-adaptive replication based algorithm which was mostly ignored in the past [4, 5].

3 Job Replication Model Based on Fuzzy Logic

The security level of a Grid site must be dynamically changing in nature, because there is no way to predict when and where a Grid will be under attack or crash. Similarly, an application's security demand is also changing with time. In our work, as in [4], we first assign a security demand (*SD*) to a user job when user submitted it. The trust model assesses the resource site's trustworthiness, namely, the trust level (*TL*). The *TL* quantifies how much a user can trust a site for successfully executing a

given job. In this paper, SD and TL are supplied as a single parameter by the user applications and the site respectively. Only the job can be successfully finished when SD and TL satisfy a security assurance condition ($SD \leq TL$) when scheduling the jobs. This is similar to the real life scenario when one surfs the net he/she is required to specify the security level of his internet explorer, such as *very high*, *high*, *middle*, *low*, *very low*, etc. The lower the security level he/she specifies, the more sites he/she can access, and the more risks he/she will take.

Here, a major challenge is that both SD and TL are dynamic quantities. The typical attributes that the user cares in determining its security demand includes job execution success rate, data integrity, access control, etc[4, 8]. The trust factors of grid site include site reputation, prior job success rate, firewall, etc. These attributes and their values are dynamically changing and depend heavily on the trust model and security policy. In this paper, we assume that there is a central server that collects job execution success rates, firewall capabilities, grid utilization, and other performance data of the sites periodically. In the initialization of the scheduling, the trust level of the site is computed through the performance data mentioned above. Then, the trust level is updated periodically with the site operation. This can be achieved by using some network or grid services like NWS (Network Weather Service)[9] and MDS (Monitoring and Discovery System)[10] when scheduling.

Concerning the above characteristics, we do not set the job replication number deterministically. Instead, we propose a Fuzzy-logic based Self-Adaptive job Replication Scheduling algorithm to handle the uncertainties of the job replication number.

3.1 System Model

In this paper, we assume that the application has been divided into sub tasks and each sub task is independent and the tasks have no deadlines and priorities. This assumption is commonly made when studying scheduling problems for grids (e.g., [2, 4,5]).However, scheduling jobs with priorities or DAG(Directed Acyclic Graph) topologies can be found in [11,12].In this paper, the terms jobs and tasks are used interchangeably.

Let $M = \{m_j \mid j = 1, 2, 3, \dots, m\}$ denote the hosts set, and $T = \{t_i \mid i = 1, 2, 3, \dots, n\}$ denote the tasks set. We define the following parameters:

(1) p_j : The speed of host m_j (MFlops);

(2) e_{ij} : Expected time to compute when schedule task t_i to host m_j . We assume that the estimates of expected task execution times on each machine in the grid sites are known. The assumption is commonly made when studying scheduling problems for grids or Heterogeneous Computing (HC) systems (e.g.,[2,11,12]).Approaches such as code profiling ,analytic benchmarking, and statistical prediction for doing this estimation can be found in [13,14,15].

(3) SD_i : The security demand of task t_i . SD_i is specified when the task is submitted, and SD_i is a real fraction in the range $[0, 1]$ with 0 representing the lowest and 1 the highest security requirement. In some grid environments, setting SD_i equal

to 1 is unnecessary although it seems to be risk-free. If SD_i is always equal to 1, maybe there are no sites could satisfy the security demand. For example, in a volunteer grid environment such as SETI@Home[16], the security demand of jobs may be lower than a computation-intensive and real-time scientific grid computing or e-commerce environment, in order to earn volunteer computing power as more as possible.

(4) TL_j : The trust level of host m_j . TL_j is in the same range $[0, 1]$ with 0 for the most risky resource site and 1 for a risk-free or fully trusted site. And TL_j can be computed through the approach in [7].

(5) q_i : The number of hosts that satisfy $SD \leq TL$ for task t_i ;

(6) $\overline{SD}$: The security demand level of the task set. $\overline{SD}$ is in the range $[0, 1]$ with 0 representing the lowest and 1 the highest security requirement. $\overline{SD}$ is computed as follows:

$$\overline{SD} = \frac{\sum_{i=1}^n (SD_i \times \sum_{j=1}^{q_i} \frac{e_{ij}}{q_i})}{\sum_{i=1}^n \sum_{j=1}^{q_i} \frac{e_{ij}}{q_i}} \quad (1)$$

The definition of $\overline{SD}$ indicates that the security demand level of the task set is correlated to the expected time to compute of the tasks in the task set.

(7) $\overline{TL}$: The trust level of the grid environment. $\overline{TL}$ is in the range $[0, 1]$ with 0 for the most risky grid environment and 1 for a risk-free or fully trusted grid environment. $\overline{TL}$ is computed as follows:

$$\overline{TL} = \frac{\sum_{j=1}^m TL_j \times p_j}{\sum_{j=1}^m p_j} \quad (2)$$

We assume that there is a central server or process that collects both SD_i and TL_j of user jobs. Then $\overline{SD}$ and $\overline{TL}$ can be computed by the scheduler through Eq.1 and Eq.2. In real grids, $\overline{SD}$ and $\overline{TL}$ can also be maintained by services like MDS [10].

(8) SE_i (Security Error ratio): The difference ratio (also called security error ratio) between $\overline{TL}$ and SD_i for task t_i , i.e. $SE_i = \frac{\overline{TL} - SD_i}{SD_i}$

where SE_i is in the interval $[-1, +\infty)$.

(9) K_i : The replication number of each job when scheduling. We set K_i in the interval $[0, 4]$ according to our previous application experiences. We choose 4 as the maximum number of replicas because in our real experiments when the number of replicas becomes larger than 4, the system performance degrades heavily.

3.2 Fuzzy Inference Process

A fuzzy set expresses the degree to which an element belongs to a set. The characteristic function of a fuzzy set is allowed to have values between 0 and 1, which denotes the degree of membership of an element in a given set. For the transformation from grid security conditions to fuzzy set, we provide the empirical membership functions in Figure 1.

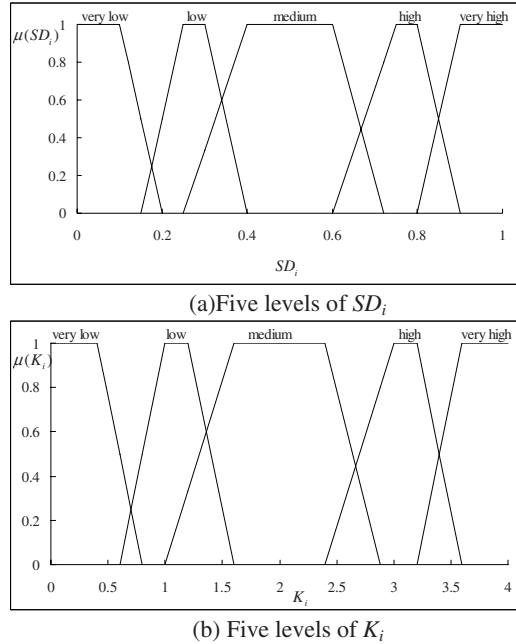


Fig. 1. Membership functions for different levels of SD_i and K_i

In this paper, fuzzy inference is a process to decide the replication number of each job in four steps:

Step 1. Compute the initial values of SD_i , TL_j , $\overline{SD}$, $\overline{TL}$ and SE_i .

