

Lecture Notes in Computer Science  
Edited by G. Goos, J. Hartmanis, and J. van Leeuwen

2437

**Springer**

*Berlin*

*Heidelberg*

*New York*

*Barcelona*

*Hong Kong*

*London*

*Milan*

*Paris*

*Tokyo*

George Davida   Yair Frankel  
Owen Rees (Eds.)

# Infrastructure Security

International Conference, InfraSec 2002  
Bristol, UK, October 1-3, 2002  
Proceedings



Springer

Series Editors

Gerhard Goos, Karlsruhe University, Germany  
Juris Hartmanis, Cornell University, NY, USA  
Jan van Leeuwen, Utrecht University, The Netherlands

Volume Editors

George Davida  
University of Wisconsin-Milwaukee  
Milwaukee, WI 53201, USA  
E-mail: [davida@cs.uwm.edu](mailto:davida@cs.uwm.edu)

Yair Frankel  
TechTegrity LLC  
P.O. Box 2125, Westfield, NJ 07091, USA  
E-mail: [yfrankel@cryptographers.com](mailto:yfrankel@cryptographers.com)

Owen Rees  
Hewlett-Packard Laboratories  
Filton Road, Stoke Gifford, Bristol, BS34 8QZ, UK  
E-mail: [owen.rees@hp.com](mailto:owen.rees@hp.com)

Cataloging-in-Publication Data applied for

Die Deutsche Bibliothek - CIP-Einheitsaufnahme

Infrastructure security : international conference ; proceedings / InfraSec  
2002, Bristol, UK, October 1 - 3, 2002. George Davida .... - Berlin ;  
Heidelberg ; New York ; Hong Kong ; London ; Milan ; Paris ; Tokyo :  
Springer, 2002  
(Lecture notes in computer science ; Vol. 2437)  
ISBN 3-540-44309-6

CR Subject Classification (1998): D.4.6, C.2.9, D.2, E.3, H.2.0, K.4.4, K.6.5

ISSN 0302-9743

ISBN 3-540-44309-6 Springer-Verlag 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-Verlag. Violations are liable for prosecution under the German Copyright Law.

Springer-Verlag Berlin Heidelberg New York  
a member of BertelsmannSpringer Science+Business Media GmbH

<http://www.springer.de>

© Springer-Verlag Berlin Heidelberg 2002  
Printed in Germany

Typesetting: Camera-ready by author, data conversion by PTP-Berlin, Stefan Sossna e. K.  
Printed on acid-free paper      SPIN 10870164      06/3142      5 4 3 2 1 0

## Preface

Infrastructure Security Conference 2002 (InfraSec 2002) was created to promote security research and the development of practical solutions in the security of infrastructures – both government and commercial – such as the effective prevention of, detection of, reporting of, response to and recovery from security incidents. The conference, sponsored by the Datacard Group and Hewlett-Packard Laboratories, was held on October 1–3, 2002. Organizational support was provided by the Center for Cryptography, Computer and Network Security Center at the University of Wisconsin-Milwaukee.

Organizing a conference is a major undertaking requiring the efforts of many individuals. The Conference President, Graham Higgins (Datacard Group), oversaw all arrangements for the conference, and the General Chair, Susan Thompson (Datacard Group), oversaw the local organization and registration. Local arrangements were directed by Jan Ward (Hewlett-Packard Laboratories) and Jamie Wilson (Datacard Group). Financial arrangements were managed by Natalie Churchill (Hewlett-Packard Laboratories). We wish to thank the organizers, without whose support this conference would not have been possible.

This conference program included two keynote speakers: Bob Evans (Office of the e-Envoy) and Vic Maconachy (Department of Defense). The program committee considered 44 submissions of which 23 papers were accepted. Each submitted paper was reviewed by a minimum of three referees. These proceedings contain revised versions of the accepted papers. Revisions were not checked and the authors bear full responsibility for the content of their papers.

We thank all the authors who submitted to the conference, without which it would not have been successful. Our thanks to the program committee and all reviewers.

July 2002

George Davida, Yair Frankel, Owen Rees

# Infrastructure Security 2002

October 1–3, Bristol, UK

## Sponsored by

*Datacard Group*  
*Hewlett-Packard Laboratories*

## Conference President

Graham Higgins, Datacard Group

## General Chair

Susan Thompson, Datacard Group

## Program Chair

George Davida, University of Wisconsin-Milwaukee

## Program Co-chairs

Yair Frankel, TechTegrity LLC  
Owen Rees, Hewlett-Packard Laboratories

## Program Committee

Alan Borrett .....	CESG
Don Beaver .....	Seagate Research
Elisa Bertino .....	Universita' di Milano
Bob Blakley .....	IBM
Matt Blaze.....	AT&T
Giovanni DiCrescenzo .....	Telcordia
David Everett .....	Microexpert
Sigrid Gürgens .....	GMD
Cynthia Irvine .....	Naval Postgraduate School
Javier Lopez.....	University of Malaga
Masahiro Mambo .....	Tohoku University
Wembo Mao .....	Hewlett-Packard Laboratories
Rene Peralta .....	Yale University
Matt Robshaw .....	Royal Holloway, University of London
Kazue Sako .....	NEC
Colin Walter.....	Comodo Group

## Table of Contents

### Biometrics

Biometric Authentication in Infrastructure Security .....	1
<i>John Armington, Purdy Ho, Paul Koznek, Richard Martinez</i>	
Denial of Access in Biometrics-Based Authentication Systems .....	19
<i>Luciano Rila</i>	

### Identification, Authentication, and Process

A Novel Approach to Proactive Password Checking .....	30
<i>Carlo Blundo, Paolo D'Arco, Alfredo De Santis, Clemente Galdi</i>	
Single Sign-On Architectures .....	40
<i>Jan De Clercq</i>	
Active Digital Credentials: Dynamic Provision of Up-to-Date Identity Information .....	59
<i>Marco Casassa Mont, Richard Brown</i>	

### Analysis Process

How to Buy Better Testing (Using Competition to Get the Most Security and Robustness for Your Dollar) .....	73
<i>Stuart Schechter</i>	
Structured Risk Analysis .....	88
<i>Neil McEvoy, Andrew Whitcombe</i>	
A Model Enabling Law Compliant Privacy Protection through the Selection and Evaluation of Appropriate Security Controls .....	104
<i>E.S. Siougle, V.C. Zorkadis</i>	

### Mobile Networks

Authentication and Authorization of Mobile Clients in Public Data Networks .....	115
<i>Prakash Reddy, Venky Krishnan, Kan Zhang, Devaraj Das</i>	
A Contemporary Foreword on GSM Security .....	129
<i>Paulo S. Pagliusi</i>	

## Vulnerability Assessment and Logs

Vulnerability Assessment Simulation for Information Infrastructure Protection.....	145
<i>HyungJong Kim, KyungHee Koh, DongHoon Shin, HongGeun Kim</i>	
Pseudonymizing Unix Log Files. ....	162
<i>Ulrich Flegel</i>	

## System Design

DPS: An Architectural Style for Development of Secure Software .....	180
<i>Pascal Fenkam, Harald Gall, Mehdi Jazayeri, Christopher Kruegel</i>	
A New Infrastructure for User Tracking Prevention and Privacy Protection in Internet Shopping.....	199
<i>Matthias Enzmann, Thomas Kunz, Markus Schneider</i>	
Different Smartcard-Based Approaches to Physical Access Control .....	214
<i>Oscar Cánovas, Antonio F. Gómez, Humberto Martínez, Gregorio Martínez</i>	

## Formal Methods

Authenticity and Provability – A Formal Framework .....	227
<i>Sigrid Gürgens, Peter Ochsenschläger, Carsten Rudolph</i>	
Protocol Engineering Applied to Formal Analysis of Security Systems ....	246
<i>Javier Lopez, Juan J. Ortega, Jose M. Troya</i>	

## Cryptographic Techniques

Applications of Multiple Trust Authorities in Pairing Based Cryptosystems.....	260
<i>L. Chen, K. Harrison, D. Soldera, N.P. Smart</i>	
Plausible Deniability Using Automated Linguistic Stegonagraphy .....	276
<i>Mark Chapman, George Davida</i>	
Virtual Software Tokens – A Practical Way to Secure PKI Roaming .....	288
<i>Taekyoung Kwon</i>	
Bit-Serial AOP Arithmetic Architectures over $GF(2^m)$ .....	303
<i>Hyun-Sung Kim, Kee-Young Yoo</i>	



## Networks

A Practical Distributed Authorization System for GARA .....	314
<i>William A. Adamson, Olga Kornievskaia</i>	

Design of a VPN Software Solution Integrating TCP and UDP Services ...	325
<i>Javier Lopez, Jose A. Montenegro, Rodrigo Roman, Jorge Davila</i>	

<b>Author Index</b> .....	339
---------------------------	-----

# Biometric Authentication in Infrastructure Security

John Armington, Purdy Ho, Paul Koznek, and Richard Martinez

Hewlett Packard Company

{john.armington, purdy\_ho, paul.koznek, richard.martinez}@hp.com

**Abstract.** This paper will identify and recommend biometric technologies that provide strong authentication, convenient usability, and versatility, in order to meet the demand of enterprise infrastructure security systems. We aim at validating suitability for one or more mainstream applications. Finger scan technology is mature and widely available. The combination of a finger scan and smart card gives true dual-factor authentication that provides a greater degree of security and simplifies authentication for end users while preserving privacy. Speaker Verification is the most natural biometric technology to use with voice-based systems. It takes advantage of the ubiquitous voice-grade telephone channel. The combination of speaker verification and token technology can provide convenient and secure access to voice portal applications. We also discuss cultural, legal, and privacy issues based on religious objections, health concerns, legal restrictions, and regulations on the use of biometric technology.

## 1 Introduction to Biometric Authentication

For many years, the use of biometrics to identify a person or verify one's identity was mainly used in government installations and Hollywood movies. With the introduction of lower cost biometric devices the technology is moving into mainstream use. MIT Technology Review Magazine recently listed biometrics as one of the "top ten emerging technologies that will change the world" [1].

Biometric authentication works at the human to machine interface, as one component of a system built of secure and trusted components. To work, it must 1) verify that the biometric came from the person at the time of verification and 2) that it matches a trusted record of that person's biometric [2].

Biometrics is used to best advantage as one factor in a multi-factor authentication system. It is one of four possible factors that include: what you know (password/PIN); what you have (smartcard/ token card); where you are (GPS locator); and what you are (biometrics). To increase security, many system designers require authentication from two or more of these factors. For example the ATM requires a difficult-to-duplicate card and a secret PIN.

### 1.1 Definition and Classes of Biometrics

A working definition for Biometrics given by Dr. Jim Wayman is “The automatic identification or identity verification of living, human individuals based on behavioral and physiological characteristics.”

Biometrics can be separated into two primary classes, behavioral and physiological. Behavioral biometrics measure habituated actions that are difficult to mimic, such as voice, gait, and signature, while physiological biometrics measure more static physical properties, such as fingerprints, and patterns on the iris or retina. Behavioral biometrics is more difficult to use because it can vary over time, while physiological biometrics typically require more intrusive measurements.

### 1.2 The Market

An excellent and well-respected source of information on the market for biometrics is the International Biometric Group’s Market Report [3]. Among the Report’s top findings are the following:

- Biometric revenues are expected to grow from \$399m in 2000 to \$1.9b by 2005.
- Revenues attributable to large-scale public sector biometric usage, currently 70% of the biometric market, will drop to under 30% by 2005.
- Finger-scan and biometric middleware will emerge as two critical technologies for the desktop, together comprising approximately 40% of the biometric market by 2005.

Although there are currently more than 200 vendors of biometric hardware or software, finger and face make up the majority of the market.

The biometrics industry has had constant growth for the past several years while the average cost of biometric devices has declined, according to Dr. Jim Wayman. Today, many fingerprint readers may be found for under \$100. Like other technology sectors, the biometrics industry has been hard hit by the economic downturn of 2001 and has seen several vendors go out of business.

### 1.3 Standards

Although forensic biometric standards such as AFIS (Automated Fingerprint Identification Standards) have existed for decades, recent adoption of biometric standards by mainstream industry standards organizations, and the emergence of organizations dedicated to the commercial biometrics industry such as the BioAPI Consortium [4] and the IBIA (International Biometrics Industry Association) [5] are evidence that the industry is maturing. BioAPI is intended to create a standard interface between biometric sensing devices and software, allowing some degree of interchangeability between devices. Most of the biometrics industry now participates in the BioAPI standard. Microsoft, a notable exception, has adopted a proprietary biometric API [6].

The ANSI standard X9.84 is developed by a subcommittee of the X9 financial industry working group and addresses secure transmission and storage of biometric data in addition to standard file and record formats to ensure authentication interoperability. The CBEFF (Common Biometric Exchange File Format) project, initiated by NIST and the US Government's Biometric Consortium has ties to both the X9.84 and the BioAPI efforts. The Organization for the Advancement of Structured Information Standards, OASIS [7], has recently formed the XCBF Technical Committee to set standards for XML encoding of biometric data.

#### 1.4 Testing

Because vendors tend to over-rate the performance of their technologies, often quoting test results that favor their products, independent test results and an understanding of testing methods and criteria should be taken into consideration when developing a biometric system. Independent testing laboratories involved in biometric device testing include the Biometrics Working Group (BWG) of the National Institute of Standards and Technology (NIST) Laboratories, and the National Physical Laboratory (UK) for the Communications Electronics Security Group (CESG).

The Center for Mathematics and Scientific Computing, National Physical Laboratory for the Communications Electronics Security Group (CESG) completed a broad-based independent test of multiple biometric technologies in March 2001. Seven biometric systems were selected for testing. These systems include 2 fingerprint systems (one using optical fingerprint capture, the other using a chip sensor), a face recognition system, a hand recognition system, an iris verification system, a vein recognition system, and a speaker (voice) verification system. All tests were conducted in a normal office environment.

For each system tested, a DET (Detection Error Tradeoff) curve was plotted to show the performance relative to different detection thresholds. The probability of False Rejection (FR) and the probability of False Acceptance (FA) were computed empirically for each threshold. A range of detection thresholds was swept over the curves. Each point on the curve represents the FR and the FA of a particular threshold. By adjusting the threshold, the tradeoff between FR and FA can be achieved and thus the performance of the system can be influenced. In general, the system is adjusted to have a low FA rate, e.g. if the decision threshold is raised, the FA rate will decrease, and vice versa. A common statistic quoted by vendors is called the Equal Error Rate, or EER, which is simply that point at which the FA rate equals the FR rate, but has little meaning without the other points on the curve. It is also important to consider how the test accounts for those who cannot use the system, called the Failure to Enroll rate. Are these counted in the FR rate, or simply ignored? How does a system accommodate these exceptions? The iris scan system showed significantly superior performance over all other types in this test. Unfortunately, because the threshold could not be varied, it was impossible to form a true DET curve. For the full report, see [8].

### 1.5 Attacks and Countermeasures

Security technology must always be concerned with subversion via false representation of digital information or data streams, commonly known as ‘spoofing’, and biometric security systems must pay particular attention to such concerns. Once a biometric has been converted to a stream of bits, it could be inserted into a legitimate system or manipulated to subvert the system’s purpose. Biometric authentication of remote users by remote devices is especially susceptible to such attack by unscrupulous persons wishing to impersonate a legitimate user. A well-designed system will take advantage of digital signatures, tamper resistant packaging, and other techniques known in the security industry to increase trustworthiness of the overall system. Trust placed in a poorly designed system may lead to security breaches that are difficult or impossible to detect. Standards being developed by the Trusted Computing Platform Association, or TCPA hold promise for enabling solutions to these trust issues in the very near future by providing a means for computing devices from smartcards, to PCs to servers, to measure and communicate their level of trustworthiness [9]. Specific attacks and countermeasures will be addressed by technology below.

## 2 Technologies

This section addresses four specific technologies investigated: Speaker Verification (voice biometrics), face recognition, finger scan, and iris scan.

### 2.1 Speaker Verification

Voice applications are becoming mainstream today thanks to technologies like VoiceXML and voice portal services. While these enabling technologies are maturing, security has not kept pace; with most voice systems today either not secured at all or secured using traditional touch-tone static passwords, which are subject to eavesdrop and can easily be stolen, thus leaving room for improvement. Voice is the most natural way of communicating over the phone channels, so speaker verification is better suited to integrate into the phone network than many other authentication technologies. Speaker verification can provide secure access to a voice portal that uses automatic speech recognition (ASR) to allow users to talk directly to sensitive systems completely hands-free using any telephone.

The task of speaker verification [10,11] is to determine a binary decision of whether or not an unknown utterance is spoken by the person of the claimed identity. Speaker verification can be implemented by three different approaches:

- Text-dependent/fixed-phrase
- Text-independent/unconstrained
- Text-dependent/prompted-phrase

which are rated below based on security, convenience, and implementation complexity.

**Text-dependent/fixed-phrase:** Users have to utter exactly the same phrase they did during enrollment and the system matches the two utterances. Advantages include simple enrollment and verification processes, and a lower error rate than the other two approaches, in general. However, this approach is subject to attack by replay of recorded speech or a template stolen from the database, and also requires memorization of a passphrase. An example of this kind of system is developed by Nuance [12], which verifies speakers by having them to say their phone numbers.

**Text-independent/unconstrained:** It verifies the speaker without knowledge of what is being said. Verification is based on specific acoustic features of the vocal tract; users can say anything they want, giving it an advantage in convenience. Also, there is no recorded passphrase that can be stolen from a database. Disadvantages are that it requires more training data and the verification is more computationally intensive. It is also subject to a pre-recorded speech attack. Systems developed by Enigma [13] and Persay [14] are using this approach.

**Text-dependent/prompted-phrase:** Users are prompted to repeat a one-time passcode (OTP)<sup>1</sup> or passphrase in real time, such that they don't need to remember passwords. The ASR module matches the passphrase with the utterance from the user, while the SV module verifies the user's voice. This approach ensures that the user is accessing in real time, countering the attack by replay of recorded speech and a stolen template. This approach can also be extensible to dual-factor authentication by providing a secure out-of-band prompting mechanism<sup>2</sup>. However, the major drawback is that a more complex infrastructure and a longer enrollment are required. VeriVoice's system [17] uses in-band OTP prompting.

**Attacks and Countermeasures.** Nevertheless, the above approaches share a vulnerability to a new technology known as text-to-speech voice cloning (TTSVC)[18,19], a highly sophisticated technique of synthesizing the speech using a person's voice and prosodic features, which will likely become more commonplace over time. With this TTSVC system, impostors can hack the text-dependent/ fixed-phrase system by synthesizing the "passphrase"; they can hack the text-dependent/ prompted-phrase system by synthesizing the prompted phrase. In the text-independent/ unconstrained case, the whole conversation can be synthesized.

<sup>1</sup> Many schemes are available today for implementing one-time passcodes. These range from public domain schemes such as S/Key [15] and SASL [16], proprietary schemes using hardware tokens such as RSA *Security*<sup>®</sup> and *ActivCard*<sup>TM</sup>.

<sup>2</sup> Out-of-band prompting leverages the trust implicit in a second communication channel to add authentication strength, e.g. Prompting via token-card leverages trust in the registration and provisioning process of the token card, and trusts that only the authorized user holds the unique token card.

**Usability.** A dual-factor authentication system combining the text-dependent/prompted-phrase approach and the out-of-band OTP prompting can effectively counteract the TTSVC attacks and provide high-level security and convenience to use over any phone channel. Users do not have to remember a PIN because the OTP is prompted for them. This also ensures real-time access and prevents impostors from using pre-recorded speech. The use of speaker verification is an effective countermeasure against the use of a stolen OTP by an unsophisticated attacker. Successful attack would require both obtaining the OTP and speaking it in the user’s voice.

## 2.2 Face Recognition

Over the past 20 years numerous face recognition papers have been published in the computer vision community; a survey can be found in [20]. Due to recent events, automatic face recognition systems received a lot of media attention and became very popular in military and aviation security. The two main commercial face recognition system makers: Viisage [21] and Visionics [22], both claimed that their systems could pick out terrorists and criminals from a scene as long as they are in the law enforcement database.

The Viisage system was used in the Super Bowl 2000. An image of each spectator was taken at a checkpoint and compared to the law enforcement database of known criminals and terrorists. Its system based on an algorithm called Eigenface [23], which was developed in the MIT Media Lab. This algorithm works extremely well on frontal faces. The Visionics surveillance system used in Tampa, FL works different from the Viisage system. An algorithm called Local Feature Analysis (LFA) [24], developed by its CEO is used. Visionics [25] claimed that LFA is better than Eigenface because it is relatively less sensitive to changes in expression, including blinking, frowning, and smiling.

**Attacks and Countermeasures.** Current face recognition technology is not adequate for unmonitored authentication systems, because there is no reliable way to test for “Liveness”. Many systems cannot distinguish between a person and a photograph of that person. Visionics counters such attacks by detecting the characteristics of a photograph, such as rectangular borders, and tests for “Liveness” by a multiple frame/ video challenge response system; it asks the user to blink or smile, but not both at the same time. These tests can deter the imposture by photos, but the system can still be spoofed by videos in which a person can blink and smile. Face recognition based on matching 3D head models can recognize faces under a variety of viewing angles and improve the performance of surveillance systems, but it can be spoofed by a plastic 3D human head model. Thus, other biometric technology, such as finger scan, which has a more effective “liveness” test (described in Section 2.3), is more suitable for unmonitored authentication.

**Usability.** Visionics claimed that [25] for head rotation less than 10-15 degrees, there is no degradation in performance. From 15 to 35 degrees, the face recognition discrimination power decreases. Face rotations beyond 35 degrees do not match well to frontal faces with its current LFA technology. There are also 3 major drawbacks of this system: 1. Glare on eyeglasses that obstruct the eyes. 2. Long hair obscuring the central part of the face. 3. Poor lighting would cause the face to be overexposed and low contrast. Also, these systems are mostly capable of recognizing frontal faces, which might be adequate in low to medium level access control applications where users are consistent from session to session. However, in surveillance applications where users are often not aware of the task, it is important for the system to handle faces rotated in depth. Other challenges included the difficulty to create a database that contains all the terrorists. Thus face recognition might not be a feasible solution for surveillance purpose.

Nevertheless, face recognition is the most non-intrusive and the easiest-to-enroll biometric method for identification. It is also considered to be one of the physical characteristics that is constantly exposed to the public. Thus, capturing face images, as opposed to scanning fingers/ irises, will not violate the Bill of Rights and will not infringe privacy. Therefore, face recognition systems are good for providing simple and convenient ways of identifying users in low to medium security applications.

### 2.3 Finger Scan

One of the more mature, well understood, and most widely accepted biometric authentication technologies, finger scan compares a real-time image of a person's fingerprint to a previously enrolled image to decide if the person is who he/she claims to be. Actually the images are neither stored nor compared; they are reduced to a set of minutia points that typically represent the relative position of endings and bifurcations (splits) of the papillary ridges (the lines which make up a fingerprint).

Fingerprints patterns are based on DNA, therefore monozygotic siblings (identical twins) will have very nearly identical fingerprints. However, state-of-the-art systems are able to distinguish between them in most cases [26].

**Attacks and Countermeasures.** There are several ways to defeat finger scan systems, including template replay, last login, adhesive tape and fingerprint replicas. A recent replica attack using gelatin, A.K.A. the Gummy attack [27], was used to circumvent a variety of optical and capacitive devices. Since gelatin is similar in electrical characteristics to human skin, some devices could not differentiate between the enrolled fingerprint and the gummy replica. Some vendors incorporate countermeasures to alleviate the possibility of these attacks. A newer type of silicon device is now available that reads the subdermal layer of skin by radiating a small electrical signal injected into the finger. Other vendors utilize "liveness tests" taking several readings of a fingerprint to detect minute differences. If the readings are exactly the same for each scan, the fingerprint is most



likely a static replica. Some systems, such as those used in military applications, incorporate a duress mode, in which a hostage user will present a different finger to make the system appear to work normally but will send a silent signal to a monitoring station [28]. Additional attacks and countermeasures, not specific to finger scan, are mentioned in section 1.5.

**Usability.** Combining finger scan with a smartcard-based public key infrastructure (PKI) [29] can enhance the overall security framework by:

- Providing stronger authentication than either a smartcard or biometric solution alone.
- Eliminating PINs and passwords that can be compromised or forgotten.
- Enhancing the user experience with a simpler interface that can be used consistently for multiple applications (single sign-on solution).

High trust access control for today’s high-value information systems requires strong authentication — stronger than that provided by single-factor finger scan biometric authentication alone. Finger scan technology provides much better assurance of identity when used for 1:1 authentication against a claimed identity than when used to identify one person from a database of tens of thousands or more. Storing the biometric template in the secure storage of a smartcard affords greater security to the overall system for several reasons:

- The template does not have to travel over the network, thus avoiding capture and replay attacks;
- If a central database were compromised, any or all of the biometric templates could be copied or replaced; and
- It alleviates many of the legal and privacy concerns associated with maintaining a central database of individuals’ biometric data.

The most secure implementation we have seen to date places a high-quality fingerprint sensor on a smartcard reader, and then passes the scanned image (actually the minutia) to a *JavaCard<sup>TM</sup>* where a card applet matches it against the enrolled template stored securely in the card. A match will unlock the card’s secrets, just as a correct PIN will unlock a traditional smartcard. Such a system is much more difficult to compromise than the PIN-based smartcard systems, which are vulnerable to Trojan horse malware capturing the PIN.

## 2.4 Iris Scan

Iris scan technology has many similarities to finger scan. Differentiation of people uses unique patterns in the iris tissue known as the *trabecular meshwork*; the patterns are reduced to a set of minutia points and compared to a stored template. Implementation for authenticating end users at the desktop can be done similarly to fingerscan; by storing and matching the biometric template on a smart card and using the biometric to unlock smartcard secrets. The differences are that: iris patterns are not based in DNA; the minutia computation and

matching algorithms are quite different; and no physical contact is required to obtain a sample image.

An iris image is captured similarly to the way that a camera takes a picture of a person's face, and typically requires a distance of less than one meter. Once captured, the iris image is parsed to create the template. A typical iris will yield approximately 173 independent points. This is more than five times the information that is derived from a fingerprint, and a much larger number than needed to distinguish a single individual among the entire population of the earth [30,31].

As of this writing, one vendor has developed a smartcard solution for iris authentication. Unlike the finger scan smartcard solution described above, it does not provide a combination sensor (camera) and smart card reader, but relies on the PC to process and communicate data between the camera and the smartcard.

Iris recognition technology is probably the most accurate method of distinguishing between individuals. Since Iris patterns have no basis in heredity (DNA), even identical twins can be easily distinguished. The probability of two individuals having just a 75% match between iris patterns is 1 in  $10^{16}$  and the probability of two irises matching 100% is 1 in  $10^{52}$ . By comparison, the estimated number of molecules in the universe is approximately  $10^{35}$ . Independent testing confirms the mathematical theory; at least to the extent that testing has been done with millions of samples. There was not a single false match, in over 6 million tries by independent testing organizations using John Daugman's algorithms [32].

**Attacks and Countermeasures.** Very little is documented of successful attacks on iris scan systems, possibly owing to the fact that the technology is proprietary not broadly implemented. John Daugman, the inventor and patent holder of many iris scan algorithms, describes a way to detect a false iris imprinted on a contact lens [33]. Iridian hints at some "liveness" tests involving detection of blinking and movement of the eye, but states, "As a matter of policy, Iridian Technologies, Inc. does not discuss the details of these properties or specific countermeasures in the open media." [34] One must recognize the risks associated with accepting the assertions of the technology's purveyors, especially in the absence of meaningful independent testing or peer review.

**Usability.** This section is more anecdotal than scientific. It represents what a typical implementation pilot might uncover when evaluating the technology. It is included here to balance the theoretical superiority of iris scan against some practical realities. While we rely on documented independent tests for specific measures, these numbers are intended to illustrate how the system might perform in a typical office environment with typical office workers.

The table below illustrates the recorded information from usability tests. Tests were done with 100 people of varying age, gender and nationality, 12 wearing glasses and 23 wearing contact lenses. Of the 100 tested, 16 could not

use the system at all because, after five tries, they were not able to successfully enroll. Testing was done using the Iridian Authenticam and the standalone authentication demo software V2.0 101987SW Rev A (July 6, 2001) from Iridian Technologies [35].

<b>Authentication Test</b>	<b>Users</b>	<b>FA</b>	<b>FR</b>	<b>Failure to Acquire</b>
Immediately after enrollment	84	0	4	0
7 Days Later	78	0	9	1
14 Days Later	72	0	6	1

Once again, this was not a scientific test, but does provide a snapshot of how people felt about using an iris recognition device. The one person who could not be acquired after the initial enrollment wore glasses. We found out later that the problem was due to overhead lighting causing glare on the glasses. Otherwise there was no correlation between false rejects of people who wore glasses or contacts and those that did not. Those who enrolled with glasses or contacts were authenticated with glasses or contacts. Some of the usability findings and user feedback appear below.

- 15 users reported feeling like there was something done to their eyes after enrolling and authenticating. (Psychological effect?)
- About half of the people tested needed at least 2 tries to enroll successfully. With 17 people who had to try 3 or more times.
- In order to get people to enroll who tried more than 3 times, they had to open their eyes very wide and 4 people had to hold their eyes open with their fingers.
- The angle of the camera seemed to have an effect on authentication.
- 8 people commented that the device was still too large and cumbersome to use on the road. (Another device to carry and set up at remote locations)

Though we concede that Iris Recognition technology is probably the most accurate and unique to individuals, we believe that the hardware could still use further miniaturization for remote applications. The equipment we used would be more useful for applications where the camera could be placed in a specific location with controlled lighting. Also, the enrollment process took too much time per subject, and in a significant number of cases (16 out of 100) was unsuccessful after five tries. This would frustrate users, especially without someone immediately available to help them. We concluded that improvements are required before iris scan can be supported for end user authentication at the desktop in widespread implementations.

### 3 Cultural Issues

Biometric cultural issues involve the way people think about biometrics in various areas include religious objections, health concerns, and the legal aspects of biometrics. Personal privacy is a central issue of the use of biometric technology

and will be used as a framework for much of this discussion. Loss of privacy refers to the loss of control of the information about the individual to whom the information pertains. The right to privacy has appeared as a legal right since 2000 years ago in the Jewish laws: “If one man builds a wall opposite his fellow’s windows, whether it is higher or lower than them . . . it may not be within four cubits. If higher, it must be four cubits higher, for privacy’s sake.” [36]. However, privacy rights are not always attained in modern society, or protected by modern governments. The right to privacy is not explicitly mentioned in the US Constitution, but is implied in Amendments 1, 3, 4, 5, 9, and 14. European Commission’s Directive on Data Protection, which went into effect in October 1998, provides more comprehensive protection, but only to citizens of EU nations. The international community recognizes privacy as a basic human right. Article 12 of the 1948 Universal Declaration of Human Rights states:

*“No one shall be subjected to arbitrary interference with his privacy, family, home, or correspondence, nor to attacks upon his honour and reputation. Everyone has the right to the protection of the law against such interference or attacks.”*

In the 1970s, as rapid computerization brought fear of surveillance, some countries sought to protect individuals from the misuse of personal data. Sweden, Germany, Canada, and France established data protection boards to protect the privacy and the integrity of records on individual citizens. The U.S. also passed a Privacy Act with a similar goal. Nevertheless, both the public and private sectors keep huge databases containing profiles of people’s personal data. According to the Center for Social and Legal Research, 52% of Americans worry about privacy invasions from government agencies and 40% of them worry about privacy invasion of the business sector [37].

Many people believe that the use of biometric systems will cause them to lose their personal privacy. The best way to generally describe the use of biometrics affect on personal privacy is shown in the quotation below.

*“Biometrics, like any technology, is defined by its usage. The broad range of biometric technologies, infrastructures, and deployment environments render any all-encompassing statements on biometrics pointless. Biometrics can be deployed in a privacy-invasive fashion, in a privacy-neutral fashion, and in a privacy-protective fashion.”* [38]

The two main types of personal privacy, information privacy and physical privacy will be used to discuss in what situations biometrics may be used to help or hinder personal privacy [38].

### 3.1 Information Privacy

The concerns for information privacy invasion on biometric systems include:

**Unnecessary or unauthorized collection.** The International Biometric Industry Association (IBIA) [5] Privacy Principal recommends that clear signage or other means of notification be used to inform everyone that video imaging and facial recognition technology are being used in any public area. The IBIA also states that the biometric images should be used only to make comparisons against known violators. In no circumstance should non-matched images be retained in a database once the comparison has been conducted.

**Improper storage and transmission.** Biometric information should be protected from unauthorized access at all times. In fact, any centralized storage of biometric data is controversial. In general, it is less accepted to store physiological biometric data, such as face images and fingerprint images, in a central server than it is to store behavioral biometric data, such as recorded speech waveforms. Our speech waveform is recorded and stored any time we leave a voice mail message. The developing consensus within the industry is that the storage of raw biometric data should be avoided; after feature extraction, the raw biometric data should be deleted. To reduce risk to privacy, a biometric system should be designed to store these feature vectors locally on the PC or on a smartcard wherever possible.

**Surreptitious database linkage.** One of the privacy threats of central storage of biometric information is that it opens up the possibility of its being used to link together personal information derived from different sources. Although the biometric cannot be used as a key, a comparison may be made between a biometric stored as part of, say, a social services program to the same person's biometric that is stored as part of a health care program to confirm that it is in fact the same person's information.

**Function creep.** Function creep is defined as using biometric data for purposes beyond that originally authorized, in violation of information privacy. Many are concerned that their biometric information may be used for other purposes and without their explicit permission. Preventing such abuse of private information would require 1) explicit approval by the information owner for collection, storage, and specific use, 2) audit practices that track the use of the information and the timeframe for retaining the information, and 3) criminal penalties for unauthorized disclosure or use of the information.

It should also be noted that most European countries' privacy policies have been strongly influenced by the Guidelines on the Protection and Privacy of Transborder Flows of Personal Data, published by the Organization for Economic Cooperation and Development (OECD) in 1980. The data privacy principles set out in the Guideline are consistent with those stated by the IBIA and general best practices outlined above.

**Legal Cases and Issues.** A controversial issue concerning unauthorized collection and privacy invasion came up after two face recognition companies, Viisage

[21] and Visionics [22], were employed by the law enforcement to identify terrorists and criminals in the Super Bowl Stadium and in Tampa, FL. It was viewed by some people as a violation of the fourth Amendment and considered an “unreasonable” search. The Supreme Court explained that government action constitutes a search when it invades a person’s reasonable expectation of privacy. This reasonable expectation of privacy does not include physical characteristics that are constantly exposed to the public, such as one’s facial features, voice, and handwriting. Thus, scanning the spectators’ facial characteristics at the Super Bowl did not constitute an unreasonable search by the government. However, it does violate the Privacy Principles set by the IBIA and the OECD.

The low accuracy of face recognition applications and other biometrics can easily create situations in which innocent people are recognized to be criminals. Richard Lawrence Sklar, a political science professor at UCLA was mistaken to be a fugitive and arrested 3 times in 12 years. On one arrest, a customs official confirmed that his birth date, height, weight, eye, and hair color, were the same as the fugitive. He was strip searched, moved from one holding cell to another, and handcuffed to several violent offenders [39].

Canada’s Personal Information Protection and Electronic Documents (PIPED) Act makes it illegal for any private company to collect personal information on an individual without his/her expressed consent or a warrant. Thus, a person walking into a bank, can request that the cameras be turned off, because it violates his/her privacy rights. It would be against the law for the bank managers to refuse to do so. This act will allow a security video recording to be handed over to police if it shows evidence of criminal activity. Of course, if the camera is shut off and a robbery is committed, there will be no proof to hand over. The act does not apply to activities of the Canadian government. In the first decision under the act, which came into effect Jan. 1, 2001, federal Privacy Commissioner George Radwanski told a Yellowknife security company that the installation of street surveillance cameras is unlawful [40].

### 3.2 Physical Privacy

Some people think that the use of biometrics is offensive, unhygienic, and harmful. Thus, people hesitate to give their biometric information.

In the past, fingerprinting has been generally associated with criminals. User education can help erase this stigma as fingerprinting may also provide simplified use and/or increased security. Other people think biometric authentication technologies that require physical contact, such as finger scanning and palm scanning are unhygienic. From our experience, finger and palm scanning do not appear to be any more unhygienic than using a door knob, currency, or other similar items that are touched by a large number of people.

Some pious people consider biometric authentication, such as the finger scan, as being similar to the “Mark of the Beast” mentioned in the Book of Revelations, Chapter 13:16-18:

*“And that no man might buy or sell, save he that had the mark, or the name of the beast, or the number of his name.”*

The method of capturing a fingerprint by computer is noninvasive in that only the image of the fingerprint is captured. No chip is inserted into and no mark is stamped on the hand or the finger.

A court ruling, *Buchanan v. Wing* (a Commissioner of the New York State Department of Social Services) (1997) [41], was about this type of religious issue. Buchanan and her family were recipients of Aid to Families with Dependent Children and food stamps from the Broome County Department of Social Services. The Department of Social Services was trying to prevent duplicated registration and required all the recipients to identify themselves by an automated finger imaging system as a condition of eligibility for benefits. Buchanan refused to do that because of her religious convictions. The Commissioner thereafter discontinued her family's aid and food stamps. Buchanan believed that finger imaging may be the same thing as the "Mark of the Beast" and refused to participate. The court ruled that Buchanan failed to demonstrate a good cause basis for exemption from the finger imaging requirement, and that Wing's decision was rational.

Some people have concerns that biometric technology, such as iris and retinal scans can be harmful to their health. Although a visible light and measurements from a short distance ( $< 3$  cm) have been required for retina scanning, it is not harmful. It measures the pattern of blood vessels in the retina. For example, the "red eye" effect in a photograph is an image of the retina. The iris scan is nothing more than taking a picture of the pattern of the iris without using a flash.

### 3.3 Proper Uses of Biometrics

Biometrics, if used properly, can help protect sensitive personal information. Replacing a PIN with a biometric can protect sensitive encryption keys and other personal information stored on a smart card while also keeping the biometric in the hands of the owner. Biometric technologies are not harmful, not a health risk, nor the "Mark of the Beast". Biometric information is subject to the same laws that govern the use of personal information, but many of these laws still have to be "tested" with biometric systems.

Since different countries have different privacy laws and standards, a biometric security system should be designed so that it will not violate the privacy guidelines upon which many laws are based, such as those set up by the International Biometric Industry Association (IBIA), and the Organisation for Economic Co-operation and Development (OECD). Because these are only guidelines, and privacy laws vary from country to country, an organization's legal team should review the implementation plans for any system making use of biometric technology.

## 4 Recommendations

The following recommendations are drawn from the investigation, research, and analysis of Biometric Authentication technologies. This investigation was in-

tended to identify uses of biometric authentication technology within IT infrastructure.

Biometrics can provide increased security for applications while increasing convenience for users, and decreasing cost. For example, unlocking a smartcard with a finger is arguably easier than having to remember and type in a PIN. Secure replacement of forgotten PINs and passwords is costly, estimated at \$40 per call, by many studies across the industry. Unlike a password, PIN, or token, biometrics cannot be shared, borrowed, or stolen, thereby limiting unauthorized delegation, improving integrity of audit information, and providing one component of a system of non-repudiation of electronic transactions. Following is a summary of specific recommendations:

**Recommendation 1:** Speaker Verification is useful for authenticating users to voice-based applications where the only connection between a user and the application server is a phone line (Plain, Ordinary Telephone Service, or POTS). It has been argued that, with the impending ubiquity of digital telephony and wireless LANs, POTS and its supporting technologies will be obsolete in less than 5 years. While this argument may be true in some areas of the world, we believe that there will be a market for SV over POTS well beyond the next 5 years, in countries that, like the US, have large rural populations with wired telephone service.

Existing systems, such as voice-mail, which use touch-tones for authentication and navigation, should be linked to the voice portal when available, to take advantage of stronger authentication and integration with other voice applications. Thus, SV is recommended to replace touch-tone passwords in voice systems. Section 2.1 details the arguments for SV over other telephone-based authentication systems.

**Recommendation 2:** The inaccuracy of SV (higher than acceptable FR and FA rates) has made it less useful in the past. This poor performance was mainly due to handset or channel mismatch. By removing the effects of handset mismatch, SV can be made useful. This can be improved by using a new technique [42] that we developed to classify the handset in use before verification.

**Recommendation 3:** Biometrics should be used as the human-to-machine authentication link, and should be implemented as close as possible to the point where a human interacts with a machine, such as the BioSwatch prototype developed by HP Labs [43], rather than transmitting biometric data across digital networks. The first computing device encountered should authenticate the user, and then machine-to-machine authentication, such as PKI or Kerberos, should be used across the network.

**Recommendation 4:** Biometric data should be kept under the control of the person to which it belongs. Centralized storage of biometric data is controversial on privacy grounds, is or may be restricted in some countries, and should be avoided where possible. Centralized storage of voice data is less controversial and more accepted than fingerprint, iris, face, or other physiological biometrics. Using a biometric to unlock a smartcard's PKI authentication function provides two-factor authentication and, by storing the biometric template on the card,



avoids the controversial requirement for centralized storage of biometric data.

**Recommendation 5:** Iris Recognition should be considered for high security authentication in the near future. Iris Recognition currently falls short in cost and user acceptance but seems to have the best combination of performance and resistance to attack, making it the technology to watch for future implementations. Iris and Retina recognition technologies are the only biometric technologies that have been found to provide sufficient accuracy for identification and single-factor authentication, provided that the overall system has no other vulnerabilities to compromise (data insertion or device spoofing, etc.).

**Recommendation 6:** Applications should not interface directly to biometric authentication subsystems. Biometric technologies and the programming standards to interface them are expected to change rapidly over the coming years, making it important to shield applications from this volatility. Applications should not interface to biometric subsystems directly, but rather work through standardized middleware that can interact with the biometric systems directly or indirectly. The middleware will allow for upgrade and replacement of authentication subsystems without requiring any changes to the application. Both the SV and the FS technologies can be implemented in this way. The SV recommendation proposes using a voice portal as the middleware to connect users to applications. The SV subsystem interfaces with the portal and not directly with any application. The FS recommendation unlocks a smartcard with a biometric so that software downstream, need not be modified to use the new technology.

**Recommendation 7:** Biometric authentication should be combined in multi-factor authentication systems to improve security. Technology seems to favor the attacker in biometric systems — examples are voice-cloning attacks on speech systems, artifact (fake fingerprint) attacks on finger scan systems, and using photographs to fool facial recognition systems. Effective countermeasures involve traditional token authentication or PKI as a second factor (for both SV and FS systems). Where Biometrics cannot stand alone as a security technology, it should be enhanced by other means to ensure trust and integrity of the overall solution. Such means may include human monitoring of the system, such as with the INSPASS system, traditional cryptographic technology such as PKI to ensure integrity of digital communications, and tamper-resistant packaging, such as smart cards or ATMs for un-monitored biometric systems.

## References

1. “Emerging Technologies That Will Change the World,” *MIT Technology Review*, January/February 2001, [http://www.technologyreview.com/articles/tr10\\_toc0101.asp](http://www.technologyreview.com/articles/tr10_toc0101.asp).
2. B. Schneier, “Biometrics: Uses and Abuses,” *Inside Risks 110, Communications of the ACM*, vol. 42, no. 8, August 1999, <http://www.counterpane.com/insiderisks1.html>.
3. “International Biometric Group,” [http://www.biometricgroup.com/e/biometric\\_market\\_report.htm](http://www.biometricgroup.com/e/biometric_market_report.htm).
4. “BioAPI Consortium,” <http://www.bioapi.org>.

5. "International Biometrics Industry Association," <http://www.ibia.org>.
6. "Microsoft and I/O Software Strengthen Industry Adoption of Biometrics," <http://www.microsoft.com/PressPass/press/2000/May00/BiometricsPR.asp>.
7. "OASIS XML Common Biometric Format (XCBF) TC," <http://www.oasis-open.org/committees/xcbf/>.
8. T. Mansfield et al, "Biometric Product Testing Final Report," *Center for Mathematics and Science Computing, National Physics Lab, CESG/BWG 2001*, 2001, <http://www.cesg.gov.uk/technology/biometrics/media/Biometric%20Test%20Report%20pt1.pdf>.
9. "Trusted Computing Platform Alliance," <http://www.trustedpc.org/>.
10. J. Campbell, "Speaker recognition: A tutorial," *Proceedings of the IEEE*, vol. 85, no. 9, 1997.
11. J. Naik, "Speaker verification: A tutorial," *IEEE Communications Magazine*, 1990.
12. "Nuance Verifier 3.0 Technical Data Sheet," [http://www.nuance.com/pdf/verifier3\\_techds.pdf](http://www.nuance.com/pdf/verifier3_techds.pdf).
13. M. Carey and R. Auckenthaler, "User Validation for Mobile Phones," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2000.
14. "The Orpheus System of Persay," <http://www.persay.com/pages/Orpheus.asp>.
15. "S/Key," *FreeBSD Handbook*, [http://www.freebsd.org/doc/en\\_US.ISO8859-1/books/handbook/skey.html](http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/skey.html).
16. "The One-Time-Password SASL Mechanism," *RFC2444*, 1998, <http://www.faqs.org/rfcs/rfc2444.html>.
17. "VeriVoice Secure Lock," [http://www.verivoice.com/verivoice\\_sl.htm](http://www.verivoice.com/verivoice_sl.htm).
18. "Text-to-Speech Voice Cloning System by AT&T Labs," <http://www.naturalvoices.att.com/>.
19. "Text-to-Speech System by Rhetorical," <http://www.rhetorical.com/>.
20. R. Chellapa, C. Wilson, and S. Sirohey, "Human and machine recognition of faces: A Survey," *Proceedings of the IEEE*, vol. 83, no. 5, pp. 705–741, 1995.
21. "Viisage Corporation," <http://www.viisage.com>.
22. "Visionics Corporation," <http://www.visionics.com>.
23. M. Turk and A. Pentland, "Face recognition using eigenfaces," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 1991, pp. 586–591.
24. P. Penev and J. Atick, "Local Feature Analysis: A General Statistical Theory for Object Representation," *Network: Computation in Neural Systems*, vol. 7, pp. 477–500, 1996.
25. "Visionics FaceIt Technology, Frequent Asked Technical Questions," <http://www.visionics.com/faceit/faqs/faqs.pdf>.
26. A. Jain et al, "Twin Test: On Discriminability of Fingerprints," in *Proc. 3rd International Conference on Audio- and Video-Based Person Authentication*, Sweden, 2001, pp. 211–216.
27. T. Matsumoto, H. Matsumoto, K. Yamada, and S. Hoshino, "Impact of Artificial Gummy Fingers on Fingerprint Systems," in *Proc. of SPIE, Optical Security and Counterfeit Deterrence Techniques IV*, January 2002.
28. "Finger-scan Vendors," [http://www.finger-scan.com/finger-scan\\_vendors.htm](http://www.finger-scan.com/finger-scan_vendors.htm).
29. "Understanding PKI," <http://verisign.netscape.com/security/pki/understanding.html>.
30. J. Daugman, "How Iris Recognition Works," <http://www.cl.cam.ac.uk/users/jgd1000/irisrecog.pdf>.
31. "International Biometric Group, Iris Recognition: The Technology," [http://www.iris-scan.com/iris\\_technology.htm](http://www.iris-scan.com/iris_technology.htm).

32. J. Daugman, "Test Reports on the Daugman's Algorithms," <http://www.cl.cam.ac.uk/users/jgd1000/iristests.pdf>.
33. J. Daugman, "Countermeasures against Subterfuge," <http://www.cl.cam.ac.uk/users/jgd1000/countermeasure.gif>.
34. G. Williams, "Iris Recognition Technology," *Iridian Technologies*, 2001, <http://www.politec.com/press%20info/white%20papers/Iris%20Recognition.pdf>.
35. "Iridian Technologies," <http://www.iridiantech.com>.
36. T. B. Batra, "Hebrew English edition of the Babylonian Talmud," vol. 1, pp. 22.
37. D. Whitfield and S. Landau, *Privacy on the Line, The Politics of Wiretapping and Encryption*, The MIT Press, Cambridge, Massachusetts, 1998.
38. "BioPrivacy FAQ's and Definitions," <http://www.bioprivacy.org/FAQ.htm>.
39. T. Forester and P. Morrison, *Computer Ethics: Cautionary Tales and Ethical Dilemmas in Computing*, The MIT Press, Cambridge, Massachusetts, 1993.
40. J. Ross, "Security Cameras In Banks, Private Business Ruled Illegal In Canada," *The Ottawa Citizen*, <http://www.rense.com/general12/sec.htm>.
41. "Connecticut Department of Social Services' Biometric ID Project," [http://www.dss.state.ct.us/digital/ny\\_test.htm](http://www.dss.state.ct.us/digital/ny_test.htm).
42. P. Ho, "A Handset Identifier Using Support Vector Machine," in *Proc. IEEE International Conference on Spoken Language Processing*, Denver, CO, USA, 2002.
43. J. Beckett, "HP + Swatch = Web Watch," November 1999, <http://www.hpl.hp.com/news/swatch.html>.