Step 2. Use the membership functions to generate membership degrees for SD_i , $\overline{SD}$, and SE_i .

Step 3. Apply the fuzzy rule set to map the input space (SD_i , TL_j , $\overline{SD}$, $\overline{TL}$ and SE_i space) onto the output space (K_i space) through fuzzy operations.

Step 4. Derive the replication number of each job through a defuzzification process.

For example, we consider initial values: $SD_i=0.7$, $\overline{SD}=0.6$, $\overline{TL}=0.5$ obtained from our applications. Two semi-empirical example fuzzy inference rules are given in the following for use in the inference process:

Rule 1: IF SD_i is *high*, $\overline{SD}$ is *medium*, and SE_i is *medium*, THEN K_i is *medium*.

Rule 2: IF SD_i is *very low*, $\overline{SD}$ is *medium*, and SE_i is *very low*, THEN K_i is *very high*.

The reason why these particular membership functions and rules are chosen is that in our experiments, the grid system can achieve best performance when we use these membership function and rules. In a real grid environment, particular membership function and rules should be chosen according to the system architecture or network topologies, grid site's dynamic security capabilities, such as security policies, intrusion detection, firewall, intrusion response capabilities, self-defense capability, site vulnerability, etc.

Through the job replication number inference process using the membership functions in Fig.1, we can deduce that K_i is 2.3. Then we round the result into an integer 2. Thus, the real job replication number of task t_i is 2.

3.3 Scheduling Algorithm

The following is the pseudo-code of the job scheduling algorithm using fuzzy-logic based self-adaptive replication.

```

1. When scheduling event occurs {
2.   Compute  $\overline{TL}$  and  $\overline{SD}$ 
3.   For each task in  $T$ 
4.     Compute  $SE_i$ 
5.     Compute  $K_i$  using fuzzy inference process
6.     Replicate the current job with  $K_i$  replications
7.     If there are available hosts for scheduling
8.       Schedule the job to  $K_i + 1$  machines with  $TL \geq SD$ 
9.       Delete the task from  $T$ 
10.    Else
11.      Insert the task into next scheduling tasks set
12.      Delete the task from  $T$ 
13.    End if
14.  End for
15. }
```

When the number of jobs in the job set becomes a fixed maximum number, like 100, we call this a scheduling event. When scheduling event occurs, FSARS first computes $\overline{TL}$ and $\overline{SD}$. For task t_i in task set T , the scheduler first compute the security error ratio (i.e. SE_i) between the job's security demand and grid system security level. Then, the scheduler computes the job replication number of t_i , i.e., K_i . After that, the scheduler chooses a set of $K_i + 1$ candidate sites with $TL \geq SD$ for job execution. Then Delete the task from T . If there is no available host for scheduling this job, insert the task into next scheduling tasks set and delete the task from T .

4 Experiment Results and Performance Analysis

4.1 Experiment Setup and Parameters Settings

We test the performance of our fuzzy-logic based self-adaptive job replication scheduling algorithm on RSBSME (Remote Sensing-Based Soil Moisture Extraction) application workload. The RSBSME workload is a typical data-intensive and high-throughput computing application on Grid system. The RSBSME is defined as a set of independent parallel jobs and each job computes a part of a large remote sensing image containing soil moisture information.

To model the heterogeneity of Grid sites, each site has different processing speed and initial trust level. The site runs various operating systems such as Linux, Windows, FreeBSD, etc. Once a job is submitted, it could be scheduled to other sites for execution, as governed by the decisions of the scheduler. The initial job security demands are normally distributed in the range [0.1, 0.9] and the initial site trust levels are uniformly distributed in the range [0.1, 1]. The job security demands and site trust levels dynamically and randomly change during the experiments.

We studied and compared the performance of the simple and frequently used heuristics such as Min-min [2], R-Min-min [4], and DT-Min-min [4], with FSARS. To evaluate FSARS, we use the following metrics:

- *Makespan*: the total running time of all jobs;
- *Scheduling success rate*: the percentage of jobs successfully completed in the system;
- *Grid utilization*: defined by the percentage of processing power allocated to user jobs out of total processing power available over all Grid sites;
- *Average waiting time*: the average waiting time spent by a job in the system.

4.2 Experiment Results and Relative Performance

The experiment results are shown in Figure.2. All the data in the figures are mean values of 10 experiment results, the waiting factor in DT-Min-min heuristic is 0.2, and the number of replication in R-Min-min is 2 for all experiments. In our experiments, a task will be dropped if it couldn't finish successfully after five times. Thus, the scheduling success rate can't reach to 100%.

The results in Fig. 2(a) suggest that the makespan order of the four scheduling algorithms from maximum to minimum is: (1) R-Min-min, (2) DT-Min-min, (3) FSARS and (4) Min-min. The makespan of R-Min-min algorithm is the largest because R-Min-min uses fixed-number of replications. More replications would be executed by R-Min-min algorithm when grid security level is high and thus make the CPU queue longer and eventually the makespan. Min-min algorithm schedules the tasks once, regardless of grid security level. The makespan of FSARS is also relatively small because of its self-adaptive job replication. The reduction of actual tasks execution makes the makespan of FSARS is relatively small.

We observed from Fig. 2(b) that FSARS has the highest successful scheduling rate in a failure-prone grid environment. FSARS and R-Min-min have higher successful scheduling rates because of job replication execution. Min-min has the lowest successful scheduling rate because it schedules tasks once, regardless of grid security conditions.