# Denial of Access in Biometrics-Based Authentication Systems

Luciano Rila

Information Security Group, Royal Holloway, University of London, UK  
`luciano.rila@rhul.ac.uk`

**Abstract.** Biometrics has been widely recognised as a powerful tool for problems requiring personal identification. Unlike traditional authentication methods, however, biometrics-based authentication systems may reject valid users or accept impostors. The accuracy of a biometric system could be defined as its combined ability to reject impostors and accept valid users. The biometrics industry places heavy emphasis on security issues relating to the rejection of impostors while denial of access remains largely neglected in the evaluation of biometric systems. In this paper, we discuss how denial of access may impact on all major aspects of a biometric system and propose solutions to reduce the probability of denial of access based on more sophisticated authentication decision-making strategies.

## 1 Introduction

Biometrics has been widely recognised as a powerful tool for problems requiring personal identification. Most automated identity authentication systems in use today rely on either the possession of a token (smartcard, USB token) or the knowledge of a secret (password, PIN) to establish the identity of an individual. The main problem with these traditional approaches to identity authentication is that tokens or PIN/passwords can be lost, stolen, forgotten, misplaced, guessed, or willingly given to an unauthorised person. Biometric authentication, on the other hand, is based on either physiological features of the body (such as fingerprint, hand geometry, and iris pattern) or behavioural characteristics of the individual (such as voice prints and written signature) and therefore does not suffer from the disadvantages of the more traditional methods.

Biometrics has the potential for increasing the security of authentication systems. In PIN/password-based authentication systems, the use of safe PINs and passwords — i.e. long, complicated ones — is rarely enforced since such passwords are difficult to remember, and therefore users tend to write them down. It has also been demonstrated that users may share a password or token when working on a group task [1]. These practices significantly weaken the security of the system. The very nature of the biometric methods overcomes such problems since biometric authentication information cannot be transferred or shared and is therefore a powerful weapon against repudiation.

Unlike traditional methods, however, biometrics-based authentication systems may reject valid users or accept impostors. The accuracy of a biometric system could be defined as its combined ability to reject impostors and accept valid users. A significant body of work has been devoted to the development of suitable metrics for evaluating the accuracy of biometrics systems [2,3,4]. Procedures for conducting technical testing for the purpose of field performance evaluation have also been proposed [5].

As pointed out in [6], the biometrics industry places heavy emphasis on security issues relating to the rejection of impostors. However, the overall performance of a biometric system cannot be assessed based only on this metric. Denial of access, i.e. rejection of valid users, is often largely neglected in the evaluation of biometric systems.

Consider as an example an online banking application employing biometrics-based authentication. A bank client wishes to make a payment of a very large amount on the pay-by date, as is often the case. For some reason, he/she is unable to provide biometric data of acceptable quality, being consequently denied access to the system. In fingerprint-based biometric systems, for example, this may happen due to injury in the finger/fingers used for authentication. In this case, it might be claimed that the bank is providing the client with an unreliable authentication method and might therefore be held responsible for any loss on the part of the client. This problem also represents a major hindrance to the user acceptance of biometrics technologies. It is thus clear that such problems should be thoroughly investigated and understood.

In this paper we consider the impact of denial of access in biometrics-based authentication systems and discuss solutions to this problem.

This paper is organised as follows. Section 2 contains a general model for biometric authentication systems. This is followed in Section 3 by a discussion of various types of failures within a biometric system. Section 4 considers the impact of false rejection of valid users on certain application domains, and, in Section 5, we propose various approaches to ameliorating the impact of false user rejections. The paper ends with some conclusions.

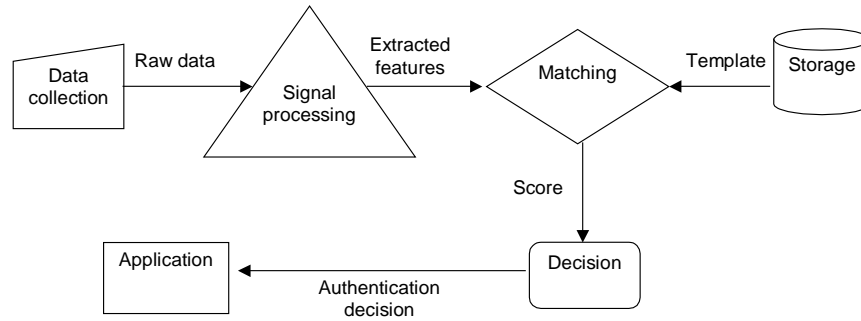
## 2 General Model for Biometrics-Based Authentication

According to [7], a general biometric system is composed of the following logical modules:

1. Data collection subsystem;
2. Signal processing subsystem;
3. Matching subsystem;
4. Storage subsystem;
5. Decision subsystem;
6. Transmission subsystem.

A block diagram for the general authentication model is given in Figure 1. The authentication process involves the raw biometric data of the claimant being captured in the data collection subsystem by an input device or sensor, and

then transferred to the signal processing subsystem where the feature extraction takes place. The matching subsystem receives the extracted features from the signal processing subsystem and retrieves the biometric reference template associated with the claimed identity from the storage subsystem. The matching subsystem then compares the submitted biometric sample with the reference template yielding a score, which is a numeric value indicating how closely the submitted sample and the reference template match. The decision subsystem receives the score and, according to a confidence value based on security risks and a decision-making policy, decides whether to accept the claimant or not. The authentication decision is finally passed on to the application.



**Fig. 1.** General model for biometric authentication.

Note that these are logical modules, and therefore some systems may integrate several of these components into one physical unit.

The biometric reference template is a crucial element in the biometric technology. A template is a small file containing distinguishing features of the user derived from his/her biometric data. The process of collecting biometric data from a user, and the subsequent processing and storing of their biometric reference template in the storage subsystem is known as enrolment. Subsequent authentication sessions compare a live biometric sample provided by the user with the user's reference template generated by the system during the enrolment procedure. Enrolment plays a critical role in the overall performance of a biometric system, as we will discuss in Section 3.3.

### 3 Evaluation of Biometrics Systems

The nature of biometric data is such that two different measurements of the same biometric feature from the same person are very likely to be different. For

example, when fingerprint recognition is used, the fingerprint image captured by the fingerprint sensor may vary, e.g., due to skin conditions, dirt, grease, or the position in which the finger is placed on the sensor. Biometric matching determines the degree of similarity between the live submitted biometric sample and the reference template. The result of this comparison is a number known as match score, which, in most systems is compared against a tolerance threshold. If the match score is above the threshold, the result of the biometric matching is a *match*. If the match score falls below the threshold, the result is a *non-match*.

It is therefore clear that the value of the tolerance threshold is a key parameter in the performance of a biometric system. The tolerance threshold establishes a lower bound for the degree of similarity necessary for a comparison to be considered a match, which, ultimately, allows access to an application. Hence, setting the tolerance threshold is a vital decision in the security policy of any biometrics-enabled application, and consequently it is fundamental to understand how the threshold affects the overall performance of the biometric system.

According to [6], the performance of a biometric system should be assessed by the analysis of three accuracy metrics: the false match rate, the false non-match rate, and the failure-to-enrol rate. These three metrics are closely related and their behaviour is determined by the value of the tolerance threshold, as we will see next.

### 3.1 False Match Rate (Accepting Impostors)

The false match rate (FMR) is the probability that an individual's template will incorrectly be considered a match for a different individual's biometric sample.

The FMR can be regarded as a function of the tolerance threshold: the higher the tolerance threshold, the lower the FMR. If the minimum degree of similarity necessary for a match is set relatively high, a match between two different fingerprints, however similar they are, becomes less likely. Conversely, if the tolerance threshold is reduced, the FMR increases.

An impostor break-in is a particularly critical security issue in applications such as access to weapons or to a bank safe. Also, for an emerging technology still seeking public acceptance, solutions providing high security must be perceived as not vulnerable to break-ins.

It is important to understand the difference between the false match rate and the false acceptance rate. The false match rate is the probability of a false match for a single comparison. The false acceptance rate is the probability that an impostor is accepted by the system. Since, in general users are allowed a certain number of attempts before being rejected by the system, the false acceptance rate is likely to be larger than the FMR. An impostor may also be able to attempt impersonating more than one valid user. Therefore the actual probability of impostor break-in is higher than the FMR.

### 3.2 False Non-match Rate (Rejecting Valid Users)

The false non-match rate (FNMR) is the probability that a valid user's biometric will incorrectly be considered a non-match for his/her reference template. The FNMR is therefore related to the issue of denial of access.

As with the FMR, the FNMR can be regarded as a function of the tolerance threshold: the higher the tolerance threshold, the higher the FMR. Thus, if the minimum degree of similarity necessary for a match is set relatively high, a valid user may have difficulties in providing a sample of his/her biometric data that matches so closely against his/her own reference template. Conversely, if the tolerance threshold is reduced, the FNMR also decreases.

A false non-match occurs when there is not a sufficiently high similarity between the user's biometric sample and his/her reference template. This may happen due to: changes in the user's biometric data (fingerprints may vary due to skin condition, ageing, injuries; voice prints may vary due to a sore throat, emotional state.); change in the biometric data presentation (placing a finger in a different position; speaking at a different volume or closer or further away from the microphone); and/or changes in the environment (variations in humidity for fingerprint recognition systems, background lighting for face-scan systems, or ambient noise for voice-scan systems).

It is important to understand the difference between the false non-match rate and the false rejection rate. The false non-match rate is the probability of a false non-match for a single comparison. The false rejection rate is the probability that a valid user is rejected by the system. Again users are generally allowed more than one attempt before being rejected by the system, and so the false rejection rate is typically lower than the false non-match rate.

Note that the FMR and the FNMR are inversely related. If an application requires high security, setting the tolerance threshold too high may eliminate the possibility of an impostor break-in but will result in an undesirably high probability of denial of access. Conversely, if denial of access is to be avoided by setting the tolerance threshold relatively low, the probability of an impostor break-in increases. The choice of value for the tolerance threshold therefore involves a trade-off between the two types of error and determines the security and convenience of a biometrics-based authentication system.

### 3.3 Failure-to-Enrol Rate

The failure-to-enrol rate (FTER) represents the proportion of users from the total population for whom the system is unable to generate repeatable templates. This will include those unable to present the required biometric feature, those unable to provide sufficiently distinctive biometric data, and those unable to match reliably against their template following an enrolment attempt. The design of the biometric solution may also make it difficult for some users to provide consistent biometric data.

Failure-to-enrol will necessitate the use of an alternative authentication method for those users unable to enrol. One possible reason for replacing



knowledge-based and token-based authentication systems with a biometrics-based authentication system is an increase in the overall security. Employing passwords as a back-up method for those users unable to enrol might therefore mean resorting to a less secure authentication method. Failure-to-enrol also potentially increases the burden of administration procedures for password management.

The FTER is affected by system design, user training, and ergonomics. It is also affected by the quality evaluation of the collected biometric data and the tolerance threshold. During enrolment — and in some systems also during authentication sessions — the quality of the collected biometric data is assessed in order to ensure reliable matches in subsequent authentication sessions. For example, in fingerprint recognition systems, a minimum number of minutiae is required for reliable matching. Lowering the quality requirements for enrolment will decrease the failure-to-enrol rate but may compromise the overall security and usability of the system. Failure-to-enrol is also related to false non-match. If the tolerance threshold is set too high, users may not be able to obtain a match following enrolment. Adjusting the value of the threshold to decrease the FTER would also affect the FMR and FNMR.

## 4 Impact of Denial of Access in Biometrics-Based Authentication Systems

Potential markets for biometrics deployment are characterised by an inherent need for user authentication to grant access to resources and/or sensitive data. In order to understand the impact of denial of access in biometric systems, we consider three such applications as follows:

- Physical access: security requirements for physical access applications can vary widely depending whether access is being granted to a bank safe or to a company's building. In an application involving access to a room or a laboratory by employees in a company, it is reasonable to assume that most authentication attempts will be genuine. In this case, denial of access would result in loss of productivity, frustration, and an increased burden on support personnel.
- Online banking: internet-based access to account and payment systems has enabled bank clients to conduct online transactions from home or office. Online banking represents an important potential market for biometrics deployment. As pointed out in the introduction, if a client attempting to make a payment of a large amount on the pay-by date is denied access and is unable to complete the transaction as a result, then the bank or the client may both incur financial loss. Also the client will most likely feel that the bank has provided an unreliable method for authentication, thus damaging the client's faith in the institution and the client's trust in the biometrics technology.
- Healthcare application: applications in healthcare are mainly concerned with access to personal information. In emergency medical scenarios, for example,

biometrics may offer a unique solution for patients without identification or unable to communicate. In this case, denial of access could represent a delay in providing healthcare service putting the patient's life at risk.

Based on the examples above, it is clear that denial of access in biometric systems greatly impacts on the usability of the system by failing to identify valid users, and consequently on the public acceptance of biometrics by harming the user trust in the emerging technology. Both aspects may represent significant obstacles to the wide deployment of biometric systems.

Denial of access also has an impact on security since it typically results in the use of alternative authentication methods such as passwords, which may be less secure.

## 5 Authentication Decision-Making

Multibiometric systems, i.e. systems employing more than one biometric technology to establish the identity of an individual, seek to improve the overall performance of the biometric system by checking multiple evidences of the same identity [9,10,11]. As pointed out in [12], multibiometrics can reduce the probability of denial of access at the expense of an increase in the FMR. Multibiometric systems also suffer from other major drawbacks. Use of multiple biometric measurement devices will certainly impose significant additional costs, will result in more complex user-machine interfaces, and will impose additional management complexity.

Another possible means of reducing the impact of denial of access is to consider more elaborate forms of authentication decision-making. It is important to observe that the authentication decision does not depend only on the FNMR. Authentication decision-making may involve use of one or more other approaches, and we now consider ways in which a more complex authentication decision-making process can reduce the likelihood of denial of access.

### 5.1 Combining Biometrics and Passwords

The outcome of the biometric matching is generally described in the literature as a binary yes/no decision [6,7,8], giving rise to accuracy metrics such as the FMR and the FNMR. Alternatively, however, the system can make a ternary matching decision: *match*, *non-match*, or *inconclusive*. So, instead of one tolerance threshold, the system would have two tolerance thresholds. A match score above the upper bound threshold would be considered a match. A match score below the lower bound threshold would be considered a non-match. A match score in the interval between the lower bound and the upper bound thresholds would be considered inconclusive.

In the event of an inconclusive matching attempt, the user would be required to resort to a password back-up procedure to proceed with authentication. In [12], Daugman discusses how to combine more than one biometric test to

arrive at a decision, and the same principles can be applied to biometrics-based authentication combined with the use of a password. Let  $FMR_{inc}$  and  $FMR_m$  denote the probabilities that an impostor scores within the inconclusive and match intervals respectively, and  $FNMR_{inc}$  and  $FNMR_{nm}$  denote the respective probabilities that a legitimate user scores within the inconclusive and non-match intervals. Also denote the probability that the use of a password is fraudulent by  $P(PWFraud)$ .  $P(PWFraud)$  corresponds to the false match rate for the password-only system since it represents the probability of granting access to an impostor. We assume that the FNMR of a password-based authentication system is zero. Following [12], the FMR and the FNMR for the combined authentication system, denoted  $FMR_{comb}$  and  $FNMR_{comb}$ , are as follows:

$$\begin{aligned} FMR_{comb} &= FMR_{inc} \times P(PWFraud) + FMR_m \\ FNMR_{comb} &= FNMR_{nm} \end{aligned}$$

Therefore, the use of a password strengthens the security of the system since the combined FMR is lower than the FMR would be if the threshold was set at the lower end of the inconclusive interval. Nevertheless, in the event of an inconclusive matching outcome, the system is operating with a probability of denial of access lower than that associated with a match outcome.

Depending on the application, successful authentication through an inconclusive matching decision might involve only giving restricted access to the desired resources.

## 5.2 Combining Single-Biometric Tests

The simplest decision policy for a biometric system that can be adopted is to accept the user's claim of identity if a match occurs and to reject the user if a non-match occurs. Such a mode of operation relies on the matching of a single biometric sample and may therefore produce unacceptable FMR and FNMR. According to [7], most biometrics-based authentication systems allow the user at least three attempts before rejecting the user. More generically, the system accepts the user's claim of identity if  $m$  out of  $n$  submitted samples match the corresponding reference template.

Another possible authentication strategy is to allow the user to attempt authentication using two or more biometric features. In this strategy, reference templates for two or more biometric features, e.g. fingerprints from different fingers or different voice passwords, are stored in the system. In order to be authenticated, the user is required to submit two or more different fingerprints, or two or more different voice passwords. The results of the different tests are then combined for the system to achieve an authentication decision, much in the same way as in multibiometric systems but employing only one type of biometric measurement. The FNMR can be reduced if the biometric tests are combined using an 'OR' operator. Note however that such a combination rule increases the FMR.

A more complex solution, using a combination of single-biometric tests, was proposed in [13]. The proposed scheme combines four different fingerprint matching algorithms to improve the performance of a fingerprint verification system.

### 5.3 User-Dependent and Transaction-Dependent Thresholds

Biometric systems typically employ a common tolerance threshold across all users. However the distinctiveness and consistency of the biometric features vary from individual to individual. Setting the tolerance threshold according to these properties for each user may therefore improve the FNMR. For example, a user able to provide very distinctive and consistent biometric data can be assigned a relatively high threshold whereas a user more prone to biometric inconsistencies can be given a lower threshold. In such an approach, the user-dependent threshold can be set according to the results of live matches during enrolment. Alternatively, user-dependent thresholds can be learned adaptively based on previous authentication sessions, such as in [14]. Note that the value of user-dependent thresholds must comply with the security requirements of the application.

The impact of denial of access can also be reduced by employing a transaction-dependent hierarchy for the thresholds. Different transactions within the same application may require different security levels and transaction-dependent tolerance thresholds can be assigned to each transaction (or class of transactions) accordingly. Relatively high thresholds are only used for granting access to privileged resources. In an online banking application, for example, account access for viewing a balance might be assigned a lower threshold than that assigned to access the means to make a large payment.

### 5.4 Context-Dependent Decision-Making

An alternative approach to user authentication involves using a more complex decision-making programme that takes into account the context of the transaction within an application. The tolerance threshold could be set dynamically according to some relevant contextual information about the transaction. For example, in a physical access application, information as to whether the claimant has accessed the room before or whether the room is usually accessed at a particular time of day — or night — could be used to set a particular value for the threshold. In an online banking application, other information would be employed such as whether the claimant has made a specific transaction before and at what time of the month/week/day, or if the payment value being requested by the claimant is close to the average value of his/her previous payments. These are subjective assessments that are highly dependent on the security requirements of the application.

## 6 Conclusions

Biometrics has emerged as a powerful tool for automated identification systems, overcoming the problems presented by the traditional methods. Unlike the tra-

ditional methods, however, biometrics-based authentication systems may accept impostors and reject valid users. In the evaluation of biometrics systems, heavy emphasis has been placed on the rejection of impostors while denial of access has remained largely neglected.

In this paper, we have discussed how denial of access impacts on: (1) the usability of the system by failing to identify valid users, ultimately representing a major hindrance to the public acceptance of biometrics; (2) the security of system since it typically results in the use of other authentication methods such as passwords which may be less secure.

We have also proposed four strategies to reduce the probability of denial of access based on more sophisticated authentication decision-making process. In the first strategy, the outcome of the biometric matching is ternary: match, non-match, and inconclusive. In the event of an inconclusive matching attempt, the user is required to provide a password to get access to the system. We have shown that combining biometrics with passwords indeed increases the security of the system at the inconclusive operating point while the probability of denial of access is lower than that at the match operating point. In the second strategy, two or more single-biometric tests are combined in order to reach an authentication decision. In the third strategy, the tolerance thresholds are set according to the user's ability to provide distinctive and consistent biometric data and/or according to the security requirements of a specific transaction. Finally, in the fourth solution, information regarding the context of the transaction is incorporated into the decision-making programme and used to set the value of the tolerance threshold dynamically. Note that an authentication decision-making programme may involve use of one or more of the proposed solutions.

**Acknowledgements.** The work described in this paper has been supported by the European Commission through the IST Programme under Contract IST-2000-25168 (Finger.Card). The information in this document is provided as is, and no guarantee or warranty is given or implied that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

The author would like to thank his colleagues in the Finger.Card Project for their encouragement and advice. Special thanks to Prof. Chris J. Mitchell for his invaluable support and his suggestions to improve this paper.

## References

1. A. Adams and M. A. Sasse, "Users are not the Enemy", *Communications of the ACM*, 42, no. 12, pp. 41–46, Dec. 1999.
2. J. L. Wayman, "Technical testing and evaluation of biometric identification devices", in *Biometrics: Personal Identification in Networked Society*, A. Jain, R. Bolle, and S. Pankati (editors), Kluwer Academic Publishers, pp. 345–368, 1999.

3. R. M. Bolle, S. Pankanti, and N. K. Ratha, "Evaluating techniques for biometrics-based authentication systems (FRR)", in *Proceedings 15th IAPR Int. Conference on Pattern Recognition*, vol. II, pp. 835–841 Barcelona, Spain, Sep. 2000.
4. J. G. Daugman and G. O. Williams, "A proposed standard for biometric decidability", in *Proceedings CardTech/SecureTech Conference*, Atlanta, GA, pp. 223–234, 1996.
5. United Kingdom Biometric Working Group, "Best practices in testing and reporting device performance", version 1.0, Mar. 2000.
6. S. Nanavati, M. Thieme, and R. Nanavati, *"Biometrics: Identity Verification in a Networked World"*. John Wiley and Sons, 2002.
7. ISO/DIS 21352: 2001, Biometric information management and security, ISO/IEC JTC 1/SC 27 N2949.
8. A. K. Jain, R. Bolle, and S. Pankanti, "Introduction to biometrics", in *Biometrics: Personal Identification in Networked Society*, A. Jain, R. Bolle, and S. Pankati (editors), Kluwer Academic Publishers, pp. 1–41, 1999.
9. L. Hong, A. K. Jain, and S. Pankanti, "Can multibiometrics improve performance?", in *Proceedings AutoID'99*, Summit, NJ, pp. 59–64, Oct. 1999.
10. A. Ross, A. K. Jain, and J. Z. Qian, "Information fusion in biometrics", in *Proceedings 3rd International Conference on Audio- and Video-Based Person Authentication*, pp. 354–359, Sweden, Jun. 2001.
11. N. Poh and J. Korczak, "Hybrid biometric person authentication using face and voice features", in *Proceedings 3rd International Conference on Audio- and Video-Based Person Authentication*, pp. 354–359, Sweden, Jun. 2001.
12. J. G. Daugman, "Biometric Decision Landscape", Technical Report No. TR482, University of Cambridge Computer Laboratory, 1999.
13. S. Prabhakar and A. K. Jain, "Decision-level fusion in fingerprint verification", *Pattern Recognition*, Vol. 35, No. 4, pp. 861–874, 2002.
14. A. K. Jain and A. Ross, "Learning user-specific parameters in multibiometric system", in *Proc. International Conference on Image Processing (ICIP)*, Rochester, New York, Sep. 2002.

# A Novel Approach to Proactive Password Checking

Carlo Blundo<sup>1</sup>, Paolo D'Arco<sup>2</sup>, Alfredo De Santis<sup>1</sup>, and Clemente Galdi<sup>3</sup>

<sup>1</sup> Dipartimento di Informatica ed Applicazioni  
Università di Salerno, 84081, Baronissi (SA), ITALY  
{carblu,ads}@dia.unisa.it

<sup>2</sup> Department of Combinatorics and Optimization  
University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada  
pdarco@cacr.math.uwaterloo.ca

<sup>3</sup> Computer Technology Institute and  
Dept. of Computer Engineering and Informatics  
University of Patras, 26500, Rio, Greece  
clegal@ceid.upatras.gr

**Abstract.** In this paper we propose a novel approach to strength password-based access control strategies. We describe a proactive password checker which uses a *perceptron* to decide whether a user's password is easy-to-guess. The checker is simple and efficient, and it works since easy and hard-to-guess passwords seem to be *linearly separable*. Experimental results show that the error rates in many cases are close to zero, memory requirements can be quantified in few bytes, and the answers to classification queries are almost immediate. This research opens new directions to investigate on the applicability of neural network techniques to data security environments.

**Keywords:** Data Security, Access Control, Proactive Password Checking, Perceptron, Neural Network.

## 1 Introduction

Authentication and identification protocols to check users' accesses to partially shared or private resources are deep problems for computer scientists involved in studies on data security. Several methods have been proposed in the recent years but, among these, password-based access control strategies are still frequently used for their simplicity.

A user, who wants to identify himself to a system, engages an interactive identification protocol: First, the user types in his login name; then, the system asks for the *password*, a secret word which proves the real identity of the user. The system stores in a database a reference string encrypted using the password as a key. So, when the user types in his password, the system re-encrypts the reference string and compares it with the stored one. If they match, the access is allowed; otherwise, it is refused. The scheme is supposed to be secure if the user keeps secret his password.

*Security of Password-based Systems.* Illegal logging onto a machine often happens by means of users' passwords. This is possible not only when the user accidentally discloses his password but even when it is easy-to-guess [8]. Indeed, if an hacker can obtain in some way the file containing the reference string encrypted with the users' passwords, used by the system to allow and refuse the access, then he can try off-line to encrypt the reference string with all the words of a dictionary until a match is found. This attack is called *exhaustive search*. Excellent softwares to accomplish this task have been developed in the recent years, and are available on-line (see, for example, [9]).

Therefore, to increase the security level of a password-based system, we have to find a *method* to reduce the efficacy of the exhaustive search attacks. This goal can be achieved if users are not allowed to choose easy-to-guess passwords.

*Previous works.* The problem of using hard-to-guess passwords has been studied in several papers. So far, four techniques [3,4] have been proposed to eliminate easy-to-guess passwords, and the more promising seems to be the *Proactive Password Checking* approach.

A proactive password checker conceptually is a simple program. It holds a list of easy-to-guess passwords that must be rejected. When the user wants to change his password, it checks for membership in the list. If the password is found, the substitution is refused and a short justification is given; otherwise, the substitution is allowed. The philosophy these programs are based on is that the user has the possibility to select a password but the system enables the change only if it is a "non trivial" one. However, a straightforward implementation of such a program is not suitable since the list can be very long and the time to check membership can be high.

Various proactive password checkers that aim to reduce the *time* and *space* requirements of this trivial approach have been proposed [10,11,5]. All these models are an improvement upon the basic scheme. An interesting approach to design a proactive password checker is the one applied in [1], and subsequently improved in [6], where the problem of password classification is viewed as a *Machine Learning Problem*. The system, in a training phase, using dictionaries of examples of easy and hard-to-guess passwords, *acquires the knowledge* to distinguish between them. This knowledge is represented by a *decision tree* that is used afterwards by the checker to accept or refuse a password change. The experimental results reported in [1,6] show a meaningful enhancement on the error rates with respect to the previous solutions.

In this paper we introduce a new idea to proactive password checking: We use a *perceptron* to distinguish easy-to-guess passwords from hard ones. During the training phase, the perceptron learns the differences between "easy" and "hard" by means of examples. Then, the perceptron is used in the testing phase to decide on the membership of a password to one of the two classes. The technique is promising since the checker shows small error rates, and requires very low memory storage (in our prototype, only 40 bytes!). It is interesting to point out that basically the perceptron codes some rules to classify passwords. Other checker which implements a direct approach (i.e., a code checking with a series of



*if-then-else* statements a bunch of rules) requires non negligible time and memory requirements. We obtain better performances evaluating a simple weighted sum and using only 40 bytes. Hence, this approach is suitable for implementations even on devices with very poor hardware facilities (i.e., smart cards).

*Easy and Hard-to-Guess.* We have informally referred to easy and hard-to-guess passwords. We define *easy-to-guess* as a condition of membership in some “easy to exhaustively search” dictionary, and *hard* by negation of easy. Notice that these notions are *computational in nature*. Hence, a password is easy if it is “guessable” in reasonable time, while it is hard if the guessing requires unavailable resources of time and space. Usually, a hard-to-guess password looks like a random string on the reference alphabet.

## 2 Mathematical Framework

The above notions can be modeled in a mathematical setting. In this section we briefly describe the problem of *choosing* a password  $p$  from a set  $\mathcal{P}$ . More precisely, along the same line of [3], we characterize the selection of *easy-to-guess* passwords from  $\mathcal{P}$  in terms of certain probability distributions.

Let  $\mathcal{P}$  be the set of all admissible passwords, let  $p$  be an element chosen from  $\mathcal{P}$ , and let  $s$  be the function used to select the password  $p$  from  $\mathcal{P}$ . Then, denote by  $p'$  a *guess* for the password  $p$  and assume that it takes a constant amount of time  $T = t(p')$  to determine whether this guess is a correct one.

We can model the *choice* of  $p$  in  $\mathcal{P}$  with a random variable  $\mathbf{S}$ , taking values in  $\mathcal{P}$ . These values are assumed according to a probability distribution  $P_S$  upon elements of  $\mathcal{P}$  that is induced by the selection function  $s$ . Moreover, the *time to guess*  $p$  can be represented with a random variable  $\mathbf{F}_{P_S}$ , which takes values in  $\mathbb{R}^+$  according to  $P_S$ .

If  $\mathbf{S}$  is uniformly distributed on  $\mathcal{P}$ , i.e.,  $P_S = U$ , and *no prior knowledge* of the authentication function (the function used by the operating system to check the equality of a guess with the true password) is available then, as pointed out in [3], to guess the selected password  $p$ , we have to try on average  $\frac{|\mathcal{P}|}{2}$  passwords from  $\mathcal{P}$ , and the expected running time is

$$E(\mathbf{F}_U) = \sum_{i=1}^{\frac{|\mathcal{P}|}{2}} T = T \frac{|\mathcal{P}|}{2}. \quad (1)$$

In this model, there is a correspondence between the set  $S$  of the selection functions and the set  $D_{\mathcal{P}}$ , the set of all probability distributions on the set  $\mathcal{P}$ . Indeed, we can characterize the bad selection functions  $s$  to choose  $p$  in  $\mathcal{P}$ , with those probability distributions  $P_S$  such that

$$E(\mathbf{F}_{P_S}) \leq kE(\mathbf{F}_U). \quad (2)$$

The parameter  $k \in [0, 1]$  defines a lower bound to the acceptability of a given selection function, represented by the distribution  $P_S$ . If  $p$  is chosen according to a probability distribution  $P_S$  that satisfies (2), we say that  $p$  is *easily guessable*.

A family of bad selection functions is represented by *language dictionaries*, where the dictionary can be seen as the image-set of a selection function  $s$ . The words in the dictionary are a small subset of all the strings that can be constructed with the symbols of a given alphabet. According to our model, the distribution induced by natural languages are *skewed* on  $\mathcal{P}$ , since they assign non zero values only to a small subset of elements, therefore  $E(\mathbf{F}_{P_S})$  is much smaller than  $E(\mathbf{F}_U)$ . Hence, it is sufficient to try a number of password smaller than  $\frac{|\mathcal{P}|}{2}$  to guess the chosen  $p$ .

To assure the security of the system against illegal accesses we have to require that the selection function does not localize a small subset of  $\mathcal{P}$ . This means that we have to find a method *to discard* those probability distributions  $P_S$  on  $\mathcal{P}$  such that  $E(\mathbf{F}_{P_S})$  is too much small. If  $E(\mathbf{F}_U)$  is very large and we can force  $P_S$  to look like  $U$ , then the goal is obtained.

A proactive password checker can be viewed as a tool to guarantee that a password  $p$  is chosen from  $\mathcal{P}$  according to a suitable distribution, i.e., a distribution that looks like the uniform one. It works like a *sieve* on the set  $D_{\mathcal{P}}$ . Actually, the proactive checker does not distinguish the different distributions but simply distinguishes among good distributions, close to the uniform one, and bad ones, close to the distributions induced by natural languages.

This is a general analysis of the password choosing problem. In order to derive practical results we need to carefully specify the password space  $\mathcal{P}$ . In our setting this set is the set of all strings of length less than or equal to 8, composed by “printable” ASCII characters. This set is reported in Section 5.

### 3 Pattern Recognition

Pattern recognition concerns with objects categorization [2]. It is a solid area of studies in Artificial Intelligence. The objects of a given universe can belong to different classes, according to their own characteristics. The recognition problem consists in associating each object to a class.

A pattern recognition system can be seen as a two-stage device: A feature extractor and a classifier. It takes as input an object and outputs the classification.

A *feature* is a measurement taken on the input object that has to be classified. The values of the measurements are usually real numbers and are arranged in a vector called *feature vector*. The set of possible feature vectors is called *feature space*. The feature extractor of a pattern recognition system simply takes measurements on the object and passes the feature vector to the classifier. The classifier applies a given criterion to establish in which class the object does belong to.

Discriminant functions are the basis for most of the majority of pattern recognition techniques. A *discriminant function* is a function that maps the feature

vector onto the classification space, and usually defines a boundary among the classes. If the discriminant function is a linear function, i.e., it defines a boundary in the classification space that looks like an hyperplane, the classifier is said linear. Of course, a linear classifier can be used if the classes themselves can be separated by means of a straight line. When this happens, we say that the problem is linearly separable.

As we will see later, the system we are looking for is a pattern recognition system which takes as input words, extracts some features from them, and then outputs a decision on their membership to the easy-to-guess class or the hard one. To classify, our device uses a perceptron which realizes a linear classifier.

*Neural Computing: The Perceptron.* Neural computing is an alternative way to do computation. Against the traditional approach of computer science, a neural machine learns solving problems by trials and errors. In a training phase the machine sees a sequence of examples with the corresponding solutions and adapts its internal parameters to match the correct behaviour. When the training phase stops, the machine is ready to solve new and unseen instances of the problem.

The approach is quite interesting since the machine holds simply the software to manage the training process. The knowledge to solve the specific problem is acquired during the learning phase and is stored in the modified values of the internal parameters. Therefore, the machine is in some sense *self-programmable*.

A formal neuron is a model which tries to capture some aspects of the behaviour of the cerebral neuron. A first model was proposed in 1943 by McCulloch and Pitts. It was a simple unit, thresholding a weighted sum of its input to get an output. Frank Rosenblatt, in 1962, in his book *Principles of Neurodynamics* introduced the name *Perceptron*.

The learning rule of the Perceptron consists of the following steps

- Set the weights and the threshold randomly
- Present an input
- Calculate the actual output by taking the threshold value of the weighted sum of the inputs
- Alter the weights to reinforce correct decisions and discourage incorrect ones

This type of learning is called *hebbian* in honour to Donald Hebb, who proposed in 1949 a similar rule starting from his studies on real neural systems.

It is well known (and not difficult to see) that the Perceptron implements a linear classifier. Indeed, the weighted sum defines an hyperplane in the Cartesian space. If the weighted sum of an input is greater than the threshold, then the pattern belongs to the class on one side of the hyperplane, otherwise it is an element of the class on the other one.

Intuitively, our problem is linear separable, i.e., easy-to-guess passwords and hard ones present characteristics which permit the separation of the two classes by means of a straight line. Under this hypothesis, we have designed the kernel of the proactive password checker. The results obtained seem to confirm this intuition.

## 4 The Checker

The proactive password checker presented in this paper is based on a perceptron. The training process uses two dictionaries, a dictionary of *hard-to-guess* words and a dictionary of *easy* ones. From each word, chosen at random in one of these dictionaries, we extract some features and use the perceptron to classify. The learning rule used to train the perceptron is the *Widrow-Hoff delta rule* [2], a specific implementation of the general learning algorithm described before.

Thus, in order to classify the passwords, we have to identify some features that are relevant for the classification. One of the first features that should be considered is the *length* of the password. Actually, it is commonly believed that, the longer is the password, the harder is to guess. However, the length is not sufficient (and sometimes is a wrong criterium) to correctly classify hard-to-guess passwords and easy ones.

Following the *intuition*, and by *trials and errors*, we have identified four features for the classification called: *Classes*, *#Strong Characters*, *Digrams*, *Upper-Lower Distribution*. More precisely:

- CLASSES: It is reasonable to consider the set of ASCII characters divided into classes of different strength. Commonly, passwords are composed by letters, this means that all the (upper and lower case) letters must have low values. In a second class, we can put the digits '0', ..., '9'. This is because it is not frequent to find a digit in a password, but it is not so unusual, too. In the last class, called the class of *strong* characters, we can put every character that does not belong to the first two classes. To mark the *distance* among these classes we have assigned to the class of letters a value equal to 0.2, to the class of digits a value equal to 0.4 and to the last class 0.6. The *overall value* of a password is computed by summing up the value associated to each character in the password. Notice that, since the feature is a sum, the longer is the passwords the higher is the value.
- #STRONG CHARACTERS: The second feature is the *number* of strong characters contained in the password.
- UPPER-LOWER DISTRIBUTION: The value of this feature is calculated by the following formula:  $|UPP - LOW|/let$ , where *UPP* is the number of upper case letters, *LOW* is the number of lower case letters and *let* is the number of letters in the password. The presence of this feature is due to the observation that passwords that contain both upper and lower case letters are lightly stronger than passwords composed by lower (upper) case letters only.
- DIGRAMS: This feature looks at the *types of digrams* present into the password. More precisely, we say that a digram is an *alternance* if the two characters of the digram belong to different classes. The checker scans the password, analyzes all the digrams from the left to the right, and assigns values to each of them. The more alternances the password has, the higher is the value.

## 5 Description of the Experiments

Many operating systems limit the length of the password. For example, Unix-like operating systems work with passwords of length at most 8 characters. At the same time, many others do not accept passwords with length less than 6 characters. For this reason, we have used, in the training and testing phases of our experiments, dictionaries of hard-to-guess passwords by generating random words with length ranging between 6 and 8 characters. Similarly, passwords contained in the easy-to-guess dictionaries, collected from several sources, have been truncated to the first eight characters.

*The Dictionaries.* We have used eight dictionaries, *strong.0*, *strong.1*, *strong.2*, *weak*, *noise.0.1*, *noise.0.2*, *noise.1.1* and *noise.2.2*, for the training phase, and nine dictionaries *test.strong.0*, *test.strong.1*, *test.strong.2*, *test.weak.0*, *test.weak.1*, *test.noise.0.1*, *test.noise.0.2*, *test.noise.1.1* and *test.noise.1.2*, for the testing phase. We briefly describe each of them. The dictionaries *strong.0*, *strong.1* *strong.2* are dictionaries of *random* words. They are composed by pooling together 30000 words of length 6, 30000 words of length 7 and 30000 words of length 8.

The difference among them is the generation rule used. The *strong.0* dictionary has been generated by randomly choosing the characters of each word in the following set:

```
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z 0 1 2 3 4 5 6 7 8 9
a b c d e f g h i j k l m n o p q r s t u v w x y z : ; < = > ? @ [ ] ^
_ ' ' { | } ! " # $ % & ( ) * + , - . / ~
```

The random selection has been done using the `random()` C-function. This function implements a non-linear pseudorandom generator that is assured to have a long period.

Recall that, we have said before that a character is *strong* if it belongs to the following set:

```
: ; < = > ? @ [ ] ^ _ ' ' { | } ! " # $ % & ( ) * + , - . / ~
```

The dictionaries *strong.1* and *strong.2* have been constructed using the same rule as the one used to construct *strong.0* with additional constraints. More precisely, each word in *strong.1* has *at least one* strong character or *at least two* characters must be digits. Similarly, each word in *strong.2* contains at least *two* strong characters or *three* digits. Intuitively, the “strength” of these dictionaries increases from *strong.0* to *strong.2*.

About the dictionaries of easy-to-guess passwords, the first one, *weak*, is composed by words having length from 1 to 18, recovered from several sources and books, truncated to the first 8 characters. All the passwords in this dictionary are composed by *lower case* letters. The dictionary *noise.0.x*, for  $x = 1, 2$ , is

constructed from the dictionary *weak* by substituting  $x$  randomly selected characters with strong ones. The dictionary *noise.1.x*, for  $x = 1, 2$ , is constructed from the dictionary *noise.0.x* by substituting *half of the lower case letters* with the corresponding upper case ones.

For the testing phase we have used dictionaries presenting the same characteristics of the training dictionaries. This similarity is reflected in the name that is the same up to the prefix *test* (i.e., *test.strong.0*, and so on). The only difference is that we have tested two weak dictionaries, *test.weak.0* and *test.weak.1* where the first one has the same characteristics of *weak*, while *test.weak.1* is constructed from *test.weak.0* by substituting half of the lower case letters with the corresponding upper case ones.

*The Intuition Behind the Experiments.* The training phase is a critical step: Based on the easy and hard-to-guess examples presented, the perceptron learns different notions of easy and hard. Hence, if the training set is not accurately chosen, the perceptron can give poor performances. The first experiment we have run is to train the perceptron with a “very easy” dictionary of easy examples, and a “very hard” dictionary of hard ones. Unfortunately, the results obtained in testing phase are not exciting applying this strategy. The main problem with this approach is that there is a big “distance” between the dictionaries. So, many noisy passwords are classified as hard-to-guess.

To avoid this phenomenon, we have used dictionaries whose distance is small enough to obtain a more refined notion of “easy” and “hard” but, at the same time, big enough to ensure that the perceptron learns these distinct notions.

*The Experiments.* We have trained the perceptron using all the possible combinations between the strong dictionaries and the weak ones described in Section 5, *fixing* the number of examples in each training.

Since this number is smaller than the number of words in the dictionaries, the training has been *randomized*. More precisely, the training procedure, in each step, randomly chooses either the hard or the easy dictionary, and then randomly selects a word in the selected dictionary. After each training, we have tested all the testing dictionaries. Since the training is randomized, we have repeated the overall process 100 times, to ensure that the results obtained were consistent and not due to a “lucky” random sequence of examples. In Table 1 of the Appendix we report the average number of modifications of the perceptron’s parameters (i.e., weights) occurred during the training phase before the convergence to a stable configuration.

Results reported in Table 1 clearly states that if the distance between the dictionaries used during the training phase is large, the process converges almost immediately. On the other hand, if the distance between these dictionaries is really small, the same process takes a while (and, maybe, does not converge).

For space limits, we report the results only for the experiments associated to the training executed using *strong.1* and *strong.2* as strong dictionaries. In Table 2 in the Appendix we report the expected error and the variance obtained on each testing dictionary.

**Table 1.** Average Number of Weight Modification over 20,000 Examples

Strong Dictionaries					
strong.0		strong.1		strong.2	
weak	102	weak	7	weak	2
noise.0.1	699	noise.0.1	385	noise.0.1	10
noise.0.2	2113	noise.0.2	1653	noise.0.2	1286
noise.1.1	3882	noise.1.1	3637	noise.1.1	10
noise.2.2	7495	noise.2.2	6980	noise.1.2	7150

**Table 2.** Testing Behaviour

	strong.1		strong.1		strong.2		strong.2	
	weak		noise.0.1		weak		noise.0.1	
Testing Dict.	Error (%)	Var. (%)	Error (%)	Var. (%)	Error (%)	Var. (%)	Error (%)	Var. (%)
test.weak.0	0.02	0.01	0.00	0.00	0.01	0.01	0.00	0.00
test.weak.1	5.36	18.09	0.02	0.00	0.01	0.01	0.00	0.01
test.noise.0.1	95.62	0.77	0.33	2.19	53.81	41.88	0.02	0.00
test.noise.0.2	99.49	0.13	79.99	0.50	91.25	8.69	79.92	0.00
test.noise.1.1	96.32	1.09	96.00	0.00	45.28	40.80	27.59	38.74
test.noise.1.2	99.59	0.12	99.56	0.00	89.70	8.51	85.91	8.13
test.strong.0	4.91	0.76	7.68	0.40	16.85	7.84	20.94	6.84
test.strong.1	0.00	0.01	1.70	0.30	11.72	8.42	16.07	7.38
test.strong.2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

It is immediate to see that if the pair of dictionaries approaches to the space of strong passwords, i.e., moving from *(strong.1, weak)*, to *(strong.2, noise.0.1)*, the error obtained on easy testing dictionaries decreases since the perceptron learns a stronger notion of “easy”. At the same time, the error obtained on the dictionaries of hard-to-guess words increases since the perceptron also learns a stronger notion of “hard”.

*Platform.* The experiments presented have been run on a 450 Mhz Pentium III machine running a Linux kernel 2.2.10 with 256 MBytes of RAM, and a 9 GBytes SCSI hard disk. On this machine, the software has checked more than 56,000 passwords/sec. Our tests show that this approach is significantly faster than all the previous proposed ones.

## 6 Conclusions and Open Problems

This paper gives more questions than answers. Following an intuition, and proceeding during the experimental phase by trials and errors, we have shown that

the hard-to-guess password set (i.e., words sampled according to a uniform-like distribution on the set of printable ASCII characters) seems to be linearly separable from the easy-to-guess set (i.e., set of “structured words” chosen, for example, according to the distribution of a natural language).

Several problems arise from this study. Strictly related to the topic of this paper, the following issues could be of interest: From an experimental point of view, it would be nice to investigate other features for the perceptron, in order to obtain better performances during the classification task. The features we have used seem to be reasonable but a more accurate choice can produce a more refined classification. Along the same line, other strategies (dictionaries) for the training phase can give an improvement as well. On the other hand, from a theoretical point of view, it would be nice to prove in a formal model that easy-to-guess and hard-to-guess passwords are really linearly separable. This intuition, at the basis of the present work, seems to be corroborate by the experimental results.

Finally, we put forward the question of the applicability of neural networks to data security problems. Further investigation on the relation among these two fields could be done by researchers belonging to both fields.

## References

1. F. Bergadano, B. Crispo, and G. Ruffo, *High Dictionary Compression for Proactive Password Checking*, ACM Transactions on Information and System Security, Vol. 1, No. 1, pp. 3-25, November 1998.
2. R. Beale and T. Jackson, **Neural Computing: An Introduction**, IOP Publishing Ltd, Institute of Physics, 1990.
3. M. Bishop, *Proactive Password Checking*, in Proceedings of 4th Workshop on Computer Security Incident Handling, 1992.
4. M. Bishop, *Improving System Security via Proactive Password Checking*, Computers and Security, Vol. 14, No. 3, pp. 233-249, 1995.
5. B. Bloom, *Space/Time Trade-offs in Hash Coding with Allowable Errors*, Communications of ACM, July 1970.
6. C. Blundo, P. D'Arco, A. De Santis, and C. Galdi, *Hyppocrates: A new Proactive Password Checker*, Proceedings of ISC01, Springer-Verlag, LNCS, Vol. 2200, Malaga, October 1-3, 2001.
7. C. Davies, and R. Ganesan, *Bapasswd: A new proactive password checker*. In Proceedings of the 16th National Conference on Computer Security (Baltimore, MD, Sept. 20-23).
8. D. Klein, *Foiling the Cracker: A Survey of, and Improvements to, Password Security*. Proceedings of the Fifth Data Communications Symposium, September 1977.
9. A. Muffett, *Crack 5.0*, USENET News.
10. J. B. Nagle, *An obvious password detector*. USENET News.
11. E. Spafford, *OPUS: Preventing Weak Password Choices* in Computers and Security, No. 3, 1992.



# Single Sign-On Architectures

Jan De Clercq

Security Consultant, HP

Authentication infrastructures have been around for many years now. They are very popular in big computing environments where scalability is a key requirement. In such environment, it's not very cost-efficient from both an implementation and an administration point-of-view to create a separate authentication system for every individual computer system, resource or application server. It is much better to outsource this functionality to an authentication "infrastructure".

The outsourcing of authentication to a specialized infrastructure also enables the enforcement of a consistent authentication policy throughout the enterprise. Another major driver behind the creation of authentication infrastructures is single sign-on (SSO). In short, SSO is the ability for a user to authenticate once to a single authentication authority and then access other protected resources without re-authenticating. The Open Group defines SSO as the mechanism whereby a single action of user authentication and authorization can permit a user to access all computers and systems where that user has access permission, without the need to enter multiple passwords.

This paper focuses on the architectural approaches one can take when designing an SSO solution for a large I.T. infrastructure and on the security technology building blocks that can be used to construct such an SSO infrastructure. This brief does not address the architecture of every SSO solution that is currently available on the software market. Many of them have a relatively small scope and only span a couple of applications, platforms or authentication methods.

## 1 Authentication Infrastructure Terminology

Before continuing let's make sure we all agree on the basic authentication infrastructure terminology.

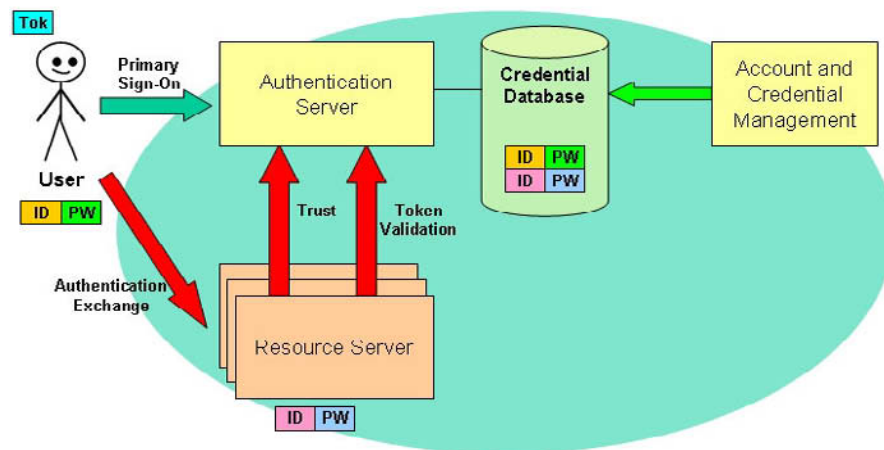
In an authentication infrastructure users trust a set of authentication authorities to provide trustworthy authentication services. These authorities are also referred to as authentication Trusted Third Parties (TTPs). Every authentication authority "reigns" over a set of resources located on machines that are part of the authentication authority's kingdom. I will call this "kingdom" from now on a "domain" (in NT

terminology) – but you may as well call it a “realm” (in Kerberos terminology), or a “cell” (in DCE terminology) - it doesn’t really matter... Anyhow, when a user logs on successfully to the authentication authority’s domain he can transparently access all resources in the domain, without re-authenticating to every individual resource server.

Note the different authentication infrastructure-related terms:

- An **authentication infrastructure** refers to a set of authentication servers and authentication authorities, providing outsourced authentication services.
- **Authentication servers** are the physical machines performing the authentication functions.
- **Authentication authorities** are a logical trust-related concept: they are the entities that are trusted by users to perform reliable authentication functions.

In order to authenticate a user shares a set of “credentials” with the authentication authority. The authentication authority stores its copy of the credentials in a secured database. For the moment, very popular types of credential databases are LDAP accessible Directories. Depending on the type of credentials the user can simply remember them or “safeguard” them in some other way (store them for example on a smart card).



**Fig. 1.** SSO in an Environment with a single Authentication Authority and a Single Authentication Server

During a typical authentication process (like the one illustrated in Figure 1) a user submits his credentials (in the example above: a User-ID and a password) or the result of a cryptographic operation involving his credentials to the authentication authority. The authentication authority then validates the credentials using the data stored in its credential database. If the credentials supplied by the user and the ones stored in the database match or if the result of a cryptographic operation on the credentials stored in the database equals the information supplied by the user, the user's identity is considered "authentic". Consequently the user is given or denied access to the authentication authority's kingdom. To prove that the user has been authenticated the authentication authority will issue a cryptographic token to the user. This token will be used as a proof of authentication in subsequent accesses to other resources in the authentication authority's kingdom.

Remember that SSO is "authentication"-related, not "access control"-related. Too many people confuse authentication and access control. "Authentication" is a security process that assures that a user's identity is authentic, or in other words that another user, application or service knows who it is talking to. "Access control" is a security process that decides what a particular user is allowed or not allowed to do with a resource.

## 2 SSO: Pros and Cons

A study conducted by the Network Applications Consortium (<http://www.netapps.org>) in large enterprises showed that users spend an average of up to 44 hours per year to perform logon tasks to access a set of 4 applications. The same study measured the content of the calls to companies' helpdesk: 70 per cent of the calls were password reset requests.

SSO is advantageous for both users and administrators. There's no need to point out that a user's and an administrator's life becomes much easier if they have to deal only with a single set of credentials – one for every user. An average user will have to provide his logon credentials only once every day and he will need to change only a single set of credentials at regular intervals. Indirectly this will increase a user's productivity. The authentication infrastructure, its administrators and helpdesk operators will only need to keep track of the changes to a single entry for every user in the credential database. A key advantage is also that all authentication data are centralized and can be accessed and manipulated using the same tools and procedures. The latter may also be a weakness: if a malicious person gets to the database and can bypass its security system, he gets access to all of the data at once.

The advantages of SSO are not only related to the ease of administration and use, it also brings important security advantages. Centralization eases the enforcement of a consistent authentication policy throughout the enterprise. And obviously, it's also much easier to secure a centralized than a distributed infrastructure. The lack of SSO services increases the risk for compromise of an authentication service's security. For example, because users need to keep track of different password credentials, they may

start writing them down on Post It notes and stick them to the back of their keyboards. Indirectly the absence of SSO can also affect the availability of an authentication service. The more passwords users have to remember or keep track of, the bigger the chances get they forget or lose them.

A good SSO solution is also platform- and/or application neutral: it can hide the authentication implementation details on different operating system platforms from the SSO user and can provide support to “outsource” the application-level authentication logic to a centralized SSO authentication authority.

An often heard argument against SSO is that SSO credentials are the “key to the kingdom”. If one can obtain the SSO credentials he gets access to all resources secured by them. This risk may be reduced when choosing SSO credentials that are not knowledge-based (a classical example of knowledge-based credentials are passwords) but rather biometric- (for example using fingerprints) or possession-based (for example using cryptographic tokens or smart cards). The use of multi-factor authentication solutions for SSO will further reduce this risk.

### 3 Simple SSO Architectures

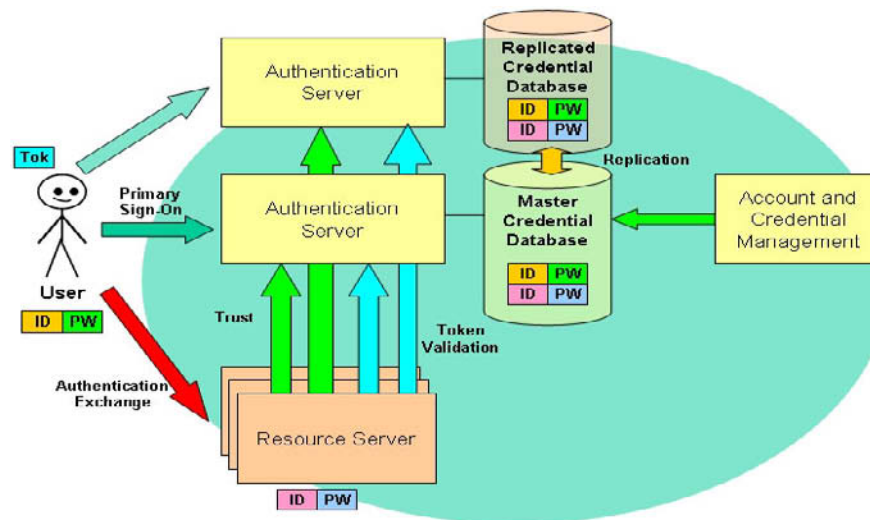
SSO is relatively easy to implement in authentication infrastructures that are using a single authentication authority. Such environment is illustrated in Figure 1. In this environment users have a single set of credentials. Remember that the concept of “users” must be interpreted in a large sense: it covers all security principals that are accessing the resources under the control of the authentication authority. Figure 1 shows a user and a resource server security principal trusting the same authentication authority. Linked to the authentication server there’s a credential database which is the primary source for account and credential management.

Over the past years operating system vendors like Novell and Microsoft have proven that SSO can easily be implemented in homogeneous LAN and intranet environments where all machines are running the same operating system and trusting the same authentication authority. Extranet Access Management System (EAMS) software vendors like Netegrity, RSA (Securant), Baltimore, Entrust and many others... have proven the same thing for homogenous web portal environments. Finally, remote access authentication-infrastructures using RADIUS, TACACS or TACACS+ showed that setting up SSO is relatively straightforward in environments using a centralized authority that communicates with a set of authentication proxies using a single well-defined authentication protocol. A non-exhaustive list of software products supporting simple SSO is given in table 1.

**Table 1.** Simple SSO Solutions (non-exhaustive list)

Simple SSO Solutions	
SSO solutions bundled with Operating System Software	
Microsoft Windows NT, Windows 2000	<a href="http://www.microsoft.com">http://www.microsoft.com</a>
Novell Netware	<a href="http://www.novell.com">http://www.novell.com</a>
SSO bundled with Extranet Access Management System (EAMS) Software	
Netegrity SiteMinder	<a href="http://www.netegrity.com">http://www.netegrity.com</a>
RSA Securant ClearTrust	<a href="http://www.rsa.com">http://www.rsa.com</a>
Obliv Netpoint	<a href="http://www.oblix.com">http://www.oblix.com</a>
IBM Tivoli Secureway Policy Director	<a href="http://www.ibm.com">http://www.ibm.com</a>
SSO using Centralized Remote Access Security Software	
Cisco (TACACS, TACACS+ solutions)	<a href="http://www.cisco.com">http://www.cisco.com</a>
Microsoft (RADIUS solutions)	<a href="http://www.microsoft.com">http://www.microsoft.com</a>

Things get much more complex if the SSO scope is extended to cover different platforms and different organizations, which are using different authentication credentials and protocols and are governed by many different authorities. Usually this also means that the infrastructure has to deal with multiple credentials per user.

**Fig. 2.** SSO in an Environment with a single Authentication Authority and Multiple Authentication Servers

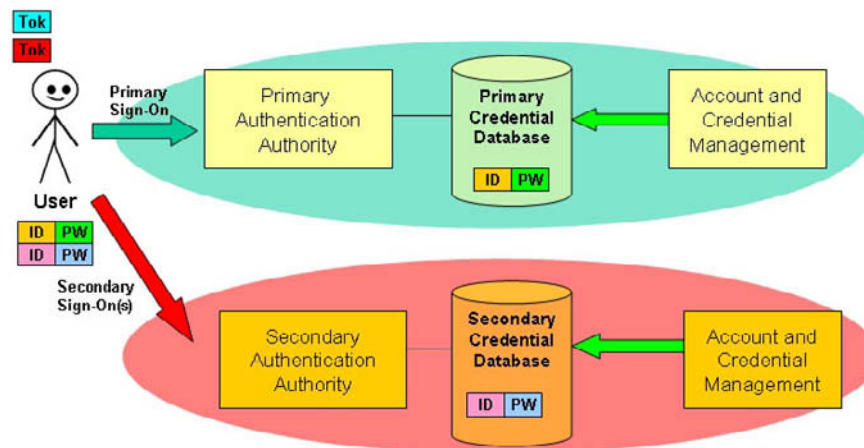
Having a single authentication authority doesn't necessarily mean that only one authentication server and a single credential database are available. For scalability and performance reasons a single authentication authority may consist of multiple authentication servers and a set of replicated credentials databases. Figure 2 illustrates SSO in an environment with a single Authentication Authority and multiple

Authentication Servers. Note that the credential database is replicated to all authentication servers. To avoid ambiguous user authentication the replication of security credentials requires a single-master replication model. For the same reason, account and credential management can only be done on the data stored in the master credential database.

## 4 Complex SSO Architectures

A big challenge in today's authentication infrastructures is to extend the SSO scope to cover many "different" authentication authorities. "Different" in this context means: implemented on different platforms and governed by different organizations. In most scenarios these infrastructures also have to deal with multiple credentials per user and many different authentication protocols.

To ease the explanation of the different SSO architectures used in complex SSO setups, let's first look at how authentication works in an environment with multiple authentication authorities but without SSO support (as illustrated in Figure 3).



**Fig. 3.** Authentication in an Environment with Multiple Authentication Authorities.

In the setup illustrated in Figure 3 the domain the user uses most often is called the user's primary authentication domain. Domains that users use less often are called secondary authentication domains. Since in the example, no SSO is available between the primary and the secondary authentication domains, when a user wishes to access resources in the secondary domains, he has to authenticate to the TTPs of those domains using his credentials as defined in that particular secondary authentication domain. Every secondary authentication domain also has its proper credential database. No need to explain that this setup creates an enormous credential-safeguarding burden for the end users. Note that in this setup the user uses different authentication tokens, one per authentication authority.

## 5 SSO Architectures Dealing with a Single Set of Credentials

The simplest complex SSO architectures are the ones using a single set of credentials. This single set of credentials is recognized by many different authentication authorities. There are two important flavors of complex SSO architectures dealing with a single set of credentials: Token-based and Public Key Infrastructure-based SSO systems.

Both SSO architectures provide SSO in a rather homogeneous environment. Homogeneous in this context means: using a single account naming format and authentication protocol that are supported by every entity, application and service participating in the SSO environment.

### 5.1 Token-Based SSO Systems

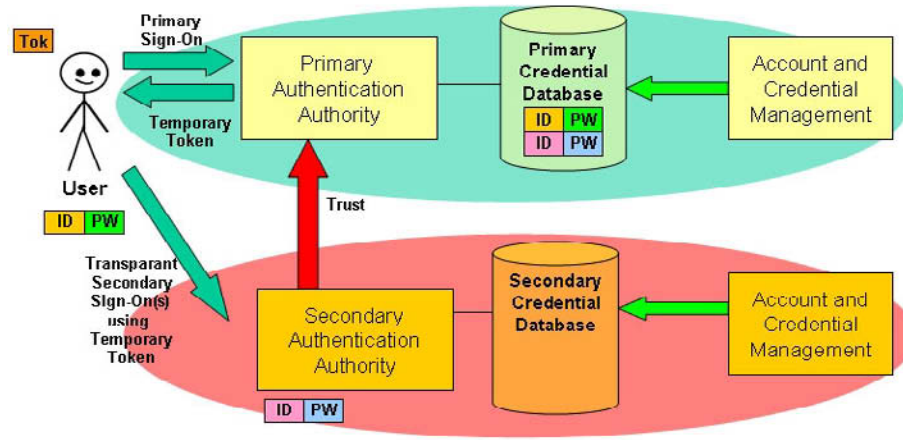
A classical example of an authentication protocol used for token-based SSO is the Kerberos authentication protocol. Kerberos is an open standard defined by the IETF that has been implemented on many different platforms.

In a token-based SSO architecture users get a temporary software token when they have been successfully authenticated to the TTP (as illustrated in Figure 4). This token can be cached on the user's machine and can be reused to prove the user's identity to other secondary authentication domain TTPs. To validate the user token the TTPs use cryptographic methods that are based on secret keys that are set up between the secondary authentication domain TTPs and the primary authentication domain TTP. This cryptographic key material represents a trust relationship between primary and secondary authentication domains.

Contrary to the tokens we will discuss in other SSO architectures, the tokens used in token-based SSO systems are valid for more than a single authentication authority. Agreed that in some token-based setups users do have more than a single token, but in that case the authentication protocols support the automatic and transparent exchange of the token of one authentication authority for a token issued by another authentication authority.

In the Kerberos case users authenticate to a central authentication service, called the Kerberos Key Distribution Center (KDC) (the authentication TTP). If their authentication credentials are valid they receive a ticket (the software token). This ticket enables the user to request other tickets from the KDC in order to access other resources in the primary and secondary authentication domains. The tickets prove to the resource servers that the user has been authenticated before by a trusted authentication service – the Kerberos KDC.

The Kerberos authentication protocol has inspired the developers of the authentication services for the Open Software Foundation's (OSF) Distributed Computing Environment (DCE). Microsoft has implemented Kerberos as the default



**Fig. 4.** Authentication in a Token-based SSO Environment

authentication protocol of Windows 2000. CyberSafe sells plug-ins that can be used to enable an operating system platform to generate, understand and validate Kerberos credentials (this process is also known as “Kerberizing”).

The Kerberos token-based SSO typically uses Remote Procedure Calls (RPCs) to transport authentication tickets. In HTTP-based environments token-based SSO can be provided using HTTP cookies. The latter mechanism is used by many Extranet Access Management Systems (EAMS) like Netegrity’s SiteMinder or Oblix’s Netpoint when dealing with multiple authentication authorities. Microsoft currently uses a similar cookie-based token system to extend the SSO functionality of its Passport web authentication solution across different web sites. In the future Microsoft wants to use a combination of the Kerberos protocol and a cookie-based token system to enable Passport SSO to span many different authentication authorities.

Token-based SSO software comes out-of-the-box with many of today’s most popular operating system platforms (Windows 2000, Netware...). Table 2 below gives some examples of software products providing token-based SSO support.

**Table 2.** Token-based SSO Solutions (non-exhaustive list)

Token-based SSO Solutions	
Kerberos-based	
Microsoft Windows 2000	<a href="http://www.microsoft.com">http://www.microsoft.com</a>
Cybersafe ActiveTrust	<a href="http://www.cybersafe.com">http://www.cybersafe.com</a>
Cookie-based	
Netegrity SiteMinder	<a href="http://www.netegrity.com">http://www.netegrity.com</a>
Oblix Netpoint	<a href="http://www.oblix.com">http://www.oblix.com</a>
RSA Securant ClearTrust	<a href="http://www.rsa.com">http://www.rsa.com</a>



## 5.2 Public Key Infrastructure-Based SSO

In a Public Key Infrastructure-based (PKI-based) SSO architecture (illustrated in Figure 5) users first register themselves at a trusted authentication authority (in this case called a certification authority (CA)) or at one of the authentication authority's registration agents (called registration authorities (RAs)). During this registration process different things occur: users identify themselves using a set of credentials; a piece of client-side software generates an asymmetric key pair; and the public key of this key pair is offered to the CA (or RA) for certification. Upon receipt of the user's credentials and the public key, the CA (or RA) will verify the user's credentials. If the credentials are valid it will generate a public key certificate and send it back to the user. The user's public key certificate and the user's private key are cached on the user's machine (or on a smart card or cryptographic token). They both are used to generate a kind of software tokens similar to the ones used in token-based SSO systems. These tokens are used to prove the user's identity to other secondary authentication authorities in subsequent authentication requests.

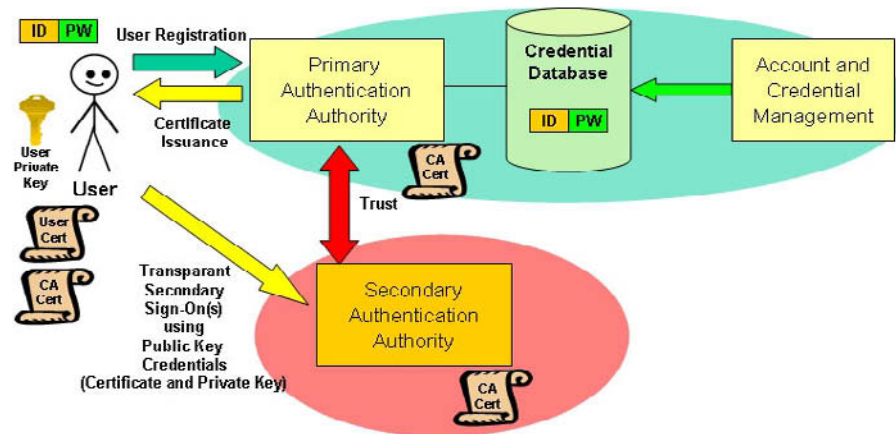


Fig. 5. Authentication in a PKI-based SSO Environment

A major difference between a token-based and a PKI-based SSO architecture is that in the PKI case the cryptographic methods that are used to validate the user token, are asymmetric cryptography-based (using public and private keys). The outcome of this validation process also largely depends on the trust relationship that is set up between the secondary authentication authorities and the primary authentication authority. In the case of a PKI-based SSO architecture the trust relationship between primary and secondary authentication authorities is represented by a secondary authentication authority's certificate (issued by the primary authentication authority). Similar to the tokens discussed in token-based SSO architectures, the tokens used in PKI-based SSO systems are valid for more than a single authentication authority.

Contrary to token-based SSO systems, PKI-based SSO systems are a relatively new technology. Early implementers of the technology experienced lots of interoperability

problems. The latter were mainly related to immature PKI standards. Over the last two years the security software industry and standardization organizations like the IETF have made important efforts to make the PKI-based SSO systems enterprise-ready. Another early adopter problem was that few applications were PKI-enabled. Although the latter problem hasn't been fully resolved, most application software vendors have modified their application to let them understand PKI-based credentials. Table 3 below gives some popular examples of PKI software solutions.

**Table 3.** PKI-based SSO Solutions (non-exhaustive list)

<b>PKI-based SSO Solutions</b>	
<b>Inhouse PKI products</b>	
Baltimore Unicert PKI	<a href="http://www.baltimore.com">http://www.baltimore.com</a>
Entrust Authority PKI	<a href="http://www.entrust.com">http://www.entrust.com</a>
Smarttrust PKI	<a href="http://www.smarttrust.com">http://www.smarttrust.com</a>
RSA Keon PKI	<a href="http://www.rsa.com">http://www.rsa.com</a>
Microsoft PKI (Windows 2000, Windows.NET)	<a href="http://www.microsoft.com">http://www.microsoft.com</a>
<b>Commercial PKI products</b>	
Verisign	<a href="http://www.verisign.com">http://www.verisign.com</a>
Globalsign	<a href="http://www.globalsign.com">http://www.globalsign.com</a>

## 6 SSO Architectures Dealing with Many Different Credentials

There are three different flavors of SSO architectures that can deal with many different credentials: architectures that use credential synchronization, architectures using a secure client-side cache and architectures using a secure server-side cache.

Contrary to token-based SSO, these three SSO architectures can provide single sign-on in a more heterogeneous environment. Besides different credential types, they can also support different account formats and multiple authentication protocols.

### 6.1 Credential Synchronization

Figure 6 shows an SSO architecture that's using a credential synchronization system. A classic example is a system that synchronizes user passwords between the credential databases of different authentication authorities. Although this architecture supports multiple credentials for every user, they are kept identical using the credential synchronization mechanism. Credential synchronization systems typically use a single master credential database. This database can be used by administrators to update the user credentials.

Because in this setup, the user is still prompted to enter his credentials by every single authentication authority, these systems are not considered true SSO systems. Many security experts consider it a very dangerous practice to synchronize credentials between the databases of different authentication authorities. Their objections are based on the "key to the kingdom" argument mentioned above. Table 4 below gives some examples of credential synchronization software products.

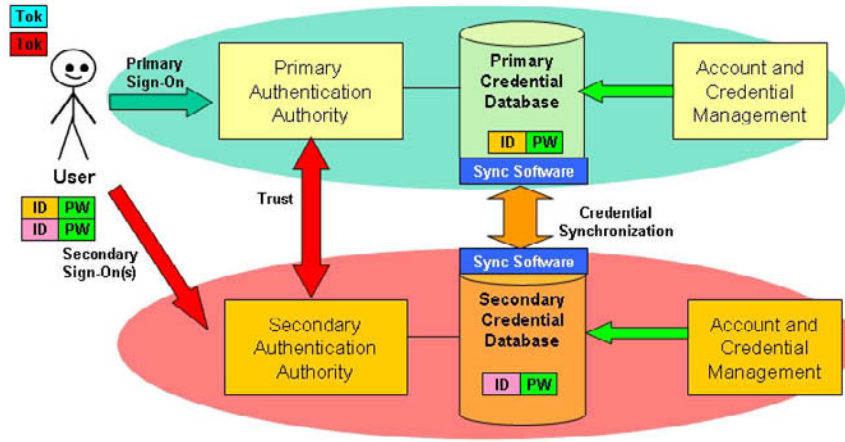


Fig. 6. Password Synchronization-based SSO

Table 4. Credential Synchronization-based SSO Products (non-exhaustive list)

Credential Synchronization-based SSO Products	
Passgo InSync	<a href="http://www.passgo.com">http://www.passgo.com</a>
Proginet SecurPass-Sync	<a href="http://www.proginet.com">http://www.proginet.com</a>
P-Synch	<a href="http://www.psynch.com">http://www.psynch.com</a>
M-Tech ID-Synch	<a href="http://www.idsynch.com">http://www.idsynch.com</a>

## 6.2 Secure Client-Side Credential Caching

Figure 7 illustrates an SSO architecture that's using a secure client-side credential caching mechanism. In this setup a set of "primary credentials" are used to unlock a user's credential cache. Later on, when the user wants to access resources requiring different authentication credentials, the other credentials are automatically retrieved from the local credential cache and presented to the authentication authority. If the credentials are valid the user will be logged on transparently to the other resource servers. Because in this setup authentication to the secondary authentication domains relies on the credentials for the primary domain to unlock access to the credentials of secondary domains, the secondary domains must trust the primary domain.

In the early days of secure client-side credential caching, it was combined with client-side scripting to automate the SSO process. No need to explain that this created a lot of administrative overhead. Nowadays credential caching is combined with more intelligent and adaptive client-side software that can automatically retrieve and provide the correct credentials to the destination server. Table 5 below gives some examples of secure client-side cache-based SSO software products.

In the context of this SSO architecture “secure storage” of the cached credentials is absolutely vital. This is certainly the case if the cached credentials are used to give access to business-critical applications or data. In the latter case it is not recommended to use this SSO architecture on portable client devices (like laptops or PDAs) or on operating system platforms that have a bad security reputation.

SSO based on a secure client-side cache can be implemented without the use of an authentication “infrastructure”. In this case the primary credentials unlocking the cache would be local credentials – “local” meaning: defined in the local machine’s security database and only valid for accessing local machine resources. In the context of an authentication infrastructure the user’s primary credentials are generally not local credentials but domain credentials.

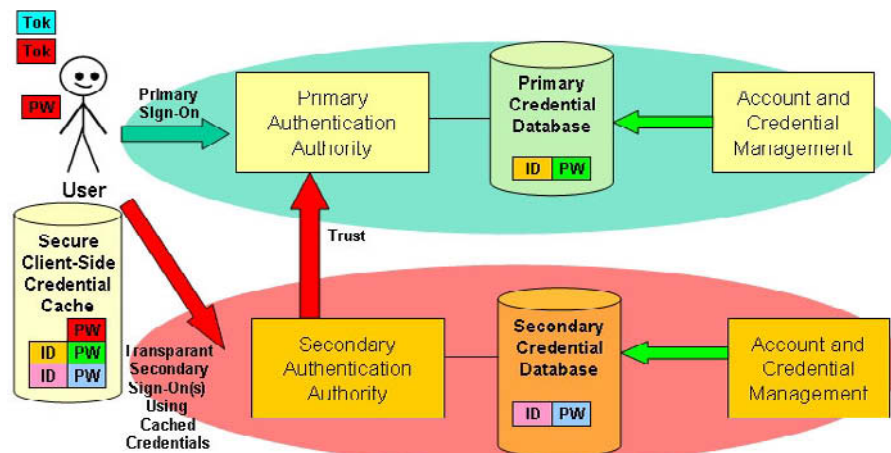


Fig. 7. Authentication in an SSO Environment using a Client-side Secure Cache

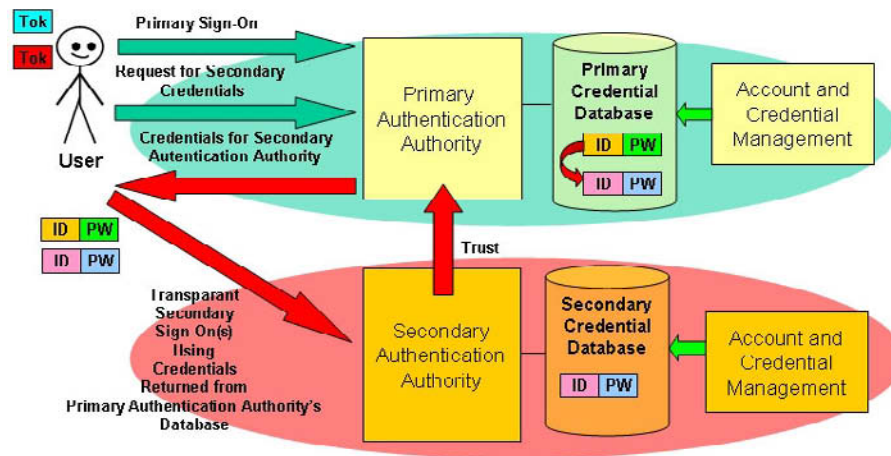
Table 5. Secure Client-Side Cache SSO Products (non-exhaustive list)

Secure Client-Side Cache SSO Products	
Bundled with OS Software	
Microsoft Windows XP, Windows.NET	<a href="http://www.microsoft.com">http://www.microsoft.com</a>
Bundled with Other Software Products	
Entrust Entelligence (PKI client)	<a href="http://www.entrust.com">http://www.entrust.com</a>
Identix BioLogon (Client-side biometrics software)	<a href="http://www.identix.com">http://www.identix.com</a>

### 6.3 Secure Server-Side Credential Caching

Figure 8 illustrates a secure server-side credential caching SSO architecture. Contrary to the use of a secure client-side cache, secure server-side credential caching SSO architectures store credentials in a central repository on the server-side. Contrary to a credential synchronization-based SSO architecture, the credentials used in a secure

server-side credential caching SSO architecture are not necessarily the same for every authentication authority.



**Fig. 8.** Authentication in a Secure Server-side Credential Caching SSO Environment

In a secure server-side credential caching SSO architecture the master credential database contains (besides the user's primary credentials) the mappings between a user's primary and secondary credentials. That's why the primary authentication authority in this architecture is sometimes referred to as the "authentication gateway". Another copy of the secondary credentials is kept in the secondary authentication domain databases.

In a secure server-side credential caching SSO setup a user always first logs on to the primary authentication authority using his primary credentials and a predefined authentication protocol. If logon to the primary authentication authority is successful, the user-side SSO software usually provides the user with a list of available applications. When accessing an application that requires a logon to a secondary authentication authority the user-side SSO software will first communicate with the primary authentication authority to retrieve the appropriate user credentials. These are then forwarded to the user in a secure way. Finally, the user-side SSO software will use the secondary domain credentials to transparently log the user on to the secondary domains.

Since in this architecture the secondary authentication authorities trust the primary authentication authority to store a copy of their credentials in the primary credential database, a trust relationship is required between every secondary and the primary authentication authority.

Server-side caching generally provides better security because the credentials are not stored on the client's hard disk (also remember the security concerns brought up for

client-side caching). The credentials may be temporarily downloaded to the client, but will disappear when the client shuts down. Also access to the server-side credential cache is only allowed after a successful logon to the central authentication authority.

An important challenge in these architectures is how to keep the copies of the credentials in the primary credential database and the ones in the secondary credential databases synchronized. Different products use different approaches to resolve this problem:

- Some products contain password synchronization services.
- Some products rely on the password synchronization services of specific software products (Passgo, Proginet...) or the password synchronization services that are built into systems management softwares (BMC Patrol, CA Unicenter, Tivoli).
- Some products do not use any password synchronization and rely on administrators or users (self-administration) to perform the credential updates.

Examples of secure server-side credential caching SSO systems are IBM's Tivoli Secureway Global Sign-On, Computer Associates eTrust and Vasco's SnareWorks Secure SSO (as listed in table 6). SecureWay Global Sign-On's primary authentication method for example is a DCE token-based. After the initial authentication, the SecureWay Client-side Global Sign-On software retrieves the correct credentials from the primary credential database and provides them transparently to the correct secondary authentication authority.

**Table 6.** Secure Server-side Credential Caching SSO (non-exhaustive list)

Secure Server-side Credential Caching SSO	
IBM Tivoli Secureway Global Sign-On	<a href="http://www.ibm.com">http://www.ibm.com</a>
Computer Associates eTrust	<a href="http://www.ca.com">http://www.ca.com</a>
Vasco SnareWorks Secure SSO	<a href="http://www.vasco.com">http://www.vasco.com</a>

## 7 SSO Architectures: Summary

So far we discussed simple SSO solutions and SSO architectures that can span multiple authentication authorities. Table 7 gives an overview of the advantages and disadvantages of the SSO architectures discussed so far. A big challenge for today's SSO products is to extend their scope to cover authentication authorities that are governed by different institutions and companies and to integrate as an authentication provider with a wide range of applications. This will be discussed in the following sections.

**Table 7.** Advantages and Disadvantages of different SSO Architectures

	Pros	Cons
Token-based	<ul style="list-style-type: none"> <li>• Single set of credentials simplifies life of user and administrator</li> <li>• Software usually comes bundled with OS software</li> </ul>	<ul style="list-style-type: none"> <li>• Requires a homogeneous authentication infrastructure environment</li> <li>• Relies on symmetric cryptography</li> </ul>
PKI-based	<ul style="list-style-type: none"> <li>• Single set of credentials simplifies life of user and administrator</li> <li>• Software usually comes bundled with state-of-the-art OS software</li> <li>• Relies on asymmetric cryptography</li> </ul>	<ul style="list-style-type: none"> <li>• Can only deal with a single set of credentials</li> <li>• Complex certificate validation logic. Requires a lot of processing on the client side</li> <li>• Requires a homogeneous authentication infrastructure environment (all services and applications must be PKI-enabled)</li> </ul>
Credential Synchronization	<ul style="list-style-type: none"> <li>• Can deal with many different credentials</li> <li>• Does not require a homogeneous authentication infrastructure environment</li> <li>• Does not impact the client-side (no extra software needed)</li> </ul>	<ul style="list-style-type: none"> <li>• Credentials are kept identical on different platforms</li> <li>• Does not provide true SSO (unless it is combined with a secure client-side caching mechanism)</li> <li>• “Key to the kingdom” argument</li> <li>• Multiple sets of credentials complicate life of user and administrator</li> <li>• Requires extra software on server infrastructure-side</li> </ul>
Secure Client-side Credential Caching	<ul style="list-style-type: none"> <li>• Can deal with many different credentials</li> <li>• Does not require a homogeneous authentication infrastructure environment</li> </ul>	<ul style="list-style-type: none"> <li>• Requires a “secure” client-side credential cache – it is not recommended to use it from portable client devices or OSs with a bad security reputation</li> <li>• Multiple sets of credentials complicate life of user and administrator</li> <li>• Has important impact on client-side (requires extra software or state-of-the-art OS)</li> </ul>
Secure Server-side Credential Caching	<ul style="list-style-type: none"> <li>• Can deal with many different credentials</li> <li>• Does not require a homogeneous authentication infrastructure environment</li> </ul>	<ul style="list-style-type: none"> <li>• Requires a credential synchronization mechanism (may be part of the SSO product)</li> <li>• Multiple sets of credentials complicate life of user and administrator</li> <li>• Requires extra software on server infrastructure-side</li> <li>• Has impact on client-side (requires extra software)</li> </ul>

## 8 Extending the SSO Scope to Cover Different Organizations

One of the driving factors behind extending SSO to cover different organizations are companies’ e-business requirements to easily authenticate users defined in other

organizations and transaction requests initiated by business partners. The demand for SSO scope extension is very high in the world of Internet portals. SSO clearly benefits the Internet experience of any portal user from an ease-of-use point-of-view.

There are two approaches when dealing with SSOs spanning different organizations. In the first approach an organization deals with the credential information and authentication of users in other organizations locally by using its proper credential database and authentication authority. In the second approach organizations agree to set up a federation agreement between their authentication authorities. In the latter case users will always authenticate to their proper organization's authentication authority.

When dealing with authentication locally an organization can define a set of local credentials or decide to synchronize credential databases. In the first case an organization simply provides every external entity with a set of new credentials that is kept in the organization's authentication database. Some organizations may even agree to delegate the administration of these credentials to an external administrator. A major disadvantage of this setup is that a user will end up with multiple credentials. In other words: there's no more SSO. In the second case an organization decides to synchronize account information between an external organization's directory and its proper directory. Even though in this scenario users may end up with a set of credentials that is identical between different organizations, users will still be prompted to enter their credentials by every other authentication authority.

A real SSO solution would accept the external entities' "foreign" credentials to authenticate to the organization's authentication authority and would not require a user to re-enter his credentials. These are the goals that are driving distributed authentication or "federation"-based authentication infrastructures.

In a federated authentication infrastructure foreign credentials have been validated by a foreign TTP and are also accepted by an organization's proper authentication authority. The reason why they're accepted is because there's an agreement, trust or federation between the foreign and a company's proper authentication authorities. This setup does not require a copy of the foreign organization's credential database. Also in this case the users only have to take care of a single set of credentials.

As long as the SSO solution behind federated authentication infrastructures supports a mechanism to set up federation- or trust- relationships between different authentication infrastructures, it can use any of the architectures we discussed earlier in this paper. They can use secure caching, credential synchronization, a token-based architecture, or a PKI-based architecture.

PKIs use the concept of CA hierarchies, cross-certification, bridge CAs or any other CA-to-CA interoperability solution to set up federations. Kerberos-based infrastructures support federations through cross-realm trust relationships. Below are some examples of commercial authentication infrastructure products and how they support "trust" or "federation":



- Novell NDS eDirectory (version 8.5 and later) uses the concept of NDS tree federations.
- Microsoft Windows NT and 2000 domains uses inter-domain trust relationships. In Windows 2000 these trust relationships are built on Kerberos cross-realm authentication.
- Microsoft Passport uses the concept of Passport federation. In the future Passport federation will use Kerberos cross-realm authentication.

An interesting new language that will help with the creation of federations in the future is the Security Assertion Markup Language (SAML). SAML is a new standard that uses XML to encode authentication and authorization information. Because its basis is XML, SAML is platform-independent. SAML is also authentication method-neutral: for example, it could be used to set up federations between PKI- and Kerberos-based authentication infrastructures. The development of SAML is driven by OASIS (the Organization for the Advancement of Structured Information Standards). OASIS is a nonprofit, international consortium that creates interoperable industry specifications based on public standards such as XML and SGML. More information on SAML is available from <http://www.oasis-open.org/committees/security/>.

The table below compares Kerberos-, PKI- and SAML- based authentication infrastructure federation.

**Table 8.** Comparing Federation Mechanisms

	<b>Kerberos-based federation</b>	<b>PKI-based federation</b>	<b>SAML-based federation</b>
Authentication Technology	Kerberos	PKI	Any
Platform support	Many	Many	Many
Support for entity authentication	Yes	Yes	Yes
Support for data authentication	No	Yes	Under development
Authorization Federation Support	Yes, but not standardized	Yes, but very few products support it	Yes
Granularity of trust relationship and security policy support	Very Monolithic, no policy support	Support for granular trust and security policies in some products	Under development
Status	Standardized	Standardized, though standardization is not complete	Under development

### 8.1 Extending SSO to Cover Different Applications

Another important authentication infrastructure feature that can help to extend the SSO scope to cover different applications are the supported authentication Application Programming Interfaces (APIs). Although this is not always an architect's primary concern, I found it useful to provide a list of popular authentication APIs. An architect should at least know about them.

The table below lists a set of well-known authentication APIs. APIs like GSSAPI, JAAS and CDSA provide vendor-neutral API's – and also provide more than just authentication APIs. SSPI, PAM, NMA and XUDA are not vendor neutral and only provide authentication services.

**Table 9.** Authentication APIs

Authentication API name	Comments
Generic Security Service API (GSSAPI)	Security services API providing authentication, confidentiality and integrity services. Defined in RFC 2078.
Security Support Provider Interface (SSPI)	Microsoft's Authentication API – has been inspired by the GSSAPI.
Pluggable Authentication Modules (PAMs)	Sun's pluggable authentication architecture.
Java Authentication and Authorization Service (JAAS)	The Java Authentication and Authorization API. Includes a JAVA implementation of Sun's PAM. Obviously the JAAS development is driven by SUN.
Common Data Security Architecture (CDSA)	Security Services API for authentication, confidentiality and integrity services driven by the Open Group.
Novell Modular Authentication Service (NMA)	Novell's pluggable authentication architecture.
XCert Universal Database API (XUDA)	XCert's API to provide strong certificate-based authentication to applications (Xcert is now a part of RSA).

## 9 SSO: The Never Ending Story?

This paper illustrated the complexity behind setting up an SSO architecture. This is the main reason why for many companies SSO will remain a holy grail for years to come. A critical new concept in the SSO world is "federation". Federation technologies like SAML may, through their tight integration with XML, revolutionize the use of SSO in authentication infrastructures.

## References and Additional Reading

- Burton Group Technical Position on “User Authentication”.
- Richard E. Smith, “Authentication: From Passwords to Public Keys”, Addison-Wesley, ISBN 0-201-61599-1.
- Burton Group Network Strategy Report on “Single Sign-on”.
- Network Applications Consortium (NAC) Position Paper: “Enterprise-wide Security: Authentication and Single Sign-on”.
- The Open Group, Security Forum on Single Sign-on:  
<http://www.opengroup.org/security/12-sso.htm>.

**Jan De Clercq** is a security consultant in HP's Technology Leadership Group (TLG), focusing on security for Microsoft platforms. Jan has written several Compaq white papers, Windows 2000 magazine articles and columns for the Duke Press Security Administrator monthly newsletter. He's co-author of the book Mission-Critical Active Directory (Digital Press). He has been a speaker at Microsoft conferences, DECUS conferences, the RSA conference, the EMA conference and the Compaq Technology Symposium. Jan has been involved in several Windows, security and PKI-related projects for large Compaq accounts. Over the last year he has been a trusted advisor on security topics for several large Windows 2000 designs and deployments, and large PKI designs. He holds a masters degree in Criminology (RUG-Gent) and I.T. (VUB-Brussels), and a special degree in Telecommunications (ULB-Brussels). Jan is based in Belgium.

# Active Digital Credentials: Dynamic Provision of Up-to-Date Identity Information

Marco Casassa Mont and Richard Brown

Hewlett-Packard Laboratories,  
Trust, Security and Privacy  
BS34 8QZ, Bristol, UK  
{marco\_casassa-mont, richard\_brown}@hp.com  
<http://www.hpl.hp.com/>

**Abstract.** Identities and profiles are important to enable e-commerce transactions. Recent initiatives, like Microsoft .MyServices and Liberty Alliance Project, aim at the provision of identity and profile management solutions along with mechanisms to simplify users' experience. These solutions must be trusted and accountable. Current PKI solutions can be used to deal with certification and trust management. Unfortunately the complexity of managing digital credential lifecycle is one of the obstacles to their adoption. This complexity is accentuated in the case of dynamic environments, where the certified information is subject to frequent changes. In this paper we address the problem of providing up-to-date certified information in dynamic contexts. We introduce the concept of active digital credential as a mechanism to provide up-to-date certified identity and profile information along with a fine-grained assessment of trustworthiness and validity. Work is in progress both to implement a prototype and assess the feasibility of the proposed model.

## 1 Introduction

E-commerce transactions on the Internet will become more and more relevant over the next few years: the current exponential growth of e-commerce sites on the Internet, new B2B initiatives and the rise of web services to underpin enterprise and government activities are going to provide new opportunities for doing business on the Internet to a larger and larger population of customers and users.

Because of the accompanying increase in the number of business interactions where there is a lack of prior knowledge about the participants, the task of establishing, managing and ensuring trust on the Internet [1] is going to be a major issue.

In this context, the problem of dealing with up-to-date identity and profile information is crucial. Personal information, profiles and rights can change quite frequently, sometimes as a direct consequence of business transactions. Each transaction, for example, can have an immediate impact on users' credit limits and their associated credit rating information.

This problem is also important in the e-commerce space. Today people buy and sell goods and services on the Internet by interacting with a multitude of e-commerce sites. Consumers traditionally need to create and manage multiple accounts, one for each web site they want to interact with. This is an issue because of the need of remembering multiple logins and passwords, the need of supplying many times the same profile information and keeping it up-to-date.

Recent initiatives, including Microsoft .MyServices [2] and Liberty Alliance Project [3], aim at the provision of infrastructure and mechanisms to ease the pain of managing profile information across multiple Internet accounts.

Both initiatives support single-sign-on across multiple service providers by using an underlying network of identity providers. Identity providers are in charge of storing profiles and identity information and providing authentication services.

Identity providers should be accountable for the services they provide and perform due diligence tasks to assess the authenticity and trustworthiness of identity and profile information they supply to relying parties. Moreover identity providers should ensure that this information is accurate and up-to-date.

In this paper we address the problem of keeping certified identity and profile information up-to-date without the burden of heavy management processes. We introduce and discuss the concept of active digital credentials, based on a novel mechanism to provide up-to-date certified identity and profile information along with an assessment of their current level of trustworthiness and validity.

## 2 Background and Requirements

The problem of assessing the authenticity of identities and profiles is not trivial as it deals with the analysis of data provenance and implies the cooperation of users and trusted authorities through the process of verification and certification.

The trustworthiness of certification authorities has a direct impact on the trustworthiness of the certified information and it directly influences the way relying parties perceive and make use of this information.

X.509 PKI systems [4], [5] provide mechanisms for dealing with certification of information and underpinning trust management. Certification authorities and registration authorities are in charge of registering, verifying and certifying identities and profiles, according to different degrees of assessment of their authenticity. These controls can range from none to face-to-face meetings.

To increase the perception of trust and transparency, certification authorities state their responsibilities and their degree of accountability by publishing Certification Practice Statements (CPS). They also use chains of certifications and cross-certification mechanisms to underpin their trustworthiness. However, this approach has negative side effects due to the lack of scalability of certification and cross-certification chains and the complexity of managing and verifying certificates.

In the last few years alternative approaches have been introduced, for example, those based on PGP [6], SPKI [7], [8] (both based on web of trust) and recently Identity-based Encryption [9] (IBE) techniques. They address part of the X.509 PKI prob-

lems for specific realms and contexts by improving their overall usability and changing the dynamics of trust assessment. In addition, trust services [10] are emerging as a viable solution to underpin trust in e-commerce and e-business areas: the management of the information that forms the basis of trust is outsourced to professional and accountable third parties. These third parties include notarization service providers, recommendation service providers, credit rating service providers and trusted storage service providers.

In all the above approaches, digital credentials are a viable mechanism to represent, certify and convey identity and profile information along with means of verifying their trustworthiness. Digital credentials are particularly important in contexts where there is no prior knowledge of the involved parties and no web of trust is in place.

Traditional X.509 and SPKI digital credentials are usually valid for a predefined period of time, ranging from a few seconds to years: their content can be used for authentication and authorization purposes. They must be revoked whenever their content is out-of-date or it has been compromised.

Unfortunately the revocation process is a burden both for relying parties and for credential issuers. Relying parties need to check credentials against certificate revocation lists (CRLs) to verify their validity or delegate this activity to third parties. Credential issuers must deal with the complexity of the overall credential lifecycle management and they are accountable for maintaining up-to-date CRLs.

The limitation of X.509 digital credentials is evident in contexts where the certified information is dynamic: in such contexts credentials (such as attribute certificates) are short-lived and they must be revoked whenever their content changes. This causes the proliferation of digital credentials with consequent implications in term of verification of their validity, correctness of their content, management of their disposal and prevention of their misuse.

X.509 and SPKI digital credentials are either valid or not valid: there is no middle ground even if the degree of trust a certification authority has in their content may vary over time or if there is a wish to vary their content. For example, in traditional X.509 certificates any variation of their attributes implies that the whole certificate must be revoked. Important attributes including credit limits and rating levels may change very often, depending on the occurrence of business transactions and owner's reputation.

To deal with dynamic information, e-commerce service providers currently use back channel communications with trusted information. Emerging mark-up languages, like SAML [11], are used to underpin the exchange of information by means of secure assertions. The disadvantage of this approach is that it requires the set-up of ad-hoc point-to-point communication channels and the exchanged assertions are meaningful in these very specific contexts.

This paper focuses on mechanisms to enable the provision of trustworthy and up-to-date information in dynamic contexts. The objective is to create a coherent and sustainable way to certify dynamic identity and profile information and reduce the burden of their lifecycle management. We introduce a new approach based on the extension of the current model of digital credentials. This approach is meant to satisfy the following requirements:

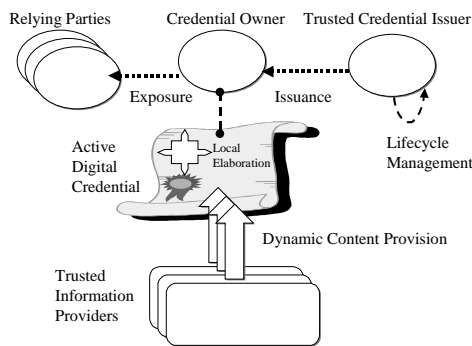
1. Provide up-to-date content of digital credentials,
2. Support up-to-date assessment of the trustworthiness and validity of digital credentials,
3. Reduce the management of digital credentials, especially in term of issuance and revocation of digital credentials.

### 3 Proposed Approach

This section introduces and describes the concept and principles underpinning *active digital credentials*. Active digital credentials are a mechanism to extend traditional static credentials (based on identity and attribute certificates) by providing means for dynamically updating their content along with an assessment of their trustworthiness.