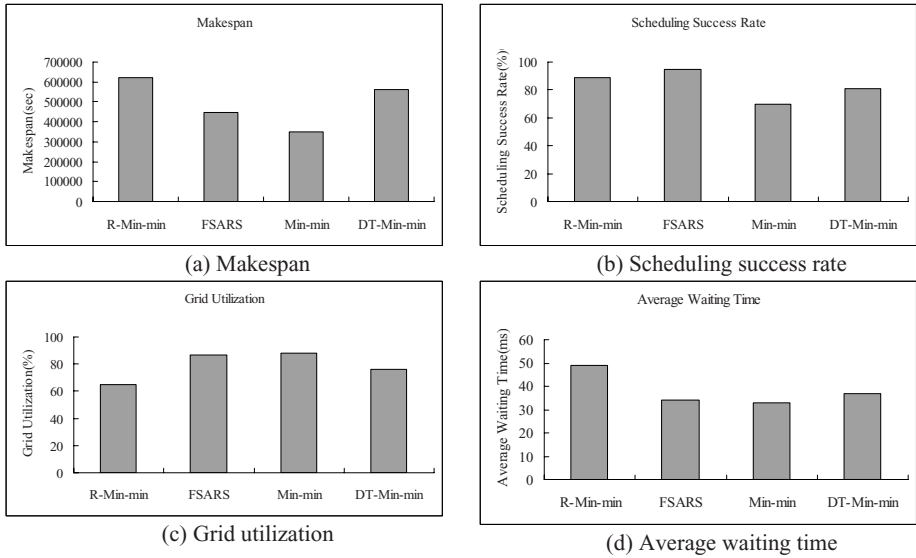


Fig. 2. Relative performances

We observed from Fig. 2(c) that Min-min has the highest grid utilization because it didn't replicate the user jobs. And FSARS have almost the same grid utilization with Min-min because it uses self-adaptive job replication which can reduce the total job replication execution significantly.

The results in Fig. 2(d) suggest that R-Min-min has the longest average waiting time because it executed excessive task replications. And FSARS has the relatively short average waiting time due to its self-adaptive job replication mechanism.

From Fig.2 we can see that no single algorithm achieves the highest performance for all metrics. However, FSARS algorithm exhibits relatively better performance with a highest success rate, moderate level of makespan, grid utilization, and average waiting time due to its adaptive job replication scheme.

In summary, FSARS changes the number of replications dynamically according to the dynamicity of the grid security conditions. So FSARS is applicable to the grid where the security conditions change frequently and can reduce the number of total job replications.

5 Conclusions

A fixed number of job replications in scheduling strategies may utilize excessive hosts or resources. This makes the makespan and average waiting time of tasks rather longer. Thus an adaptive replication strategy is necessary in a real grid with dynamic security conditions. We have studied Fuzzy-logic based Self-Adaptive job Replication Scheduling (FSARS) algorithm that are of use when minimizing job replications and improving fault tolerance in grid environments. We applied fuzzy theory to handle the fuzziness and uncertainties when deciding the replication number of user jobs. And

we have compared makespan, success rate, grid utilization and average waiting time of FSARS with other strategies and shown that the task scheduling in a real grid can dramatically be improved by introducing fuzzy-logic based self-adaptive job replication scheduling algorithm. Thus FSARS is applicable to security-driven and fault-tolerant grid job scheduling.

Acknowledgments

The funding support of this work by Innovation Fund of Huazhong University of Science and Technology (HUST) under contract No.HF04012006271 is appreciated.

References

1. Foster, I., Kesselman, C., Tuecke, S.: The anatomy of the grid: enabling scalable virtual organizations. *International Journal of High Performance Computing Applications* 15, 200–222 (2001)
2. Braun, T.D., Hensgen, D., Freund, R., Siegel, H.J., et al.: A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. *J. Parallel Distrib. Comput.* 61, 810–837 (2001)
3. Dogana, A., Özgüner, F.: Scheduling of a meta-task with QoS requirements in heterogeneous computing systems. *Journal of Parallel and Distributed Computing* 66, 181–196 (2006)
4. Song, S., Hwang, K., Kwok, K.: Risk-resilient heuristics and genetic algorithms for security-assured grid job scheduling. *IEEE Trans. on Computers* 55, 703–719 (2006)
5. Abawajy, J.H.: Fault-tolerant scheduling policy for grid computing systems. In: *Proceedings of IEEE 18th International Parallel and Distributed Processing Symposium*, pp. 238–244. IEEE CS Press, Los Alamitos (2004)
6. Azzedin, F., Maheswaran, M.: Integrating trust into grid resource management systems. In: *Proceedings of International Conference on Parallel Processing (ICPP'02)*, pp. 47–54. IEEE Computer Society Press, Los Alamitos (2002)
7. Song, S., Hwang, K., Kwok, Y.: Trusted grid computing with security binding and trust integration. *Journal of Grid Computing* 3, 53–73 (2005)
8. Arenas, A. (ed.): State of the art survey on trust and security in Grid computing systems. CCLRC Technical Report, RAL-TR-2006-008, pp. 9–21 (2006)
9. Wolski, R.: Forecasting network performance to support dynamic scheduling using the Network Weather Service. In: *Proceedings of the 1997 6th IEEE International Symposium on High Performance Distributed Computing (HPDC97)*, pp. 316–325. IEEE Press, Piscataway (1997)
10. Schopf, J.M., D'Arcy, M., Miller, N., et al.: Monitoring and Discovery in a Web Services Framework: Functionality and Performance of the Globus Toolkit's MDS4. Available at <http://www-unix.mcs.anl.gov/~schopf/Pubs/mds-sc05.pdf>
11. Kim, J., Shiple, S., Siegel, H.J., et al.: Dynamically mapping tasks with priorities and multiple deadlines in a heterogeneous environment. *J. Parallel Distrib. Comput.* 67, 154–169 (2007)
12. Zhao, H., Sakellariou, R.: Scheduling multiple DAGs onto heterogeneous systems. In: *Proceedings of 20th International Parallel and Distributed Processing Symposium*, pp. 1–14. IEEE Press, Piscataway (2006)

13. Iverson, M.A., Özgüner, F., Follen, G.J.: Run-Time Statistical Estimation of Task Execution Times for Heterogeneous Distributed Computing. In: Proceedings of 5th IEEE International Symposium on High Performance Distributed Computing (HPDC96), pp. 263–270. IEEE Computer Society Press, Los Alamitos (1996)
14. Ali, S., Siegel, H.J., Maheswaran, M., et al.: Task Execution Time Modeling for Heterogeneous Computing Systems. In: Proceedings of 9th Heterogeneous Computing Workshop (HCW 2000), pp. 185–199. IEEE Computer Society Press, Los Alamitos (2000)
15. Iverson, M.A., Özgüner, F., Potter, L.: Statistical prediction of task execution times through analytic benchmarking for scheduling in a heterogeneous environment. *IEEE Trans. Comput.* 48, 1374–1379 (1999)
16. SETI@home Project, <http://setiathome.ssl.berkeley.edu>

An Enhanced DGIDE Platform for Intrusion Detection

Fang-Yie Leu, Fuu-Cheng Jiang, Ming-Chang Li, and Jia-Chun Lin

Department of Computer Science and Information Engineering, Tunghai University, Taiwan
leufy@thu.edu.tw

Abstract. In this article, we propose a grid-based intrusion detection platform, named Enhanced Dynamic Grid Intrusion Detection Environment (E-DGIDE), which is an extension of our previous system DGIDE. The DGIDE exploits a grid's dynamic and abundant computing resources to detect intrusion packets. The E-DGIDE provides two types of standby/backup mechanisms, one online and one offline, to prevent itself from crashing. With the two mechanisms, the reliability of the system is improved.

1 Introduction

Most network administrators deploy intrusion detection systems (IDSs) to protect their network infrastructures. Although some traditional IDSs' detection accuracies are high in offline tests, when facing enormous traffic, they lose a portion or most of their detection capability or even crash since tremendous amounts of packets often markedly slow down detection speed, consequently disabling their detection power [1,2].

From an engineering viewpoint, reliability of a network can be improved in one of two ways: by increasing the reliability of each component or by introducing redundant components. Either approach imposes costs on the whole system, and thus the design problem can be reduced to optimizing reliability subject to cost constraints.

In this paper, we propose a grid-based intrusion detection platform, named the Enhanced Dynamic Grid Intrusion Detection Environment (E-DGIDE), which is an extension of our previous system DGIDE [3]. However, detection subsystems of the E-DGIDE may crash or overload anytime due to some management or maintenance issues. Thereby, two standby/backup mechanisms (one online and one offline) are provided to improve its reliability.

The rest of this paper is organized as follows. Section 2 presents relevant research. Section 3 illustrates system framework and describes the functions of the E-DGIDE's subsystems. Section 4 concludes this article and addresses our futures work.

2 Related Work

Grid computing, aggregating distributed resources and technologies to form a dynamic and distributed virtual organization over LANs or WANs, is frequently employed to process difficult and complex tasks and solve large-scale and complicated problems [4] aiming to enable dynamic selection, sharing, and aggregation of distributed autonomous resources.

Bannister and Trivedi [5] proposed a task-allocation model for fault-tolerant systems where the fault tolerance originates in redundant copies of modules. They optimized the task allocation by balancing the load over a homogeneous system. However, their model does not provide an explicit system-reliability measure, nor does it consider failures of communication links. Hariri and Raghavendra [6] proposed a method to solve the problem of task allocation for reliability. By introducing multicopies of modules, the system achieves high reliability when the task assignment maximizes the number of trees which are capable of processing the entire task. But this approach does not provide an explicit reliability expression in terms of system parameters. Shatz et al. [7] also proposed a method to solve the problem of task allocation for maximizing reliability. In their model, an explicit cost function is provided and is used to measure system reliability and to guide the search for an optimal/suboptimal task assignment.

3 The Enhanced Subsystems in the E-DGIDE

The functions of the DGIDE are described in [3]. The E-DGIDE enhances the DGIDE in that it increases two standby/backup dispatchers. The detectors' backup strategy is also preserved. In the E-DGIDE, dispatchers are in charge of retrieving packet summaries from packet headers and dispatching them to detectors. Note that in the DGIDE the backup broker backs up packets rather than packet summaries. This wastes too much storage space. Therefore, in the E-DGIDE, we move the head processor from a detector to a dispatcher. Dispatchers can be classified into three types: online, standby and backup. The former performs the tasks same as those in the DGIDE. The latter two take over for the former when the former can not function properly.

Subnets in a network management unit can also be classified into two types: key and ordinary. Those critical to a network system are key subnets. The subnets serving the main office of an organization and the computer center of a university are two examples of the former. In the E-DGIDE, the reliability enhancement for these two types of subnets is associated with different reliability strategies. An ordinary subnet is served by an online dispatcher, whereas a key subnet is served by an online dispatching pair (i.e., a fully parallel redundancy configuration), which consists of two identical dispatchers, one acting as an online dispatcher and the other as the standby, so that packets mirrored (duplicated) from those sent to a key subnet can be reliably collected and transferred to a detector, even when one of the pair crashes. In the following, the online dispatcher that serves an ordinary subnet is also called an ordinary dispatcher.

In an online dispatching pair, the two dispatchers' inputs are "wired" together so as to receive a packet from their switch synchronously. In addition, packets mirrored (from a switch) to the online dispatcher are backed up in the standby dispatcher rather than in backup broker, a subsystem for backing up packets. All dispatchers other than online dispatching pairs are backed up by one offline subsystem, named the general backup dispatcher. After taking over for a failed dispatcher, the general standby dispatcher functions like an ordinary dispatcher.

Fig. 1 shows the communication paths among subsystems of the E-DGIDE before detector k fails. Fig. 2 illustrates that when realizing that it will fail, detector k actively sends a “will crash” message to the scheduler and delivers the heap tables it has produced for subnet i to the backup broker. The remaining steps, steps 3 to 6 and 6’, are respectively the same as steps 4 to 7 and 7’ in Fig. 1. A heap table is a table accumulating inbound packets for a subnet so as to discover attacks and to identify who the probable hacks are.

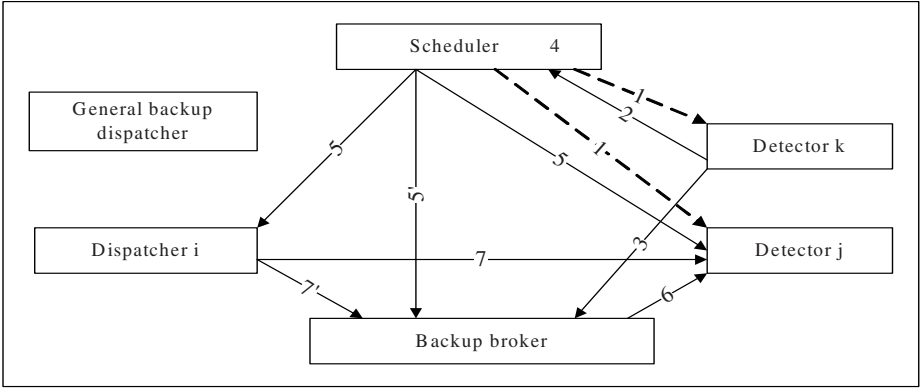


Fig. 1. The communication paths among the subsystems of the E-DGIDE before detector k fails. (1: periodically or on receiving a query, detectors j and k calculate their total scores TSC_j and TSC_k , respectively; 2: sending “may crash” when TSC_k is lower than a threshold; 3: sending heap tables Hs; 4: choosing an appropriate detector; 5: notifying dispatcher i to dispatch packets to detector j, and detector j to receive packets from dispatcher i; 5’: requesting the backup broker to send Hs to detector j with a message (T-H, k, i, j) [3]; 6: sending Hs; 7 and 7’: sending packet summaries).

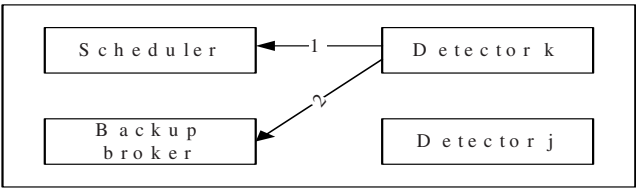


Fig. 2. Realizing that it will fail, detector k actively sends a “will crash” message to the scheduler and delivers the heap tables it has produced for subnet i to the backup broker. The remaining steps, steps 3 to 6 and 6’, are respectively the same as steps 4 to 7 and 7’ in Fig. 1.