#### 3.1 Active Credential Model

Figure 1 shows the high level model of active digital credentials:



**Fig. 1.** High-level active digital credential model and roles

In contrast with traditional digital certificates – which have static content and a predefined period of validity – active credentials provide certified mechanisms to dynamically retrieve, calculate and update their content and state their current level of trustworthiness and validity. This includes dynamic evaluations of:

1. Values of credential attributes;
2. Validity and trustworthiness of these attributes;
3. Validity and trustworthiness of the whole digital credential.

The proposed method is based on *late binding* of values to credential attributes.

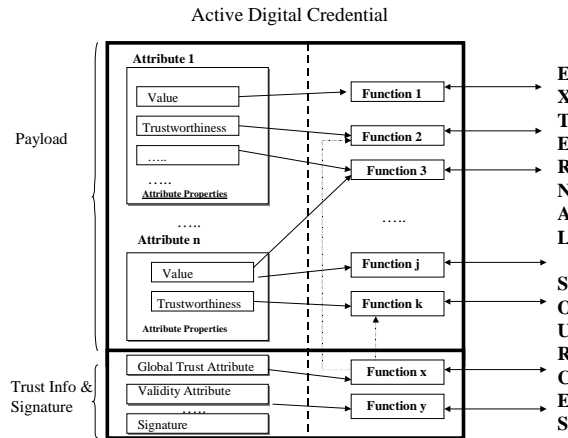
A key aspect of active digital credentials is that they not only provide certified mechanisms to retrieve their up-to-date content but they also can contain mechanisms to perform local elaboration of this information. Local elaboration can correlate in-

formation coming from heterogeneous sources, calculate the aggregated value of attributes, their validity and trustworthiness.

Credential issuers certify the trustworthiness of these *mechanisms*. They need to understand the functions underpinning these mechanisms and verify their correctness, before embedding them in an active digital credential. They might directly implement these functions after agreements with the information providers. The relying party uses active digital credentials to obtain up-to-date information and evaluate their trustworthiness and validity. This contrasts with traditional approaches, in which the credential issuers only certify the validity and trustworthiness of *data*.

A local interpretation of active digital credentials (at the relying party site) ensures that specific security and privacy requirements are fulfilled and that the interactions between the involved parties happen in a predefined and controlled way.

The basic model of an active digital credential is showed in Figure 2:



**Fig. 2.** Model of the content of an active digital credential

An active digital credential is a certified collection of *attributes* along with embedded *functions*. Its purpose is to represent identity and profile data, along with tools to assess their validity and trustworthiness.

In general, a credential attribute is characterized by a set of *properties* whose values can be determined dynamically by executing embedded functions. Attributes can represent any identity and profile information: name, address, public key, credit card information, credit rating, driving license information, etc.

The properties of an attribute include the value of the attribute, its default value and its level of validity and trustworthiness.

The functions embedded in an active digital credential are certified (and possibly directly written) by digital credential issuer(s) as trusted methods to compute property values. This computation might involve information contained within the credential and external dynamic information, retrieved from local systems or the Internet. Active credential functions are agreed between the involved parties, including the owner, issuers and the information providers.



An example of an active digital credential is a digital credit card credential (traditional PKI would make use of an attribute certificate). The list of associated attributes might include a credit card number, an expiration date, a credit limit and a credit rate along with references to the legitimate bearer. For example, the credit limit attribute might have a default value of \$10000 but the current value is determined by an embedded function, which retrieves this information directly from the credential owner's bank account.

Pursuing the example of the credit rating attribute, an associated function can dynamically determine the level of risk and trustworthiness associated to the credential owner, for instance by interacting with a certified credit rating agency.

In another example, an active digital credential can be used for authorization purposes. It contains an access attribute whose value (PERMIT, DO NOT PERMIT) is determined by a function based on the employee's role within an enterprise and the current date and time.

This mechanism is appropriate not only for the attributes contained in the credential "payload" but also for "management" attributes stating the validity and trustworthiness of the whole digital credential. Trust functions can be used to calculate the validity of a credential and its expiration date, along with the current level of trust. Levels of trust and validity could have any value in a numeric range, like  $[0,1]$  or values from a pre-defined set of values (for example, "High Trust", "Medium Trust", "Low Trust", "No Trust").

A simple application of this property is the definition of "decaying" certificates whose levels of trustworthiness and validity depend on the time elapsed since their issuance. Any type of function can be used to calculate this information at different levels of granularity. As a consequence, a digital credential may still be valid and trustworthy even if some of its attributes are not.

In general the level of trust and validity of credentials can be determined in a fine-grained way, ranging from the whole credential to specific attribute properties.

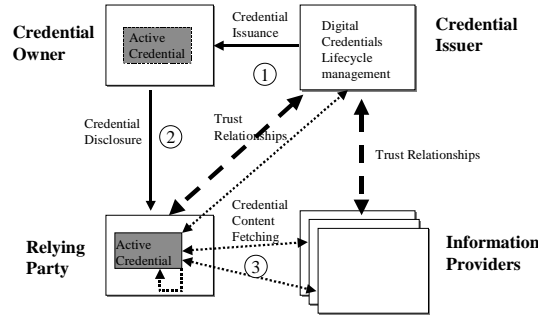
Relying parties have to make an upfront decision about which certification authorities they want to trust, exactly as they do with traditional PKI systems. Relying parties have to trust that a certification authority did their due diligence in assessing the functions embedded in an active credential.

Active digital credentials do not substantially change the current PKI certification model: they make it more dynamic. In the traditional PKI model, certification authorities issue X.509 identity and attribute certificates after retrieving and assessing their content. The content of issued credentials can be seen as a snapshot of attributes' values at the issuance time. At the time of issuance they are believed to be trustworthy and valid, according to certification authorities' policies and standards. In our model the values of attributes can be retrieved at run time. The validity and trustworthiness of the certified attributes (or the whole certificate) might vary overtime. Instead of a binary decision making process (either valid and trusted or not valid or not trusted) they can be dynamically assessed, according to certification authorities' criteria. The final decision has still to be made by the relying parties.

The next sections describe two scenarios involving active credentials along with the infrastructure necessary to properly use them.

### 3.2 Scenarios

Figure 3 shows a scenario where active digital credentials are exchanged between a credential issuer, a credential owner and a relying party.



**Fig. 3.** Active credential scenario 1

This scenario involves the following players:

1. **Active credential issuer:** it is a trusted certification authority that issues active digital credentials and manages their lifecycle. This entity has multiple trusted relationships with third parties (including banks, credit rating services, recommendation services, etc.) who provide up-to-date content for active digital credentials. The credential issuer certifies the functions which dynamically retrieve this content.
2. **Trusted information provider:** it is an information provider that supplies up-to-date information about identity and profiles, under well-defined constraints and agreements. A trusted identity provider has a trust relationship with active credential issuers.
3. **Credential owner:** it is the owner of the active digital credential. At the issuance time, the credential owner may specify which information provider must be used to retrieve identity and profile information. The credential issuer must have trust relationships with those information providers.
4. **Relying party:** it is the entity that supplies products and services on the Internet. It receives active credentials from purchasers. Access might be granted to a user depending on the dynamic evaluation of the content of these active digital credentials.

Users either directly supply their identity and profile information to the certification authority or point to trusted information providers, which must be contacted to collect this information. Active credential issuers might also play the role of trusted information providers.

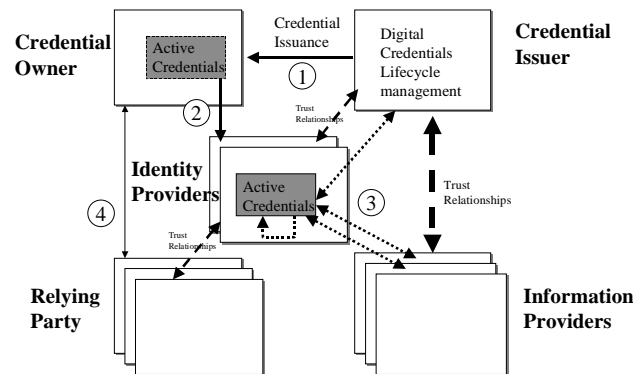
A certification authority (credential issuer) issues an active digital credential to a user after a due diligence process involving the verification of user's information (Figure 3 – step 1).

Credential owners supply their active credentials to service providers to access services (Figure 3 - step 2). Service providers accept credentials issued by the credential issuers they trust. They evaluate these credentials to retrieve up-to-date content and determine their current level of trustworthiness and validity, based on principles either defined or certified by those credential issuers (Figure 3 – step 3).

Service providers can make their final decision about the validity and trustworthiness of these credentials by taking into account dynamic information and their opinion of the trustworthiness of the certification authority. The indication of trustworthiness is useful in that one service provider might be happy to accept a trustworthiness of 50% whereas another might not. However, it is the same credential that is offered to both.

The information retrieved from information providers can be digitally signed (by the information providers) and potentially encrypted (so that it can only be decrypted by the specific service provider).

Figure 4 shows a variant of the above scenario where identity providers act as trusted authentication services and store identity and profile information. They supply this information to relying parties, on behalf of their owners.



**Fig. 4.** Active credential scenario 2

The parties and relationships in this scenario are conceptually similar to those in scenarios painted by Microsoft MyServices and Liberty Alliance initiatives. We analyze the interactions between the involved parties from the perspective of providing certified and up-to-date information.

People still own active digital credentials, issued by certification authorities (Figure 4 – step 1). These credentials might be directly exposed to trusted identity providers, that act as proxies on behalf of the owners (Figure 4 – step 2).

The fact that active credentials have been assessed and certified and their content is up-to-date increases the overall perception of trust and accountability.

Third party information providers may still retain profiles and information about the credential owners. They can disclose (part of) this information to identity providers

under terms and conditions defined by the owners, through active digital credentials (Figure 4 – step 3).

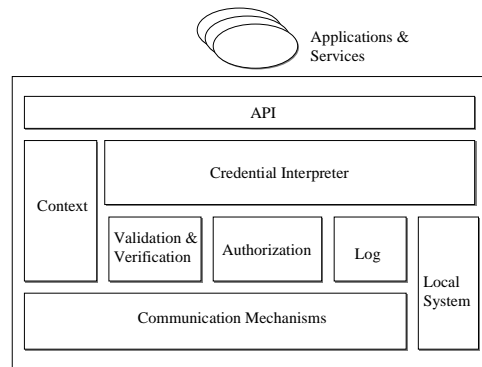
Identity providers are enabled to supply identity and profile information to service providers according to credential owners' policies, through the evaluation of active digital credentials (Figure 4 – step 4). They retrieve up-to-date certified information, evaluate its current level of trustworthiness and supply it to the relying parties.

In particular circumstances identity providers might also play the role of credential issuers.

### 3.3 Infrastructure

Because of the nature of active digital credentials, the entities that evaluate these credentials (evaluators, e.g. relying parties, identity providers, etc.) need to use an appropriate infrastructure. This section describes high-level aspects of an infrastructure for the interpretation and execution of active credentials.

Figure 5 shows the basic components of this infrastructure:



**Fig. 5.** High-level infrastructure model

The set of components includes:

1. *Credential interpreter*: it is an engine that interprets active digital credentials. The interpreter is in charge of coordinating the traditional verification and validation processes (about the trustworthiness of the credential issuer) that are executed by validation and verification sub-components. The interpreter creates an internal representation of the credential including its attributes, their properties and related functions. It locally executes credential functions in order to retrieve up-to-date information from remote information providers and it calculates the values of attribute properties.
2. *Context manager*: it is the component that aggregates and manages contextual information about the credential owner and the entity that is evaluating the credential (evaluator). Specifically the evaluator context might contain information about the identity and profile of the evaluator, which might be passed (as parameters) to

the functions embedded in the credential, during their execution. This could be requested to enable the interaction with remote information providers and fulfill constraints dictated by the credential owner.

3. *Validation and verification component*: it is the component that provides policies and processes necessary to deal with the validation and verification of active digital credentials [16]. It contains a list of trusted (identity) certificates of third parties, including those owned by trusted certification authorities. The credential interpreter uses this component to make decisions about the integrity, validity and trustworthiness of active credentials including the trustworthiness of the issuers and their digital signatures. The functionalities of this component can also be used by the functions embedded in active digital credentials, to verify digitally signed or encrypted information supplied by information providers.
4. *Authorization component*: it is the component that provides authorization and access control policies and an enforcement engine [16]. The credential interpreter uses this component to make decisions about the information that can be disclosed to third parties and which access can be granted to local resources, while interpreting active credentials.
5. *Communication component*: it is the component providing basic communication mechanisms to interact with external systems and services (via common Internet protocols, including HTTP, SSL, etc.). It is in charge of establishing secure connections with remote parties.
6. *APIs*: it is a set of high-level APIs to the infrastructure that allow external applications and services to manipulate active credentials in a way that is transparent to the underlying mechanisms.
7. *Logging system*: It is the component that logs all the activities and events that happen during the evaluation of active credentials. Logged data can be used for auditing purposes and as evidence in case of disputes. This component is fundamental for storing and providing access to non-repudiable data, such as signed information supplied by information providers to active credential functions.

The above infrastructure is meant to provide a safe environment to interpret active credentials and execute their functions. The interpreter acts as a virtual machine for these functions and it makes sure that their executions happen in a controlled way, according to predefined policies.

For privacy and confidentiality reasons the communication between the interpreter and the remote information providers might be encrypted. The credential evaluator might be asked to authenticate itself in order to satisfy credential owner's privacy policies. The interpreter mediates all the interactions during the authentication process by providing support for secure connections, for example including traditional SSL two-way handshakes.

The interpreter verifies the validity and trustworthiness of the credential issuer and provides fine-grained information to the credential evaluator according to local policies, defined by the evaluator. If the active credential issuer is trusted, the execution of trust functions within the active credential allows the retrieval and elaboration of information about the validity and trustworthiness of the credential and their attributes.

The evaluator can take into account this information along with its opinion of the trustworthiness of the credential issuer, in order to make an informed decision.

Depending on the specification of the active digital credential, the information retrieved by the embedded functions can be digitally signed by the information providers, for non-repudiation purposes. This information might also be encrypted with the evaluator's public key along with nonces to avoid reply attacks.

The certification authority could associate to each attribute (within an active digital credential) a list of the providers' identity certificates from which up-to-date information is collected. This gives extra information to the evaluator to verify the correctness of the sources of up-to-date information. The infrastructure also provides mechanisms to allow the active credential functions to access contextual information and verification functionalities.

The above infrastructure can be deployed in multiple contexts, ranging from browser plug-ins to back-end middleware processes.

## 4 Discussion

Relevant prior work in this area is described by patent [15]. It introduces the concept of static references to external information within a credential. Those references are usually implemented by an identifier (label) to retrieve information stored elsewhere. The solution described in [15] does not provide any certified mechanism to locally elaborate the retrieved information.

Relevant work and technologies have been developed in the context of mobile code, as a way to locally execute operations that might affect remote resources or retrieve information from third parties. This work includes traditional Java applets and applications, Microsoft ActiveX components, etc. Applets, applications, Microsoft components can be digitally signed by trusted certification authorities and their integrity can be validated and verified by common web browsers.

These technologies can be used as a foundation of our work. In addition, we introduce mechanisms to dynamically evaluate the content of digital credentials by defining and certifying those mechanisms (functions) within the credentials themselves, in a fine grained way: an active digital credential is a tool for describing, certifying, retrieving, elaborating and assessing dynamic identity and profile information.

X.509 PKI identity and attribute certificates are usually perceived as entities that are generated once and then relied upon offline later. Part of their attractiveness relies just on this. Actually, this is not a correct perception of the reality. For verification purposes, relying parties still have to download certification revocation lists from online sites, maintained by certification authorities. Moreover, in case of dynamic information, traditional certificates only provide a snapshot of this information at the issuance time.

Active digital credentials extend the online interaction model in order to retrieve and provide up-to-date information. Information can be retrieved from different information providers and it can be locally correlated, according to embedded policies: this is a feature that is not provided by common PKI certificates.

An active digital credential can be seen as a “contract” agreed between the credential owner, information providers and the credential issuer(s). As credential owners can dictate the criteria for the disclosure of their data, active credentials enable them to retain control over their data. There is no system-induced requirement for actual data values to be disclosed to relying parties; this privacy characteristic is especially important when the credential owner has no prior trust in the relying parties.

Active digital credentials still need to go through traditional credential lifecycle management processes. Moreover, their active functions need to be assessed and certified by the issuers (certification authorities). Issuers and information providers need to stipulate agreements, along with trust relationships. However these trust relationships are also required in scenarios that do not involve active credentials, in order to underpin trust and accountability. We believe that active digital credentials improve the lifecycle management of digital credentials by diminishing the dependency of credentials’ validities on the unchanged nature of all their contents. This is particularly true for very dynamic environments.

On one hand, the value of attributes can be retrieved dynamically and their validity and trustworthiness assessed on the fly. This allows active credentials to be valid and trustworthy even if part of their attributes are not anymore. The effect is to reduce both the number of revoked credentials and the need for short credential lifetimes, at least in contexts where the objective is to supply identity and profile information instead of authentication or authorization rights.

On the other hand, the content of active credentials depend on the availability of external systems, services and Internet connections. When those entities are not available, active credential content cannot be retrieved. This can be an issue. Risks can be partially mitigated by introducing default values for attribute properties and local elaborations. It must also be said that the content of an active credential does not necessarily depend on external information providers but it might depend on local elaboration of information like date and time (for example in case of decaying certificates).

Active credentials need a proper infrastructure in order to be evaluated. However, this is also true for traditional digital certificates, to deal with their validation and verification. The advantage of the proposed active credential infrastructure is that it provides certified and agreed mechanisms to assess the validity and trustworthiness of active credentials and up-to-date content. This can be achieved in a safe and secure way thanks to the mediation of the active credential infrastructure.

On one hand it is possible to enforce privacy and data protection constraints over credentials’ content. The content of an active credential can be disclosed and made available to a relying party only after the authentication of this relying party and the fulfillment of criteria dictated by the credential owner.

On the other hand, functions within active digital credentials need to be carefully built in order not to introduce vulnerabilities within the systems and services they access at the information providers sides. Those risks can be mitigated by constraining the kind of access and activities those functions can perform on the remote sites. Credential issuers must be involved in building those functions, in cooperation with the information providers.

Active credentials automate the process of retrieving information: the mechanisms for doing this are embedded and available within active credentials. This reduces the need for ad-hoc back-channel connections between the involved parties, as predefined and certified communication links are made available. On the other hand credential issuers must assess the trustworthiness and suitability of active credential functions: in some cases they might write these functions. There is a shift of accountability from the relying party to the credential issuers.

Current technologies can be used to implement active digital credentials. For example digital signed XML [12], [13] can be used to describe such credentials and WSDL [14] to describe embedded functions. Java classes, scripts and Microsoft (.NET) web services can possibly be used to implement those embedded functions.

Although technology is already available, the involved parties must agree on the format of active credential, both in term of semantic of the attributes (and their properties) and definition of the embedded functions.

## 5 Current Work

We are in the process of implementing a prototype that includes the basic infrastructure components to support active digital credentials, as described in our model.

We are planning to make experiments in a realistic distributed environment, in an incremental, step-by-step way.

The first step will involve a few simulated web commerce sites, at least two identity providers and a few credential issuers and information providers. Subject to availability, we are going to adopt a standard federated identity management infrastructure, for example the one that is currently under specification by the Liberty Alliance Project.

The objective is to increase our understanding of integration details, obtain more experimental data about the implications of issuing, managing and evaluating active digital credentials and refine requirements in terms of security and privacy.

Next steps will involve experiments with a broader number of players (owners, credential authorities, relying parties and information providers) to understand the implications of operating active digital credentials at a realistic scale.

## 6 Conclusion

The provision of up-to-date and trustworthy identity and profile information is important to enable e-business transactions.

Digital credentials are a viable way to certify and verify identities and profiles but their limitations are due to their static content and the complexity of the underlying infrastructure to manage them. When dealing with dynamic environments, current digital credentials introduce further complexity at the management and usability level. We introduced and discussed the concept of active digital credentials, as a way to couple certified attributes with mechanisms to retrieve their up-to-date values. Em-



bedded active credential functions are used to evaluate (by retrieval and local correlation) not only the current content of a credential but also its validity and trustworthiness, in a fine-grained way.

We believe this approach simplifies the management of credentials in dynamic environments, by reducing the need for certification revocation practices and short-lived certificates. We also believe that it boosts accountability and trust because of the definition of clear mechanisms for retrieving information, the assessment and certification of these mechanisms by trusted third parties and the provision of up-to-date content, along with a dynamic assessment of its validity and trustworthiness.

Work is in progress. We are building a prototype of our model, gather further experimental information and potentially refine our proposal in the context of a federated identity management scenario.

## References

1. Camp, L. J.: Trust and Risk in Internet Commerce .The MIT press (2000)
2. Microsoft: Microsoft .MyServices: a platform for building user-centric applications. <http://www.microsoft.com/myservices/> (2002)
3. Liberty Alliance: Project Liberty Alliance. <http://www.projectliberty.org/> (2002)
4. Housley, R., Ford, W., Polk, W., Solo, D.: RFC2459: Internet X.509 Public Key Infrastructure Certificate and CRL profile. IETF (1999)
5. Farrell, S., Housley, R.: An Internet Attribute Certificate Profile for Authorization. IETF (1999)
6. IETF: An Open Specification for Pretty Good Privacy (PGP). <http://www.ietf.org/html.charters/openpgp-charter.html> (2001)
7. Ellison, C.: SPKI Requirements, RFC 2692. IETF (1999)
8. Ellison, C., Frantz, B., Lampson, B., Rivest, R., Thomas, B., Ylonen, T.: SPKI Certificate Theory, RFC 2693. IETF (1999)
9. Boneh, D., Franklin M.: Identity-based Encryption from the Weil Pairing. Crypto 2001 (2001)
10. Baldwin, A., Beres, Y., Casassa Mont, M., Shiu, S.: *Trust Services: A Trust Infrastructure for E-Commerce*. HPL-2001-198 (2001)
11. OASIS: SAML 1.0 Specification Set - <http://www.oasisopen.org/> (2002)
12. Eastlake, D., Reagle, J., Solo, D.: XML-Signature Syntax and Processing, draft-ietf-xmlsig-core-08. IETF (2000)
13. Bray, T., Paoli, J., Sperberg-McQueen, C.M.: Extensible Markup Language (XML) 1.0. W3 Recommendation (1998)
14. W3C: Web Services Description Language (WSDL) 1.1. W3C (2001)
15. Zubeldia, P., Romney, G.: Digital Certification Systems. Patent: EP 0869637 A2 (1998)
16. Casassa Mont, M., Brown, R.: PASTELS project: Trust Management, Monitoring and Policy-driven Authorization Framework for E-Services in an Internet based B2B environment. HPL-2001-28 (2001)

# How to Buy Better Testing

## Using Competition to Get the Most Security and Robustness for Your Dollar

Stuart Schechter

Harvard University  
stuart@post.harvard.edu

**Abstract.** Without good testing, systems cannot be made secure or robust. Without metrics for the quality and security of system components, no guarantees can be made about the systems they are used to construct. This paper describes how firms can make the testing process faster and more cost effective while simultaneously providing a reliable metric of quality as one of the outputs of the process. This is accomplished via a market for defect reports, in which testers maximize profits by minimizing the cost of finding defects. The power of competition is harnessed to ensure that testers are paid a fair price for the defects they discover, thereby aligning their incentives with those of the firm developing the system. The price to find, demonstrate, and report a defect that is set by the market serves as the measure of quality.

## 1 Introduction

A market for lemons is one in which consumers cannot determine products of quality from defective goods. In such a market, consumers assume they will be sold a product of the lowest quality and so become unwilling to pay a price higher than they would for such a ‘lemon.’ As a result, it becomes economical to produce only products of the lowest quality [1]. Anderson laments that security is a trait that consumers cannot measure and as a result firms have had little incentive to produce more secure systems, so that from the standpoint of security, systems tend to be lemons [2].

This paper proposes a means to make the process of testing systems faster and more cost effective while integrating into the process a reliable metric of the quality of the tested system. This metric of quality, which can be measured throughout the testing process, is the market price to find, demonstrate, and report a previously undetected defect in the system. Previously referred to as the *cost to break* (CTB) of a system [3,4], this metric is impractical if not impossible to calculate using technical means but can be measured using a market for defect reports. In this market a firm buys defect reports from testers, at a market price governed by the presence of competition among those testers.

Comprehensive testing is an essential ingredient to the process of creating secure and robust systems and components. While testing provides a means of

evaluating the progress of the development process, it is also important to be able to evaluate the integrity of the testing process itself. Measuring how well a system has been tested is essential for determining when a system is to be released and for comparing the system to other systems.

However, the industry has struggled to quantify security, robustness, and other key measures of quality. Existing metrics, such as Mean Time Between Failures (MTBF), fail to measure how systems respond to extreme conditions and the presence of adversaries. The industry also has yet to discover how to optimize the incentives of developers and testers to ensure that systems are well designed, built, and tested. Firms need a way to ensure that systems have reached a quantifiable level of security and robustness and that they are getting value for their testing dollar. Until this goal can be achieved, firms will overspend on testing while continuing to release bug-ridden systems.

Having a metric of security and robustness as an output of the testing process has further benefits. Consider that a system is only as secure and robust as the weakest of the hardware and software subsystems from which it is built. For example, a system running the world's most secure web server is far from secure if it is running the world's least secure operating system. Before building a system with a desired level of security, it is extremely beneficial to be able to measure the strength of each potential subsystem.

This paper describes desirable properties of a market for purchasing defect reports in Section 2, with rules for the market introduced in Section 3. A number of simplifications are introduced at the beginning of the analysis in Section 4 and are later discussed and revised in Section 5. Applications are discussed in Section 6. Future work and concluding remarks are presented in Section 7 and 8, respectively.

## 2 What Makes a Good Market?

Given the opportunity, what properties would a firm build into an ideal market for purchasing defect reports from testers?

VALUE *The price paid for each defect found should be minimized.*

If the testing budget is fixed, *value* is obtained by maximizing the number of defects found. If the number of defects found is fixed, *value* is obtained by minimizing the amount of money spent in finding them.

SPEED *Testers should find and report defects to the manufacturer as quickly as possible.*

The sooner a defect is reported, the sooner it can be fixed and the less likely it is to cause damage before it is repaired. Given trade-offs between *value* and *speed*, this paper will introduce solutions that place priority on *value* and then show opportunities for exchanging *value* for *speed*.

ORDER *The easier a defect is to find (or the cheaper the defect report is to produce), the earlier it should be reported.*

The most obvious and most commonly occurring defects are both the most easy to find and the most likely to result in damage. Defects that are difficult to find are the least likely to occur when the system is operating and thus are the least likely to cause damage [5]. Product quality can be improved more quickly if the most common defects are discovered and fixed first. For any class of security vulnerabilities that cause the same amount of damage when exploited, locating and fixing the cheapest vulnerabilities to find will increase the *cost to break* of the system faster than if harder to find vulnerabilities are given higher priority.

For example, a defect in a router that corrupts one of every thousand packets is of more concern than a defect that corrupts one packet per billion. A software company prevents more theft if adversaries must spend millions of dollars to break a cryptographic algorithm than if a vulnerability remains that can be found and exploited for a thousand dollars. That the most frequently occurring problems are often easy to detect is one of the few natural laws that help the system's developer.

Before a product's release, order is the least important of these properties.

### 3 Designing a Market

A number of papers have already made a great deal of headway into means of classifying defects and vulnerabilities [6,7,8]. Rather than worry about the different consequences of each class of defect, we start by assuming all defects are equally hazardous. Section 5.6 describes how different defect classes are handled.

We design a market for defects in which there is one buyer, the firm charged with improving the quality of the system, and many sellers, the testers charged with finding and reporting defects. This one-buyer assumption may seem misguided, especially if the defects are security vulnerabilities that, if left unrepaired, would be of value to an adversary. In Section 5.7 we will see why an adversary is not likely to buy information about a vulnerability at any price that the firm is willing to pay, and thus need not be treated as a competing buyer.

With this in mind, the firm may set the rules of the market as follows:

**Access**            *All testers have full and equal access to the system to be tested.*

The average price of a defect report cannot fall below the average cost of finding a defect, otherwise no rational tester will search for defects. To maximize *value* and *speed* it is in the firm's best interest to minimize the testers' costs by giving them full and complete access to the system.

**Pricing**            *A tester reporting a unique and verifiable defect at time  $t$ , complete with test case, will receive a reward  $r(t)$ . A tester reporting a non-unique defect (one that was reported earlier) receives nothing.*

The firm chooses any reward function  $r(t)$  such that it increases continuously with time, starting with the lower bound on the cost of finding and reporting a

defect and ending with CTB, the *cost to break* of the system that represents the firm's maximum willingness to pay for a defect.

This rule frees the firm from any need to understand the testers' costs in order to set prices and maximize *value*. Rather, the firm will rely upon competition between the testers to minimize its costs.

While not essential to this analysis, I suggest supplementing the pricing rule to require testers to pay the transaction costs of processing defect reports (still receiving their reward if the report is valid). This discourages reports that are erroneous, badly detailed, or duplicates of published existing reports. Testers are expected to integrate these costs into the price they demand for each defect reported.

**Selling**      *A tester may search for defects at any time, and may report a defect immediately or at any time after finding it.*

While forcing immediate reporting would appear to maximize *speed* and improve *order*, it would be impossible to do so. More importantly, doing so would interfere with the testers' ability to make a fair profit and reduce the incentive to test. We will instead rely (yet again) on competition to achieve our goals.

**Information**      *All testers receive the other testers' defect reports immediately after they are received by the firm.*

After we've ensured that testers receive the maximum information available about the system (via the Access rule), testers' efficiency can be further improved by ensuring that they are not wasting effort searching for known defects. The information rule stated above is most sensible before a product's final release, when defects are common and there is a reasonable probability that additional testers would discover a previously reported defect before the firm can fix it. After product release, a firm may want to sacrifice *value* by delaying the release of security defects until after they are repaired in order to ensure that users have a chance to patch their systems before the vulnerability is made publicly available. This will be discussed in more detail in Section 5.8.

## 4 Simplifying Assumptions

The market for defects has been constructed so that once the the firm has declared its desired *cost to break*, CTB, it has no further strategic choices to make. The strategy lies completely in the hands of the testers.

In order to analyze how the testers will behave, we first simplify the nature and rules of the game until the analysis becomes trivial and then remove the simplifications in order to make the game better reflect reality.

The following story summarizes our simplified game.

Frank's system has a defect which he is willing to pay \$1,000 to locate. Frank invites Ann and Bob to test the system over a thousand hour period and to provide a test case that identifies the defect. The reward for reporting the defect in the first hour is a dollar. As each hour passes,

the reward for reporting the defect is increased by a dollar. However, only the first player to describe the defect will receive the reward.

Ann can find the defect with \$300 of effort. Bob can find the defect with \$500 of effort. No other tester can find the defect for less than \$500 of effort. All of these facts are known to all the testers.

The simplifications implied in the above example can be formalized in the following statements:

- (1) *There is one and only one defect.*
- (2) *The act of finding a defect is atomic and takes no time.*
- (3) *Each tester knows her cost of finding the defect.*
- (4) *Each tester knows the other testers' cost of finding the defect. Each tester knows that the other testers know each other's cost of finding a defect, and so on.*

If each player knows that a defect exists and knows everyone's cost of finding the defect (perfect knowledge as defined in simplifying assumptions 3 and 4), the market can be modelled as a cooperative game with multiple sellers (the testers) and one buyer (the firm). All testers are offering an information good that is a perfect substitute for the others' good as all are describing the same defect. The firm is only willing to pay for this information from one of the testers.

The marginal contribution of all testers except for the one with the lowest cost is 0. The marginal contribution of the low cost tester is the difference between her cost of production (finding the defect) and that of the tester with the second lowest cost of production. The tester cannot demand a price higher than the cost of production of the second lowest cost tester. Since the firm does not have the ability to negotiate in this market, we can assume that it will have to pay the highest price that is less than the cost of the second lowest cost tester. In our story, the smallest discrete unit of currency is a dollar.

Ann believes that Bob and the other testers are rational and that they will not sell the defect at a loss. Since no other tester can find the defect for less than \$500, Ann can sell the defect at any price below \$500. To maximize her profit, Ann finds the defect and reports it when the reward reaches \$499.

## 5 Approaching Reality

In the real world, products have multiple defects, finding a defect is a time consuming operation requiring a large number of subtasks, and each tester's knowledge is far from perfect. We'll now try to replace the above simplifying assumptions with ones that more closely match the realities of testing.

### 5.1 The Presence of Multiple Defects

This market for defects would be of little value if it could only be used to find a single flaw in the system. This assumption must be removed, which we will represent by writing it again and crossing it out.

(1) ~~There is one and only one defect.~~

In order for the market described to accommodate reports of multiple defects we must substitute a new assumption for assumption 1.

(1a) *All defects are independent. Finding one defect  $d$  in the set of all defects  $D$  neither aids nor hinders the search for another defect  $d' \in D$ , nor does it indicate that a tester is more or less likely to find another defect.*

With this new assumption in place, we can open markets for any number of defects in parallel, even if we don't know how many defects there are. It is easiest to view these parallel markets as one single large market for defects.

There are now two defects  $d_1$  and  $d_2$ , each of which Frank is willing to pay \$1,000 to locate. Ann's costs of finding defects  $d_1$  and  $d_2$  are \$300 and \$400 respectively. Bob's costs of finding defects  $d_1$  and  $d_2$  are \$500 and \$200 respectively. Ann reports  $d_1$  for a reward of \$499 and Bob reports  $d_2$  for \$399.

### 5.2 Knowledge about Others' Costs (Part One)

(4) ~~Each tester knows the other testers' cost of finding the defect.~~

It's quite unlikely that a tester can know how hard it is for every other tester to find a defect when she doesn't even know what the defect is yet. We will replace this assumption with one that is somewhat less far fetched. This will allow us to make progress in the analysis before relaxing this assumption further in Section 5.4.

(4a) *Once given knowledge of a defect  $d$ , a tester with one of the two lowest costs of finding  $d$  will know the other lowest cost tester's cost to find  $d$ .*

There are now two sets of moves for each tester. First, a tester must decide if and at what price she will search for a defect. Then, if the tester has found a defect, she must decide at what price she will sell it. She can calculate the appropriate times for these actions from her matching price choices by using the inverse function of  $r(t)$ .

If Ann knows her cost of finding a defect (Assumption 3), and we define  $c_a$  to be this cost, we say that Ann will find a defect when the price covers her cost; when  $r(t) \geq c_a$ . At this time  $t$  she can immediately sell the defect and break even<sup>1</sup>, or wait until just before the next lowest cost tester will find the defect and sell it for a profit. She can do this because once she has found the defect she

<sup>1</sup> We ignore the unlikely event that Ann's cost is exactly the same as Bob's, and that they both attempt to report the defect at the same unit of time, as this becomes increasingly unlikely as our measurement of time becomes increasingly less discrete.

will know the next lowest cost testers's cost of finding the defect. Though the story changes slightly under these new assumptions, we always reach the same end.

Ann and Bob's costs haven't changed, but neither player starts out with any knowledge about the other's costs. When the reward price reaches \$200, Bob spends \$200 to find defect  $d_2$  knowing that he can sell it immediately to break even regardless of what he discovers Ann's cost of finding  $d_2$  to be. After spending his \$200, Bob learns everything he needs to know about  $d_2$ , including that it would cost Ann \$400 to find  $d_2$ . Bob now knows, just as he did in the previous example, that he can wait to report the defect until the reward is \$399. Using the same logic, Ann will find defect  $d_1$  once the reward reaches \$300 and will once again report it when the reward is \$499.

### 5.3 The Time and Cost of Searching

- ~~(2) The act of finding a defect is atomic and takes no time.~~
- ~~(3) Each tester knows her cost of finding the defect.~~

Finding a defect can require both time and expense, and it is hard to determine the cost of finding something until you know what it is you are looking for. The following assumptions are much more realistic.

- (2a) A unit of work  $w$  spent searching for a defect is atomic, has a fixed cost  $c_w$ , and takes no time.
- (3a) Each player can estimate the probability  $p_w$  that a unit of work she engages in will reveal a previously unreported defect.

After 500 hours into testing the reward for reporting a defect has reached \$500. Ann has a test that she can design and run for \$5. The test has a 1% chance of revealing an unreported defect. Running the test has the expected value no less than  $0.01 \cdot \$500 = \$5.00$ . Ann decides to perform the test.

More formally, we say that a tester will search for a defect when her expected minimum reward justifies the cost of searching. The equation below represents this by stating that the expected value of searching, which is the product of the reward and the probability that searching yields a reward, must be greater than or equal to the cost of searching.

$$r(t) \cdot p_w \geq c_w$$

Simplification 2a will not be further refined in this paper, and so our analysis will continue to be predicated on the assumption that a unit of work takes no time. There is no doubt that since even small units of work take time, a trade-off



exists between the *speed* and *value* in testing. The choice of testing period is one which needs to be made based on the time and budgetary constraints of firms with knowledge of the state of the art in testing tools and technology. It is thus outside the scope of this paper.

#### 5.4 Knowledge about Others' Costs (Part 2)

~~(4a) Once given knowledge of a defect  $d$ , a tester with one of the two lowest costs of finding  $d$  will know the other lowest cost tester's cost to find  $d$ .~~

Removing this assumption about knowledge makes the players' second move strategy less clear. All we can do is analyze the strategies of each player based on his or her beliefs about the costs of the other players.

Once Ann has found a defect, the crux of the problem is that she no longer knows the likelihood that another tester will report that defect at time  $t$  for reward  $r(t)$ .

We will represent Ann's estimate of the probability that no other player will have reported defect  $d$  at time  $t$  as  $p_a(T \setminus a, D, D_r, d, t)$ , where  $T \setminus a$  is the set of all testers except Ann, and  $D_r$  is the set of defects that have been reported.

Ann will report the defect at a time that maximizes her expected payoff function, which is once again the product of the reward times the probability of receiving the reward:

$$r(t) \cdot p_a(T \setminus a, D, D_r, d, t)$$

The better Ann's expected payoff function approximates the knowledge of assumption 4a and the behavior of the other testers who might find the same defect, the more money Ann will be able to make. The uncertainty, however, should encourage Ann to report earlier than she might if given the perfect knowledge she had with assumption 4a.

#### 5.5 Defect Dependencies and Learning about the Skills of Others

There is a strong case for removing simplifying assumption 1a. For starters, we wouldn't think Ann was very smart if she discovered a number of defects, watched Bob report each defect at a price below what she believed his costs were, and refused to revise her estimates of Bob's costs to find additional defects.

We also can't ignore the possibility that the discovery of one defect will simplify the discovery similar defects. What's more, it is often difficult to even determine where one defect ends and another begins.

~~(1a) All defects are independent. Finding one defect  $d$  in the set of all defects  $D$  neither aids nor hinders the search for another defect  $d' \in D$ , nor does it indicate that a tester is more or less likely to find another defect.~~

Once we acknowledge that the discovery of a defect changes every player's cost of finding another defect, a new assumption won't fix the rules of the game. Rather, it is the rules of the market and not the simplifying assumptions that need repair. The following example shows how the firm may fail to obtain *value* under the pricing rules from Section 3.

Frank's system has three defects which are all closely related. Ann can find the first of three defects for \$500, and can find each of the two remaining defects for \$10 each given knowledge of the first defect. Bob can find a defect for \$2000 but given knowledge of any of the defects can find the other two for only \$5. Charles can find any defect for \$600, but isn't aided by information about defects that have already been discovered.

Ann finds the first defect when the reward has reached \$500 and then instantly finds the two other defects. She reports all three defects at once when the price reaches \$599, collecting \$1,797.

The above example demonstrates two problems with the rules as they stand. Since Bob has a lower cost of finding the incremental defects and Ann is the one reporting them, the most efficient tester is not the one testing for the incremental defects. Since the reward function increases monotonically, the firm must grossly overpay for the incremental defects rather than paying no more than the cost of the second lowest cost tester. By the second lowest cost rule, Ann should collect \$599 for reporting the first defect, and Bob should collect \$9 for each of the additional defects, allowing the firm to pay \$617 for the three defects instead of \$1,797.

To fix this flaw I propose that the market be reset each time a single defect is reported, with the reward price being reset down to the transaction cost of reporting a defect (\$0 in our story). If the reward is reset to \$0 when Ann reports the first defect, Bob will report the next defect when the reward price reaches \$9. The reward will once again be set to \$0 and after it climbs to \$9 again he will report the third defect. While this approach sacrifices a modest amount of *speed*, without it *value* is all but nonexistent.

The firm may want to adjust the reward function so that it grows very slowly (or remains at \$0) while an outstanding defect is being fixed. This prevents a slew of related defects and test cases, which are trivial modifications of the first reported defect, to be reported at high frequency causing strain on the reporting system. Alternatively, allowing the reward to increase and a flood of related test cases to be reported may be just what is needed to build a good internal regression testing suite.

There is also a rather practical benefit to setting the reward back to \$0 each time a defect is reported in that it allows for more predictable budgeting. If a dollar is added to the reward 'pot' every hour, and reporting a defect allows the tester to take the money in the pot, then the rate at which defects are reported does not affect the rate at which money flows out of the firm's coffers. The disadvantage is that fixing the budget removes any guarantees about the cost to find and report a defect (cost to break) after any given testing period. A compromise solution exists at which the firm may decide to increase the flow of money into the pot in order to cause defects to be reported more quickly.

With the new rule for the reward function in place, Ann may now wish to recalculate  $p_a(T \setminus a, D, D_r, d, t)$  for her unreported defects each time a defect is reported and the reward is reset. In doing so, she may actually determine that

she may want to take a loss if she has underestimated the other testers in her past calculations of  $p_a$ . However, the guiding equation behind Ann's strategy need not change as she is still working to maximize the expected reward for each defect she has found.

### 5.6 Some Defects Are More Important than Others

Different classes of defects have different consequences and so the firm may be willing to pay more for some defects than others. Rather than create a market for each class of defect, it is easier to pick one class of defect as the baseline class and measure the other classes relative to the baseline class. If the firm is willing to pay only half as much for a defect of a new class as it is for the baseline class, the reward for finding a defect from that class at time  $t$  can be set to  $\frac{1}{2}r(t)$ . Generally, if a defect is from a class that the firm is willing to pay  $x$  times more for than it would for a defect from the baseline class, the reward for reporting such a defect should be set to  $x \cdot r(t)$ .

This approach is equivalent to running a parallel market for each class of defect, except that all the rewards are reset each time a defect is reported. Since defects may not be independent of defects from other classes, and finding one in one class may make it easier to find one in another class, resetting all rewards is indeed necessary.

### 5.7 The Presence of Additional Buyers

We began our analysis by assuming that the firm is the only buyer of defects during the testing process. One might be concerned that adversaries may purchase security defects. Assume that, following the testing process, the firm continues to offer a reward equal to its claimed cost to break (CTB) to anyone reporting a new security vulnerability. Under what conditions will an adversary buy a vulnerability during the testing process?

A security defect has no value to an adversary if the flaw is discovered and fixed by the firm before the product is released. An adversary can buy a security vulnerability by publicly offering to purchase a vulnerability at the end of the testing period at a price just higher than the the reward offered by the firm, or  $\text{CTB} + \epsilon$ . Neglecting  $\epsilon$  and assigning to  $p$  the probability that the defect will not be discovered by the end of the testing process, we note an adversary would not pay more than  $\text{CTB} \cdot p$  for a security vulnerability during the testing process.

However, the tester selling the defect also has the option of waiting until testing is over before making the transaction, and knows the firm will always willing to pay CTB for a previously unreported vulnerability even if the adversary is not. The tester will demand a price that is greater than  $\text{CTB} \cdot p$ . This means that the sale is a zero sum game, between the tester selling the vulnerability and the adversary buying it, in which the player who can best estimate the true probability  $p$  wins. Since the adversary does not know anything about the vulnerability until he buys it, he will never be able to estimate  $p$  as well as the tester who discovered the vulnerability and knows how hard it was to find.

What's worse, if the tester has found multiple vulnerabilities she can maximize her profits by selling the adversary the one that is most likely to be discovered by another tester (and thus is the least valuable.) As a result we can conclude that adversaries buying security vulnerabilities will be faced with a market for lemons for the duration of the testing process, and will find it in their best interest to wait until testing is complete to make a purchase. When they finally do purchase a vulnerability, they will have to pay at least as much as CTB, the price offered by the firm.

Can the tester sell at a price below CTB and make up for the difference in volume by selling to multiple buyers? In such a case, a buyer is able to engage in arbitrage by purchasing the defect and selling it to the firm. This will result in the defect being fixed, and any exploits of the defect becoming worthless. What's more, once the first buyer has purchased the exploit he would also be able to sell the exploit to other buyers. As there would now be multiple sellers of the same information good, the price of the good would drop to zero. For both these reasons, a tester cannot expect to be able to sell a security defect more than once.

In fact, the tester's ability to resell security defect information adds an additional barrier to the sale of security defects to adversaries at *any* price. Any adversary must worry that after he has paid to learn the details of the vulnerability, the seller will resell that information to the firm (or other buyers). If the transaction protects the anonymity of the seller, the tester can claim that another tester must have discovered and reported the defect. Since the only way to ensure the testers won't resell is to take them out of the game, testers who wish to avoid sleeping with the fishes will be wary of selling security defects to anyone other than the firm.

## 5.8 Delaying Release of Discovered Defects

It may not always be advantageous to release defect reports immediately to all testers. This is especially true when the test is open to the public, the product is in use by customers, and the defect affects the security of the product.

If defects are not immediately released, the firm should continue to pay only the first reporter the full reward or it should split the reward among each tester. This will cause testers to demand more for each defect, as they must bear the risk of not receiving the full reward for their efforts in searching for defects. Testers will also be wary that the firm might be tempted to create a phantom tester which it could claim reported defects before the true reporter. If the testers do not trust the firm, the firm may need to employ a trusted third party to manage the entire defect reporting and reward process. If only the first reporter receives the reward, the trusted third party may only be needed to sign and time stamp a message authentication code of each defect report submitted, thereby providing proof of who found the defect first.

The firm should never give a full reward  $r(t)$  to every tester who reports a defect before the public is made aware of the defect's existence. This may seem like a good way to put *speed* before *value* and encourage as much testing as

possible, but it will more likely cause the testers to collude, sharing defects to bleed cash from the firm. This is particularly serious if adversaries pose as testers and learn about vulnerabilities by taking part in such collusion.

## 6 Applications

Testing processes that result in metric of quality, such as we've seen above, have applications that expand beyond improving testing. These include improving product security, better management of release processes, and improving the effectiveness of incentives for both in-house developers and outside contractors.

### 6.1 Differentiating System Security

In previous work I proposed that security should be measured economically by the cost to break into a system and that, by introducing this metric into the marketplace, the providers of more secure software can drive out the lemons [3]. The metric fixes the information asymmetry and addresses the lemons uncertainty problem.

My previous work also shows how to measure the *cost to break*, or CTB, of the firm's system  $F$  against that of a competing system  $C$  [3]. To place an upper bound of  $u$  dollars on  $C$ 's cost to break, the firm offers a reward of  $u$  dollars to the first tester to report a vulnerability in  $C$ . To place a lower bound of  $l$  dollars on  $F$ 's cost to break, a reward of  $l$  dollars is offered for each and every unique vulnerability reported in  $F$ . By showing that  $l > u$ , the firm demonstrates that its system  $F$  is more secure than the competitor's system  $C$ .

The bounding approach is effective if the true cost to break is higher than the lower bound, but exceedingly expensive otherwise, as the firm must pay a large reward for a large number of vulnerabilities. Ideally, the firm would like to measure the cost to break at lower expense during testing and throughout the development and release process. The techniques described in this paper, when applied to a set of testers that includes the public, allow the firm to use competition to avoid overpaying for the vulnerabilities that must be repaired to reach a desired cost to break.

If the firm can assume that an adversary who wants to exploit its system has access to the same public that was invited to test, it is reasonable to expect that the adversary will have to spend at least as much as the firm did in order to find an exploit.

The *cost to break* metric also aids the construction of more secure large-scale systems built from component subsystems. Recall the example of the highly secure web server running on an insecure operating system. System builders, faced with the choice of how much to invest in the security of a component, often err by paying too much for one component and too little for another. If the security of each can be measured, then the firm can be sure to invest in the security of the weakest link in the system. For systems that are only as strong as their weakest component, such a strategy will result in the maximum increase in the security of the composite system.