3.1 Detectors

In the following, we describe the six different types of messages a detector (e.g., detector j) in the E-DGIDE may receive and the corresponding activities that occur. Also, a detection unit is a collection of inbound packets that a subnet receives in a

specific second and a detection unit is further partitioned into several detection subunits, each consists of 200 consecutive packets.

- (1) A notice from the scheduler telling detector j to serve dispatcher i : Detector j then creates two sockets, one connecting itself and dispatcher i and the other connecting itself and the backup broker.
- (2) A notice from the scheduler telling detector j to take over for detector k to serve dispatcher i : this is the case where detector k can not work properly. Detector j also creates two sockets, one connecting itself and dispatcher i and the other connecting itself and the backup broker.
- (3) A packet summary, whose destination IP is g , from dispatcher i : then detector j accumulates the packet summary in heap table H_{ig} . The logical detector [3] immediately checks to see if the packet is an attack or not.
- (4) A checkpoint cpt or an EOT from dispatcher i : that means the end of a detection subunit. Detector j 's flood detector [3] checks all heap tables. If any intrusion is found, it disconnects the corresponding connections/sessions, requests that the intruder analyzer [3] analyze which source IP has issued the attack, and finally sends cpt/EOT to a standby/backup mechanism (backup broker or standby dispatcher i). If the received message is an EOT, detector j further deletes the underlying heap tables.
- (5) An inquiry message for total score TSC_j from the scheduler: that indicates the scheduler has not received the last total score from detector on schedule, which is once per second. Detector j 's performance agent collects its own features, calculates TSC_j and checks whether $TSC_j < \text{low-threshold}_j$ or not. If yes, it sends a "may crash" message to the scheduler and delivers heap tables to the standby/backup mechanism. Otherwise, it sends TSC_j to the scheduler.
- (6) Heap tables H_s from the backup mechanism: detector j saves H_s .

3.2 Reliability Analysis for Detectors

A. Notations and Definitions

The notations and definitions are specified below. T_d : The task, the set of detection units to be detected; m : The number of detection units forming the task T_d ; u_i : i th detection unit of task T_d ; D : The set of detectors in the E-DGIDE; n : The number of detectors; D_k : k th detector in D ; e_{ik} : The accumulative execution time for task u_i running on detector D_k ; X_d : An $m \times n$ binary matrix corresponding to a task assignment; x_{ik} : An element of X_d where $x_{ik} = 1$ if and only if u_i is assigned to D_k in the assignment represented by X_d , otherwise $x_{ik} = 0$; $x_{ik}(t)$: x_{ik} at time t ; λ_k : The failure rate of detector D_k ; $R_k(T_d, X_d)$: The reliability of detector D_k , which is the probability that D_k is operational for detecting detection units assigned to it under assignment X_d during the detection; $R(T_d, X_d)$: The reliability of the E-DGIDE during the detection when task T_d is allocated by assignment X_d . This is defined as the probability that T_d can run successfully on the E-DGIDE during the detection in the assignment represented by X_d .

B. Reliability Analysis

The reliability of a distributed system depends not only on the reliability of the communication network, but also on the processing node reliability and the

distribution of the resources in the network [8,9]. The allocation problem here is to allocate each of the m detection units to one of the n detectors such that an appropriate objective cost-function (i.e., total detection time) is minimized subject to stated resource limitations and constraints imposed by the application or environment. Besides, we assume that all the subsystems in the E-DGIDE are time-dependent. The future life of all subsystems is assumed to follow a Poisson process with a constant failure rate. Failures of detectors, dispatchers and schedulers are individually assumed to be statistically independent [10,11].

Although a node or link may fail during an idle time, we do not consider such failures because they only affect the subsystem's completion time, not the system's reliability. Also, we do not consider link failure, only node failures are addressed.

- (1) Calculating $R_k(T_d, X_d)$: The reliability of a detector D_k in time interval t ($0 < t \leq 1$) is $\exp(-\lambda_k t)$. Under X_d , the reliability of D_k for detecting the detection unit assigned to it during the detection is [7]

$$R_k(T_d, X_d) = \exp(-\lambda_k \sum_{i=1}^m x_{ik} e_{ik}) \quad (1)$$

- (2) Calculating $R(T_d, X_d)$: The system reliability is the probability that T_d can run successfully on the E-DGIDE during the detection under X_d . Shatz and Wang [12] assumed that r identical processors are functioning simultaneously. For simplicity's sake, let $r=1$ (i.e., a detector consists of only one computer), then

$$R(T_d, X_d) = \text{Prob}(T_d \text{ can run successfully under } X_d \text{ during the detection})$$

$$= \prod_{k=1}^n R_k(T_d, X_d) = \exp(-\text{COST}(X_d)) \quad (2)$$

Where

$$\text{COST}(X_d) = \sum_{k=1}^n \sum_{i=1}^m \lambda_k x_{ik} e_{ik} \quad (3)$$

3.3 Dispatcher and Backup/Standby Dispatcher

To balance the detection load among heterogeneous detectors, a dispatcher in the E-DGIDE, e.g., dispatcher i , computes an exponential average [3] at time t as the predicted traffic volume for $t+I$, denoted by $TV(i)_{t+I}$.

Initially, dispatcher i sets a timer X_D to infinity to wait for a wake-up message issued by the scheduler telling the dispatcher to send packet summaries to the corresponding detector, e.g., detector j . On receiving the message, dispatcher i sets X_D to 1 sec to synchronize itself with the scheduler, then creates two sockets to connect itself to detector j and to the backup broker, respectively. There are three other types of messages that a dispatcher may receive.

- (1) A packet (not packet summary) from switch i : dispatcher i retrieves the packet summary, sends the summary to the backup broker and detector j , and increases the corresponding counter by one. When the counter 200 (i.e., check to see if it is the end of underlying detection subunit), the dispatcher sends a checkpoint cpt

to the backup broker and detector j telling them this is the end of the current detection subunit, and resets the counter to zero.

- (2) A query from the scheduler: that is, the scheduler does not receive $TV(i)_{t+1}$ on schedule. Dispatcher i predicts traffic $TV(i)_{t+1}$ and sends it to the scheduler.
- (3) A notice from the scheduler telling dispatcher i to send packet summaries to detector j instead of detector k , i.e., detector k is not working properly. Dispatcher i sets X_D to 1 sec if $X_D > 1$ sec, otherwise X_D remains unchanged.

When X_D times out, dispatcher i sends an EOT to the backup broker and detector j indicating this is the end of the underlying detection unit.

3.3.1 General Backup Dispatcher

To take over the task performed by an ordinary dispatcher, the general backup dispatcher's only input INP is connected to p switches with a p -input multiplexer, where p is the number of ordinary dispatchers. The multiplexer's i^{th} input, switch i 's mirror port and dispatcher i 's input are "wired" together. Usually, the multiplexer is disabled and its only output is connected to INP. When the scheduler finds out that dispatcher i has failed, the scheduler enables the multiplexer, directs network traffic from the multiplexer's i^{th} input to the general backup dispatcher, and notifies this dispatcher to send packets to the detector originally detecting intrusion for subnet i so that the detection can proceed. The algorithm that the general backup dispatcher performs is the same as that of an online dispatcher.

3.3.2 Reliability Analysis for Ordinary Dispatchers

For simplicity's sake, we assume that all dispatchers are homogeneous. This is a "k out of n" redundant system.

A. Notations and Definitions

The notations and their definitions are as follows. k : Redundancy level of the dispatching system. For ordinary dispatchers, $k=1$ and $n=g$ where g is the number of ordinary subnets; T_o : The task, the set of detection units to be dispatched to detectors for ordinary subnets; p : The number of detection units forming the task T_o ; u_i : i th detection unit of task T_o ; $e_{i,j}$: The accumulative time for dispatching u_i to a detector by dispatcher j during the one-second dispatch; X_o : A $p \times p$ binary matrix corresponding to a task assignment; x_{ij} : An element of X_o where $x_{i,i} = 1$ (i.e., u_i is dispatched by dispatcher i) and $x_{i,j} = 0$ when $j \neq i$; λ_j : The failure rate of dispatcher j ; λ_{gbd} : The failure rate of the general backup dispatcher; $R_j(T_o, X_o)$: The reliability of dispatcher j , which is the probability that dispatcher j is operational for dispatching the detection unit assigned to it under assignment X_o ; $R_{diff}(T_o, X_o)$: The reliability of the first failed ordinary dispatcher together with the general backup dispatcher; $R(T_o, X_o)$: The reliability of all ordinary dispatchers together with the general backup dispatcher.

B. Reliability Analysis

(1) Calculating $R_j(T_o, X_o)$: Under X_o , the reliability of dispatcher j for the dispatch of detection units assigned to it during the dispatch is [7]

$$R_j(T_o, X_o) = \exp(-\lambda_j \sum_{i=1}^p x_{ij} e_{i,j}) = \exp(-\lambda_j \cdot e_{j,j}) \quad (4)$$

(2) Calculating $R_{diff}(T_o, X_o)$: For the dispatcher that fails first, e.g., dispatcher j , the reliability is

$$\begin{aligned} R_{diff}(T_o, X_o) &= \exp(-\lambda_j \cdot e_{j,j}) + (1 - \exp(-\lambda_j \cdot e_{j,j})) \cdot \exp(-\lambda_{gbd} \cdot e_{j,gbd}) \\ &= \exp(-\lambda_j \cdot e_{j,j}) + \exp(-\lambda_{gbd} \cdot e_{j,gbd}) - \exp(-\lambda_j \cdot e_{j,j}) \cdot \exp(-\lambda_{gbd} \cdot e_{j,gbd}) \end{aligned} \quad (5)$$

in which $e_{j,j} + e_{j,gbd}$ is the turnaround time for dispatching u_j to a detector. If the general backup dispatcher and ordinary dispatchers have the same failure rate, λ_{gbd} is equal to λ_j .

$$R_{diff}(T_o, X_o) = \exp(-\lambda_j \cdot e_{j,j}) + \exp(-\lambda_j \cdot e_{j,gbd}) - \exp(-\lambda_j \cdot (e_{j,j} + e_{j,gbd})) \geq R_j(T_o, X_o) \quad (6)$$

Others' reliabilities remain on $R_j(T_o, X_o)$. $R(T_o, X_o)$ can be defined in two ways. The first is that all the packets should be safely delivered to detectors, then

$$R(T_o, X_o) = \exp(-\sum_{j=1, p, j \neq diff} \lambda_j \cdot e_{j,j}) \cdot R_{diff}(T_o, X_o) \geq \exp(-\sum_{j=1, p} \lambda_j \cdot e_{j,j}) \quad (7)$$

The second is the situation where at least one of the ordinary dispatchers works properly. Its reliability is $1 - R(T_o, X_o)$. However, the second definition is inappropriate for the E-DGIDE since some subnets may now be under attack, but the E-DGIDE does not know that.

3.3.3 Online Dispatching Pairs

On receiving an allocation message with the ID of a detector, e.g., detector j , from the scheduler, the two dispatchers of an online dispatching pair contend to lock a flag F . The winner, denoted by D_{win} , will act as an online dispatcher. The loser, denoted by D_{los} , will be a standby dispatcher which also backs up network traffic in backup table i for an underlying subnet (i.e., subnet i). The schema of a backup table is shown in Table 1.

Table 1. An example of a backup table [3]. (DSU: detection subunit).

Checkpoint	DSU
*20060220123_1	packet 1_1, packet 1_2, ..., packet 1_200
*20060220123_2	packet 2_1, packet 2_2, ..., packet 2_200
20060220123_3	packet 3_1, packet 3_2, ..., packet 3_200
...	...

The algorithms that D_{win} and an online dispatcher perform are similar to each other, but different in that: (1) no socket is established between D_{win} and the backup broker; (2) no packet summaries are sent to the backup broker; (3) D_{win} periodically sends messages to disable D_{los} 's timer X_L , which is an indicator showing whether D_{win} is

still alive or not (described below); (4) after dispatching the last packet summary of a detection unit, i.e., X_D times out, D_{win} contends to lock flag F with D_{los} . If D_{win} wins again, it continues performing the D_{win} algorithm; otherwise, it will act as D_{los} .

D_{los} performs the following tasks. As with the algorithm that D_{win} executes, D_{los} initially sets a timer X_L to ∞ to wait to receive a wake-up message from the scheduler requesting the two dispatchers of online dispatching pair i to send packets to a detector, e.g., detector j . On receiving such a message, D_{los} sets X_L to 1 sec and creates a backup table for subnet i . There are seven other types of messages D_{los} may receive.