## 6.2 Compensating White Box Testers and Software Engineers

Defect markets also can be used directly for compensating all types of test engineers. Public testing does not eliminate the need for professional internal testers, as they will be able to perform white box testing using information not available to the public to find defects at lower cost. Internal testers are also best used early on as the defect reports they produce are of higher quality and are thus cheaper for the firm to verify and process. Such costs may be a significant component of the transaction when defects are inexpensive to find.

Testing often suffers because the job is considered unrewarding. Test engineers are often paid less than developers, and their positions are less esteemed, since building systems is more highly regarded than finding problems with them. All too often, good testers are promoted out of testing. This would be less likely to happen if testers' compensation more directly matched the contribution they make to the product. Markets for defects create a meritocracy in which the best testers are rewarded in a way that makes their jobs more desirable while ineffective testers are encouraged to find more suitable positions. One might even wonder whether great testers lost to development teams will start testing systems again, in their free time, to pick up extra money.<sup>2</sup> Some might even learn that their true calling is to return to testing.

Testers are not the only members of the team who can be evaluated based on the cost of finding defects in systems. It has often been difficult to provide engineers with incentives for developing high quality work (rather than fast work) without an objective measure of the level of defects. The cost for a tester to find a defect when the engineer releases his work for testing provides the measure of quality that can be integrated into the engineer's incentives.

## 6.3 Release Management

The cost of finding a defect may be used as a measure for determining when systems are ready to ship, especially when other metrics, such as Mean Time Between Failures (MTBF), are hard to measure or not applicable. Many other metrics fail to account for extreme conditions such as extreme load or attack by an adversary with extensive knowledge of the system, that would not otherwise occur in the laboratory and aren't detectable in typical real world use.

## 6.4 Contracts

The question of when a software development contractor has fulfilled its obligation to provide a system of a reasonable quality is one that has been the subject of innumerable disputes. Placing a lower bound on the cost of finding a defect in the completed system may be just the metric needed to allow parties to reach agreement on quality before work begins. A wise client will pay more to ensure

---

<sup>2</sup> Developers should not be allowed to collect rewards for testing their own systems as doing so compromises their incentive to create as few defects as possible.

that the testing budget comes out of the contractor's pocket, thus ensuring that the contractor will have the maximum incentive to provide initial quality. The client should also insist that the testing itself be open to third parties that cannot be influenced by the contractor.

## 7 Future Work

There are many ripe areas of research to explore as the ideas I have outlined are put into practice. A starting point is to relax the assumption that work spent searching takes no time (part of Assumption 3a). One approach would be to quantify the trade-offs between the value and speed of testing, perhaps as a function of the number of available testers and size of the system to be tested. Much can also be gained by formalizing rules for determining whether a defect report is indeed valid, and to explore the role played by third parties. More light needs to be shed on how testers behave given their limited knowledge of the abilities of others. This may arise through empirical study, or by adding additional assumptions so that a Bayesian-Nash Equilibrium analysis may be performed.

## 8 Conclusion

The security and robustness of systems has traditionally suffered because testers are not properly rewarded. I have described an economic approach to rewarding testers that aligns their goals with those of the firm developing the system. This approach allows firms to cost-effectively test systems for individual flaws and simultaneously measure the quality of the overall system.

Using testing based on defect markets as described in this paper, subsystems can be built with measurable security and can be used to construct more complex systems which can then also then be measured.

**Acknowledgements.** This paper could not have been completed without the advice, comments, and suggestions from Adam Brandenburger, L. Jean Camp, David Molnar, David Parkes, Michael D. Smith, and Omri Traub.

This research was supported in part by grants from Compaq, HP, IBM, Intel, and Microsoft.

## References

1. Akerlof, G.A.: The market for 'lemons': Quality uncertainty and the market mechanism. *The Quarterly Journal of Economics* **84** (1970) 488–500
2. Anderson, R.: Why information security is hard, an economic perspective. In: 17th Annual Computer Security Applications Conference. (2001)
3. Schechter, S.E.: Quantitatively differentiating system security. In: The First Workshop on Economics and Information Security. (2002)

4. Silverman, R.D.: A cost-based security analysis of symmetric and asymmetric key lengths. <http://www.rsasecurity.com/rsalabs/bulletins/bulletin13.html> (2001)
5. Brady, R.M., Anderson, R.J., Ball, R.C.: Murphy's law, the fitness of evolving species, and the limits of software reliability. <http://www.ftp.cl.cam.ac.uk/ftp/users/rja14/babtr.pdf> (1999)
6. Camp, L.J., Wolfram, C.: Pricing security. In: Proceedings of the CERT Information Survivability Workshop. (2000) 31–39
7. Aslam, T., Krsul, I., Spafford, E.: Use of a taxonomy of security faults. In: 19th National Information Systems Security Conference. (1996)
8. Landwehr, C.E., Bull, A.R., McDermott, J.P., Choi, W.S.: A taxonomy of computer program security flaws. *ACM Computing Surveys* **26** (1994) 211–254



# Structured Risk Analysis

Neil McEvoy and Andrew Whitcombe

Consult Hyperion, 8 Frederick Sanger Road, Surrey Research Park, Guildford, Surrey,  
GU2 7EB, United Kingdom  
{neil, andrew}@chyp.com

**Abstract.** Decisions concerning the security of information are frequently taken on a different basis to other business decisions. Necessary investments may not take place on the grounds that the returns are not easily quantifiable, when compared with other investments, or in the blithe hope that “it will never happen to us”. Conversely, “fear, uncertainty and doubt” can lead organisations to divert funds towards information security that might be better spent elsewhere. This paper presents Structured Risk Analysis (SRA), a method to help organisations take rational steps to improve their information security. Using SRA, an organisation can place information security in the context of the business as a whole, to determine the appropriate level of resources to be directed toward improvements. Given this budget, SRA enables the organisation to identify how it should be spent to generate the best possible security ‘return’.

Whilst other risk analysis methods exist, we believe SRA to be more closely integrated with other business processes, systems development and operations, giving a much firmer context for the analysis and more confidence in the results. It is non-proprietary, does not require specialist software tools and is inherently ‘tuneable’ in the level of detail that is applied: all of which means that SRA is particularly cost-effective.

We believe SRA gives business owners the information they need to make truly informed decisions about the security, viability and future direction of their Information Systems.

## 1 Background

Information security is progressively becoming headline news, particularly as ordinary people start to rely on electronic means to carry out routine transactions such as making payments from a bank account or filing a tax return. Barely a week goes by without scares about the latest virus, or a report of an embarrassing lapse by a large organisation, hitting the front page and radio and television new bulletins. In the week this paper was drafted, the biggest noise was about the UK Inland Revenue suspending their online self-assessment service following reports that users had seen snippets of information relating to other taxpayers [1].

All of this understandably has the tendency to leave the lay user confused or, worse, frightened. What is more surprising is that so many large organisations leave themselves wide open to appear in tomorrow’s headlines. That is not to say that money and effort is not being spent. For example, in the wake of September 11<sup>th</sup>, many airports and airlines are deploying biometric technology to screen travellers –

despite informed academic comment that many systems are incapable of delivering the claimed benefits.

A number of organisational cultures can lead to irrational stances towards information security. The most negligent would be the assumption that “it could never happen to us”. A more prevalent response to the issue is to appoint a person or team to be responsible for information security, but without the breadth of experience, or the ‘clout’ to identify and solve the most serious security issues and to successfully deploy and promote the solutions. For example, a team with a strong software background may understand the security flaws with the server operating systems and database management systems, but may under- or over-estimate the impact such flaws might have on the business. Conversely, a business person charged with the same responsibility may have an acute awareness of potential damage to the organisation, without understanding how a hacker might be able to penetrate the systems to realise his worst fears.

The successful management of information security risks has many components. Policies must be developed and applied that define who can have what kind of access to which information and infrastructure components. Procedures must define the controls around such access. Technical means must be deployed to enforce policies and procedures. Active monitoring must detect serious or systematic attempts to circumvent the policies and procedures.

None of the above should be defined in a vacuum. Otherwise, (for example) overly bureaucratic procedures might be defined to protect access to equipment which is not critical to maintaining information security; or sensitive traffic on a wireless network may be available, un-encrypted, on the street outside the building. The key to taking optimal decisions in all these areas is to understand the risks to which the organisation is exposed by doing nothing, and the degree to which these could be mitigated by the deployment of particular countermeasures.

Of course, recognition of the value of risk analysis in this area is not new, and methods to conduct such analyses have been proposed. However, uptake of these methods has been slow, leading to *ad hoc* analyses, where they take place at all. We find that most methods for risk analyses have one or more of the following deficiencies:

- They are not rooted in the wider disciplines of information analysis and system development, which leads to doubt about the completeness of any analysis.
- They do not provide a means whereby information security risks can be compared with other business risks (or indeed business opportunities), which would allow business people to take rational decisions about the application of funds.
- They fail to distinguish clearly between *threats* to the business and the *vulnerabilities* of systems, leading to a muddled analysis
- They are either too superficial (nothing which is not obvious is revealed) or too detailed (can’t see the wood for the trees)
- They are ‘static’ in terms of the foreseen vulnerabilities or countermeasures, whereas in reality information security is a fast-moving discipline
- They are inflexible or prescriptive
- They are proprietary, or rely on specific software packages.

## 2 Requirements for SRA

Recognising these deficiencies, we decided to define our own method, Structured Risk Analysis (SRA) that would address these issues.

What then were the requirements for SRA?

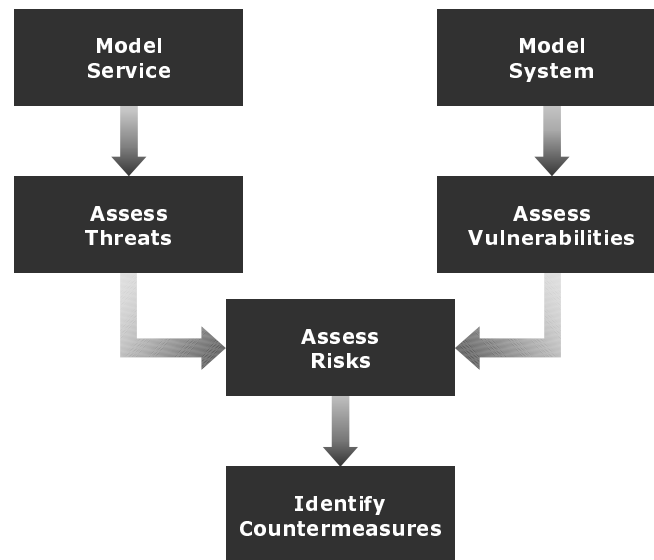
- *Business context.* The method must enable business people to take business decisions in the real world, where there are other calls on resources, and not the 'ideal' world of limitless resources to improve information security.
- *Technical grounding.* The method must be rooted in 'best practice' for systems analysis and design, with a defined place in the systems development lifecycle, whilst not dependent on any particular techniques, methods or tools in this area.
- *Separation of concerns.* The method must clearly distinguish between business and technical concerns, and bring both together leading to a robust analysis.
- *Support for quantitative analysis.* While not appropriate in all circumstances, the method must be capable of quantifying exposure to risk, in order to determine an overall information security budget and the optimal allocation of that budget.
- *'Tuneable' analysis.* It must be possible to control the level of detail of the analysis, appropriate to particular corporate circumstances.
- *Evolution.* It must be possible for the method to evolve, for example to recognise changes in methods of attack and countermeasures
- *Maintainability.* It must be possible to generate a 'living' risk analysis model, which will allow changing circumstances to be tracked without undue expenditure.
- *Openness.* It must be possible to use the method without payment of licence fees or purchase of specific software.

## 3 How Does It Work?

### 3.1 Overview

Figure 1 below outlines the basic steps undertaken in order to carry out a successful Structured Risk Analysis.

SRA builds on existing Systems Analysis methods and best practice to ensure that the service and systems under consideration are represented clearly, in a way which is understandable to stakeholders and hopefully in a manner that is consistent with the rest of the organisation. This enables the results of the SRA to be viewed in an overall business context, allowing security decisions to be taken as part of a complete picture of organisational structure, spend and development. These representations, or service and system 'models' are then used to identify the threats and vulnerabilities that exist within the system.



**Fig. 1.** The basic steps undertaken during a Structured Risk Analysis

All the threats and vulnerabilities identified are then cross-referenced to ascertain whether it is possible that a particular threat might be realised by a particular vulnerability. If this is possible, then we have discovered a real risk.

A number of factors are considered in order to help identify the extent to which such risks expose the organisation to damage, leading to the selection of appropriate countermeasures.

### 3.2 Understanding the Environment

In order to provide a complete picture it is important to understand both the service being offered and the system via which this service is provided. The representation of this understanding, through standard systems analysis modelling techniques, provide the model service and model system that form the foundation of the SRA process.

The ‘Model Service’ process in Figure 1 identifies the information assets that the service must manage. This is achieved by drawing up a Logical Data Structure (LDS), to identify data entities, the keys that uniquely identify each instance of the entity, the entities attributes, and the relationships between the entities. Ideally, an LDS will have been produced during the system development process in the context of a structured method such as SSADM [2] or one of the flavours of the ‘Yourdon’ method [3]. We find that since an LDS can be presented as a set of tables, it is particularly suited to review and discussion by a wide variety of stakeholders. Alternatively, an object model can be derived if Object-Orientated Analysis (OOA) [4] is the preferred method of analysis within an organisation. Whatever method is used, it is crucial that the outcome represents ‘real world’ entities about which the system holds information, and is uncluttered by any detail concerning the implementation. Therefore, even

for a large information system, the number of such entities is usually limited to 20 or so.

The 'Model System' process is used to understand the physical architecture of the system within which this information is stored and manipulated. It must identify processing elements, networking elements and storage elements. Normally, such models will be generated during the 'system architecture' phase of system development process. The depth, or level of detail, of such a model is of course variable. In our experience, it is best to work (at least initially) at a high level, as long as the model is complete at that level. In this way, the later stages of analysis will remain tractable and the 'big picture' will emerge. If necessary, particular features of the big picture can be examined in greater detail later.

Sometimes obtaining the service and system models is straightforward, as the logical and physical architectures have been laid down as part of the overall design of the system, and have been well adhered to during implementation. However this is not always the case, and the key challenge at the start of any SRA is often producing models that bridge the gap between the system an organisation thinks it has, and what is actually in place.

As a final step before undertaking specific security analysis it is useful to construct a cross-reference between the two models. This will identify for each data entity (or, at a more detailed level, for each attribute), which physical system elements are responsible for its capture, processing, transmission and storage.

### 3.3 Assess the Threats

Threats are things that might damage the business or organisation providing the service. This damage might be indirect, in the sense that most immediate loss might be inflicted on (say) a customer or employee. Threats are, in themselves, nothing to do with the technology which provides the system. A threat might be 'the reputation of the business may be damaged by revealing sensitive customer information'. On the other hand, 'an attacker outside the building might eavesdrop on 802.11b wireless traffic' is definitely not a threat in SRA's taxonomy. (As we shall see, it is a good example of a 'vulnerability'.)

It follows that threats are derived from the Service Model. This is done by examining every entity in the model from the familiar 'CIA' angles: confidentiality, integrity and availability. For each entity, it is necessary to determine the worst consequences of the loss of Confidentiality, Integrity and Availability. To do this it is necessary to examine all entities for an attribute. For example, imagine a medical database system. For the 'patient' entity, the attribute 'HIV status' might be most sensitive from the confidentiality viewpoint, whilst 'blood group' might be most important from the integrity viewpoint. However, rather than carry forward the security characteristics of every attribute for every entity, it is generally more convenient to carry forward the 'worst cases' for each of C, I and A, at the entity level. Thus if there are *IE* information entities, the number of potential threats to carry forward in the analysis is  $(3 \times IE)$ .

Of course to perform this assessment, it is necessary to place the threat on some kind of scale. The key factor here is the Damage, *D*, which would be inflicted on the organisation by a single incident. Of course, this could be quantified fully and measured in pounds or some other currency. This is usually necessary in a commercial or-

organisation where it is required to derive a security budget which is proportionate in terms of other potential use of funds. The direct cost is usually straightforward to estimate. However, indirect costs could predominate, for example the loss of custom which might arise from unfavourable publicity, and the marketing effort required to re-establish customer trust. The indirect costs are much harder to estimate. In some cases, for example where human safety is concerned, it may be considered inappropriate to quantify  $D$  in hard cash (for example by putting a price on a human life). For these reasons, in the interest of quickly identifying the big picture, we are often asked to perform semi-quantitative analyses: for example  $D$  might be assessed on a three-point scale: high, medium or low.

The second factor to assess for each threat is the Gain,  $G$ , which an attacker could derive from making a threat a reality. Of course the first step to doing this is to identify what types of attacker might have something to gain. Once again, this assessment could be made in hard cash (say if the most likely attackers are credit card fraudsters or an unscrupulous business competitor) or in a less quantitative way (say if the attackers are teenage hackers, motivated only by perverse self-satisfaction or kudos amongst their 'peers').

### 3.4 Assess the Vulnerabilities

A vulnerability is a property of a system component, which might be exploited to mount an attack. It has nothing to do with what that component might be used for in any particular system, at any particular time. As with threats, the 'CIA' classification is used, in this case to characterise the security properties of each component.

For each vulnerability, there are two properties to be considered. Firstly, the Cost,  $C$ , to an attacker of exploiting the vulnerability must be assessed. In considering this, it is important not only to include the marginal cost to a particular person of mounting an attack, but also to gauge the total amount of resources which need to be deployed. For example, an attack against a smart card chip might require a scanning electron microscope such as might be found in a university electronics department. Although a student might be able to gain access to such a device for minimal personal outlay, the fact that specialist (and expensive) equipment is required serves to limit the opportunity for an attack.

Secondly, the Likelihood,  $L$ , that an attacker will be caught during or after an attack should be assessed. For example, an attack that can be mounted using web-mail from a cyber-café, is much less risky for the attacker than breaking into an office (or bribing an employee) to plant a line monitor.

### 3.5 Identifying and Assessing the Risks

At this point in any SRA we now have depressingly long lists of threats and vulnerabilities to consider, but how many of them actually result in real risk to the organisation?

A threat presents no risk if it can never be realised, and a vulnerability only presents a risk if it can be exploited in a way that exposes an asset to a threat. Restated, a real risk exists if an identified threat can be realised by a vulnerability.

For a system managing IE information entities utilising SC system components, we have potentially  $3 \times IE \times SC$  risks. For most systems, not all components handle all entities, so the cross-reference described above can be used to reduce the number of combinations to be considered.

For each risk, we wish to identify the degree to which the organisation is exposed, known as Exposure ( $E$ ). This we take to be given by the Damage ( $D$ ) done by any single incident, multiplied by the Probability ( $P$ ) of an incident taking place in a given period:

$$E = D \otimes P \quad (1)$$

We use the symbol  $\otimes$  to denote a function that behaves like multiplication, to allow semi-quantitative approaches to be formulated. For example, if  $D$  or  $P$  is zero, then so will  $E$ . We will see this in the worked example presented below. Clearly, straightforward multiplication can be used when a monetary calculation is being performed.

We will already have assigned a value for  $D$ ; as we have seen it is a property of the threat associated with the risk. How do we assess  $P$ ? In SRA, we make the assumption that the probability of an attack being carried out is proportional to the attacker's expectation of gain. This we define to be the Profit ( $Pr$ ) he might derive from a particular attack multiplied by the probability that he is not caught ( $PNC$ ).

$$P = Pr \otimes PNC \quad (2)$$

$Pr$  is calculated from the attacker's gain ( $G$ ) and his costs ( $C$ ). Using the same symbolic convention as before:

$$Pr = G \Xi C \quad (3)$$

where the symbol  $\Xi$  is used to denote an operator with similar properties to 'minus' (and which is 'minus' in the fully quantitative case). Notice that  $G$  and  $C$  are already known: they are properties of the threat and vulnerability respectively.

Of course  $PNC$  is just the inverse of the likelihood that the attacker will be caught:

$$PNC = 1 \Xi L \quad (4)$$

Notice that  $L$  is defined as a property of the vulnerability. We have now defined the exposure to each risk in terms of basic properties of the threat (damage to organisation and gain to the attacker) and vulnerability (cost to the attacker and his probability of being caught). When applied to every risk, we can generate an ordered list of information security risks. When the analysis is fully quantitative, the exposure to each risk can be compared to other types of risks run by the organisation.

### 3.6 Countermeasures

For the greatest risks, the organisation will wish to select countermeasures to reduce its exposure. Countermeasures can be technical (for example, the use of encryption to protect a communications link) or procedural (for example, the implementation of dual controls) or physical (for example better locks on doors). A countermeasure usually acts to reduce a vulnerability: either by increasing the attacker's cost or increas-

ing the chance that he is caught. Given new estimates for these characteristics, the new (reduced) exposures can be calculated. In the fully quantitative case, it can be checked that the selected countermeasures fulfil the organisation's standard 'rate of return' criteria. Where no suitable countermeasure exists, insurance can be considered. Insurance is generally specific to a threat rather than a vulnerability; but the underwriters will of course be interested to know how each threat might be realised. In any case, we believe that presentation of a thorough SRA will help underwriters to provide competitive premiums.

## 4 Worked Example

The following is a worked example of the application of SRA and is provided for illustrative purposes only. In this example, a hypothetical service whereby an organisation offers its customers up-to-the-minute company share price and business news via satellite broadcast.

No information service serves any other purpose than to make information which is both accurate and timely, available to legitimate users only. If the information is not accurate or timely, or is made available to people with no right to access the information, then the system has been in some sense subverted.

Therefore, to understand the threats which apply to the service, it is essential to have a model of exactly what information the service is designed to maintain. A Logical Data Model contains the following elements:

- Entities, usually a 'real-world' item, which can be distinguished from other entities of the same type, for example 'person'.
- Attributes, which are a property of an entity, for example 'eye colour' or 'height'
- Relationships, which are links between entities.

In practice, it is the one-to-many relationships that are most important to data modelling. A one-to-one relationship is trivial, and indicates that the two entities can really be considered as one. Many-to-many relationships are complex and should be analysed further. They can always be resolved into two one-to-many relationships via a 'link entity'. Figure 2 and Table 1 identify the Information Entities used in our example, the relationships between them and their key attributes.

For the purposes of this paper an extremely simple model is used, one that assumes that all 'Customers' receive the same generic service and get data relating to all listed 'Companies'. Also it has been assumed that the 'Price Report' and 'News Report' entities are so similar in terms of the types of threat they may be exposed to and how they are physically processed, that they are considered as one data entity, 'Company Report'.

The Service model is used to produce a Threat Catalogue identifying each threat, the damage its realisation would cause the organisation, and the maximum gain that any attacker could achieve. In some cases it is important to fully quantify the damage and gain as part of a budget allocation process. In this worked example a semi-quantitative assessment of the impact of a realised threat is used, based on a three-point scale of 'High', 'Medium' and 'Low'.



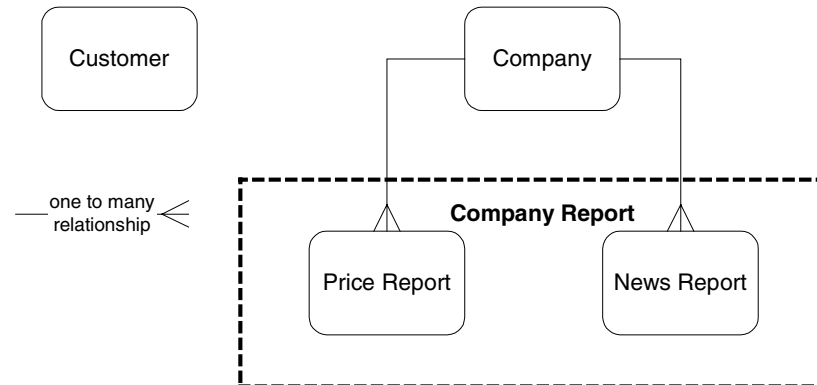


Fig. 2. Logical Data Model of Service

Table 1. Information Entities

Entity	Major Attributes
Customer	Customer ID Customer Name
Company	Company ID Company Name
Company Report	Company ID Date Time Price/News Item

This worked example is merely an illustration of how SRA is performed, and not a definitive statement of risk in such a system. Indeed it may be that the reader disagrees with the values allocated in this example. If this is the case, the value of such a structured approach becomes all the more clear, as it can be seen how simple assessments can be altered, and the effect rippled through the whole process to show the ultimate impact a small change has on the exposure of the organisation to risk.

Figure 3 shows the processing nodes, network links and data storage devices used within our example system. Each of these physical entities is considered in turn to produce a vulnerabilities catalogue such as that shown in Table 3. For the purposes of this simple example, the News Wire/Price Feeds are considered to have the same vulnerabilities, as are the Price Server, News Server and Broadcast Engine. Also the Satellite itself and broadcast receivers are not considered within this example, however, depending upon the implementation (e.g. is the broadcast receiver provided to the customer as part of the service?) these may require further attention if this were a real SRA.

Table 2. Threat Catalogue

Information Entity	Threat Type	Damage ( <i>D</i> )	Gain ( <i>G</i> )	Likely Attacker
Customer	Confidentiality	Medium	Medium	Business Rival
Customer	Integrity	Medium	Low	Hacker
Customer	Availability	Medium	Low	Business Rival
Company	Confidentiality	Low	Low	Hacker
Company	Integrity	High	Low	Business Rival
Company	Availability	Medium	Low	Hacker
Company Report	Confidentiality	Medium	Medium	Business Rival
Company Report	Integrity	High	High	Market Manipulator
Company Report	Availability	High	Low	Hacker

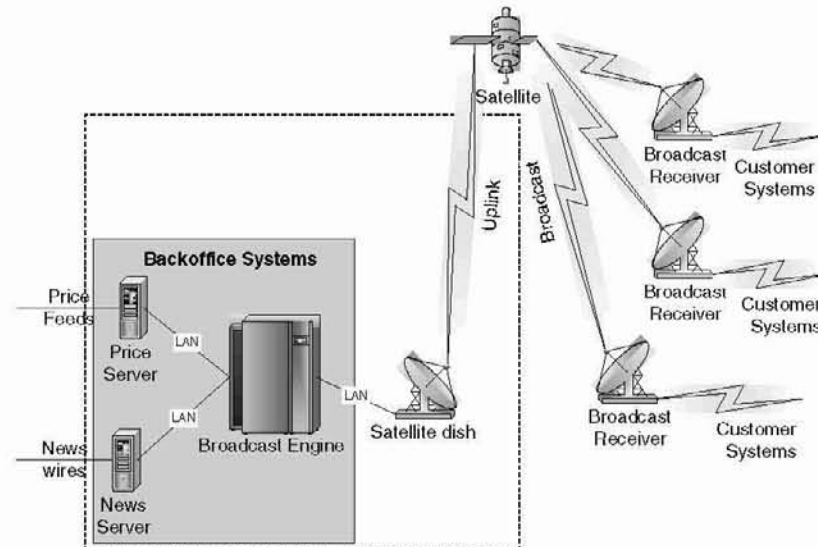


Fig. 3. Physical Model

Vulnerabilities are associated with the data links and processing nodes defined in the physical model. In both cases, they are categorised to correspond with the three different classes of generic threat—to confidentiality, integrity or availability.

For each vulnerability, the cost to the attacker of exploiting the vulnerability, and his probability of being caught are estimated. As with the damage and gain parameters associated with threats, these are given on a three point scale—high, medium and low.

For cost, these have the following definitions:

- HIGH – requires the resources of medium or large corporation, or organised crime syndicate

- MEDIUM – requires the resources of a small corporation or criminal gang
- LOW – within the resources of an individual hacker.

For the probability of the attacker being caught, the following definitions apply:

- HIGH – 0.5 to 1.0
- MEDIUM – 0.2 to 0.5
- LOW 0 – to 0.2.

In this example, it is assumed that no special security precautions are taken. Appropriate precautions will be derived following the risk analysis, when considering economic counter measures.

**Table 3.** Vulnerabilities catalogue
















Physical Entity	Vulnerability	Cost to attacker	Likelihood of capture
Price Feed/News Wire	Confidentiality	Medium	Low
Price Feed/News Wire	Integrity	Medium	Low
Price Feed/News Wire	Availability	Low	Low
Back Office Systems	Confidentiality	Medium	Medium
Back Office Systems	Integrity	Medium	Medium
Back Office Systems	Availability	Medium	Low
LAN	Confidentiality	Medium	Low
LAN	Integrity	Medium	Low
LAN	Availability	Low	Low
Satellite Dish	Confidentiality	Medium	Low
Satellite Dish	Integrity	Medium	Low
Satellite Dish	Availability	Low	Medium
Uplink/Broadcast Channel	Confidentiality	Medium	Low
Uplink/Broadcast Channel	Integrity	Medium	Low
Uplink/Broadcast Channel	Availability	Medium	Low

Once Threats and Vulnerabilities have been catalogued, the risk analysis finds areas where vulnerabilities of the system can be exploited to realise a business threat, and ranks each such risk in terms of *exposure*.

As a precursor to this analysis, a cross-reference (e.g. Table 4) needs to be built between the logical and physical models defined earlier. This cross-reference shows which information entities are handled (i.e. stored, processed or transmitted) by which system components.

For each valid combination of entity and component, risks are identified (classified, as usual, by confidentiality, integrity and availability) from the corresponding threats and vulnerabilities. The severity of each risk is calculated as follows:

**Table 4.** Service/System model cross reference

CROSS REFERENCE		INFORMATION ENTITIES		
		Customer	Company	Company Re- port
SYSTEM COMPONENTS	Price Feed/ News Wire			
	Back Office Systems			
	LAN			
	Satellite Dish			
	Uplink/ Broadcast Channel			

Firstly calculate the ‘profit’ to an attacker ( $Pr$ ) as defined in Equation (3) previously. In a semi-quantitative analysis, such as this example,  $Pr$  can be calculated using a lookup table:

**Table 5.** Attacker Profit lookup table

ATTACKER PROFIT		ATTACKER GAIN		
		HIGH	MEDIUM	LOW
ATTACKER COST	HIGH	Low	Negligible	Negligible
	MEDIUM	Medium	Low	Negligible
	LOW	High	Medium	Low

This amounts to subtraction of the cost from the gain to calculate the profit.

Next Assess the *probability that an attack will take place* ( $P$ ), as defined in Equations 2 and 4.

This amounts to multiplying the attacker profit by the likelihood that he does *not* get caught in committing the attack.

This then enables us to construct a Risk Catalogue, where each valid combination of entities and components on the model cross-reference (Table 5) is assessed once again under the three categories of Confidentiality, Integrity and Availability.

**Table 6.** Attack Probability lookup table

ATTACK PROBABILITY		ATTACKER PROFIT		
		HIGH	MEDIUM	LOW
DETECTION PROBABILITY	HIGH	Low	Negligible	Negligible
	MEDIUM	Medium	Low	Negligible
	LOW	High	Medium	Low

**Table 7.** Risk Catalogue

Entity	Component	Attacker Profit ( <i>Pr</i> )	Attack Probability ( <i>P</i> )	Damage ( <i>D</i> )
CONFIDENTIALITY				
Customer	Back Office Systems	Low	Negligible	Medium
Customer	LAN	Low	Low	Medium
Customer	Satellite Dish	Low	Low	Medium
Customer	Uplink/ Broadcast Channel	Low	Low	Medium
Company	Price Feed/News Wire	Negligible	Negligible	Low
Company	Back Office Systems	Negligible	Negligible	Low
Company	LAN	Negligible	Negligible	Low
Company	Satellite Dish	Negligible	Negligible	Low
Company	Uplink/Broadcast Channel	Negligible	Negligible	Low
Company Report	Price Feed/News Wire	Low	Low	Medium
Company Report	Back Office Systems	Low	Negligible	Medium
Company Report	LAN	Low	Low	Medium
Company Report	Satellite Dish	Low	Low	Medium
Company Report	Uplink/Broadcast Channel	Low	Low	Medium
INTEGRITY				
Customer	Back Office Systems	Negligible	Negligible	Medium
Customer	LAN	Negligible	Negligible	Medium
Customer	Satellite Dish	Negligible	Negligible	Medium
Customer	Uplink/Broadcast Channel	Negligible	Negligible	Medium
Company	Price Feed/News Wire	Negligible	Negligible	High
Company	Back Office Systems	Negligible	Negligible	High
Company	LAN	Negligible	Negligible	High
Company	Satellite Dish	Negligible	Negligible	High
Company	Uplink/Broadcast Channel	Negligible	Negligible	High
Company Report	Price Feed/News Wire	Medium	Medium	High

Entity	Component	Attacker Profit ( $Pr$ )	Attack Probability ( $P$ )	Damage ( $D$ )
Company Report	Back Office Systems	Medium	Low	High
Company Report	LAN	Medium	Medium	High
Company Report	Satellite Dish	Medium	Medium	High
Company Report	Uplink/Broadcast Channel	Medium	Medium	High
<b>AVAILABILITY</b>				
Customer	Back Office Systems	Negligible	Negligible	Medium
Customer	LAN	Low	Low	Medium
Customer	Satellite Dish	Low	Negligible	Medium
Customer	Uplink/Broadcast Channel	Negligible	Negligible	Medium
Company	Price Feed/News Wire	Low	Low	Medium
Company	Back Office Systems	Negligible	Negligible	Medium
Company	LAN	Low	Low	Medium
Company	Satellite Dish	Low	Negligible	Medium
Company	Uplink/Broadcast Channel	Negligible	Negligible	Medium
Company Report	Price Feed/News Wire	Low	Low	High
Company Report	Back Office Systems	Negligible	Negligible	High
Company Report	LAN	Low	Low	High
Company Report	Satellite Dish	Low	Negligible	High
Company Report	Uplink/Broadcast Channel	Negligible	Negligible	High

Finally exposure to the risks is calculated by combining the Damage ( $D$ ) (a function of the threat) with the attack probability:

In essence, the Damage is multiplied by the Probability of attack to generate the Exposure. It is exposure that is the key to aiding Organisations in making business decisions as to how they can make their systems economically secure. Table 6 indicates the Risks that open up our example service to medium or high exposure. The interesting point of note from this process is that the confidentiality of the data is relatively unimportant, as the kind of customer that requires this service is unlikely to go to the trouble of tapping it for free, as they can well afford the fees charged considering the value of the information to them. The integrity of the information provided about companies to customers is shown to be vitally important, as any actions taken on incorrect information from this trusted source is bound to have serious repercussions, both in monetary terms and in terms of reputation. In addition, false information inserted by an attacker could be used to create a false market, which is likely to lead to recriminations from financial regulatory bodies.

## 5 Conclusions

Before describing the SRA method we listed the requirements that such a methodology should meet to be truly valuable as a risk assessment tool. Now let us consider how SRA meets these requirements.

**Table 8.** Exposure lookup table

EXPOSURE		DAMAGE ( <i>D</i> )		
		HIGH	MEDIUM	LOW
PROBABILITY OF ATTACK	HIGH	Extreme	High	Medium
	MEDIUM	High	Medium	Low
	LOW	Medium	Low	Slight

**Table 9.** Greatest Exposure

Entity	Component	Risk Type	Exposure
Company Report	Price Feed/News Wire	Integrity	High
Company Report	Back Office Systems	Integrity	Medium
Company Report	LAN	Integrity	High
Company Report	Satellite Dish	Integrity	High
Company Report	Uplink/Broadcast channel	Integrity	High
Company Report	Price Feed/News Wire	Availability	Medium
Company Report	LAN	Availability	Medium

- *Business context.* By providing the means for quantifying the exposure a particular risk opens an organisation up to, SRA enables business owners to make informed decisions as to whether the benefits of security measures justify their expense.
- *Technical grounding.* SRA uses existing systems analysis techniques to ensure the information service, and the physical system via which it is delivered are completely and openly defined.
- *Separation of concerns.* By separately considering the logical service to identify threats, and the physical architecture to identify vulnerabilities, SRA enables technical and business issues to be considered clearly and separately before the results are cross-referenced to identify where they combine to present a risk.
- *Support for quantitative analysis.* Depending on the information available, the time for analysis and the complexity of the system, SRA enables empirical estimations of cost in order to allow determination of an overall information security budget and the optimal allocation of that budget.
- *'Tuneable' analysis.* Implementing SRA need not be a massive undertaking. Depending on corporate size, circumstances and budget, performing such an analysis, even on simple high-level models of system and service, will provide some shape and direction to security expenditure and focus.

- *Evolution.* As the foundation of the method is based upon best practice in the field of systems analysis it is possible for SRA use the most appropriate techniques to model the logical service and the physical system, as such practice evolves and improves. Also because the method is not based on static logic (e.g. in a proprietary software package) but relies on the real knowledge and experience of real people, this enables dynamic response to advances in IT in general and IT security in particular.
- *Maintainability.* One of the benefits of performing a Structured Risk Analysis in this way is that when services and systems change, it is possible to tweak the models and see whether simple changes to information models, or physical architecture, effect the exposure to risk faced by the organisation.
- *Openness.* SRA is based on a unique combination of open standards in the area of systems analysis and security, requiring no payment of licence fees or purchase of specific software.

Security controls, and the actual dangers presented by the risks these controls are introduced to protect against, are fundamental to the economic success of an IT system. It is common sense, therefore, to give the deployment of such security controls the same sort of rigorous analysis and consideration as other areas of systems analysis and design. Such a statement is hardly controversial, however, the fact remains that quantifying risk is not easy, and is, therefore, often put aside in favour of more easily defined and controlled economic factors, such as development cost and revenue return.

SRA provides an approach that allow the same disciplines of structure and objective observation that are well established in other areas of systems analysis and design to be applied to security, thus placing security costs and savings on a level playing field with other factors that can effect the ongoing success of an information service.

As indicated in the introduction of this paper, we believe the key to taking optimal decisions in all these areas is to understand the risks to which the organisation is exposed by doing nothing, and the degree to which these could be mitigated by the deployment of particular countermeasures.

We believe SRA gives business owners the information they need to make truly informed decisions about the security, viability and future direction of their Information Systems.

## References

1. Rogers, James, Simons, Mike.: Revenue's security glitch a warning for e-government, Computer Weekly Magazine, Thursday 6th June 2002.
2. Weaver, Philip L, Lambrou, Nicholas, Walkley, Matthew.: Practical SSADM Version 4+, Financial Times Pitman Publishing, ISBN 0-273-62675-2, 2<sup>nd</sup> Edition, 1998.
3. Yourdon, Inc.: Yourdon Systems Method: Model-Driven Systems Development, Prentice Hall, ASIN 013045162, 1<sup>st</sup> edition (March 4, 1993)
4. Yourdon, Edward, Coad, Peter.: Object-Orientated Analysis, Yourdon Press, ISBN 0-13-629981-4, 2<sup>nd</sup> Edition, 1991



# A Model Enabling Law Compliant Privacy Protection through the Selection and Evaluation of Appropriate Security Controls

E.S. Siougle and V.C. Zorkadis

Hellenic Data Protection Authority, 8 Omirou St., P.C. 10564, Athens, Greece  
{sefrosini,zorkadis}@dpa.gr, [www.dpa.gr](http://www.dpa.gr)

**Abstract.** The broad adoption and increasing reliance on computing and communication systems in applications domains such as health services, insurance, telecommunication and direct marketing leads to the creation, collection and processing of enormous amounts of personal data. Responding to this development, international bodies, the European Union and various countries established personal data protection laws and Authorities to regulate and control their application. The legal framework imposes the taking of appropriate security measures, that may be different compared with those specified by data controllers based on their business needs, since personal data are assets with, possibly, different values for the data subjects and the controllers. In this paper, we propose a security controls selection model, that supports data controllers in their effort to methodologically choose security measures compliant to privacy protection laws being in force. Also, we propose a process to assess (methodologically) the privacy protection requirements according to the related legal provisions and the selected and implemented security controls.

## 1 Introduction

Emerging computer and communications technologies have radically altered the ways in which the communication and exchange of personal data is performed. These technologies offer many advantages, such as speed or efficiency, but there also arise many questions related to the protection of personal information traversing the global communications infrastructure. The transmission of vast quantities of personal data within seconds across national frontiers and indeed across continents in combination with the threats they are exposed to, such as traffic analysis and user profile creation has made it necessary to consider privacy protection.

Advances in information technology and the Internet have changed the way that companies conduct business. Over the past decade, there has been unprecedented growth in the ability of organizations to create, collect, analyze, combine and disseminate personal data. The telecommunication and mobile communication industry, insurance, direct marketing and health companies are processing an ever-increasing volume of personal data. Computers store and exchange enormous amounts of personal information, such as medical data and financial data.

Data subjects (users, consumers, etc) expect their personal data to be protected and their privacy to be respected by the organizations (data controllers) they conduct business with. On the other hand, data controllers have realized that protecting personal data and developing data subject's trust promise to become a competitive advantage. Thus, protecting privacy has become a new business imperative.

Privacy protection laws were enacted in many countries in the last three decades. They have been introduced to regulate the processing of personal data and to prevent what are considered to be privacy violations, such as unlawful storage or storage of inaccurate personal data and abuse or unauthorized disclosure of personal data. In addition to national data protection laws, several legal instruments related to privacy protection have been adopted at an international level. Among the most influential are the European Union's Directives 95/46/EC and 97/66/EC, the Council of Europe's Convention of the Protection of individuals with regard to Automatic Processing of Personal Data, and the OECD's Guidelines Governing the Protection of Privacy and Transborder Flows of Personal Data [3,4].

Therefore, data controllers are obliged to respond to privacy protection requirements imposed by the above-mentioned legislation. Specifically, these laws impose obligations to data controllers, which relate to the secure processing of personal data through the selection of the appropriate set of security controls, so that the desired protection of personal data is achieved.

The aim of this paper is twofold. The first objective is to propose a privacy protection model to support data controllers in their effort to respond to the requirements of the laws. This will be achieved through the selection of an appropriate set of security controls that will meet both the legal privacy protection requirements and any particular requirements of the data controllers regarding the processing of personal data. The second objective of the paper is the evaluation and certification of the selected security controls in order to enhance confidence that an acceptable level of privacy protection has been achieved. If the security controls are evaluated as meeting the privacy protection requirements imposed by the legislation then trust is established that the data controller has implemented privacy in this rapidly changing, information-driven economy. Thus, the proposed privacy protection and security evaluation model aims at instilling confidence and assurance that an appropriate set of security controls has been identified and implemented for the protection and secure processing of personal data.

The selection and evaluation processes reflect the auditing practices followed by the Hellenic Data Protection Authority regarding the compliance of data controllers to Greek and European legal framework. The models proposed are concerned with security issues from the perspective of privacy protection. They can be integrated in the overall approach of a data controller for the formulation of a complete security plan.

The paper is organized as follows. In the second section, the privacy protection requirements imposed by the legal framework are analyzed. The third section is devoted specifically to the analysis of the requirements of the security principle. In the fourth section the security controls selection model is presented. In the fifth section the evaluation model is briefly presented. Finally, the paper is concluded.

## 2 Privacy Protection Requirements

According to European data protection legislation, every organization has an explicit obligation towards the protection of personal data. Data controllers processing personal data have to demonstrate that they have implemented an effective approach with respect to their own IT security arrangements as far as the secure processing of personal data is concerned.

The data protection laws are based on the principles of lawfulness and fairness, minimality, purpose specification, accuracy, anonymity, security, individual participation and accountability. These principles are analyzed as follows:

- *Principle of lawfulness and fairness:* personal data should be gathered by fair and lawful means. Unfair or unlawful collection may result in business loss and deceptive business practices.
- *Principle of minimality:* the amount of personal data collected should be adequate, relevant and not excessive in relation to the purposes for which they are collected and/or further processed. The collection of personal information should be limited only to those that are necessary for the fulfillment of the purposes identified by the data controller. A connection should be established between the personal data collected and the purposes identified for collecting information.
- *Principle of purpose specification:* personal data should be collected for specified, explicit and legitimate purposes and not be further processed in ways incompatible with those purposes. The identification of the purposes for which personal data is needed is a critical step in defining the kinds of personal data that should be collected and processed for the fulfillment of the activities of the organization. The data processing purposes should be defined according to the context of the business and the reasons for collecting information should be well documented and legitimate. Failure of a data controller to identify the purposes for collecting personal data will make it difficult to adequately manage the effective and lawful processing of personal data.
- *Principle of accuracy:* personal data should be accurate and up to date. Inaccurate or incomplete data should be erased or rectified. The identification of the degree of accuracy, completeness and timeliness of data should be considered according to the requirements of the identified data processing purposes.
- *Principle of anonymity:* personal data should be preserved in a form, which permits identification of the data subjects for no longer than is required for the purpose for which those data are stored. This principle imposes responsibility on data controllers to retain personal information only as long as it is needed for the fulfillment of the identified purposes.
- *Principle of security:* appropriate security measures, technical and organizational, should be taken to protect personal data from unintended or unauthorized disclosure, destruction or modification.
- *Principle of individual participation:* data subjects should be informed of the information held on them by data controllers, should be given the right to access and the right to rectify, erase or block this information concerning them, if it is inaccurate or misleading. This principle allows users to exercise control over their personal data and be able to correct inaccurate or incomplete information.

- *Principle of accountability*: entities responsible for processing of personal data (data controllers) should be accountable for complying with the above principles. This principle focuses on identifying and assigning responsibility for compliance. Appropriate accountability will ensure effective implementation, policy development, adherence, evaluation, and refinement of privacy protection throughout the organization. Privacy policies and practices need to apply to all personal data in the custody of the data controller.

The above principles constitute the privacy protection requirements of the legislation and define a regulatory framework for the protection of personal data. Data controllers must comply with the privacy protection requirements and implement privacy protection based on them.

### 3 Security Principle Requirements

The main security obligations are outlined in the security principle. This principle clearly defines that appropriate security measures, both technical and organizational, should be taken by the data controller for the protection of personal data against accidental or unauthorized destruction or accidental loss as well as against unauthorized access, alteration or dissemination.

According to data protection laws based on the Directive 95/46/EC [3], the obligations of the data controller regarding the secure processing of personal data are:

1. Establishment of the appropriate security standards and procedures. Without appropriate security measures, unauthorized parties (both within and outside an organization) may be able to access, use, copy, disclose, alter and destroy the personal data of the data controller. Such action could create significant harm to the individual to whom the data relates, as well as potential liability for the data controller. Without appropriate access controls mechanisms, unauthorized individuals may access personal data for unauthorized purposes. Without appropriate audit trails for access to personal data, security breaches may not be detected and remedied. Furthermore, data controllers are obliged to periodically assess the security of their information and communication infrastructure considering the security technology evolution. The establishment of organizational measures is related to security management and to allocation of resources to security, the identification of roles and responsibilities, the establishment of rules ensuring compliance with the security procedures.
2. Selection of personnel based on their skills and ethics and the provision of appropriate training in security issues: there is a requirement for computer security awareness training for personnel at all levels throughout the organization. Training should focus into making them aware of the security and privacy protection requirements and to prevent them from making unnecessary errors and to make them aware of their responsibilities for security. On going educational, awareness and training programs should be put in place within the organization. A third factor affecting personal reliability indicates the establishment of a work environment, which meets health and safety standards.
3. Management of outsourcing contracts and the selection of a processor according to the technical and organizational security measures governing the processing. The security and organizational measures that should be taken by the processor

correspond to those implied by the security principle, and are equivalent to those imposed to data controller. Furthermore, the processor is contractually obliged to process personal data only on instructions of the controller and to take the required staff-related precautions. The parts of the contract or the legal act relating to data protection and the requirements relating to the technical and organizational measures should be in writing or in another equivalent form.

4. Whenever personal data are transferred outside Europe, the data controller must consider the security measures taken and the legislation concerning data protection in the country or territory where the data are transferred.

Consequently, the selection of the appropriate set of security controls meeting the privacy protection requirements imposed by the data protection legislation is imperative for the protection of personal data. The appropriate security measures must be selected and implemented so that an acceptable level of privacy is achieved.

## 4 Security Controls Selection Process

The selection of an appropriate set of security controls to provide an adequate level of privacy protection is of major importance and difficulty. It may be accomplished through the exercise of a risk assessment strategy. Following a risk assessment exercise, security controls are identified to reduce risks to an acceptable level, based on the value given to the assets of an organization.

However, in the case of the protection of personal data, the value of personal data and the risks associated with them are appraised differently according to the relation of personal data to the party valuing them. Users, usually, place greater value to their personal data than the data controllers, who are, probably, more interested in processing personal data for conducting business. Users are not willing to overlook the failure of a data controller to protect their privacy.

Therefore, a risk assessment exercise should be performed in case the objective of an organization is unique and specific issues and risks need to be addressed. Also, a number of baseline control manuals are available that can be used for the selection of an appropriate set of security controls. For example the ISO security standard (ISO 17799) constitutes a comprehensive reference document for identifying a set of security controls that will meet the security requirements of the majority of organizations across all functional domains. Furthermore, the Common Criteria (CC) standard, which addresses protection of information from unauthorized disclosure, modification or loss of use and is, also, applicable to non-human threats, can be used for the selection of the appropriate security controls. Additionally, sample security plans can be used for this process.