- (1) A packet comes from switch i . D_{los} retrieves the packet summary and appends the summary to a DSU (detection subunit) field of the current tuple in backup table i , increases the corresponding counter by one, and checks whether this is the end of a detection subunit. If yes, it inserts a checkpoint into the checkpoint field of the current tuple.
- (2) A specific checkpoint is sent by detector j , indicating detector j has completely detected the underlying detection subunit. D_{los} marks the checkpoint in backup table i with an “*”.
- (3) Heap tables H_s are sent by detector k , which means detector k is going to crash. D_{los} saves H_s .
- (4) A message (T-H, k , j) is sent by the scheduler, telling the dispatching pair that detector j will take over for detector k where T-H stands for “transmitting heap tables”. D_{los} then transmits the items backed up for subnet i to detector j , including detector k ’s heap tables H_s , uncommitted packets and their checkpoints, and unanalyzed packets which are contained in unfinished and/or unanalyzed detection units that online dispatching pair i has received.
- (5) A message (T, k , j) is sent by the scheduler, i.e., detector k has failed and H_s have been lost where T stands for “transmitting”. D_{los} unmarks all marked checkpoints, delivers all items backed up in backup table i for subnet i to detector j . Detector j will generate new heap tables.
- (6) An EOT is sent by detector j . D_{los} clears backup table i previously constructed for detector j (for subnet i), and deletes previous heap tables if any exist.
- (7) A message from D_{win} of the underlying dispatching pair to disable X_L , indicating D_{win} is working properly. D_{los} sets X_L to ∞ .

After processing a received message, D_{los} checks to see whether D_{win} is alive or not. If not, it proclaims itself as D_{win} and executes the algorithm D_{win} with $X_D = X_L$, instead of $X_D = \infty$, for synchronously taking over for the original D_{win} .

Due to detection delay, after both X_D and X_L time out, their detector, e.g., detector k , may still be detecting a detection unit. Therefore, even if the online dispatching pair is reallocated to dispatch packets to another detector, the previous D_{los} , which may now be D_{win} or D_{los} , has to keep performing the D_{los} algorithm in order to continue backing up packets for detector k until the corresponding EOT issued by detector k arrives.

3.3.4 Reliability Analysis for Online Dispatching Pair

We use the same model for reliability as in [12]. An explicit system-reliability expression is also derived for the reliability of an allocation.

A. Notations and definitions

The notations and definitions are as follows. T_{ks} : The task, the set of detection units to be dispatched to detectors for key subnets; g : The number of detection units forming the task T_{ks} ; u_i : i th detection unit of task T_{ks} ; $e_{i,k}$: The accumulative time for dispatching u_i to a detector by dispatcher D_{win_k} during the one-second dispatch; X_{ks} : A $g \times g$ binary matrix corresponding to a task assignment; x_{i,win_j} : An element of X_{ks} . $x_{i,win\_i} = 1$ if and only if u_i is assigned to D_{win_i} in the assignment represented by X_{ks} ; $x_{i,win\_j} = 0$, when $j \neq i$; λ_{win_i} : The failure rate of D_{win_i} ; λ_{los_i} : The failure rate of D_{los_i} ; $R_{win\_i}(T_{ks}, X_{ks})$: The reliability of dispatcher D_{win_i} , which is the probability that D_{win_i} is operational for dispatching detection units assigned to it under X_{ks} during the dispatch; $R_{win\_i,los\_i}(T_{ks}, X_{ks})$: The reliability of online dispatching pair i ; $R(T_{ks}, X_{ks})$: The reliability of all online dispatching pairs.

B. Reliability analysis

Consider an online dispatching pair that consists of two identical dispatchers functioning simultaneously, therefore λ_{win_i} is equal to λ_{los_i} , then [12]

$$R_{win\_i}(T_{ks}, X_{ks}) = \exp(-\lambda_{win\_i} \sum_{j=1}^g x_{j,win\_i} \cdot e_{j,win\_i}) = \exp(-\lambda_{win\_i} \cdot e_{i,win\_i}) = R_{los\_i}(T_{ks}, X_{ks}) \quad (8)$$

The reliability of one online dispatching pair is

$$R_{win\_i,los\_i}(T_{ks}, X_{ks}) = \exp(-\lambda_{win\_i} \cdot e_{i,win\_i}) + \exp(-\lambda_{los\_i} \cdot e_{i,los\_i}) - \exp(-\lambda_{win\_i} \cdot (e_{i,win\_i} + e_{i,los\_i})) \quad (9)$$

where $(e_{i,win\_i} + e_{i,los\_i})$ is the turnaround time of u_i . The reliability of all online dispatching pairs is

$$R(T_{ks}, X_{ks}) = \prod_{j=1}^g R_{win\_j,los\_j}(T_{ks}, X_{ks}) \quad (10)$$

4 Conclusion and Future Work

This paper proposes the E-DGIDE platform, which is a dynamic environment. When system detection and analytical performance very frequently become unbearably low, we can dynamically deploy several powerful computers as newly joined detectors to improve system performance. The E-DGIDE also provides a fault-tolerant environment in which a portion of a DoS/DDoS attack or some logical attack packets may be unanalyzed due to node crash or low performance. If a detector can not continue its detection, the scheduler will choose another one to take over its detection task. If a dispatcher is not working properly, a standby/backup dispatcher will take over the dispatching task. Therefore, using standby/backup mechanisms, the E-DGIDE can effectively improve its system reliability. This eliminates the drawbacks that occur in a static detection environment.

Furthermore, it is important to derive the E-DGIDE's mathematical performance and cost models so that users can validate the system formally. The prospect of a universal standby, which can take over for any failed subsystem, is intriguing and worth investigating. Also, in this paper, we assume that the backup broker and the scheduler are reliable. However, they may practically fail. Their standby/backup mechanisms should be further studied. These topics will constitute our future research.

References

- [1] Haggerty, J., Shi, Q., Merabti, M.: Beyond the Perimeter: the Need for Early Detection of Denial of Service Attacks. In: The IEEE Annual Computer Security Applications Conference, pp. 413–422 (2002)
- [2] Du, Y., Wang, H.Q., Pang, Y.G.: Design of a Distributed Intrusion Detection System based on Independent Agents. In: The IEEE International Conference on Intelligent Sensing and Information, pp. 254–257 (2004)
- [3] Leu, F.Y., Li, M.C., Lin, J.C.: Intrusion Detection based on Grid. In: International Multi-Conference on Computing in the Global Information Technology, pp. 62–67 (2006)
- [4] Foster, I., Kesselman, C.: The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, San Francisco (1999)
- [5] Bannister, J.A., Trivedi, K.S.: Task Allocation in Fault-Tolerant Distributed Systems. *Acta Informatica* 20, 261–281 (1983)
- [6] Hariri, S., Raghavendra, C.S.: Distributed Functions Allocation for Reliability and Delay Optimization. In: The IEEE/ACM Fall Joint Comput. Conference, pp. 344–352 (1986)
- [7] Shatz, S.M., Wang, J.P., Goto, M.: Task Allocation for Maximizing Reliability of Distributed Computer Systems. *The IEEE Trans. on Computers* 41(9), 1156–1168 (1992)
- [8] Shooman, M.L.: Reliability of Computer Systems and Networks: Fault Tolerance, Analysis and Design. In: Shooman, M.L. (ed.) *Electronic Version*, John Wiley and Sons, West Sussex (2002)
- [9] —: Probabilistic Reliability : An Engineering Approach, 2nd edn. Krieger, Melbourne Fla (1990)
- [10] Srinivasan, S., Jha, N.K.: Safty and Reliability Driven Task Allocation in Distributed Systems. *The IEEE Trans. on Parallel and Distributed Systems* 10(3), 238–251 (1999)
- [11] Lawless, J.F.: Statistical Models and Methods for Lifetime Data. John Wiley & Sons, West Sussex (1982)
- [12] Shatz, S.M., Wang, J.P.: Models & Algorithms for Reliability-Oriented Task-Allocation in Redundant Distributed-Computer systems. *The IEEE Trans. on Reliability* 38(1), 16–27 (1989)

Author Index

- An, Gaeil 179
Apduhan, Bernady O. 385
Arbaugh, William A. 489

Baek, Myung-Sun 269
Balfe, Shane 123
Benz, Michael 191
Botvich, Dmitri 146
Brancovici, George 169
Bresson, Emmanuel 395

Cao, Wanhua 529
Casola, Valentina 82
Chan, Keith C.C. 1
Chang, Chin-Chen 333
Chen, Binglan 449
Chen, Danwei 519
Chen, Hsiao-Hwa 313
Chen, Jieyun 42
Chen, Ruichuan 203
Chen, Shuyi 352
Chen, Wei 519
Chen, Zhong 203
Choi, Donghyun 135
Choi, Yoon-Hwa 539
Chu, Xiaowen 313
Chung, Kyo IL 286
Cremene, Marcel 61

Ding, Li 259

Eckert, Claudia 191
Ehrig, Jörg 458

Fu, Jianming 449
Fu, Zhi (Judy) 489

Gui, Chunmei 239

Han, Dong-Guk 286
Han, Wei-Hong 71
Han, Zongfen 42
Hao, Jingbo 468
Hermanowski, Martin 191
Hori, Yoshiaki 249
Hu, Guyu 259
Hu, Jianbin 203

Huang, Dazhi 103
Huang, Kun 499
Huang, Runhe 385
Huang, Shou-Husan Stephen 276
Huang, Xinyi 13, 22
Hung, Kuo Lung 333

Jain, Nitin 489
Jia, Yan 71
Jiang, Congfeng 549
Jiang, Fuu-Cheng 559
Jiang, Yixin 313
Jin, Hai 42
Jun, Sungik 410

Kang, Hogab 135
Kim, Ho Won 286
Kim, Mooseop 410
Kim, Seungjoo 135
Kim, Sung-Kyoung 286
Kim, Yong Ho 441
Kim, Youngse 410
Kong, Fanyu 52
Kuang, Wenyan 296

Lee, Dong Hoon 441
Lee, Hwaseong 441
Lee, Jinheung 420
Lee, Myeong-Hyeon 539
Lee, Sanggon 420
Lee, Yunho 135
Leu, Fang-Yie 559
Leung, Ho-fung 323
Li, Daxing 52
Li, Fagen 114
Li, Jun 529
Li, Ming-Chang 559
Li, Wenjia 499
Li, Yunfa 42
Lim, Jongin 286
Lin, Jia-Chun 559
Lin, Shieh-Shing 344
Linner, David 94
Liu, Jiangchuan 313
Liu, Xiaohu 549

- Löhr, Hans 372
 Lü, Jian 216
 Lu, Xicheng 478
 Lu, Yansheng 529
 Luo, Jun 259
 Lychev, Robert 276

 Ma, Jianhua 385
 Ma, Wenping 306
 Mancini, Emilio Pasquale 82
 Manulis, Mark 395
 Mazzocca, Nicola 82
 McGibney, Jimmy 146
 Meling, Hein 156
 Mohammed, Anish 123
 Mu, Yi 22, 32
 Müller-Schloer, Christian 169

 Nguyen, Son Thanh 3
 Ni, Guiqiang 259
 Noh, Sanguk 361

 Ouyang, Kai 313

 Pan, Peng 323
 Pan, Zhisong 259
 Park, Jong Hyuk 441
 Park, Joon S. 179
 Pfeffer, Heiko 94
 Pietzowski, Andreas 458
 Prehofer, Christian 226

 Qin, Baodong 52
 Qin, Bo 32

 Radusch, Ilja 94
 Rak, Massimiliano 82
 Ram, Vishnu 489
 Ramasamy, HariGovind V. 372
 Riveill, Michel 61
 Rong, Chunming 3
 Ryou, Jaecheol 410

 Sadeghi, Ahmad-Reza 372
 Sakurai, Kouichi 249
 Satzger, Benjamin 458
 Schmeck, Hartmut 2
 Schulz, Stefan 372
 Schunter, Matthias 372
 Seberry, Jennifer 22

 Shi, Bin 323
 Shin, Minho 489
 Shin, Sanguk 420
 Shiratori, Norio 385
 Song, Hyoung-Kyu 269
 Steglich, Stephan 94
 Strassner, John C. 489
 Stübke, Christian 372
 Stumpf, Frederic 191
 Sun, Guozi 519
 Sun, Ying 114
 Sun, Yuqing 323
 Susilo, Willy 22, 32

 Takata, Katsuhiko 385
 Tanaka, Masataka 385
 Tang, Liyong 203
 Tang, Yong 478
 Tao, Ye 216
 Trumler, Wolfgang 458
 Tu, Gang 529

 Ungerer, Theo 458

 Villano, Umberto 82

 Wang, Cheng 549
 Wang, Huaimin 239
 Wang, Jun 71
 Wang, Shulin 468
 Wang, Xinmei 306
 Wang, Yuan 216
 Wang, Yufeng 249
 Wei, Li 296, 509
 Wen, Yingyou 352
 Won, Dongho 135
 Wu, Qianhong 32
 Wu, Quanyuan 239
 Wu, Wei 22

 Xia, Nan 296
 Xiao, Bin 478
 Xu, Feng 216
 Xu, Guangbin 296

 Yan, Min 103
 Yan, Zheng 226
 Yang, Bo 13, 114
 Yang, Chen 306
 Yang, Fumin 529

Yang, Jianhua 276
 Yang, Laurence T. 441
 Yau, Stephen S. 103
 Yeo, So-Young 269
 Yin, Jianping 468
 Yoo, Kee-Young 430
 Yoon, Eun-Jun 430
 Yu, Jia 52
 Yu, Ping 216
 Yu, Yong 13, 114

 Zhang, Boyun 468
 Zhang, Dafang 499

Zhang, Dingxing 468
 Zhang, Huanguo 449
 Zhang, Mingwu 13
 Zhang, Yaoxue 296, 509
 Zhang, Yingzhou 519
 Zhao, Hong 352
 Zhao, Xuan 203
 Zhao, Yinghui 549
 Zheng, Di 71
 Zhou, Yuezhi 296, 509
 Zhu, Xianshu 499
 Zou, Deqing 42
 Zou, Peng 71