Baseline security supports the selection of a set of security controls, which will provide good protection against most threats and under most circumstances. However, baseline manuals provide little guidance on how to determine the set of controls to provide adequate security for the particular business situation or according to legal, regulatory requirements.

The security principle of the data protection law states that the data controllers so as to protect the personal data in their custody should adopt appropriate security measures, technical and organizational. Also, the security of their information and

communication infrastructure should be accessed considering the security technology evolution and the business dependency on information technology.

Therefore, we propose a model for the selection of the most appropriate set of security controls that will satisfy the privacy protection requirements regarding the secure processing of personal data. Baseline manuals, the CC standard, sample security plans, possibly, in combination with the findings of risk assessment exercises, support the selection of the security controls

Our proposed security controls selection model is depicted in Figure 1.

The principle of purpose specification serves as the axis of the selection model. Thus, the first step of the model is to identify the purposes for which the data controller collects and processes personal data, according to the range of activities of the organization. The result of this step identifies the data processing purposes of the data controller.

According to the principle of purpose specification, personal data should be collected and processed for legitimate purposes only. However, non-legitimate purposes may be identified at this step of the model. In such case the non-legitimate purposes are removed and the steps of the model are repeated for each legitimate data purpose of processing. At this point it is worth mentioning that the purposes of collecting and processing personal data for security reasons, such as auditing and logging, are legitimate. Such data processing purposes should, also, meet the privacy protection requirements of the legislation.

In the next step, the personal data or categories of personal data, whose processing is necessary for the fulfillment of the identified data processing purposes, are defined. The identification of the personal data corresponding to each data processing purpose is based on the application of the principles of lawfulness and fairness and the principle of minimality.

In the following step, the protection degree of personal data corresponding to each data processing purpose is identified based on the privacy protection requirements imposed by the data protection legislation and any particular privacy requirements deriving from the activities of the organization. For each purpose specification and the corresponding data the privacy protection principles accuracy, anonymity, individual participation and accountability are applied so that a specific (concrete) privacy protection requirements set is defined. The concrete privacy protection requirements set includes the identification of the law compliant operations that can be performed on personal data for the fulfillment of the specific data purposing purpose, the sources (origins) for collecting personal data and the recipients to whom the personal data are disclosed and communicated. Furthermore, the identification of the time period during which it is intended to conduct the data processing operations on the personal data, the interconnections of the personal data, the staff members that are responsible for the processing of personal data and accountable for compliance with the established privacy principles.

The next step involves the application of the security principle for the selection of the appropriate security controls from one or a combination of baseline manuals, the CC standard, sample security plans and the findings of risk assessment exercises. The security controls are selected according to the concrete privacy protection requirements. The security principle is specifically applied in this step so that the security controls selected are the appropriate ones to assure the law provisions regarding data disclosure, destruction and modification, both in technical and organizational domain. The computational and telecommunication infrastructure of

the organization is taken into consideration in this step so that the security controls selected are in accordance with the existing infrastructure, the procedures in operation and the possible security-related technological solutions.

The technical control requirements of the security principle range from authentication, integrity, access control mechanisms, confidentiality, accountability and digital signature creation and verification to privacy-enhancing requirements such as anonymity, pseudonymity, unlinkability and unobservability. To organizational control requirements belong among others security and privacy protection planning and strategy, security and privacy policy creation and maintenance and disaster recovery and business continuity planning.

Additionally, the selection of the appropriate security controls is based on the requirements regarding staff selection and security and privacy related training and the possible outsourcing contracts, in case the data controller involves third entities for the collection and processing of personal data. Finally, possible transfers of personal data outside European Union should be considered with respect to the security measures taken and the legislation concerning data protection in the country or territory where the data are transferred.

The security controls selected along with the privacy protection requirements determine the privacy protection profile corresponding to each data processing purpose. Thus, the privacy protection profile defines a set of appropriate privacy and security requirements and objectives. The privacy protection profiles should guide the eventual product selection and evaluation. The model steps are repeated so that a privacy protection profile is determined for all the processing purposes of the data controller.

Finally, the last step refers to a periodical monitoring of the security publications and vulnerability analysis and generally of technological solutions and advances that should be considered so that privacy protection profiles and implementations are adjusted properly.

The following example will contribute to the clarification of the model. The data processing purposes of a hospital may include, among others, the provision of health services, the administration of personnel and the notification of infectious diseases to the authorities responsible for monitoring such diseases. Each of these purposes has certain privacy protection requirements regarding the secure processing of personal data, which dictate the adoption of different security controls. To the personal data for the purpose of administration less strict security controls may be applied in comparison to the personal data for the purpose of providing health services. Furthermore, to the personal data of the third purpose the control of anonymity should be applied when transmitted to the authorities monitoring infection diseases.

Therefore, the data processing purposes have different privacy and security requirements even if some of them are processing the same personal data. Thus, the privacy protection profile corresponding to a processing purpose is determined based on the privacy protection principles of the legislation and the privacy requirements both of the business and the specific data processing purpose.

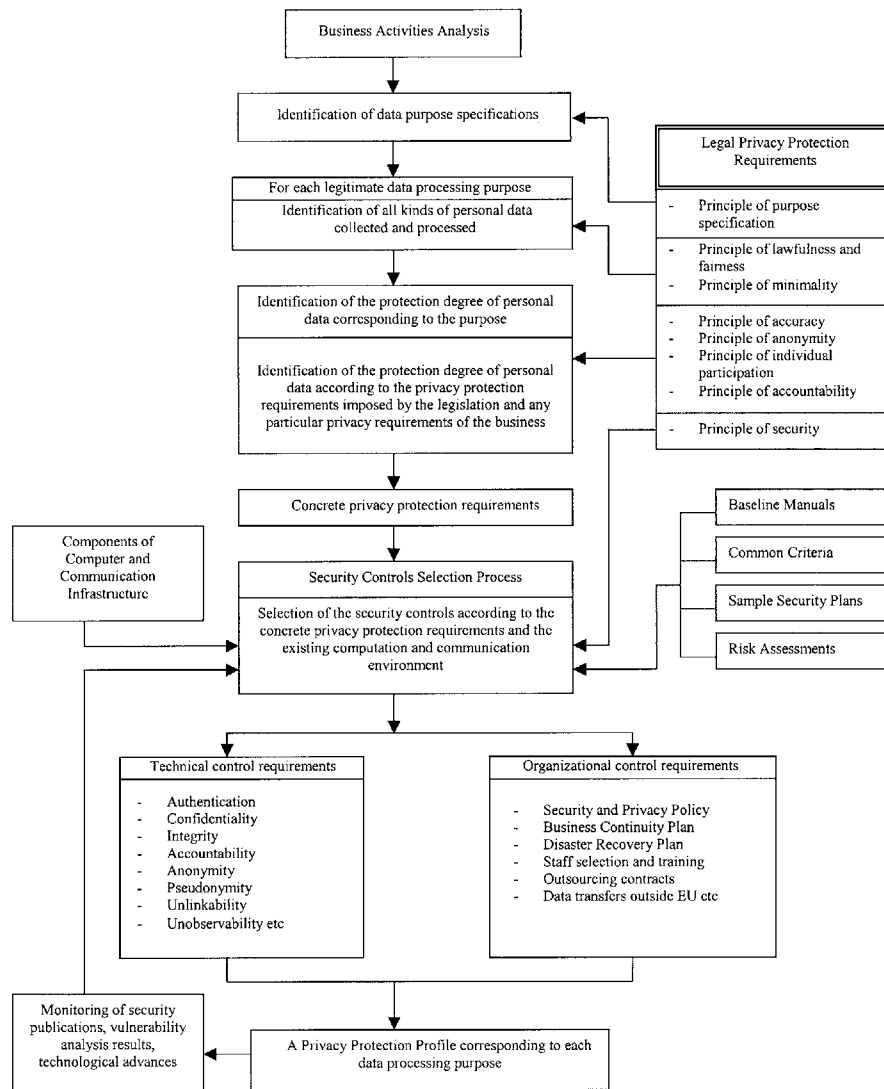


Fig. 1. Privacy Protection Model

## 5 Privacy and Security Evaluation Process

In order to gain confidence that the data controller has achieved the desired level of privacy protection, an evaluation of the security controls selected and implemented should be performed based on some security standard [5, 6]. The Common Criteria (CC) standard provides a flexible evaluation framework that can deal with security and privacy protection requirements. It provides a common set of requirements for the



security functions of products and systems and for assurance measures applied to them during a security evaluation. The evaluation process establishes a level of confidence that the security functions of products and systems and the assurance measures applied to them meet these requirements.

Thus, the evaluation of the selected and implemented security controls will take place against the CC standard in order to ensure that an appropriate set of security controls is identified, that the selected controls are indeed adequate to satisfy an acceptable level of security and privacy protection and that these controls are properly installed and operated. The evaluation of the privacy protection requirements will be based on the privacy protection principles discussed on the previous sections.

Our proposed privacy and security evaluation model is depicted in Figure 2. It is based on a two-stage evaluation process. The privacy protection profile is examined against both the privacy protection requirements and the selected security controls. The model steps are repeated for each data purpose of processing.

In the first step of the model the privacy protection level of each data processing purpose is examined as complying with the legal provisions of the privacy protection principles. The principles of purpose specification, lawfulness and fairness, minimality, accuracy, anonymity, individual participation and accountability are checked with respect to the collection and processing of the personal data corresponding to each data processing purpose.

In the next step of the model the selected and implemented security controls of the privacy protection profile are assessed with respect to CC-based security functional and security assurance requirements. The principle of security is checked regarding both the technical and the organizational control requirements of each data processing purpose.

In the final step of the evaluation process tests are carried out to ensure that the selected technical and organizational security controls are permanently in operation and that all operational procedures are adhered to. Furthermore, penetration tests should be used to determine whether the security controls implemented are adequate to protect from known vulnerabilities. [8]

The successful completion of the evaluation process ensures trust regarding the privacy protection of the data controller and may lead to a certification. In case the evaluation process fails, corrective actions should be taken according to the problem determined. The security and privacy policy and the privacy protection profiles may be revisited and the security controls selection model may be repeated.

## 6 Conclusion

In this paper, we proposed a security controls selection model, that aims at methodologically supporting data controllers to comply with privacy protection laws. This model reflects the security auditing practices followed by the Hellenic Data Protection Authority in its efforts to evaluate and probably approve the privacy protection degree achieved by data controllers. The model uses as an axis of the selection process the processing purpose dictated by the data controller's business activities. The data controller's practices regarding creation, collection, processing

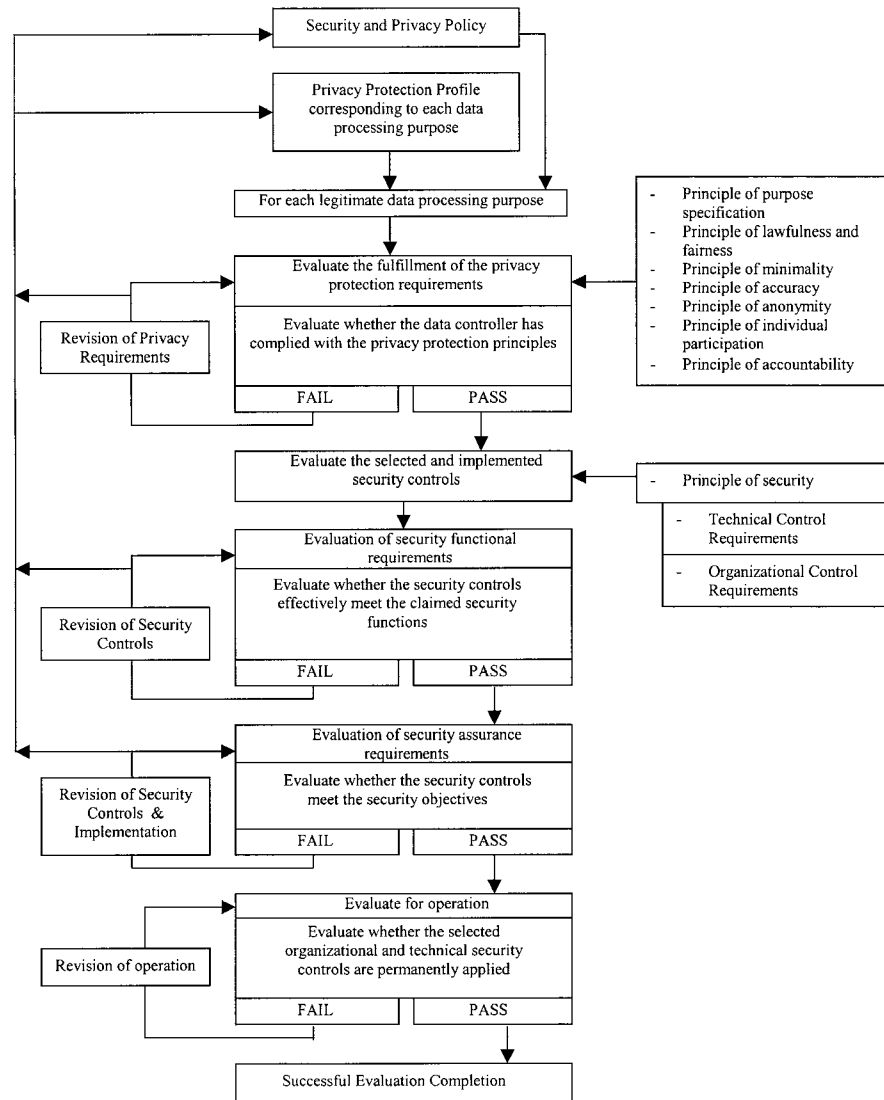


Fig. 2. Privacy and Security Evaluation Model

and communication of personal data are determined in the light of the privacy protection law principles lawfulness and fairness, minimality, accuracy, anonymity, security, individual participation and accountability. The concrete privacy protection requirements comprise the input to the next step of the methodology along with a security baseline manual, the computation and communication infrastructure and existing vulnerabilities and security solutions resulting to a detailed set of security controls (privacy protection profile) for each processing purpose. The final step of the model anticipates the periodical reviewing and modification of the protection profiles according to new vulnerability analysis results and technological security solutions.

Furthermore, we, also, proposed a security and privacy evaluation model, which takes as input the privacy protection profile that is the output of the final step of the security controls selection process. This model is repeated for all the data processing purposes of the data controller. It examines both the privacy protection requirements of the privacy protection profile and the selected and implemented security controls. Evaluation of the privacy protection level of personal data corresponding to the data processing purposes is based on the privacy protection principles of the legislation. Evaluation of the selected and implemented security controls is based on the Common Criteria standard regarding both the security functional and assurance related requirements. The final step of the model evaluates the selected security controls for operation. In case the evaluation is completed successfully trust is established concerning the level of privacy protection of the data controller, otherwise corrective actions should be taken.

## References

- [1] C. Pounder, 'Security And The New Data Protection Law', *J. Computers & Security*, 17 (1998), pp. 124–128.
- [2] Institute for Certification of Information Technology (ICIT), 'Scheme for self-assessment and certification of information security against BS 7799', 1997.
- [3] Directive 95/46/EC on the protection of individuals with regard to the processing of personal data and on the movement of such data and 97/66/EC on the protection of personal data in telecommunications, [www.eu.int/comm/.../media/dataprot/](http://www.eu.int/comm/.../media/dataprot/).
- [4] OECD, Guidelines Governing the Protection of Privacy and Transborder Flows of Personal Data, Paris, 1980.
- [5] Lynette Barnard and Prof. Rossuw von Solms, 'A Formalized Approach to the Effective Selection and Evaluation of Information Security Controls', *J. Computers & Security*, Vol. 19, No. 2, pp. 185–194, 2000.
- [6] V. C. Zorkadis, E. Siougle, Ph. Mitleton, 'Technical and Legal Reports on Evaluation of Security and Privacy Protection', Hellenic Data Protection Authority, 1999–2000.
- [7] V. Zorkadis, E. Siougle, 'Information Security and Privacy Audit Modeling', *Proc. of the 5<sup>th</sup> World Multiconference on Circuits, Systems, Communications and Computers*, Crete, July 2001
- [8] A. Jones, 'Penetration testing and system audit – Experience gained during the investigation the investigation of systems within the UK', *J. Computers & Security*, 16 (1997), pp. 595–602.

# Authentication and Authorization of Mobile Clients in Public Data Networks

Prakash Reddy, Venky Krishnan, Kan Zhang, and Devaraj Das

HP Labs  
1501 Page Mill Road, Palo Alto Ca USA

**Abstract.** We present a protocol that enables mobile clients to be authenticated and authorized in data networks that are deployed in public places otherwise referred to as hotspots! The three key elements of a hotspot network are the mobile client, the hotspot server and the service provider. A mobile client is any device that can be used to access the internet. The hotspot server is a node in the data network that is a bridge between wireless clients and wired broadband network. The service provider is an entity that has an existing service relationship with the client and the hotspot server. The protocol discussed in this paper shows how three parties: Client, hotspot server and the service provider come together in a mutually un-trusted environment, authenticate each other and upon authentication exchange authorization tokens that are used in subsequent service requests. The most common use of this protocol is for clients to gain internet connectivity in public places, specifically in hotspots. The hotspot server provides the equivalent of cellular network roaming functionality. The service provider allows added features to its clients.

## 1 Introduction

Mobile access to the web is rapidly growing in popularity. You see people accessing the web at airports, coffee shops, malls etc. Internet connections are being made available in more and more public places. There are several reasons for the sudden growth of wireless access to the web in public places. One of the reasons is the standardization of the wireless local area network (WLAN) around the 802.11b. This standard referred to as Wi-Fi operates at speeds up to 11 Mega Bits per second. As a result of this standardization there has been an explosion of wireless devices equipped with WLAN access cards. The cost of wireless networks is coming down and they are easier to set up than a wired network.

By year-end 2007, 60 percent of the population in the U.S. and Europe will carry wireless devices, according to a recent Gartner report. By 2010, the percentage will leap to 75 percent.

Even though the public wireless networks are seeing rapid growth, they are not in the mainstream yet. The public wireless networks are mushrooming along different lines

- An individual deploying their own access point and making it available for public use
- Companies trying to build their own infrastructure to provide access for fees
- A few other companies are trying to build a virtual network by aggregating

existing public wireless networks d) A true roaming model. If we examine the evolution of cellular networks, each carrier started out by building their own network and servicing their own customers. If the customer were to travel outside the providers network, the service was not available. The carriers soon realized that they could make their network appear wider and generate more revenue by setting up service level agreements (SLA) to allow customers to access each other's networks leading to the birth of roaming. The carriers had to figure out how to authenticate, authorize and bill each other's customers.

We see a similar model developing with perhaps one major difference. As the cost of deploying a public wireless network infrastructure is considerably smaller than when compared to deploying a cellular infrastructure, we anticipate lot more infrastructure providers in this space.

Given this model, we need an authentication and authorization (AA) model that can deal with the three parties that are involved in connecting a mobile node to the public wireless network.

Current schemes do not span trust boundaries, because WLAN/LAN infrastructure providers from unrelated enterprises have not established a mechanism for authentication and authorizing a mobile node. This paper defines an AA protocol which can be deployed in a public wireless networks.

In section 2 we will introduce the terminology and we will discuss the AA schemes currently being used. Section 3 will describe the network model and walk thru the steps of authentication and authorization and describe in detail the message sequence. We will walk thru the implementation details of the protocol in section 4. We will conclude by discussing the results and identifying future work.

## 2 Authentication and Authorization

Authentication and authorization are partly independent. Authentication is necessary before any authorization can happen. Authenticating clients in un-trusted domains requires adopting a scheme/protocol that will allow verifying the identity of the involved parties without revealing shared secrets or keys.

### 2.1 Definitions

Authentication is the act of verifying a claimed identity. It should be possible for the receiver of a message to determine its origin and an intruder should not be able to assume someone else's identity.

Authentication is a technique to ensure that the stated identity of the user is correct. As a first step, the entity interested in authenticating itself with some service will introduce itself and declare its identity. The service provider should be able to verify that the contacting party (sender) is the one it claims to be. The initiating party has to present some verification to prove its identity. The initiating party on the other hand would want to ensure the validity of the contacted party. The contacted party has to present some identification about itself to the initiator.

Upon successful authentication, the service provider (contacted party) is assured that the service is available only to users who have a right to access the service and the user can be sure that the correct service provider is being used.

Authorization is the process of determining if the presenter of certain credentials is authorized to access a resource or make use of a service. Typically a user is authenticated by the service and an authorization token is supplied to the user, which is used for any future service requests. The authorization token could be as simple as a random number, or could be encoded with other information like expiration time, users identity etc. In order for authorizations to be effective, they should be carefully constructed, and protected from intruders when services are requested. One can easily weaken a sound authentication scheme with a weak authorization mechanism. Authorization tokens should not be easily reproducible, they should be protected when making service requests and finally should minimize the time for which they are valid so as to prevent them from being used in replay attacks.

## 2.2 Evaluation Criteria

We will be examining some of the AA schemes currently in use in public wireless networks. Before describing and analyzing these solutions, this section considers the various requirements that a viable solution needs to address.

- First and foremost, the authentication and authorization scheme needs to work at a practical level. From a users perspective it should be easy to use, perhaps non-intrusive.
- It needs to scale.
- The solution should allow for new service providers and infrastructure providers to join and participate in the emerging public network.
- The solution should facilitate access to services minimizing redundant interactions for authentication and provide a single-sign on regardless of the users primary service provider.
- It must support the need to access the service independent of the users physical location.
- Clients should be allowed to use multiple client devices to gain access.

From a service provider point of view, their customers should be able to access public networks that are their own or belong to those of its partners. This implies that once they have implemented the server side of the scheme, they would be able to expand/participate in the public network by either deploying their own infrastructure or by establishing SLAs with existing service providers.

The primary responsibility of the infrastructure providers is to verify the authenticity of the authorizations issued by their service provider partners prior to allowing the users access to services. In addition, they can keep track of usage statistics and present them to the users service provider for accounting/billing purposes.

### 2.3 Related Work

In this section we will examine some of the AA schemes used in the public wireless networks. Any AA scheme has to be evaluated within the context of its operation. The context is defined to be the number of parties involved and the interaction needed to authenticate and authorize a user.

The first model we will examine is where an entity has deployed a WLAN in a public place. Any person with a mobile node entering the place has the ability to access the network provided the user has established a relationship with the owner of the network. For example this relationship could be in the form of registering with the provider or giving out the credit card at the time of access. It is quite possible to establish a relationship at the time of access by providing the proper credentials. Once a relationship exists, the user is required to access a particular web page using their browser. This page requires the users to authenticate themselves by entering their user id and the associated password. Any access to the network resources is prevented until the user has been authenticated. The authentication step needs to happen at each location the user wants to access the network. Lets examine this model by first breaking it into onetime steps and steps that are to be repeated each time.

Registration is a one time process and involves the following steps:

- The user must go thru a registration process with the provider and establish a user id and password. This may involve user modifying some settings on the mobile device.
- The user must know how to reach the web page that allows them to authenticate themselves.
- Authentication is a process that has to be repeated every time users enter a physical location where they would like to access the public WLAN. The process is made up of the following steps:
- When the user is interested in accessing the public WLAN, he or she must direct the web browser to the authentication page of the provider.
- Enter the user id and the associated password. Which would be sent to the provider's infrastructure and verified.
- Upon successful authentication, the user is allowed to use the network.

Let us examine to see how well this scheme matches the criteria we have laid out. The scheme outlined above seems to be practical and scalable from the provider's point of view. In this case the network and service is provided by the same entity. The network provider can expand the network and automatically allow all its existing users to access the expanded network. From the users point of view, they can use the network where their provider has a deployed infrastructure. The users have the responsibility to authenticate themselves every time they enter a physical location. Entering user id and password information using a resource-constrained device becomes an issue, however a different implementation of this scheme could overcome this limitation.

**Table 1.** shows the messages that are exchanged between the user and the network owner as part of the authentication process.

Mobile Client	Service provider
1. User access providers web page	1. Provider sends a authentication form
2. User enters user id and password	2. Provider verifies the presented credentials against its user database and approves or rejects
3. User is approved and is allowed to access the network resources.	3. Provider keeps track of usage

The user id and the password are passed in the clear or can be encrypted. If they are passed in the clear it is very easy for sniffers to steal the credentials. Encryption scheme would require additional software to be installed on the client device and all authentication requests have to be handled by this software. A more serious issue with this scheme is if another rogue provider pretends to be the provider known to the user, they can easily steal the users credentials and any other information being accessed by the user. This scheme does not allow the user to verify the authenticity of the provider.

Privacy in this scheme is limited to what is agreed upon between the user and the provider at the time of signup. The provider has the ability to keep information about all of the services being accessed on a per-user, per physical location.

The virtual public WLAN uses a similar scheme as above. The main difference is that network is made up of several different infrastructure providers and a coalesced together and made to appear as a single network by the service provider. The user signs up with the service provider and is allowed to access any of the public WLANs deployed by the service provider partners. The major difference in this scheme from the user perspective is the availability of the expanded network.

In terms of protecting the user credentials and the privacy this scheme suffers from similar limitations as identified above. Even though the user has a better coverage, the user is still required to authenticate at each new physical location.

The service provider has to deal with wide range of infrastructure owners and enter into SLAs with each of them. The service provider cannot guarantee the performance or the reliability of the network as each infrastructure owner may have distinct standards.

The infrastructure provider by signing up with a service provider may be forced to restrict the use of the infrastructure to just the clients of the service provider. They may be limited or restricted from offering local services and charging for it as they have no way to authenticate or bill the client directly.



**Table 2.** shows the three entities involved in a public network and the messages that are exchanged between them.

Mobile Client	Infrastructure	Service provider
1. User connects to infrastructure	1. Detects user, notifies service provider. Returns the form sent by provider to user.	1. Provider sends a authentication form
2. User enters id and password	2. Does minimal validation and forwards it to service provider. Saves the authorization token and accepts the user.	2. Provider verifies the presented credentials against its user database and approves by sending an authorization token.
3. User is approved is able to use the service.	3. Keeps track of users usage. Sends usage info to service provider.	3. Receives user specific usage data.

### 3 Our Protocol

We use the following authentication and authorization protocol for service access at hotspot servers. In our system, a nomadic user/client gets network services by subscribing to an Internet Service Provider (ISP). There can be many ISPs each running its own authentication/authorization server (AS) for user authentication and authorization. There also exist wireless-based hotspot servers (HS), which provide connectivity as well as local services to the nomadic user. Hotspot servers could be deployed either by ISPs or by third parties.

Our protocol has two phases, the client authentication phase and the access setup phase. In the client authentication phase, the client and his/her ISP (AS server) mutually authenticate each other and derive a shared session key. In the access setup phase, the AS server sends client authorization information to the HS server. Additionally, the AS server chooses a service access key and sends it to the HS server and the client in a secure way. The client can then use the service access key to access HS server securely. Before describing the protocol, we first outline some of the assumptions we make about the environment.

#### 3.1 Assumptions

Here are some of the assumptions we make while describing the protocol.

- a. There are many Hotspot (HS) servers.  $HS_{[1...l]}$
- b. There are many ISPs that run Authentication/Authorization Servers (AS) –  $AS_{[1...m]}$
- c. There are several nomadic users – referred as Client (C) –  $C_{[1...n]}$

- d. The number of clients are much greater than the hotspot servers, which in turn are more than authorization servers.
- e. Any given nomadic client  $C_i$  has a Service Level Agreement (SLA) with one or more ISPs (ASs).
- f. A nomadic client  $C_i$  will request service access through  $HS_k$  (where  $k$  is based on  $C_i$ 's location).
- g. In-order for hotspot providers to be viable they will have SLAs with ASs. When  $C_i$  requests a service from  $HS_k$ ,  $HS_k$  needs to ensure that it [ $HS_k$ ] has an SLA with  $C_i$ 's  $AS_j$ .
- h. Anonymity of  $C_i$  should be maintained wherever possible. Thus  $HS_k$  does not need to know  $C_i$ 's identity.  $C_i$ 's identity is verified only by  $AS_j$ .
- i. There is a Discovery protocol/mechanism that enables the  $C_i$  to establish a communication channel to  $HS_k$ .
- j. The User & the personal client device are viewed as one ( $C_i$ ).  $C_i$  gets access to  $HS_k$  using wireless local area network interfaces like 802.11,  $C_i$  is uniquely referred by an identifier (like the Wireless LAN card's MAC address).
- k. The Client Device can range from a Laptop to a single functionality device with minimal UI.
- l. We assume that each pair of AS and HS servers have a secure communication channel between them. These secure communication channels can be set up using a PKI infrastructure or other existing methods. Since AS and HS servers are expected to online, we think this is a reasonable assumption.

### 3.2 Client Authentication Phase

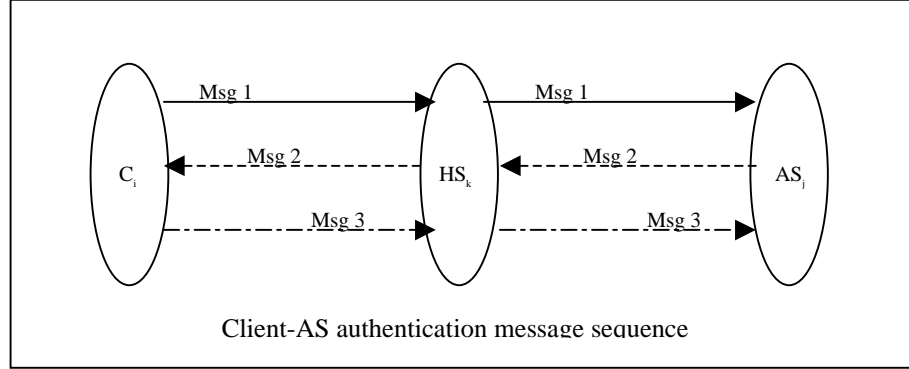
Before a mobile client can access any service offered by a HS, they must be authenticated. The pre-requisite for client authentication is that the client has established a relationship with an AS and the HS also has an SLA in place. We assume that the client shares a secret authentication key with each AS that he/she has an SLA with.

There are three messages exchanged between an authentication server AS and client  $C_i$  in this phase. All three messages are sent via a hotspot server HS. The role of HS is simply forwarding messages between the client and the appropriate AS.

- **Msg 1 ( $C_i \rightarrow AS_j$ ):  $C_i\_id$ ,  $AS_j\_id$ , Nonce <sub>$i$</sub> ,  $HS_k\_id$ , Request specific data**  
where Nonce <sub>$i$</sub>  is a number randomly chosen by  $C_i$  and sent to  $AS_j$  as a challenge.  $C_i\_id$  is how the client is identified by  $AS_j$ .  $AS_j\_id$  and  $HS_k\_id$  are identifiers of the authentication server and the hotspot server, respectively.

Upon receiving Msg 1,  $AS_j$  verifies that the request is from a valid client (by looking up its client database). If the verification fails,  $AS_j$  aborts the protocol.

- **Msg 2 ( $AS_j \rightarrow C_i$ ):  $C_i\_id$ ,  $AS_j\_id$ , Nonce <sub>$j$</sub> , Nonce <sub>$i$</sub> , Response specific data, MAC <sub>$j$</sub>**   
where Nonce <sub>$j$</sub>  is a number randomly chosen by  $AS_j$  and sent to  $C_i$  as a challenge, and MAC <sub>$j$</sub>  is the message authentication code computed on the whole message using the authentication key  $K_{ij}$  shared between  $C_i$  and  $AS_j$ . MAC <sub>$j$</sub>  serves as  $AS_j$ 's response to  $C_i$ 's challenge in Msg 1.



**Fig. 1.** Shows the three messages that are exchanged between the client ( $C_i$ ) and authorization service ( $AS_j$ ) to establish mutual authentication. Hotspot ( $HS_k$ ) simply forwards the messages.

Upon receiving Msg 2,  $C_i$  verifies that  $MAC_j$  is correctly computed using key  $K_{ij}$ . If the verification fails,  $C_i$  aborts the protocol.

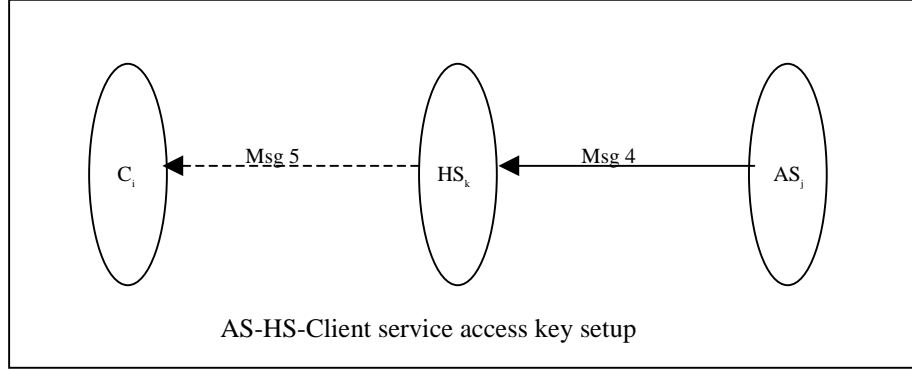
- **Msg 3 ( $C_i \rightarrow AS_j$ ):**  $C_i\_id$ ,  $AS_j\_id$ ,  $Nonce_i$ ,  $Nonce_j$ ,  $MAC_j$ , **Response specific data**,  $MAC_i$   
 where  $MAC_i$  is the message authentication code computed on the whole message using the authentication key  $K_{ij}$  shared between  $C_i$  and  $AS_j$ .  $MAC_i$  serves as  $C_i$ 's response to  $AS_j$ 's challenge in Msg 2.  $MAC_j$  is included in this message as a way of integrating the previously exchanged data.

Upon receiving Msg 3,  $AS_j$  verifies that  $MAC_i$  is correctly computed using key  $K_{ij}$ . If the verification fails,  $AS_j$  aborts the protocol.

When all the verifications are successful, client  $C_i$  and authentication service  $AS_j$  have mutually authenticated each other. They can now privately compute the shared session key  $K_s$  from the message authentication code computed on  $(MAC_i, MAC_j)$  using the shared authentication key  $K_{ij}$ , i.e.,  $K_s = MAC(MAC_i, MAC_j)$ .

### 3.3 Access Setup Phase

After successfully authenticating client  $C_i$ ,  $AS_j$  obtains authorization information for  $C_i$  and send it to  $HS_k$  using the secure channel between  $AS_j$  and  $HS_k$ . Additionally,  $AS_j$  choose a service access key  $K_a$  and send it to both  $C_i$  and  $HS_k$  so that  $C_i$  can use it for secure service access with  $HS_k$ . The following two messages are exchanged in this phase.



**Fig. 2.** Shows the messages exchanged by the client, authorization service and hotspot in the access setup phase.

- **Msg 4 ( $AS_j \rightarrow HS_k$ ):** Authorization Data,  $K_a$ ,  $E_{K_s}[K_a]$ ,  $AS_j\_id$ ,  $MAC_s$  where  $E_{K_s}[K_a]$  denotes the encryption of  $K_a$  using  $K_s$  and  $MAC_s$  is the message authentication code computed on data  $(E_{K_s}[K_a], AS_j\_id)$  using  $K_s$ . This message is sent over the secure channel between  $AS_j$  and  $HS_k$ .

After receiving Msg 4,  $HS_k$  gets the necessary authorization information and the service access key  $K_a$ .  $HS_k$  then forwards  $(E_{K_s}[K_a], AS_j\_id, MAC_s)$  to the client  $C_i$ .

- **Msg 5 ( $HS_k \rightarrow C_i$ ):**  $E_{K_s}[K_a]$ ,  $AS_j\_id$ ,  $MAC_s$

After receiving Msg 5, client  $C_i$  verifies that  $MAC_s$  is correctly computed using key  $K_s$  associated with the received  $AS_j\_id$ . When successful, client  $C_i$  decrypts  $E_{K_s}[K_a]$  using  $K_s$  to obtain  $K_a$ . Client  $C_i$  can then use the shared service access key  $K_a$  to access  $HS_k$  securely.

### 3.4 Discussion

We have described a protocol that allows clients to authenticate themselves with authorization services in public un-trusted environments. The protocol does not require the client to send its secrets over the network and also allows it to verify the credentials of the authorization service. Several of the popular protocols either require the client to send its secrets over the network or provide no way to authenticate the credentials of the service provider. In public un-trusted networks it is critical that client not be required to send its keys over the network. For example in user-name, password based protocol, it is trivial for a rogue hotspot to save the credentials of users and later use them.

Kerberos[11] is an application-level security and authentication system that was developed as part of MIT's Project Athena is another protocol that is quite popular, however it requires that all clients and services be part of a single kerberos authentication database.

## 4 Implementation

We will briefly discuss the various alternatives we considered and give an overview of our implementation

### 4.1 General Discussion

The hotspot AA protocol's intent is to establish a secure environment where there is mutual distrust between the three parties involved, i.e., the Client, the hotSpot server and the Authentication Server. NASREQ & Mobile IP extensions to Diameter<sup>1</sup> [1] have similar goals, i.e. secure authenticated remote access, but address them in different domains. Therefore some of their features do not satisfy the requirements defined for our domain as defined in the introduction section.

Currently, Internet access is the only application these extensions support. The hotspot AA protocol also supports AA for application services that may be potentially distributed.

The *NASREQ extension* is tuned to provide IP access, logging-in service etc. It does not support a 3-party authentication scheme. The *Mobile IP extension* is very similar to hotspot AA protocol conceptually. This is because the three entities - mobile node (MN), foreign agents and the home agents share a security association. New session keys are generated for each combination MN-Home, MN-Foreign, Foreign-Home whenever a user tries to access a home agent. The hotspot-AA protocol deals with mutual authentication between User (MN) and AS (Home agent) while in the mobile IP case the MN authenticates with the Home Agent; Home Agent authentication (by the user) is not required.

### 4.2 Hotspot-AA Protocol Implementation

Section has 2 sub-sections: the first part discusses the implementation details of the AA protocol & sub-section 2 is about the encryption options between client & HS.

#### 4.2.1 AA Protocol

Our implementation currently supports authenticated access for HTTP-based access & services. The protocol has been implemented over both Diameter (as an Diameter application) and HTTP [13]. For the Diameter based implementation, the publicly available implementation<sup>2</sup> of the Diameter base protocol from Sun [8] has been used (which also implements the Diameter protocol API specification [9]). The hotspot-AA protocol messages are transported in the format as defined in the EAP specification (in the NASREQ extension). The protocol was implemented over HTTP to also cater

---

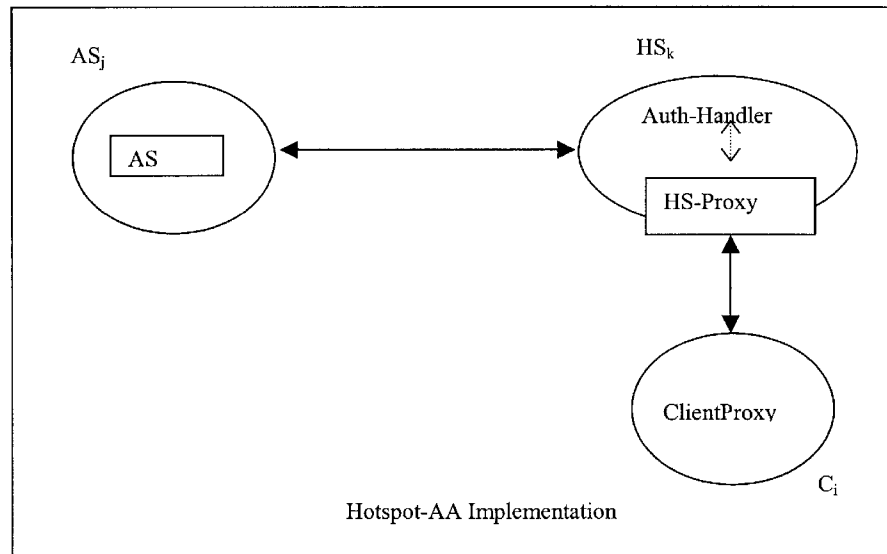
<sup>1</sup> The Diameter protocol, an IETF Internet Draft based on RADIUS, is defined to provide a base protocol that can be extended in order to provide AAA services to new access technologies. The base protocol is not stand-alone & is designed to be extended with a Diameter application. Two Diameter applications, currently defined in the IETF drafts, of interest are: the Mobile IP & NASREQ extensions.

<sup>2</sup> Only binaries are made available.

to entities not supporting the Diameter release from Sun. It also enabled access to systems across firewalls (by exploiting the presence of http proxies).

The implementation of the hotspot-AA protocol is a proxy-based solution and hence the execution of the protocol is transparent to the user being authenticated. The implementation was done keeping Internet Access AA in mind mostly because one of the primary functionality of the hotspot server is providing access to the Internet at the hotspots.

We also provide implicit authorization for local web services deployed at the hotspots. For example, certain services should not be accessible to a certain class of users whereas certain services should be accessible to only administrators. These kinds of Access Control can be specified at the HS.



**Fig. 3.** Shows the four parts of our implementation

#### 4.2.1.1 Client Side Protocol Handler (ClientProxy)

The client side protocol handler is implemented as a proxy (referred to as ClientProxy). The user is expected to set his browser's HTTP/HTTPS proxy to the ClientProxy. ClientProxy modifies each request in a way such that the entities at the HS can verify that the client is authentic. Specifically, the ClientProxy signs the request URLs with the authorization key and the signature is appended to the URL request before sending it to the HS. Apart from the signature, the Client-ID is also appended.

The first URL request triggers the hotspot-AA protocol execution after the ClientProxy discovers that it does not possess the authorization key for accessing the

entities at the HS. The ClientProxy sends and receives the hotspot-AA protocol messages as HTTP POST messages and responses respectively. The protocol message exchanges happens with the Auth Handler (described in the next section) at the HS.

#### 4.2.1.2 HS Side Protocol Handler (Auth-Handler & HS-Proxy)

The HS side has two components:

- Auth-Handler

The implementation of the HS side protocol handler (referred to as *Auth Handler*) is implemented as a web service. There are two variations of this:

- Diameter based Auth Handler

This implements a subset of the NAS side of the Diameter NASREQ extension. Simply put, the Auth Handler converts the POST HTTP messages from ClientProxy to corresponding Diameter messages. These messages are then sent to the AS and the responses are converted to HTTP POST response and sent to the client.

- HTTP based Auth Handler

In this case, the hotspot-AA HTTP POST messages from the ClientProxy are forwarded to the intended AS over HTTP.

Access to the Auth Handler for hotspot-AAA protocol execution itself does not require user authentication. In both the above cases, the HS and the AS gets mutually authenticated (and a session key is generated) before the client messages are forwarded. The HS and the AS also undergoes the shared key based authentication. The shared key is established as a result of the SLA. (There are other alternates possible for the HS ↔ AS authentication, using PKI, for example.) The mutual authentication between any two principals is limited to a definite time interval after which they have to undergo the authentication again. The Authorization key that is distributed by the AS (Message #4 in the protocol description) is stored.

- HS-Proxy

For providing Internet access to clients the HS hosts a proxy. In order to provide access to only authenticated users, the proxy needs to have some hooks to query the Auth Handler, etc. In other words, the proxy has to invoke the AA infrastructure for each client request.

Every client URL request contains a signature and the client-id (refer the section on Client side protocol handler). HS-Proxy invokes the Auth-Handler service with the arguments - URL, signature and client-id. The Auth-Handler validates the signature and responds back with the authorization status.

The Authorization key given by the Auth-Handler is used by the HS-Proxy to do encryption/decryption.

### 4.3 AS Side Protocol Handler

The AS has also been implemented over both Diameter and HTTP. The Diameter based implementation is a standalone Diameter server (implementing the Diameter base protocol) which dynamically loads the AS side of the hotspot-AA protocol (implemented as a library). The library registers a set of callbacks to process the Diameter messages dealing with hotspot-AAA. On receipt of a message, the Diameter server parses the message and invokes the registered callbacks.

The HTTP based implementation is a web-service. The AS maintains two databases – one for the clients and one for the HSs that it has SLAs with.

### 4.4 Encryption Options

Data encryption is optional and is left to the client to decide whether he wants encrypted data transfer or not. Encryption is provided only between the ClientProxy and the HS-Proxy. The HS-Proxy decrypts the data before forwarding it to the next network hop. Encryption/decryption algorithms use the Authorization Key (that the AS hands over to the client and the HS) as the key. The ClientProxy and the HS-Proxy does the encryption on only a part of the data (bodies of the HTTP messages). Note that this encryption is over and above the encryption due to data transfer with a secure web-site (SSL communication).

## 5 Summary

We have outlined several AA schemes that are in practice today and shown that none of these adequately address the three party mutually un-trusted network model. We propose a protocol that can be used to authenticate and authorize clients, infrastructure and service providers that operate in an un-trusted environment. This protocol can play a significant role in the development of public data networks that support true roaming. The current implementation supports only HTTP, we would like to extend the implementation to support other transport mechanisms like FTP and RTSP. We also would like to investigate how this protocol can co-exist with mobile-ip.

## References

- [1] Diameter Base Protocol  
[http://www.interlinknetworks.com/references/technical\\_materials/docs/draft-ietf-aaa-diameter-mobileip-07.txt](http://www.interlinknetworks.com/references/technical_materials/docs/draft-ietf-aaa-diameter-mobileip-07.txt)
- [2] Diameter Mobile IPv4 Application  
[http://www.interlinknetworks.com/references/technical\\_materials/docs/draft-ietf-aaa-diameter-mobileip-07.txt](http://www.interlinknetworks.com/references/technical_materials/docs/draft-ietf-aaa-diameter-mobileip-07.txt)
- [3] Diameter NASREQ Application  
[http://www.interlinknetworks.com/references/technical\\_materials/docs/draft-ietf-aaa-diameter-nasreq-07.txt](http://www.interlinknetworks.com/references/technical_materials/docs/draft-ietf-aaa-diameter-nasreq-07.txt)



- [4] PPP Extensible Authentication Protocol (EAP)  
<http://www.faqs.org/rfcs/rfc2284.html>
- [5] Diameter implementation  
<http://playground.sun.com/diameter>
- [6] The Diameter API  
[http://www.interlinknetworks.com/references/technical\\_materials/docs/draft-ietf-aaa-diameter-api-01.txt](http://www.interlinknetworks.com/references/technical_materials/docs/draft-ietf-aaa-diameter-api-01.txt)
- [7] cbserver- A fast and tiny Coolbase Appliance Server  
<http://www.cooltown.com/dev/reference/coolbase/cbserver/cbserverPaper.pdf>
- [8] OpenSSL, <http://www.openssl.org/>
- [9] Hypertext Transfer Protocol, <http://www.w3.org/Protocols/rfc2616/rfc2616.html>
- [10] tinypoxy, <http://tinypoxy.sourceforge.net/>
- [11] B. Clifford Neuman and Theodore Ts'o. Kerberos: An Authentication Service for Computer Networks, *IEEE Communications*, 32(9):33-38. September 1994

# A Contemporary Foreword on GSM Security

Paulo S. Pagliusi

Information Security Group, Royal Holloway, University of London  
Egham, Surrey TW20 0EX, UK  
p.s.pagliusi@rhul.ac.uk  
<http://www.isg.rhul.ac.uk>

**Abstract.** This article contains a current outline of the GSM system security, with focus on the air interface protocol. It presents the terminology and describes the GSM security operation, including its principles and features. This document also discusses the effectiveness of GSM authentication and the strength of GSM encryption. It includes therefore the most significant physical and cryptanalytic attacks on GSM security mechanisms, such as the up to date optical fault induction and partitioning attacks. GSM security features retained and enhanced for the 3G Security and further applications in network (Internet) remote access are also contemplated. This article aims primarily at contributing to a progressive research in mobile systems security and at reviewing the security solutions implemented in this area for further applications.

## 1 Introduction

This article contains an up to date overview of the European GSM<sup>1</sup> cellular phone system security, with focus on the air interface protocol. It presents the terminology and describes the GSM security operation, including its principles and features, such as subscriber identity confidentiality and authentication, stream ciphering of user traffic and user-related control data, and use of triplets and SIM module.

This document also discusses the effectiveness of GSM authentication and the strength of GSM encryption. It includes therefore the most significant physical and cryptanalytic attacks against GSM security mechanisms, for instance the new optical fault induction and partitioning attacks, and the GSM features retained and enhanced for the Third Generation Security (3GPP)<sup>2</sup>. Further applications in network remote access are also contemplated. This article aims primarily at contributing to a progressive research in mobile systems security and at reviewing the security solutions implemented in this area for further applications in other correlated areas, such as authentication for Internet remote access supporting ubiquitous mobility.

---

<sup>1</sup> GSM was formerly acronym for Groupe Spéciale Mobile (founded 1982). Now is acronym for Global System for Mobile Communications (<http://www.gsmworld.com>).

<sup>2</sup> 3GPP (3rd Generation Partnership Project) is a partnership project including: ETSI (Europe), ARIB & TTA (Japan), TTC (Korea) and T1P1 (USA).

### 1.1 Terminology

- **Mobile Station (MS):** a Mobile Equipment (ME or “mobile telephone”) with its GSM Subscriber Identity Module (SIM). Each MS has a contractual relationship with a network, called the home network but may be allowed to roam in other visited networks when outside home network coverage area (Figure 1.).

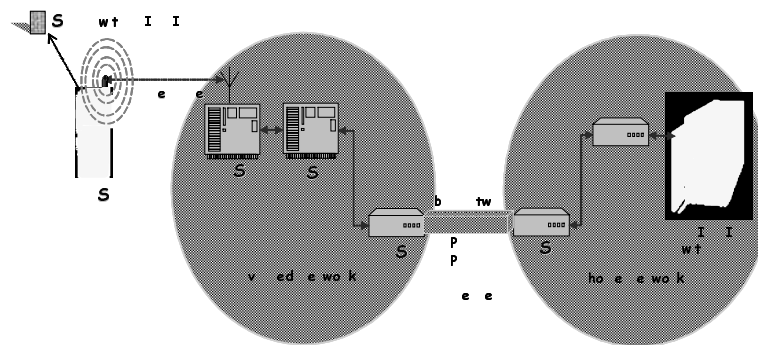


Fig. 1. GSM System Overview.

- **International Mobile Subscriber Identity (IMSI) and Authentication Key (Ki):** at the subscription time, the home network assigns the MS a unique and permanent identifier, the IMSI, together with a unique 128-bit secret key (Ki). Each customer's Ki is also stored in an Authentication Centre (AuC) in the home network. Ki plays two roles in GSM: authentication consists of proof that MS possesses Ki and encryption is performed with the use of a cipher key derived from Ki.
- **GSM Subscriber Identity Module (SIM):** module implemented on a smart card that must be inserted into the ME for service access. The IMSI and the authentication key Ki of the MS should be “securely stored” in the SIM.
- **Public Land Mobile Network (PLMN):** network that currently provides service or “is visited” by a MS. A MS is registered with the PLMN which it is currently visiting. A PLMN contains, among others components: a Base Station (BS) and a Visited Location Register (VLR).
- **Base Station (BS):** the Base Transceiver Station belonging to a PLMN serving the MS. Base stations form a patchwork of radio cells over a given geographic coverage area. Base Stations are connected to base station controllers (BSC).
- **Base Station Controller (BSC):** is a node controlling a number of BS, coordinating handovers and performing BS co-ordination not related to switching. The BSC to BS link is in many cases a point to point microwave link. BSC are also connected to mobile switching centres (MSC) via fixed or microwave links. MSC are connected to the public networks (e.g. PSTN, PDNS, ISDN and Internet) [6].
- **Visited Location Register (VLR):** used to record information about all MS “visiting” a specific PLMN.

- Home PLMN (HPLMN): each MS has a home PLMN with which shares an IMSI and a Ki. The HPLMN and the visited PLMN have a bilateral agreement, under which the visited PLMN trusts the HPLMN to pay for the services that the visited PLMN provides to the MS. Each HPLMN maintains a Home Location Register (HLR) and operates an Authentication Centre (AuC) to support its MS.
- Home Location Register (HLR): used to record the most recent known location of all MS belonging to a specific HPLMN.
- Authentication Centre (AuC): used by a HPLMN to generate random challenges (RAND) and to store secret key information (Ki) relating to each of its MS. The AuC can be integrated with other network functions, e.g. with the HLR.
- Air Interface: synonym for radio path. The MS ‘visits’ a PLMN by communicating with the serving BS across an air interface and receiving an entry in the VLR.

## 2 Security Features for GSM

The purpose of security for GSM system is to make the system as secure as the public switched telephone network and to prevent phone cloning. The use of air interface as the transmission medium allows a number of potential threats from eavesdropping. As stated by [4], “it was soon apparent in the threat analysis that the weakest part of the system was the radio path, as this can be easily intercepted”. In fact, there was no attempt to provide security on the fixed network part of GSM. And it should be noted that the GSM security was designed with three constraints in mind [13]:

- Concern of grant too much security and so bringing export problems upon GSM;
- GSM did not have to be resistant to “active attacks” where the attacker interferes with the operation of the system, perhaps masquerading as a system entity; and
- The trust between operators for the security operation should be minimized.

The technical features for security are only a small part of the GSM security requirements; the greatest threat is from simpler attacks such as disclosure of the encryption keys, insecure billing systems or even corruption. A balance is required to ensure that these security processes meet these requirements. At the same time a judgment must be made of the cost and effectiveness of the GSM security measures.

The principles of GSM security are, according to [6] and [13]:

- Subscriber identity confidentiality;
- Subscriber identity authentication;
- Stream ciphering of user traffic and user-related control data; and
- Use of SIM as security module.

It is also important to emphasize the GSM feature of use of triplets. The GSM principles and this special feature are described in the following sections.

## 2.1 Subscriber Identity Confidentiality

- **Purpose:** to avoid an interceptor of the mobile traffic being able to identify which subscriber is using a given resource on the air interface.
- **Description:** The use of temporary identifiers provides anonymity, so that it is not easy to identify the GSM user. This protects against the tracing of a user's location by listening to exchanges on the radio path. Instead of using the IMSI, a new temporary mobile subscriber identity (TMSI) is allocated by the PLMN at least on every location update and used to identify a MS on the air interface.
- **Operation:** When a MS attempts access with a PLMN with which it is not presently registered, the MS uses its IMSI to identify itself. The IMSI is then authenticated by the PLMN, which results in the sharing of a cipher key ( $K_c$ ). When the PLMN switch on encryption, the VLR generates a TMSI to the MS, storing the association of TMSI and IMSI in its database. The TMSI is then sent to the MS, encrypted with  $K_c$ . The next time the MS attempts access in that PLMN, it uses the TMSI previously allocated by the VLR instead of its IMSI. Then the PLMN looks up its table of TMSI to IMSI mapping to find the MS permanent identity. After a successful authentication and once an encrypted channel have been established, the PLMN assigns to the MS another TMSI. It is frequently given a new TMSI to that a MS cannot be previously identified and followed around. After a handover to a new VLR, or a successful re-authentication with the same VLR, the PLMN always sends a new TMSI to the MS. Then the MS stores the new TMSI and removes the association with any previously allocated TMSI. In turn, the VLR removes the association with the old TMSI and the IMSI from its database.

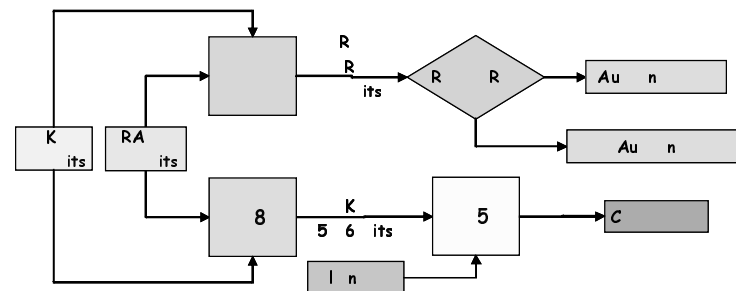


Fig. 2. GSM Authentication, Cipher Key Generation and Encryption.

## 2.2 Subscriber Identity Authentication

- **Purpose:** The authentication is used to identify the MS to the PLMN operator.
- **Description:** It consists in the guarantee by the land based part of the system that the MS identity presented across the air interface is the real one originally embedded in the SIM. The PLMN then knows who is using the system for billing pur-

poses. This protects the PLMN from illicit use. GSM authentication is a one-way process, i.e., the visited PLMN is not authenticated.

- **Operation:** Authentication is performed by a challenge and response mechanism. Ki in the HPLMN is held in the AuC. A random challenge (RAND) is generated by the AuC and issued to the MS, via PLMN. The MS encrypts RAND using Ki and the authentication algorithm A3 implemented within the SIM, and send a signed response (SRES) back to the PLMN. AuC performs the same process with RAND to compute the expected response (XRES), which is sent to the PLMN. The PLMN now can check that the MS has Ki and that the response received is correct by comparing the value received from the HPLMN (XRES) with what it receives from MS (SRES). Eavesdropping of the radio channel should reveal no useful information, as the next time a new RAND will be used (Figure 2).

$$\begin{aligned} \text{SRES} &= \text{A3}_{\text{Ki}}(\text{RAND}) \\ \text{XRES} &= \text{SRES?} \end{aligned} \quad (1)$$

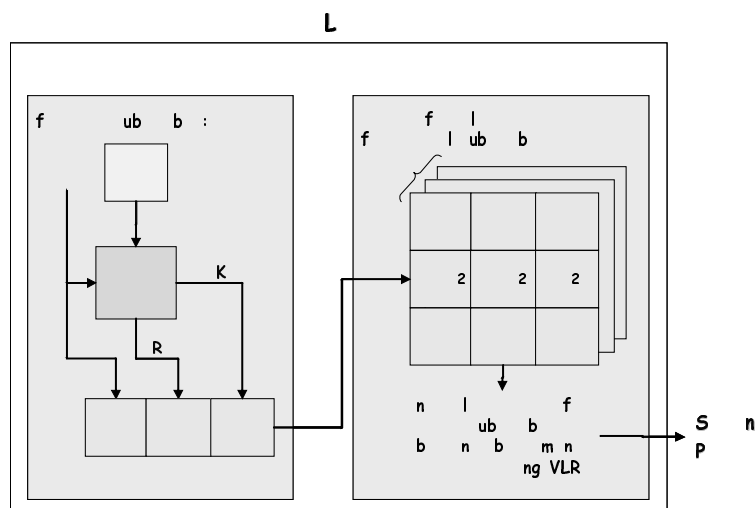
### 2.3 Stream Ciphering of User Traffic and User-Related Control Data

- **Purpose:** stream cipher encryption is used in GSM system to protect sensitive information against eavesdropping on the air interface.
- **Description:** This provides protection to the user data passing over the radio path on physical connections or connectionless and to the sensitive information on the signalling channel (e.g. phone numbers, TMSI) from eavesdropping on the air interface. Confidentiality is achieved by the use of encryption at physical layer, the choice being influenced by: speech coder, error propagation, delay and handover.
- **Operation:** At the same time that XRES and SRES are calculated, RAND and Ki are passed through algorithm A8 by both the MS (SIM) and the HPLMN (AuC) to derive the cipher key Kc. Then Kc is delivered from the HPLMN (AuC•HLR) to the serving PLMN (VLR•BS). Typically, algorithms A3 and A8 are combined into one called A3/8 that is residing within the SIM and the AuC. Kc is used for encrypting the signalling and messages to provide privacy through the use of A5 series algorithms (Figure 2). The BS tells ME which A5 algorithm it has (if it has one) and sends a cipher command. At the same time, the BS starts decrypting. The ME starts encrypting and decrypting when it receives the cipher command. The BS starts encrypting when it receives back the cipher command acknowledged. A fresh cipher key Kc is generated for each call. When a handover occurs during a call, the necessary information is transferred by the PLMN to the new BS, and the encryption continues using the same Kc.

$$\begin{aligned} \text{Kc} &= \text{A8}_{\text{Ki}}(\text{RAND}) \\ \text{Ciphertext} &= \text{A5}_{\text{Kc}}(\text{Plaintext}) \end{aligned} \quad (2)$$

## 2.4 Use of Triplets

- **Purpose:** with the use of the triplets, authentication can be performed in the 'visited' PLMN without the network operator (BS, VLR) having knowledge of  $K_i$ .
- **Description:** A random challenge (RAND) and the resulting expected response (XRES) and the cipher key ( $K_c$ ) produced by A3/8 form a "triplet" (224 bits).
- **Operation:** An AuC will produce a batch of triplets for a MS, each entry with a different RAND, all at once and pass these for distribution to the associated HLR of the same HPLMN. When a MS attempts to make a call or a location update in either its HPLMN or in a visited PLMN, the SIM passes its identity to the VLR. The VLR makes a request to the subscriber's HPLMN for a batch of triplets for the identity claimed by the MS (i.e. SIM) and the HLR of the HPLMN responds with a batch of (n) triplets for that claimed identity (Figure 3).



**Fig. 3.** GSM Triplet Generation, Distribution and Subscriber Management.

The serving VLR then authenticates the SIM by sending  $RAND(i)$  of the batch of triplets to the MS, via BS, and by comparing the value of the stored expected response  $XRES(i)$  for that  $RAND(i)$  with the received SRES produced by the SIM. If they match, the MS claimed identity is deemed to be authenticated and so the VLR can pass the cipher key  $Kc(i)$  from the triplet to the serving BS.  $Kc(i)$  and the secret key  $Kc$  calculated by the SIM, both with the same value, can be used respectively by the BS and the MS to protect the air interface (Figure 4). When the PLMN has run out of triplets, it should request more from the home HLR, though the PLMN is allowed to re-use triplets if it cannot obtain more from the HPLMN.

$$\text{Triplet}(i) = (RAND(i), XRES(i), Kc(i)); 1 \leq i \leq n. \quad (3)$$

Where:  
 $n$  = number of entries stored in a batch of triplets for a subscriber.  
 $i$  = entry chosen by the serving VLR to be send to the MS via BS.

## 2.5 Use of SIM as Security Module

- **Purpose:** Key distribution, authentication and cipher key generation.
- **Description:** SIM is implemented on a smart card (which should be “tamper proof”) to make it “infeasible” to extract the Ki. Although the SIM is required at the start of a call only, In GSM a call must close if the SIM is removed from the ME during a call to avoid parallel calls using a unique SIM (i.e., a stolen SIM).
- **Operation:** As described before, the ME passes the RAND received from the VLR to the SIM. Then SIM passes its Ki value and the received RAND through algorithm(s) A3/8. The resulting SRES produced by the SIM is passed back to the ME and then to the VLR, that verify if the SIM claimed identity can be authenticated. If the SIM is authenticated, the VLR passes Kc from the triplet to the serving BS. Then SIM passes Kc to the ME and as a result the BS and the ME can begin ciphering communication using Kc and the A5 algorithm (Figure 4).

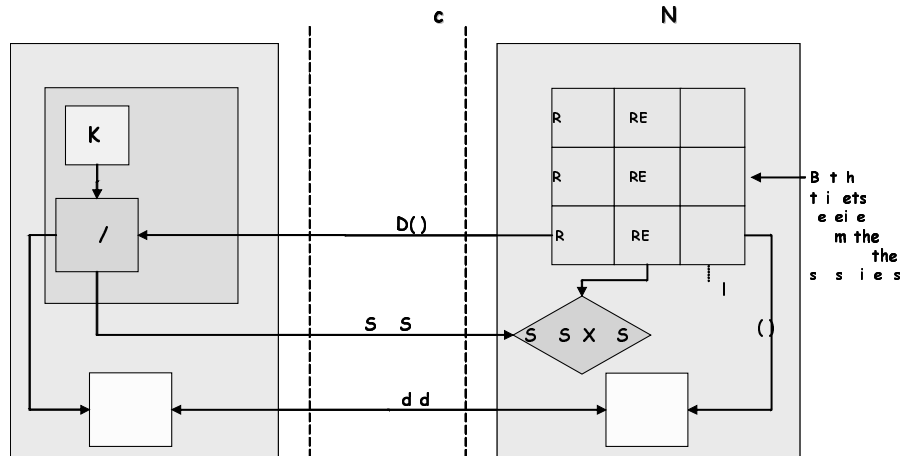


Fig. 4. Authentication and Encryption for GSM using triplets and SIM.

## 3 Effectiveness of GSM Authentication

The efficacy of GSM authentication relies on a number of algorithm requirements:

1<sup>st</sup>) it is statistically near impossible for an impostor to guess what the correct SRES should be and therefore masquerade as another subscriber. As the MS has only one chance to return SRES for a particular RAND, and the parameters SRES/XRES are 32 bits long, such an impostor has only a 1 in  $2^{32}$  chance of guessing SRES correctly. Since SRES must be indistinguishable from any other 32 bit number that might be returned instead of SRES, then this is not a realistic attack.

2<sup>nd</sup>) an impostor cannot derive Ki from collecting a number of RAND-SRES pairs obtained from eavesdropping the air interface. This means that A3/8 must resist a known plaintext attack. Further, as an attacker could steal a SIM, send chosen RAND



to the SIM and collect the SRES returned A3/8 must be resistant to a chosen plaintext attack (this latter requirement was shown not to be satisfied by the algorithm COMP128, used as A3/8 by many operators).

3<sup>rd</sup>) an impostor cannot derive a particular Kc from the RAND and SRES in the same triplet or by collecting a number as RAND-SRES pairs. This means that SRES and Kc, though derived from the same RAND and Ki, must be completely unrelated.

4<sup>th</sup>) Ki was not to be shared with the serving PLMN. Even A3/8 does not need to be known by the VLR, as this algorithm is used only where Ki is present (i.e. in the AuC and SIM). The same occurs with A5, not used in the VLR but in the BS.

5<sup>th</sup>) AuC is resistant to penetration.

6<sup>th</sup>) The SIM should be tamperproof.

## 4 Strength of GSM Encryption

A5 series algorithms are contained within the BS and the ME, but not within the SIM, as they have to be sufficiently fast and are therefore hardware. There are two defined algorithms used in GSM known as A5/1 (only members of CEPT<sup>3</sup>) and A5/2 (exportable). Neither A5/1 nor A5/2 has been officially published. The cipher key Kc, related with algorithm A5, is 64 bits long but the top 10 bits are forced to 0 in SIM and AuC. Then there are only 54 bits of effective key length. There is a new Kc in each call and 22 bit message key changes every frame (4.615 ms), giving a 5.4 hour repeat cycle. Encryption operates at the physical layer (layer 1) in the protocol stack. Ciphering only exists between MS (ME) and BS, as it was assumed that most other links afterwards would be along fixed lines. The decision to put encryption at the layer 1 had some consequences, as described below:

- The maximum amount of data, both user and signalling data is encrypted;
- Since ciphering takes place after error correction and deciphering takes place before error correction, then a stream cipher must be used for GSM because of the relatively high uncorrected error rate (of about  $10^{-3}$ ) in wireless environments;
- The frame counter, normally used for synchronisation at layer 1, is used with an expanded length as an input to the key stream generator. The frame counter is 1024 times longer than the longest frame aggregation required for non-encryption purposes to avoid repetition during a call and so causes encryption weakness; and
- The encryption algorithm can be implemented in hardware.

As a stream cipher, A5 works on a bit by bit basis (and not on blocks, as DES and AES). So, an error in the received ciphertext will only result in the corresponding plaintext bit being in error, as shown in the diagram for A5 operation (Figure 5).

As a function of Kc and frame counter, a key stream generator produces a string of pseudo-random bits. This string of bits is XORed with the plaintext to produce the ciphertext. At the decrypting end, the same key stream is produced and XORed with

---

<sup>3</sup> CEPT - Conférence Européenne des administrations des Postes et des Télécommunications.

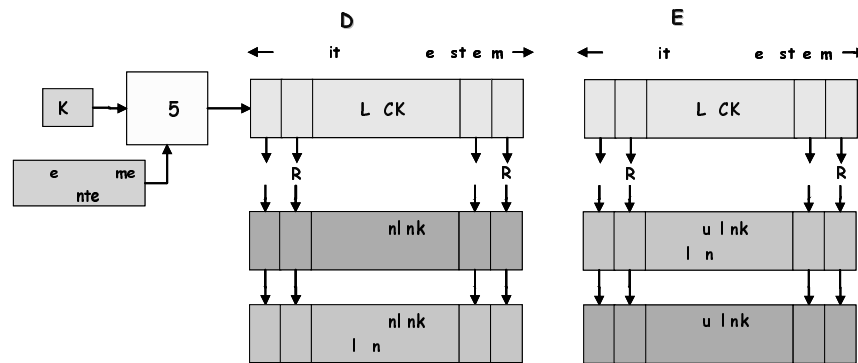


Fig. 5. Operation of A5 at the mobile station [13].

the ciphertext to produce the plaintext. In GSM both sides can transmit simultaneously (it is full duplex), so within a frame, a MS or BS both transmits and receives a frame. The first 114 bit block (BLOCK1) of the string is used to encrypt the plaintext data being transmitted. The second 114 bit block (BLOCK2) is used to decrypt the data received in that frame, as shown in the diagram (Figure 5). At the other end of the air interface, the first block is used to decrypt the received ciphertext and the second block of the same string is used to encrypt the plaintext to be transmitted.

## 5 Attacks on GSM Security

The most significant physical and cryptanalytic attacks on GSM are given below; [9], [10] and [13] are also good references on this subject.

### 5.1 Microwave Links

The fact that the BS to BSC link is in many cases a point to point microwave link is a clear gap in GSM security. This link can be eavesdropped upon as data is at this point un-encrypted. At the time of GSM design, it was expected that the BS to BSC link would be across fixed links and therefore that encryption would not be required. In 3GPP, the encryption extends as far as the Radio Network Controller (RNC), the 3GPP equivalents of the BSC - microwave links are therefore protected [13].

### 5.2 SIM/ME Interface

When a call is not in progress, it is possible for the SIM to be removed from one ME and put in another with which it has no previous association. As the SIM-ME interface is unprotected, it is possible for the SIM to be connected to terminal emulators instead of "genuine" ME and consequently for messages on the SIM-ME interface to be

tapped. However, unless the algorithms on the SIM are sub-standard, there is no advantage in compromising the SIM-ME interface in this way.

### 5.3 Attacks on the Algorithm A3/8

Wagner and Goldberg announced in April 1998 that they had cracked COMP128, an algorithm taking the function of A3/8 in the SIM of many operators. COMP128 had a weakness which would allow complete knowledge of Ki if around 160 000 chosen RAND-SRES pairs could be collected (“chosen plaintext” attack). There are active attacks that can be used to obtain these pairs. The quickest attack would be to steal the user’s mobile phone, remove the SIM and connect it to a phone emulator that can be used to send 160 000 chosen RAND to the SIM and receive the SRES. SIM tend to have relatively slow clock speeds and it can therefore take up to 10 hours to obtain the 160 000 pairs (with faster SIM, it would take 2 and a half hours).

Another method is to use a false BS to send the RAND over the air interface. The rate at which pairs can be collected is slower and would take a number of days; however the attacker does not need physical possession of the SIM. After these efforts, the attacker has the Ki and can masquerade as the user and run calls on her bill, and also determine the Kc for the user’s calls and therefore eavesdrop upon them [13].

### 5.4 Attacks on the Algorithm A5/1

The only attack on an algorithm that has been confirmed to be A5/1 was that by Biryukov and Shamir [2], later improved by Wagner [3]. The technique used is known as “time-memory trade-off”. In a pre-processing phase, a large database of algorithm states and related key stream sequences is created. In the attack phase, the database is searched for a match with sub-sequences of the known key stream. If a match is found, then with high probability the database gives the correct algorithm state. It is then simple to compute Kc and decipher the rest of the call. Shamir and Biryukov made an attack feasible in practice, if 2 minutes of known key stream could be obtained. Wagner spotted a further optimization which would allow Kc to be obtained with only 2 seconds of known plaintext (from both uplink and downlink).

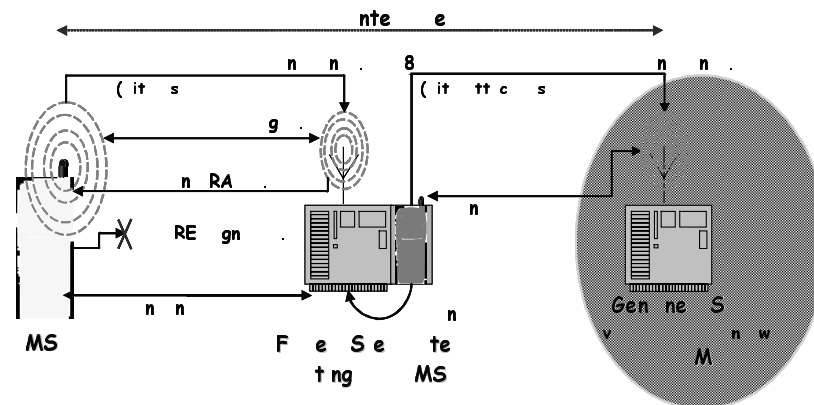
This is not trivial, because the precise sequence of bits that is encrypted must be obtained. It is a fine piece of cryptanalytic work, but in reality it would probably not be used, as the false BS attack represents an easier method of eavesdropping [13].

### 5.5 Attacks on the SIM Card: Optical Fault Induction and Partitioning Attacks

Since the SIM module is implemented on a smart card, then any discovery of a common vulnerability in smart cards immediately affects the security of the information stored in the SIM (e.g., IMSI and Ki). Thus, it is vital to emphasize a new class of attacks on smart cards called optical fault induction, revealed by Skorobogatov and Anderson [12]. They discovered the flaw after Skorobogatov found that he could inter-

They were able to expose the circuit to the light by scraping most of the protective coating from the surface of the microprocessor circuit embedded in each smart card. With more study, they were able to focus the flash on individual transistors within the chip by beaming the flash through a microscope. By sequentially changing the values of the transistors used to store data, they were able to ‘reverse engineer’ the memory address map, allowing them to extract the secret data from the smart card. The authors asserted that they have developed a technology to block these attacks.

The new IBM approach seems to be a much more useful method than either breaking the cryptographic algorithms (COMP128) used by the SIM card or by intrusive attacks, such as the optical fault induction. The IBM researchers' report also offers advice to the smart card industry on how to protect against vulnerabilities.



**Fig. 6.** The False BS Threat.

### 5.6 False Base Station

According to [9], “the MS is authenticated to the BS, but the BS is not authenticated to the MS”. That is, GSM provides unilateral authentication. This allows attacks where a malicious third party masquerades as a BS to one or more MS. One of the assumptions when GSM was designed was that the system would not be subject to active attacks, where the attacker could interfere with the operation or impersonate one or more entities. It was believed such attacks would be too expensive compared to other methods of attacking GSM as a whole and wiretapping the fixed links or even just bugging the target [13]. But the cost of BS devices has fallen quickly and it is easy to find BS emulators. Moreover the use of encryption on a call does not happen automatically – the call begins un-encrypted and the MS is instructed to begin (or not) ciphering at a particular point in call set-up. The start encryption instruction is given un-encrypted and with no origin authentication by the PLMN and it can be subject to tampering in transit; i.e., the BS instructs the MS to begin encryption but this is manipulated in transit to be an instruction not to cipher. The problem of the PLMN expecting encryption and the genuine MS not is realized by the false BS acting as a MS and setting up a call to the genuine BS itself. The un-encrypted call from the target MS to the false BS is connected to the encrypted call from the false BS to the PLMN, so it seems to the caller that they have the call they requested. But the call from the target MS to the false BS is not encrypted so the link can be eavesdropped at this point. And since the call between the false BS (but true SIM) and the PLMN is an encrypted genuine call, the PLMN does not see that anything is awry.

The Figure 6 shows the steps to be followed in order to achieve the false BS attack. One effect of this attack is that the call is made on the false BS subscription and not that of the MS's. So the BS attack can be detected later if an itemized bill is checked. The 3G prevention of this attack is carried out by integrity protection of the start encryption command, and cannot be achieved merely by mutual authentication.

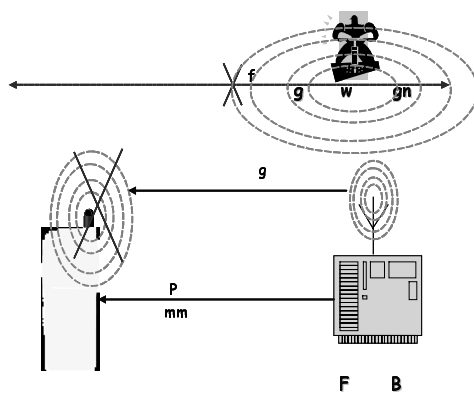


Fig. 7. Denial of service threats to GSM.

### 5.7 Other Threats

As mentioned in section 5.3, the PLMN is allowed to re-use triplets if it cannot obtain more from the HPLMN. But there is nothing to stop the BS using the same triplet repeated times. Thus if a single triplet is compromised, a false BS can impersonate a genuine PLMN to a MS indefinitely. Moreover, as the false BS has  $K_c$ , it will not be necessary for the false BS to suppress encryption on the air interface. While the genuine BS is using the compromised triplet, an attacker could impersonate a MS and obtain calls at a valid user's expense [9].

Another threat that always exists is denial of service (DoS), which can be achieved if an attacker sends a signal at high power on the GSM frequencies to 'jam' the radio path. Another kind of DoS is for a false base station to capture a MS and prevent the MS communication (Figure 7).

### 5.8 Measurement and Fraud Detection

As illustrated in [4], a properly designed billing system can be used to detect fraud patterns from normal GSM usage. Different types of fraud often produce a distinct pattern that can be detected. Therefore what is necessary to detect are: (i) multiple calls at the same time; (ii) large variations in revenue being paid to other parties; (iii) large variations in the duration of calls, such as very short or long calls; (iv) changes in client usage, indicating that a MS has been stolen or is being abused; and (v) monitor the usage of a GSM client closely during a "probationary period".

There are some "Fraud Engines" on the market that can provide these features, enabling patterns in billing data to be analyzed, and give time for swift effective action. With fraud detection capability, and security procedures in place, it is possible to minimise the effect of fraud on a billing system.

## 6 Third Generation Security – 3GPP

The 3G radio access link security was built on the security of GSM. 3GPP, that developed the 3G/UMTS standards, adopts the security features from GSM that have proved to be needed and robust and try to ensure compatibility with GSM to ease inter-working and handover. 3G security tries to correct the problems with GSM by addressing security weaknesses and by adding new features. The 3G security has the following security features: mutual authentication and key agreement between MS and network; encryption of user traffic and signalling data over the air interface; and integrity protection of signalling data over the air interface. The GSM security features to retain and enhance in 3GPP are, according to [1] and [6]: (i) maintain a smart card as a subscriber identity module (UMTS SIM or USIM); (ii) authentication of the MS to the network; (iii) encryption of the user traffic and the signalling data over the air interface; and (iv) user identity confidentiality over the air interface.

The new security features required for 3GPP includes mandatory integrity protection for critical signalling commands (e.g. for the start encryption command), which

provides enhanced protection against false BS attacks (section 5.6) by allowing the MS to check the authenticity of certain signalling messages. This feature also extends the influence of MS authentication when encryption is not applied by allowing the PLMN to check the authenticity of certain signalling messages. Mutual authentication and key agreement also provides enhanced protection against false BS attacks by allowing the MS to authenticate the PLMN. 3G authentication provides authentication of MS (USIM) to PLMN and PLMN to MS, establishes a cipher key (CK) and an integrity key (IK), and assures to the MS that the keys were not used before.

In 3G systems a new sequence number (SQN), generated in the AuC, is attached to the triplets to prevent the threat resulting from the triplets re-use (section 5.7). USIM has a scheme to verify freshness of received SQN (the SQN attached to a triplet must exceed the most recently received subset). A Message Authentication Code (MAC) is also attached to show that the authentication vector (or ‘quintet’) really came from the HPLMN and to integrity protect the SQN recently attached.

The encryption of the user traffic and the signalling data over the air interface is performed through an algorithm called KASUMI [5] (based on the Japanese MISTY1 [7]), which had an open design process, taking a longer cipher key length (128-bit) derived during authentication. The encryption terminates at the RNC, a 3G entity similar to the BSC. The links BS-RNC that may be over microwave are thus ciphered (section 5.1). KASUMI is also used for the integrity protection of commands (critical signalling) between MS and RNC [13].

3G specifications also introduced protection of network signalling (e.g. quintets) transmitted between and within networks, which can be used to eavesdrop upon MS traffic or to masquerade as valid MS. Further, to prevent false messages transmitted along the network, it is vital that the origin of such commands can be authenticated so that only authorised parties can send them.

## 7 Further Applications in Network (Internet) Remote Access

The use of triplets is an significant GSM feature that permits delegation of the authentication function as well as cipher key distribution from the HPLMN to the serving PLMN through a minimal trust relationship between operators. That is, the home network does not need to reveal the information most sensitive, such as the secret key (Ki), to any intermediate entity in the PLMN currently ‘visited’ by the MS. Another benefit with the use of triplets is that subsequent authentications do not require additional round trips with the HPLMN and this gives a big performance advantage. This kind of solution (enhanced in the 3GPP with the use of quintets), can be “exported” to applications like network remote access supporting ubiquitous client mobility.

In particular, in the scenario where a user’s access device wishes to access the Internet via different multiple access media and network interfaces (e.g. PPP, Bluetooth, IEEE 802.1x), leading to the use of a number of network operators, such as the

recently inaugurated IETF PANA work<sup>4</sup>. In this scenario, the backend authentication server (home AAA<sup>5</sup> server) can make use of triplets (or 3G quintets) to delegate client (e.g. PANA client) authentication function to any intermediate authentication agent (e.g. PANA authentication agent) in the access network, achieving minimal trust relationship between home and access network operators as well as performance benefit.

Moreover, the same triplet (or quintet) distribution operation can include cipher keys distribution to any network access point, such as NAS (Network Access Server), wishing to establish an encrypted channel with a visiting access device (e.g. PANA Client). We can imagine a feasible scenario where the interface between the client and the NAS is wireless and the NAS must use a signalling message, such as a start encryption command, to activate the encryption in the air interface. In this case, beyond the use of triplets, the introduction of mandatory integrity protection for critical signalling protocol instructions is crucial to avoid the typical masquerading or ‘false entity in-the-middle’ attack, as occurs in the false BS attack, of which prevention cannot be achieved solely by mutual authentication. Thus, the mandatory integrity protection for critical signalling messages is another solution arising from the mobile telecommunications sphere that may be clearly adapted on security mechanisms for the Internet remote access area.

## 8 Conclusions

This article reviewed the GSM cellular phone system security, with focus in the air interface protocol. This document described the GSM security operation, the effectiveness of GSM authentication and the strength of GSM encryption. The GSM security threats and attacks were included in this paper for a better understanding of the GSM features retained and enhanced for the Third Generation Security (3GPP).

Some simple but useful and robust security solutions implemented in GSM and enhanced in the 3G security, such as the use of triplets and quintets and the mandatory integrity protection for critical signalling commands, can be “exported” and adapted for application in other correlated areas, such as authentication for network (Internet) remote access supporting ubiquitous mobility.

## References

- [1] 3GPP TS 33.102 V3.11.0, “Security Architecture”, 3rd Generation Partnership Project, Technical Specification Group, 3G Security, Valbonne, France, 2002, [http://www.3gpp.org/ftp/Specs/2002-03/R1999/33\\_series/33102-3b0.zip](http://www.3gpp.org/ftp/Specs/2002-03/R1999/33_series/33102-3b0.zip).
- [2] A. BIRYUKOV, A. SHAMIR, “Real time cryptanalysis of the alleged A5/1 on a PC”, preliminary draft, December 1999.

---

<sup>4</sup> PANA is acronym for “Protocol for carrying Authentication for Network Access”. See more details in the PANA workgroup link: <http://www.ietf.org/html.charters/pana-charter.html>.

<sup>5</sup> AAA is acronym for Authentication, Authorization and Accounting.



- [3] A. BIRYUKOV, A. SHAMIR, D. WAGNER, "Real time cryptanalysis of A5/1 on a PC", in *FSE 2000*, LNCS No. 1978, Springer Verlag, Berlin, 2000.
- [4] C. BROOKSON, "GSM (and PCN) Security and Encryption", 1994, <http://www.brookson.com/gsm/gsmdoc.htm>.
- [5] ETSI TS 35 202 V4.0.0, "Universal Mobile Telecommunications System (UMTS); Specification of the 3GPP confidentiality and integrity algorithms; Document 2: Kasumi algorithm specification", <http://www.etsi.org/dvbandca/3GPP/3gppspecs.htm>.
- [6] P. HOWARD, "GSM and 3G Security", lecture notes, Royal Holloway, University of London, 19 Nov 2001, <http://www.isg.rhnc.ac.uk/msc/teaching/is3/is3.shtml>.
- [7] M. MATSUI, "New block encryption algorithm MISTY", in *Fast Software Encryption '97*, Lecture Notes in Computer Science No. 1267, Springer-Verlag, 1997, pp. 54–68.
- [8] C. MITCHELL et. al., "Link 3GS3 Technical Report 2: Security Mechanisms for Third Generation Systems", Vodafone, GPT and RHUL, 15/05/96, pp. 25 and 92.
- [9] C. MITCHELL, "The security of the GSM air interface protocol", Technical Report, RHUL-MA-2001-3, 18 August 2001.
- [10] J. R. RAO, P. ROHATGI AND H. SCHERZER, "Partitioning Attacks: Or How to Rapidly Clone Some GSM Cards", IBM Watson Research Center, in *2002 IEEE Symposium on Security and Privacy*, Oakland, CA, May 2002.
- [11] R. SCHMITZ, "SHAMAN Deliverable D02 – Intermediate Report: Results of review, Requirements and reference Architecture", Information Society Technologies, 08 November 2001, pp. 41-42.
- [12] S. SKOROBOGATOV, R. ANDERSON, "Optical Fault Induction Attacks", University of Cambridge, in *2002 IEEE Symposium on Security and Privacy*, Oakland, May 2002.
- [13] M. WALKER AND T. WRIGHT, "Security", in F. Hillebrand, editor, *GSM and UMTS: The Creation of Global Mobile Communication*, pp. 385-406, John Wiley & Sons, New York, 2002.

# Vulnerability Assessment Simulation for Information Infrastructure Protection

HyungJong Kim, KyungHee Koh, DongHoon Shin, and HongGeun Kim

Korea Information Security Agency, Seoul, Korea  
{hjkim, khko, dhshin, hgkim}@kisa.or.kr

**Abstract.** The main contribution of this research is to present the method that make it possible to assess the vulnerability in the information infrastructure using simulation technology. To construct the vulnerability assessment simulation system, we make use of the knowledge-based simulation modeling methodology and we designed expressions to estimate the degree of vulnerability of host. Also, we show newly defined vulnerability analysis method that is applicable to vulnerability assessment simulation system. As research results, the vulnerability assessment simulation system and VDBFS(Vulnerability DataBase For Simulator) are constructed.

**Keywords.** Vulnerability Assessment, Vulnerability Analysis, Information Infrastructure, Modeling and Simulation, Knowledge-Based Simulation

## 1 Introduction

Recently, as entire area of social activity becomes more related with the information infrastructure, the importance of its protection against threats increases. Since most of attacks are caused by the vulnerability in the network and system, vulnerability assessment is a prerequisite to protect the information infrastructure. In the vulnerability assessment area, the scanning tool and penetration test are widely used. Scanning tools have their vulnerability database that is used to investigate whether there exist vulnerabilities in the information infrastructure. Penetration test is performed by human-attacker who is good at exploiting the vulnerabilities. These two methods have their own merits, but they have several shortcomings that make it unsuitable to assess the vulnerabilities in the information infrastructure. The representative shortcomings that should be supplemented are as follows.

First, the two methods can cause damages or performance degradation in the information infrastructure, because their activities are based on real networks. Especially, when some essential services or systems in the information infrastructure are stopped, it can be badly damaged. Second, when we make use of scanning tools or penetration tests, it is impossible to assess the vulnerability of networks that doesn't exist currently. There are needs to assess the network that will be constructed in near future and to assess the network whose design will be altered. Third, scanning tools and penetration tests assess the vulnerability of the network just based on the current security related state. So, it is very difficult to get the difference after security manager applies new defensive mechanism to the network. Especially, although the

results are different according to the manager's experience and management period, these two methods cannot consider these attributes.

The simulation technology is appropriate to overcome these limits. When we make use of the simulation technology, the relation between previous and subsequent behaviors can be defined in the model, and selection of attacks based on the result of previous attacking is possible. Also, vulnerability assessment using simulation has no bad effect on functionality and performance of the information infrastructure, and makes it possible to evaluate the network that doesn't exist currently. Additionally, since attacks are generated in various time axis, it is possible to assess vulnerabilities at the various security level and management period.

As a result of this research, we present a vulnerability analysis method that is appropriate to the vulnerability assessment simulation and vulnerability assessment method using simulation technology. As a result of the vulnerability analysis, VDBFS(Vulnerability DataBase For Simulator) that contains 126-attack, 110-vulnerability, and 384-solution information is constructed and the knowledge-based simulation methodology is used in the construction of the system.

## 2 Related Work

Fred Cohen [1] proposes a cause-effect model to simulate cyber attacks and defense efforts. When simulation is started, the cause-effect model is applied to network model and the process of attacks and defenses is simulated. The cause-effect model is designed for the purpose of simulation and analysis and it is based on a set of 37 classes of threats, 94 classes of attack mechanisms, and about 140 classes of protective mechanisms. Those are interlinked by database, which associates threats with attacks and attacks with defenses. In this simulation, users can specify defender strength, strategy, and the number of simulation runs. In CMU/SEI [2], attack-modeling method is suggested for information security and survivability. In the research, attack tree is used to represent an event that could significantly harm the enterprise's mission. Also, the attack patterns are defined to represent a deliberate, malicious attack that commonly occurs in specific contexts. The attack pattern consists of the overall goal, a list of pre-conditions, the steps for carrying out the attack, and a list of post-conditions that are true if the attack is successful. Related attack patterns are organized into attack profile that contains a common reference model, a set of variants, a set of attack patterns, a glossary of defined terms and phrases. When the attack patterns that have a set of variants is properly instantiated, we say such patterns are applicable to the enterprise attack tree. The instantiation is to substitute the variants for domain specific values. Bishop [3] defines a characteristic of a vulnerability as a condition that must hold for the vulnerability to exist. Each vulnerability can be represented by its set of characteristics, called the characteristic set. He hypothesizes that every vulnerability has a unique, sound characteristic set of minimal size i.e., the basic characteristic set of the vulnerability. A set of characteristics for a set of vulnerabilities can be determined and the size of a complete set of characteristics for a system is smaller than the set of vulnerabilities. Each characteristic suggests a tool to analyze the system or system programs to determine if the condition exists. This analysis can be used to eliminate vulnerabilities; Simply look for situations in which the characteristic conditions hold, and negate them.

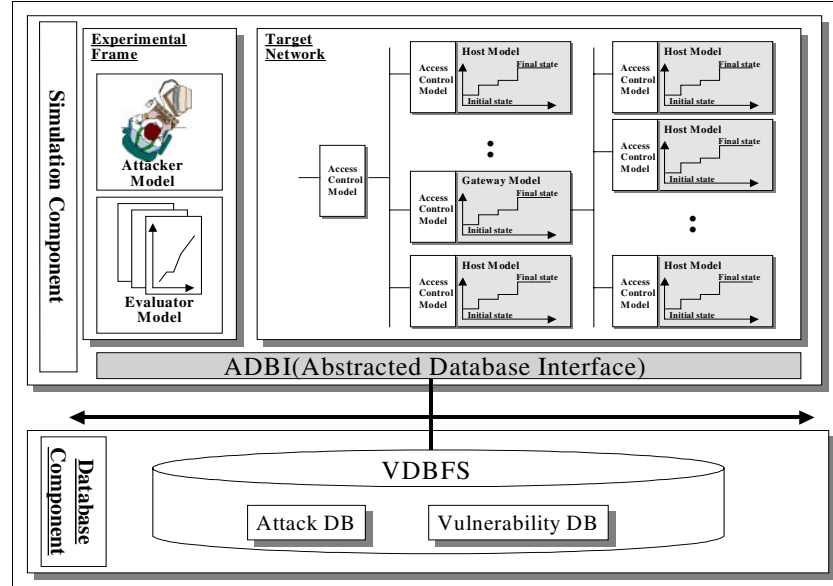


Fig. 1. The overall structure of system

### 3 Modeling of Computer Network for Vulnerability Assessment

#### 3.1 Overall Structure of Simulation Model

Fig. 1 shows the overall structure of the simulation system. It consists of simulation component and database component. The simulation component consists of EF(Experimental Frame) which is vulnerability assessment environment and target network model. The database component contains data, which are used at simulation execution time. The database component is named as VDBFS(Vulnerability DataBase For Simulator) and it consists of Attack DB and Vulnerability DB. The ADBI(Abstracted DataBase Interface) is designed to interface simulation component with VDBFS.

Left side of Fig. 2 shows the simulation processes - model design, model execution and model analysis. At the model design process, the models in model base are extracted and overall simulation model is constructed, and the data in VDBFS are used to execute the model. During the execution process, the data related with vulnerability assessment are gathered, and they are used at the model analysis process. The right side of Fig. 2 shows that the whole scope of process related with vulnerability assessment simulation. The processes in the cross shape are for end user of simulation system and the other processes are research scope of this paper. We have two main research topics, one is construction of VDBFS and the other is design and implementation of models in model base. The remainder of this chapter is related with VDBFS and model base construction. In the VDBFS construction, the cause-effect model and vulnerability analysis method is presented. In the model base construction, the design of each model is explained focusing on the EF.

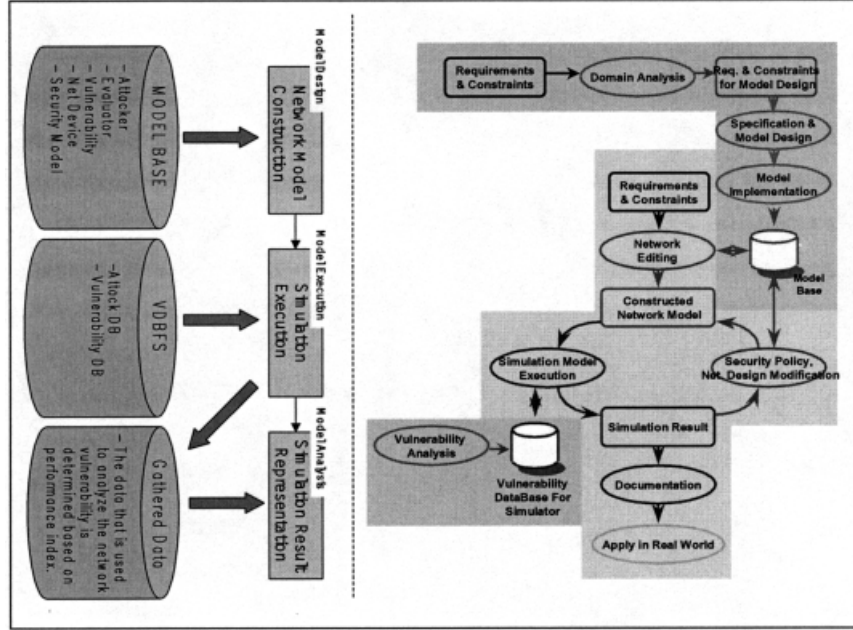


Fig. 2. Modeling and Simulation Process

### 3.2 VDBFS (Vulnerability DataBase For Simulator) Construction

**Cause-Effect Model for Simulation:** Cause-effect model for security simulation is to model what events inevitably cause security policy violations. In this research, we define causes as vulnerabilities and attacks, and effects as target system's state transitions. At a higher level, target system's state changes initial state into final compromised state. At a lower level, an attack is composed of a sequence of actions. Each action exploits system's vulnerability, and this causes state transition. In our simulation, attacks are modeled by attacker model component, and vulnerabilities, states and state transitions are modeled by vulnerability model component. To meet our research goal, we introduce CV (Compound Vulnerability) and AV (Atomic Vulnerability) as a analysis method. In the design of CV and AV, we refer Bishop's approach. While Bishop's approach is a kind of speculation, we extend and implement these concepts to actual simulation system. We classify AVs using Category and Type attributes. Also we represent CV by logical composition of AVs not by a simple set. The cause-effect model serves as behavioral knowledge of the simulation model components and this is implemented as VDBFS (Vulnerability DataBase For Simulator).

**CVs and AVs:** The definitions of CVs and AVs are as follows:

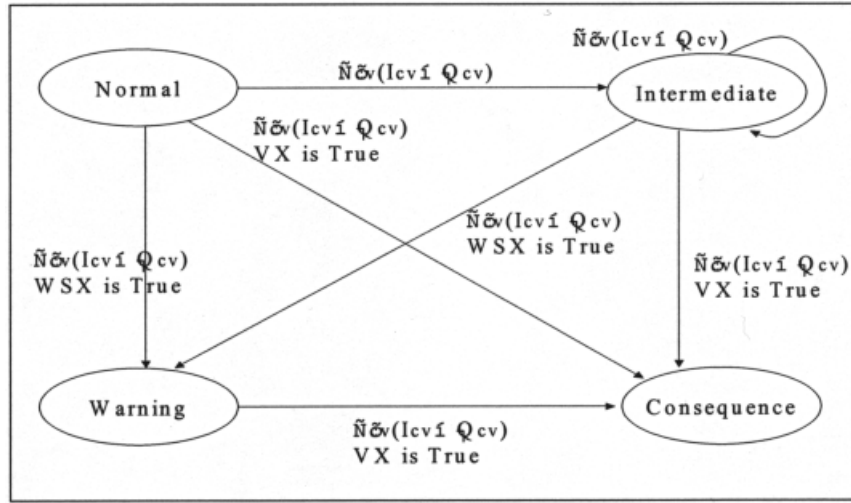
Compound Vulnerability:  $CV = \{I_{cv}, Q_{cv}, \delta_{cv}, WSX, VX\}$

Where,

$I_{cv} = \{I_{cv_1}, I_{cv_2}, \dots, I_{cv_n}\}$   
 $Q_{cv} = \{\text{Normal, Intermediate, Warning, Consequence}\}$   
 $\delta_{cv}: I_{cv} \times Q_{cv} \rightarrow Q_{cv}$   
 WSX: warning state vulnerability expression  
 VX: vulnerability expression

In the definition of CV,  $I_{cv}$  is a set of attack input sequences.  $Q_{cv}$  has four essential states that are meaningful in the simulation. Normal state is a state in which a target system is waiting for input packets. When the target system is under attack, system's state is Intermediate. The Warning state means that probability of exploitation occurrence is beyond a specific level, and the system can be an abnormal state by a simple malicious action. Consequence state is a goal state, which means the target system is exploited by attacker.  $\delta_{cv}$  is state transition function and defined as shown in Fig. 3.

A CV is represented by logical composition of AVs. VX holds the expression. An expression is composed of AVs and five binary logical operators. We evaluate the expression by calling AV objects. If this expression is evaluated as TRUE, it means that the target system that has this vulnerability is exploited by attack action sequence and state transition to compromised state occurs in the model.



**Fig. 3.** State and state transition of CV

WSX is warning state vulnerability expression. Syntax of WSX is the same as VX's. If this expression is TRUE, state transition to warning state occurs.

Atomic Vulnerability :  $AV = \{I_{av}, Q_{av}, \delta_{av}, \text{Type}, \text{Category}\}$

Where,

$I_{av} = \{I_{av_1}, I_{av_2}, \dots, I_{av_n}\}$

$Q_{av} = Q(\text{initial state}) \cup Q(\text{final state})$

$\delta_{av} : I_{av} \times Q(\text{initial state}) \rightarrow Q(\text{final state})$

Type: {Fact, NonProb, Prob}

Category: {Generic, Application-Specific, System-Specific}

**Table 1.** Logical operators for VX

AND Relation (AND)	To represent vulnerability exploited if both AVs are true
OR Relation (OR)	To represent vulnerability exploited if either or both AVs are true
Probabilistic OR Relation (POR)	To represent vulnerability exploited if either or both AVs are true. But each AV has weight value that accounts for vulnerability of target system for each AV(from 0 to 1).
Sequential AND Relation (SAND)	To represent vulnerability exploited if one AV at front is true and then the other AV is true sequentially.

In the definition of AV,  $Q_{av}$  is a set of states.  $Q$  (initial state) is a subset of  $Q_{av}$  and has special states NONE and ANY.  $Q$  (final state) is a subset of  $Q_{av}$  and has a special state NONE.  $I_{av}$  is a set of attack input sequences to AV.  $\delta_{av}$  is a state transition function. Identification of states and attack inputs is relevant to future application [3]. If goal of vulnerability analysis is to develop automated tools such as vulnerability scanner, abstraction level should be low. Low level requires precise and well-defined identification. If the tools are to be run with much human intervention and analysis, abstraction level may be higher. Abstraction level of this research is related with simulation methodology. We are modeling systems at a conceptual level of Ye's process control approach [6]. The conceptual level describes security-critical states and state transitions of an entity. Attack inputs are formalized in the form of [command parameter] like `lpd [file begin with "\cf\" and execute command]`.

Type and Category are bases for classifying the corresponding AV. An AV is one of three type; Fact, NonProb or Prob. A Fact AV has no input (NONE) and no state (NONE). Therefore, a Fact AV's  $\delta_{av}$  is  $\delta_{av}(\text{NONE}, \text{NONE}) = \text{NONE}$ . This type explains the corresponding AV's origin. NonProb and Prob type AVs explain whether these AVs are exploited probably or deterministically. Category is Generic, Application-Specific for specific application, System-Specific for specific OS or H/W.

We present two examples to show how the CV and AV are defined. For each example, we describe vulnerability and then define it as one CV, several AVs and relation among the AVs.

**Example One.** libedit searches for the `.editrc` file in the current directory instead of the user's home directory, which may allow local users to execute arbitrary commands by installing a modified `.editrc` in another directory [7]. CVE code is CVE-2000-0595. Vulnerable S/W is FreeBSD 4.0 and earlier. An attacker exploit this vulnerability by executing `/usr/sbin/nslookup` or `/usr/bin/ftp` which are statically or dynamically linked with libedit. This vulnerability is decomposed into 3 AVs. First, AV#1 represents that libedit searches for the `.editrc` file in the current directory instead of the user's home

directory. AV#1 explains why libedit in FreeBSD 4.0 is vulnerable. AV#2 represents that an attacker modified .editrc in another directory. In AV#3, an attacker gains root access by execute arbitrary commands in a modified .editrc.

$CV_{CVE-2000-0595} = \{Icv, Qcv, \delta cv, WSX, VX\}$   
 $Icv = \{\text{libedit} [\text{.editrc}], \text{libedit} [\text{arbitrary command}]\}$   
 $WSX = AV\#1 \text{ and } AV\#2$   
 $VX = AV\#1 \text{ and } AV\#2 \text{ and } AV\#3$   
 $Qcv, \delta cv \text{ and } AVs \text{ are shown in Fig. 4.}$

**Example Two.** Buffer overflow in Berkeley automounter daemon (amd) logging facility provided in the Linux am-utils package and others[7]. Vulnerable S/W is FreeBSD 3.2 and RedHat Linux 6.0. CVE code is CVE-1999-0704. There is a remotely exploitable buffer overflow condition in the amd daemon under several operating systems. Amd is a daemon that automatically mounts filesystems whenever a file or directory within that filesystem is accessed. Filesystems are automatically unmounted when they appear to have become quiescent. The vulnerability is in the log functions of the daemon [8]. This is an example of buffer overflow. This vulnerability comes from a failure to check bounds when copying data into a buffer(AV#4). At the same time this can be occurred because an attacker is allowed to alter the return address in stack(AV#5). If an attacker calls RPC amd with synthesized buffer, an attacker can gain root access(AV#6).

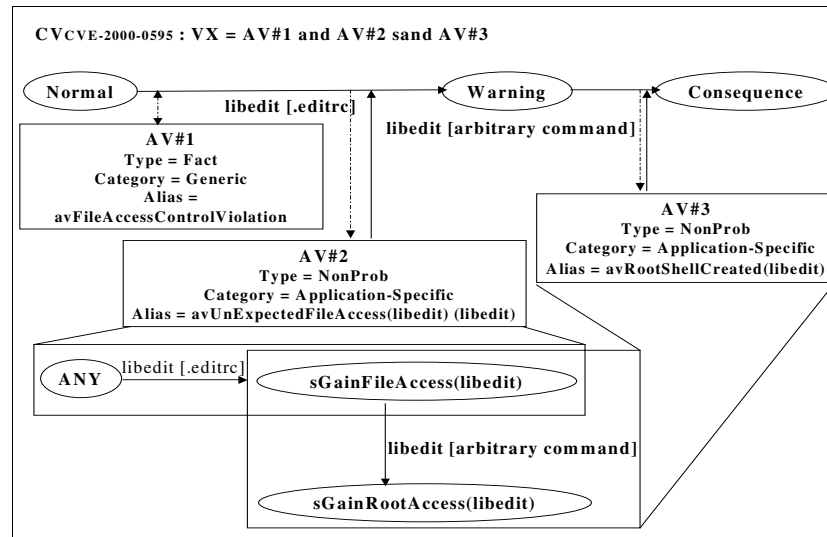


Fig. 4. Example one. State transitions of CV and AVs

$CV_{CVE-1999-0704} = \{Icv, Qcv, \delta cv, WSX, VX\}$   
 $Icv = \{\text{amd} [\text{shellcode}]\}$



WSX = AV#4 and AV#5

VX = AV#4 and AV#5 and AV#6

Qcv, δcv and AVs are shown in Fig. 5.

**VDBFS:** VDBFS is composed of Attack DB and Vulnerability DB. In Vulnerability DB, there are four tables: VictimInfo table for OS, OS version, H/W platform, installed S/Ws and their versions; VulnerabilityInfo table for vulnerability information such as CVE code, published date; AtomicVulnerabilityInfo for AVs; SolutionInfo for information about patch and remediation. In Attack DB, there are also four tables: AttckInfo for information about attacker; InputTypeInfo for making attack packets; CommandInfo for commands; AttackUseVulnerability table for relationship between attack and vulnerability. We select carefully and analyze more than one hundred vulnerabilities based on CVE code to validate simulation model components, to generate useful attack scenarios and to be worthy of vulnerability database itself.

### 3.3 Model Base Construction

In this section, The Design of the model components is explained, the model base includes EF (Experimental Frame) Model, Host Model and Security Model.

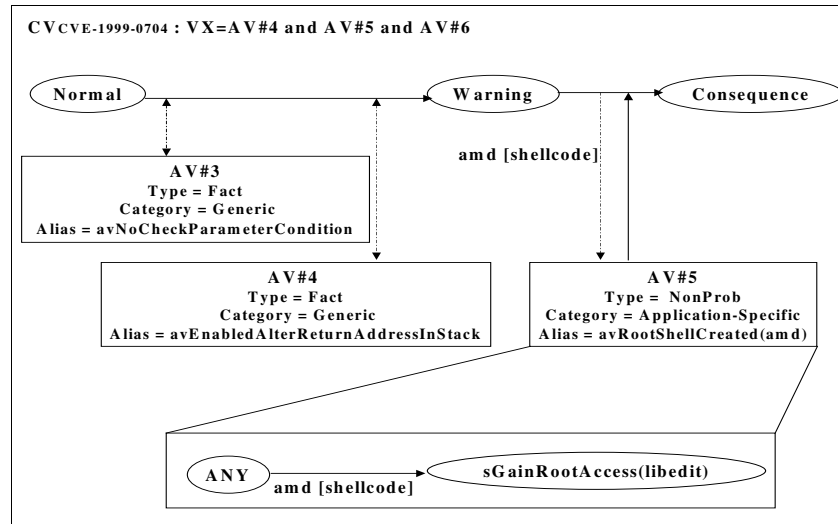


Fig. 5. Example two. State transitions of CV and AVs

**Experimental Frame Design:** EF (Experimental Frame) is a specification of the conditions under which system is observed or experimented [4]. Usually, EF is a frame that interacts with evaluation target to obtain the data of interest under specified

condition. The EF usually has two type of component: generator, which generates input segments to the system; transducer, which observes and analyzes the system output segments. Fig. 6 represents the EF that is used for vulnerability assessment. It consists of Attack Simulator model, which generates attacks against the target network and Evaluator model, which observes and analyzes the model's reaction. The Attack Simulator consists of Attacker model and Input Generator model. The Evaluator Model receives generated attack information from Attacker Model and reaction of network model through port 'solvd'.

The core of Attacker Simulator is knowledge processing facility in Attacker model. To make it possible for the Attacker model to process knowledge, we make use of the knowledge-based modeling methodology that is explained as interruptible atomic-expert model [10].

The Evaluator Model has two roles to get to the objectives of simulation. One is to gather the data from the target network and the other is to analyze the data and to show the security-related characteristics of the network. In the model design, we can see the simulation-based security analysis method, its limits and the method of overcoming it. In the analysis method description, we should categorize two types of method, because the simulation time unit is different and there is definitely difference in cause of exploitation. The two analysis methods are as follows.

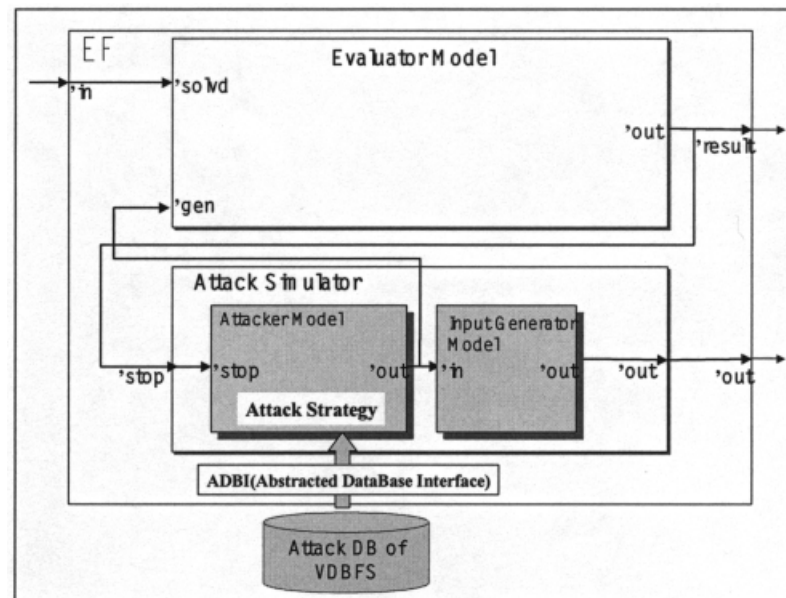


Fig. 6. Experimental Frame Design

*Flooding Attack Analysis:* Flooding attack is that attacker effectively disables server by overwhelming it with connection requests or transferring of data. The achievement of the attacker's goal (Denial of Service) is determined by the resource of the server and the severity of attack (number of attack input per time unit). In this circumstance, we defined performance index that represents the host's vulnerability against flooding attack. The degree of flooding attack vulnerability is described as (1).

$$\begin{aligned}
 V_{\text{flooding}}^k &= \frac{1}{T_{\text{CapaSatura tion}}^k} \\
 T_{\text{CapaSatura tion}}^k &= \frac{R^k}{A^k - (A^k \times FR^k)} \\
 V_{\text{flooding}}^{\text{host}} &= \frac{\sum_{k=1}^{NP} \beta_k V_{\text{flooding}}^k}{NP}
 \end{aligned}
 \tag{1}$$

Where,

- $V_{\text{flooding}}^k$  : Degree of Vulnerability of  $K_{th}$  protocol in host
- $V_{\text{flooding}}^{\text{host}}$  : Degree of Vulnerability of Host
- $NP$  : Number of Protocol
- $\beta_k$  : Coefficient to Calculate the Vulnerability Contribution of  $K_{th}$  protocol
- $A^k$  : Attack Input Generation Rate (input/msec) of  $K_{th}$  protocol
- $T_{\text{CapaSatura tion}}^k$  : Capacity Saturation Time (msec) of  $K_{th}$  protocol
- $R^k$  : Allotted Resource of  $K_{th}$  protocol
- $FR^k$  : Filtering Rate of Access Control List of  $K_{th}$  protocol

where,  $0 \leq FR^k \leq 1$

*S/W Vulnerability Exploitation Analysis:* In the S/W vulnerability analysis, we should assume the problem domain of simulation as a selected cause-effect set in VDBFS. Since there are a number of vulnerabilities in the S/W in the information infrastructure, it is impossible to estimate all the vulnerabilities in all S/Ws. Another assumption of this simulation is that the configuration of the s/w related with vulnerability is default value and it recommends sound configuration as a simulation result. The second assumption is inevitable in this simulation that does not utilize any scanning tools. But, if we make use of scanner to gather the configuration data, we can overcome this limitation. But, limitation in the first assumption still exists though we make use of the scanning tools, because there is same limitation in scanning tools whose problem domain is limited in vulnerability database. Since we do our simulation under these assumptions, we should analyze the simulation result based on the limited data. So we should define several expressions to analyze the vulnerability of host. Especially, in the S/W vulnerability analysis, we should consider the relation between universal set of vulnerability in real world and selected set of vulnerability in simulation world.

$$\begin{aligned}
 V_{s/w}^{universal} &= \sum_{k=1}^n I_k \bullet e(k) \\
 V_{s/w}^{selected} &= \sum_{k=1}^m I_k \bullet e(k) \\
 V_{s/w}^{unknown} &= V_{s/w}^{universal} - V_{s/w}^{selected} = \sum_{k=m+1}^n I_k \bullet e(k) \\
 e(k) &\begin{cases} 1 & \text{When } K_{th} \text{ Vulnerability is exploited} \\ 0 & \text{Otherwise} \end{cases} \quad (2)
 \end{aligned}$$

Where,  $n$ : Number of Element of Universal Vulnerability Set  
 $m$ : Number of Element of Selected Vulnerability Set  
 $V_{s/w}^{universal}$ : Degree of Universal S/W Vulnerability in a Host  
 $V_{s/w}^{selected}$ : Degree of Selected S/W Vulnerability in a Host  
 $V_{s/w}^{unknown}$ : Degree of Unknown S/W Vulnerability in a Host  
 $I_k$ : Impact of the  $k_{th}$  Vulnerability in a Host

In the (2), the three types of vulnerability and the relation among them are presented. In our assumption, our target system is more vulnerable than the simulation result, since it has unknown vulnerabilities that don't contain in VDBFS.

$$\begin{aligned}
 V_{s/w}^{universal} &= V_{s/w}^{selected} \times \left(1 + \frac{V_{s/w}^{unknown}}{V_{s/w}^{selected}}\right) \\
 UnknowVulFactor(\gamma) &= \frac{V_{s/w}^{unknown}}{V_{s/w}^{selected}} = \frac{\sum_{k=m+1}^n I_k \bullet e(k)}{\sum_{k=1}^m I_k \bullet e(k)} = \frac{\sum_{k=m+1}^n e(k)}{\sum_{k=1}^m e(k)}, \text{ where, assume all } I_k = 1.
 \end{aligned}$$

The value of  $\gamma$  can be simplified in these three case,

$$\gamma = \begin{cases} 0 & \text{When, all value of } e(k) \text{ in } unknown \text{ case is 0 or } n \text{ is equal to } m. \\ \frac{n-m}{m} & \text{When, average of } e(k) \text{ in } unknown \text{ case is same as } e(k) \text{ in selected case} \\ \frac{n-m}{\sum_{k=1}^m e(k)} & \text{When, all } e(k) \text{ in } unknown \text{ case is 1} \end{cases} \quad (3)$$

To calculate the unknown vulnerability effects, the unknown vulnerability factor( $\gamma$ ) is defined in simulation analysis, as shown in (3). We assume all value of  $I_k$  as 1, because our evaluating target network is the information infrastructure and all exploitation has large impact to them. Also, we want to avoid the subjective definition of the severity of exploitation. (3) presents the unknown vulnerability factor( $\gamma$ ) as a function of  $n$  (number of universal vulnerabilities),  $m$  (number of known vulnerabilities) and the summation of  $e(k)$ .

Though theses expressions are simple, we can see the relation of unknown vulnerability and known vulnerability through it. Also, these expressions should be specialized, when this simulation system is applied to assess a specific information infrastructure. In the next chapter, we show the characteristics of these expressions using our simulation result.

**Host Model Design:** The formal description of Host Model is as follows.

Host Model:  $H = \{T, I, \Omega, Y, Q, \Delta, \phi\}$

Where,

T: Simulation Time.

I:  $I = \{I_1, I_2, I_3, \dots, I_m\}$ : Attack Set, that have m Attack elements.

$\Omega$ : (I, T): Enable input segment set.

Y: Output set. Output values are response of external input.

Q:  $Q = \{Q_1, Q_2, Q_3, \dots, Q_n\}$ : Host model state set

$\Delta$ :  $\Omega \times Q \rightarrow Q, Q \rightarrow Q$ : State Transition Function, The state transition is based on the vulnerability expression and the state of protocol buffer.

$\phi$ :  $I \times Q \rightarrow Y, Q \rightarrow Y$ : Output Function.

Host Model consists of these four components.

- Set of host characteristics that describe the characteristics of host.
- Access Control Model that has facility of user authentication and network access control.
- Protocol Buffer Model that used to indicate the state of resource such as backlog size in system.
- Vulnerability Model that represents the vulnerabilities in host.

Set of host characteristics contains abstracted information of host such as hardware platform, IP address, OS and application information. Protocol Buffer Model is used to indicate the resource occupation rate in Host Model, when DoS attack is occurred. Buffer Occupation Rate is calculated by the expression (4),

$$BufferOccupationRate = \frac{CurrentBufferSize}{TotalBufferSize} \times 100 \text{ (%) } \quad (4)$$

The Buffer Occupation Rate and threshold value are used to make decision to transit the state from normal state to abnormal state. Vulnerability Model consists of CV List and AV List. The AV list consists of system-AV and application-AV and CV has vulnerability expression that is composed of AVs and logical operators.

Fig. 7 shows the timing diagram of host model. As shown in the Fig. 7, there are critical attack inputs at t1 and t2, and the state transition occurs from normal to warning and from warning to abnormal. In this case, the vulnerability expression is fired by the critical attack input at t3. Also, there is flooding attack from t4 to t6, which affects the Buffer Occupation Rate in (4), and there are state transitions from normal to warning and from warning to abnormal. In the flooding type attack, the amount resource of the buffer and reset timer are critical factor to decide the state transition.

There are four types of security models that have facility that controls the access of the external input. First type of model is user authentication model and second type of model is network access control model, which are components of OS Security Model. The third type of model is static and dynamic packet filter model and the fourth type of model is application proxy model. These packet filter and proxy model are components of Firewall Model. Fig. 8 shows the behavioral feature of this mechanism. When an external input is insert into the model, it infer with security rules and external input. The result of the inference is to drop or accept the input.

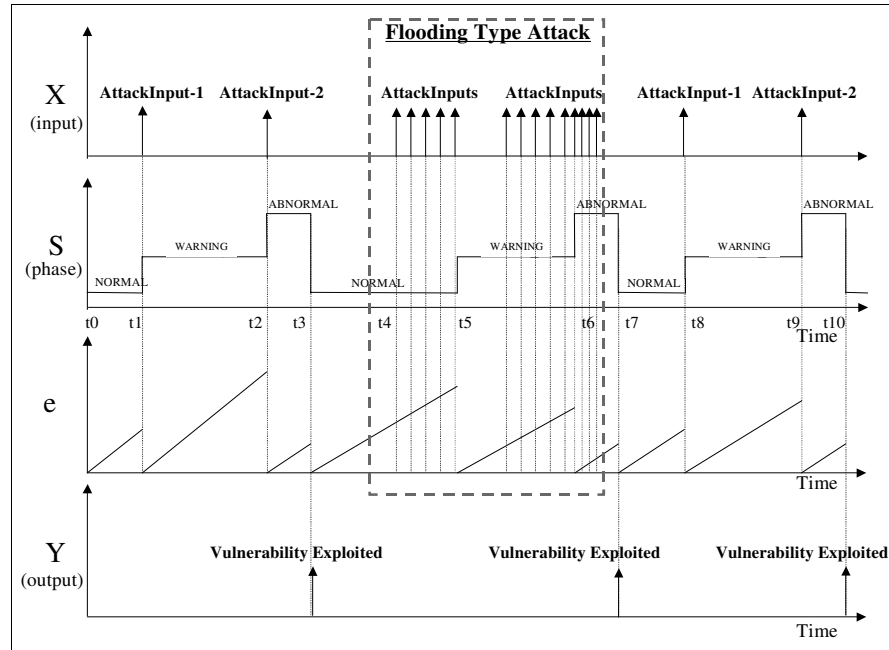


Fig. 7. Timing Diagram of Host Model

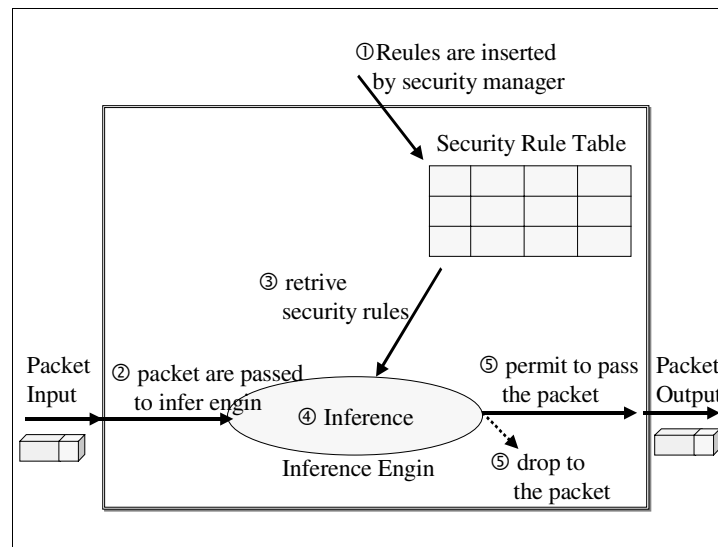


Fig. 8. Behavior of Security Model

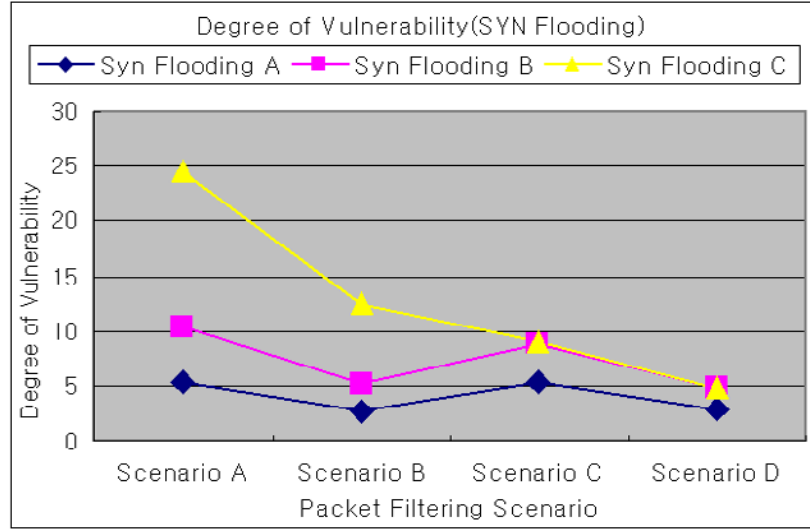


Fig. 9. Degree of Vulnerability (SYN Flooding )

## 4 Simulation Executions and Analysis

### 4.1 Flooding Attack Simulation

In this section, we show a simulation example to validate our simulation model. Fig. 9 shows a simulation result in SYN Flooding attack. We select two attacker-systems in two different domains. The SYN Flooding attack C is more severe than attack B by two times and attack B is more severe than A by two times. In our simulation model, there are two points that access control rules can be applied to. The one is Firewall Model that is located at the front of the network and the other is Host Security Model that is component of Host Model. We specify four security scenarios: First, in the scenario A, we do not use any access control rules. Second, we deploy rules that filter the SYN packet from unreliable hosts in scenario B. Third, we deploy access rate rules in scenario C. Fourth, we deploy both access control rules in scenario B and C in the scenario D.

In this simulation, the inter-arrival time of attack input is taken to be exponential random variables with a mean of each value in Table. 2. We executed 5 times using different seed values for each simulation. The scenario B and D of Fig. 9 shows that when SYN packets from unreliable host are filtered, about a half of the vulnerability degree decreases. Also, when we deploy the access rate rules in scenario C and D, there is limit in the decrease of vulnerability degree. The threshold of access rate rules causes this limit. When the threshold value is large, there is no effect in the degree of vulnerability at the case of SYN Flooding A and B. Also, as the defense scenario becomes weak, the variance of the degree of vulnerability becomes larger.

**Table 2.** Inter-arrival time of attack input in attack scenarios

Attack Scenario Attacker Host IP	SYN Flooding A	SYN Flooding B	SYN Flooding C
xxx.xxx.160.4	Exponential dist. Mean = 100msec	Exponential dist. Mean = 50msec	Exponential dist. Mean = 25msec
yyy.yyy.150.4	Exponential dist. Mean = 100msec	Exponential dist. Mean = 50msec	Exponential dist. Mean = 25msec

#### 4.2 S/W Vulnerability Exploitation Attack Simulation

In our simulation system, we defined a vulnerability estimation expression that includes the unknown vulnerability factor in (3). In this section, we will show characteristics of the expression through the simulation result.

In the Fig. 10, the three case of  $e(k)$  in (3) are plotted. As shown in Fig. 10, the vulnerability assessment result is 15 degree of vulnerability. The rate of exploitation is about 48.4% in this simulation. At the end of simulation, user can examine the Atomic Vulnerability that contributes to the exploitations.

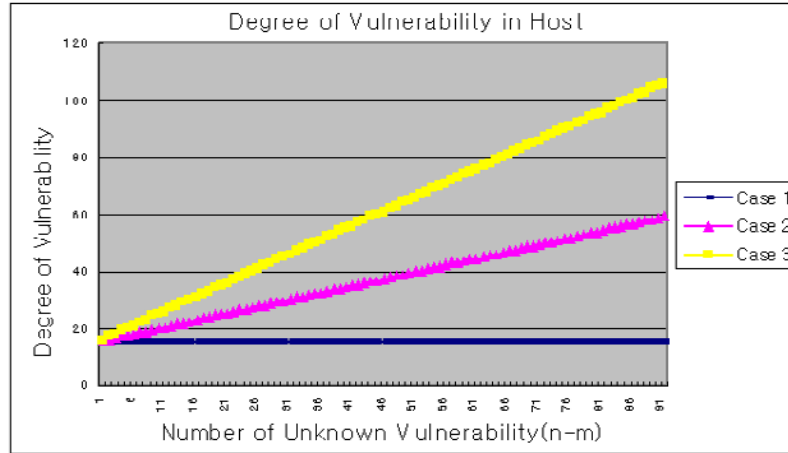
Using this simulation result, we can plot the variance of degree of vulnerability as the number of unknown vulnerability increases. In the Fig. 10, the Case 1 means that the probability of unknown vulnerability exploitation is zero and the degree of vulnerability has no change. But, in the Case 2 and 3, the degree of vulnerability increases, as the number of unknown vulnerability becomes larger. The degree of vulnerability in universal set is between the case 1 and case 3 for each number of unknown vulnerability. If we make smaller the slope of the Case 3 or Case 2, the difference between each case becomes smaller. As shown in (3), there are two methods to achieve small slope, the one is to increase the value  $m$  in Case 2 and Case 3 or increase the summation of probability value,  $e(k)$  in Case 3.

Fig. 11 shows the variance of unknown vulnerability factor ( $\gamma$ ), as the number of selected vulnerability increases. In the plotting of Fig. 11, we assume that the number of element in universal set is 1000, and the rate of exploitation is 48.4%. The Case 1 is omitted, because the unknown vulnerability factor is zero. Fig. 11 shows that the unknown vulnerability factor decreases steeply at first, as the number of selected vulnerability increases, and the value converges to 0, as the number of selected vulnerability increases. Since the unknown vulnerability factor ( $\gamma$ ) implies the correctness of the simulation result, if we construct a VDBFS containing appropriate number of vulnerability information, we can get reliable simulation results under our assumption in (3).

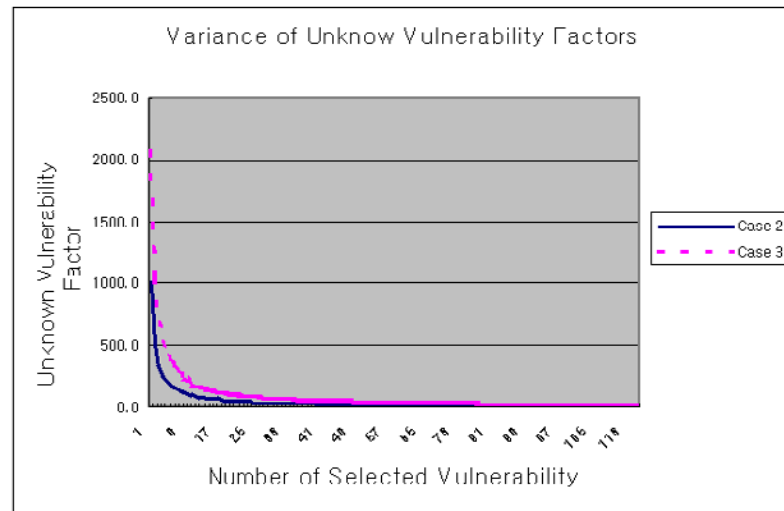
## 5 Conclusions and Discussion

Our research objective is to construct a vulnerability assessment simulation system for protection of critical information infrastructure. To achieve it, we should analyze and





**Fig. 10.** Degree of Vulnerability in Host (S/W Vulnerability Exploit Case)



**Fig. 11.** Variance of Unknown Vulnerability Factors ( $\gamma$ )

customize the vulnerability information and construct VDBFS. In the construction of VDBFS, the Compound Vulnerability, Atomic Vulnerability and Vulnerability Expression are defined to represent the vulnerability information adequate to the simulation model. Also, we construct a model base of simulation system. The components of model base are designed and implemented such as EF, host model, network device model, security model and so on. Especially, in the EF model, interruptible atomic-expert model is exploited in the design of attacker model, and the Evaluator model is designed to estimate the degree of vulnerability of system.

In the VDBFS construction, the future work is to extend the VDBFS and makes its reserved information as much as that of scanning tools. Also, we should apply this

simulation system to a specific information infrastructure and specialize the Evaluator Model. The simulation result should be compared with real world phenomenon and we should tune the presented expressions. In the EF design, we should develop other performance indexes that can be extracted through the execution of the simulation models.

## References

1. F. Cohen, "Simulating Cyber Attacks, Defences, and Consequences," *Computer & Security*, Vol.18, pp. 479–518, 1999.
2. A. P. Moore, R. J. Ellison and R. C. Linger, "Attack Modeling for Information Security and Survivability," Technical Report No. CMU/SEI-2001-TR-001, Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, March, 2001.
3. M. Bishop, "Vulnerabilities Analysis", *Proceedings of the Recent Advances in Intrusion Detection*, September, pp. 125–136, 1999.
4. B. P. Zeigler, H. Praehofer and T. Kim, "Theory of Modeling and Simulation, Second Edition", Academic Press, 2000.
5. A. M. Law and W. D. Kelton, "Simulation Modeling and Analysis, Third Edition", McGraw Hill, 2000.
6. N. Ye and J. Giordano, "CACA - A Process Control Approach to Cyber Attack Detection", *Communications of the ACM*, Vol.44(8), pp. 76–82, 2001.
7. <http://icat.nist.gov>
8. <http://online.securityfocus.com>
9. Avolio and Blask, "Application Gateways and Stateful Inspection : A Brief Note Comparing and Contrasting," *Trusted Information System*, Inc., 1998.
10. T. Cho and H. Kim, "DEVS Simulation of Distributed Intrusion Detection System," *Transactions of the Society for Computer Simulation International*, vol. 18, no. 3, September, 2001.

# Pseudonymizing Unix Log Files<sup>\*</sup>

Ulrich Flegel

University of Dortmund, D-44221 Dortmund, Germany  
Ulrich.Flegel@udo.edu

**Abstract.** Unix systems in many cases record personal data in log files. We present tools that help in practice to retrofit privacy protection into existing Unix audit systems. Our tools are based on an approach to pseudonymizing Unix log files while balancing user requirements for anonymity and the service provider's requirements for accountability. By pseudonymizing identifying data in log files the association between the data and the real persons is hidden. Only upon good cause shown, such as a proceeding attack scenario, the identifying data behind the pseudonyms can be revealed. We develop a trust model as well as an architecture that integrates seamlessly with existing Unix systems. Finally, we provide performance measurements demonstrating that the tools are sufficiently fast for use at large sites.

## 1 Introduction

In times when companies appreciate the value of personal information, customers will increasingly appreciate the protection of their valuable personal data. Survey after survey shows that privacy concerns with respect to the use of the internet are on the rise [5,6,7]. A gap has been shown between what kind of identifying data users think ought to be collected versus what kind they think actually is collected when accessing a web site. The biggest disagreement is shown for some of the data commonly logged by web servers such as machine name or IP address, browser and OS [7]. Most users would rather not access a site than reveal personal information asked by the site for registration [7]. The vast majority of users strongly values being able to anonymously use internet services [7]. This situation calls for privacy enhancing technologies that can be used today and that help service providers to comply with user expectancies as well as with legal regulations concerning personal data. Service providers that can credibly assure their customers the protection of their personal data may gain a competitive advantage over those that can't. Consider as a simple example a web site where the service provision requires no user registration and no billing. There is a reasonable user expectancy to be able to access the site anonymously.

Furthermore, prevalent statutory regulations confine the processing<sup>1</sup> of personal data. Considering our example web site the users would not only have

---

<sup>\*</sup> This work is currently partially funded by the German Research Council (DFG) under grant number Bi 311/10-2.

<sup>1</sup> Processing, in relation to personal data, covers virtually the entire data life cycle from collection, through to erasure of the data when no longer required.

reason to expect, but they would also have a right to be able to use the site anonymously, based on the pertinent legal situation in many western countries. Refer to [3] and [8] for a more detailed discussion of the related legal issues. Unfortunately it is insufficient to merely legally interdict the misuse of personal data. Once personal data has been collected its processing is hard to control in a non-technical way, since any processing can occur on arbitrary copies of the data. Considering the example web site it is practically futile for a user to determine or even to prove that the organizational controls are ineffective wrt. the protection of his personal data.

Therefore we observe an increasing demand for strong technical enforcement of privacy principles. Multilaterally secure systems take into account and balance conflicting security requirements of all involved parties. Accordingly a multilaterally secure internet service allows customers to act pseudonymously as long as violations of the service security policy are still detectable. In return the service provider is allowed to recover the identities of customers upon good cause shown, which supports the hypothesis of violations of the security policy. Translated to the example web site this means that the IP addresses and potentially other identifying data are pseudonymized when the web server logs are generated. Only if a client exploits a vulnerability of the web server its IP address and other identifying data is recoverable.

This paper demonstrates that the approach to threshold-based identity recovery we designed in [4,3] is actually implementable and applicable in practice. Specifically the contributions of this paper are the following:

- the development of a practical architecture to integrate our pseudonymizer with existing *syslog* auditing infrastructures;
- the exploitation of a trust model supported by actually deployed systems;
- the provision of tools that implement our concepts and architecture;
- a detailed evaluation of the performance of our pseudonymizer, indicating that it lives up to the requirements posed by the audit data volume generated by a busy server system at the Center for Communication and Information Processing at the University of Dortmund.

The remainder of this paper is organized as follows: Section 2 justifies our design decision to pseudonymize *syslog* audit data and introduces the relevant concepts. In Sect. 3 we detail the design decisions for integration of the pseudonymizer, explain the trust model and outline our approach to pseudonymization. The architecture of our implementation also is illustrated in Sect. 3. In Sect. 4 we evaluate the performance of the implementation. We position our contribution wrt. related work in Sect. 5 and conclude in Sect. 6.

## 2 A Case for *syslog*-Pseudonymization

Most modern operating systems offer comparable audit components, e.g. AIX, BSDs, HP-UX, Linux, Solaris, Windows-NT and its descendants. All of these record the identity of the actor of an event or at least identifying features of

users. Audit components specifically designed for compliance with the Trusted Computer System Evaluation Criteria (TCSEC) (C2 or higher) [9,10] or the Common Criteria (FAU\_GEN1.2 and FAU\_GEN2) [11] are required to record information identifying the originator of an event. Hence, all of these audit components record personal data and need to be considered for pseudonymization. In addition to the audit components of the operating system, application processes record audit data containing personal data, such as web servers and mail exchangers.

## 2.1 Event Sources

When considering pseudonymization of audit data all relevant event data sources mentioned above should be covered. With our approach we aim at a non-invasive, easy to install tool that covers as many categories of audit data as possible. We therefore decided to build a pseudonymizer for *syslog* audit data. Due to its uniformity and availability in most significant Unixes many event sources in the Unix world leverage *syslog* for recording audit data. Also active network components such as routers provide for remote audit recording using *syslog*. Finally there are several third party products available for Windows driven systems to integrate with an existing Unix *syslog* infrastructure. By supporting *syslog* we can pseudonymize host-based, network-based and out-of-band audit data. Refer to [1,12,13] to learn more about *syslog*.

Event sources not using *syslog*, but directing their audit data to dedicated files are covered by our *redirector*. In practice with our tools we cover most of the audit data usually collected in Unix systems. For a more elaborate discussion on the diverse sources for audit data in Unix systems refer to [3].

## 2.2 Audit Data

As mentioned above, host-based, network-based and out-of-band audit data converge in *syslog*. The audit data collected via *syslog* describes events and conditions monitored by diverse components of the IT system. These components are categorized as *facilities* in *syslog* parlance. A facility that monitors an event or condition supplies a rating of its *severity* [13,12]. The *priority* of an event record is defined as the pairing of its generating facility and its severity. The severity ranges from debugging and informational over error conditions to emergency conditions. Mostly informational and diagnostics messages from the kernel and from network services are recorded via *syslog*. Refer to [1] for some annotated examples.

Also serious actions of users associated with login sessions are recorded by *syslog*. Depending on the system configuration a variety of other sorts of events may be reported to *syslog*. As an example the intrusion detection system *Snort* [14] is capable of reporting attack detections to *syslog*. Basically, *syslog* audit data is collected and used for troubleshooting and limited manual intrusion detection. Also some intrusion detection systems analyze *syslog* audit data, such as the *STAT Tool Suite* [15].

### 3 Embedding the Pseudonymizer

There are several opportunities for pseudonymization of *syslog* audit records along the path from the audit component to the final recipient. While it would be possible to integrate pseudonymization with all audit components, such an approach does not scale well. Also audit records should be pseudonymized before they reach the final recipient, which may be a privacy adversary.

Firstly, a straightforward approach is to pseudonymize the output of `syslogd` by reconfiguring `syslogd` to write into named pipes being read by pseudonymizers. The problem with this solution is, that a pseudonymizer is not able to determine the priority of the local output records of `syslogd`. This information is lost, but may be required for effective log analysis.

Secondly, there are two ways to pseudonymize audit records before they enter `syslogd`. We can wrap the *syslog* API function calls, but this approach does not catch audit records received from the kernel or via the network. Instead, a *wrapper* may pick up audit records from the *syslog*-sockets, while `syslogd` is configured to receive the pseudonymized audit records from the wrapper. We implemented such a pseudonymizing wrapper (see Fig. 1).

Thirdly, we can embed pseudonymization within `syslogd` by patching its source code or by replacing the existing `syslogd` by a pseudonymizing `syslogd`. The disadvantages of these solutions are that either the access to system-dependent source code of `syslogd` is required, which may not be readily available for all Unixes, or the existing implementation of `syslogd` may differ significantly from the replacement version. In addition to the wrapper we provide source code patches to optionally embed pseudonymization within `syslogd`.

Some audit components write audit records directly to files without involving `syslogd`. As long as our pseudonymizer can parse these records, it may also read from these files. Sometimes it is useful to consolidate the audit records of as many audit components as possible in one place, f.i. to use the same pseudonymizer for these records. We supply a *redirector*, which picks up the audit records from a named pipe or a growing file and deposits them by means of the *syslog* API.

#### 3.1 Trust Model

When considering audit data generated on the local host we are always confronted with the risk of integrity loss in case an attacker achieves sufficient privilege to manipulate audit data or the process of its generation. Even in the face of these problems *syslog* audit data today is still one very important source of information that is used to resolve attack situations or to gather early indications thereof.

In the same way in which an attacker can manipulate audit data or its generation, he may also corrupt the integrity of pseudonyms in pseudonymized audit data to evade later identification. We argue that we can rely on the pseudonyms generated on a host under attack as long as we can rely on the *syslog* data generated on that host. Stronger techniques for pseudonym generation are outside the

scope of this approach. Refer to Sect. 5 for related work. Pseudonymization can be used to make inferences on the pseudonymized data impractical. However, we cannot avoid inferences that take into account additional information outside the scope of pseudonymization.

Our approach to pseudonymization balances conflicting security requirements of at least two parties. On the one hand the users of an IT system wish to use it anonymously to protect their privacy. On the other hand the IT system owner's representative responsible for the system security requires accountability in the case of policy violations. Other interests in identifying users are not considered here, such as direct marketing by means of user profiling.

To put the mutual security requirements for anonymity and accountability not at risk, neither of the above mentioned parties can be allowed to control the audit components, `syslogd` and the pseudonymizer. Hence we introduce a third party denoted as the personal data protection official (PPO), who controls above mentioned system components (see the PPO domain in Fig. 1). The PPO is trusted by the users to protect their pseudonymity and he is trusted by the site security officer(s) (SSO) to ensure accountability in the face of a security incident.

In the current implementation audit components (with the help of *redirectors*) feed audit records via a *wrapper* or via *syslog* into the pseudonymizer (see Fig. 1 and Sect. 3.2). The pseudonymizer substitutes predefined types of identifying information, henceforth denoted as *features*, by pseudonyms. We exploit Shamir's threshold scheme for secret sharing to generate the pseudonyms. Owing to the properties of the threshold scheme we use, the features behind the pseudonyms can be recovered only under certain conditions later on (see Sect. 3.2). Thus, attacks on the pseudonymity of the system users are anticipated and resisted as soon as the pseudonymized audit records leave the pseudonymizer. Note that audit records should be pseudonymized on the same physical component where they are generated by the audit components. This may not be possible in some situations, f.i. when consolidating audit data generated by routers or Windows driven systems as is proposed in Sect. 2.1. In such cases additional measures must be taken by the PPO to provide a secure channel between the audit component and the pseudonymizer.

The pseudonymized audit data is delivered to and analyzed by the SSO to acquire indications for policy violations (see the SSO domain in Fig. 1). In order to support aforementioned analysis the pseudonyms of a given identifying feature are linkable wrt. a given suspicion. Given sufficient suspicion, the pseudonyms can be used to reveal the originally substituted features in the hope to gain more information helping to identify the perpetrator. In our approach sufficient suspicion is defined by a threshold on weighted occurrences of potentially attack-related events. The recovery of a feature is possible if and only if the corresponding threshold is exceeded. Thus, the SSO's requirement for accountability is met if the definition of sufficient suspicion is satisfied, but not otherwise, conforming to the user requirement for pseudonymity. The recovery of identifying features

is done by the *reidentifier*, which is – unlike the pseudonymizer – fed with the output files of `syslogd` (see Fig. 1).

### 3.2 Pseudonymizing *syslog*

The pseudonymizer receives audit records from `syslogd` containing identifying features. Since each feature may contribute with a specific weight to a scenario legitimating feature recovery, the pseudonymizer is able to distinguish different event types and associates each embedded feature type with one or more scenarios and the respective weight(s). The PPO specifies apriori knowledge about the syntax of events and features to be pseudonymized. In cooperation with the SSO, the PPO models suspicions justifying reidentification by specifying associations between features and scenarios, defining the weight(s) for the contribution of each feature to its associated scenario(s), as well as the scenario thresholds. This apriori knowledge is stored in the configuration files of our tools. Consequently, feature recovery is tied to the scenarios that are configured when the features are pseudonymized.

**Parsing:** To locate and pseudonymize identifying features in *syslog* audit records, we have to parse them. Since the syntax of the records is mainly determined by the originating audit component, recognition of different event and feature types is based on the evaluation of syntactical contexts, e.g. by means of regular expressions for pattern matching as defined in POSIX 1003.2 Sect. 2.8. Refer to [1] for the details.

When parsing audit records we use the apriori knowledge stored in the configuration files for our tools. By sequentially matching audit component, event type context, and feature type contexts, we isolate the features of the record that are to be pseudonymized. According to the assigned weight and scenario, a number of pseudonyms is generated, replacing the respective feature. The pseudonymized audit records retain the format of the unpseudonymized version.

**Generating Pseudonyms:** The basic idea of our approach to generating pseudonyms is to have the pseudonymizer split an identifying feature into as many pseudonyms as are needed to pseudonymize audit records containing this feature. The pseudonyms have the property, that given a number of pseudonyms equal to the associated scenario threshold or more, but not less, the reidentifier is able to recover the corresponding feature. Secret sharing threshold schemes are suitable to fulfill these requirements.

For our purposes we exploit Shamir’s threshold scheme, as described in detail in [16]. Owing to different conditions of deployment of Shamir’s threshold scheme, we make some modifications with regard to its application. For a more technically inclined description of these modifications and the basic algorithms used by the pseudonymizer and the reidentifier, refer to [4].

In a nutshell, the pseudonymizer encrypts each identifying feature and the corresponding decryption key is split into shares, which are used as pseudonyms



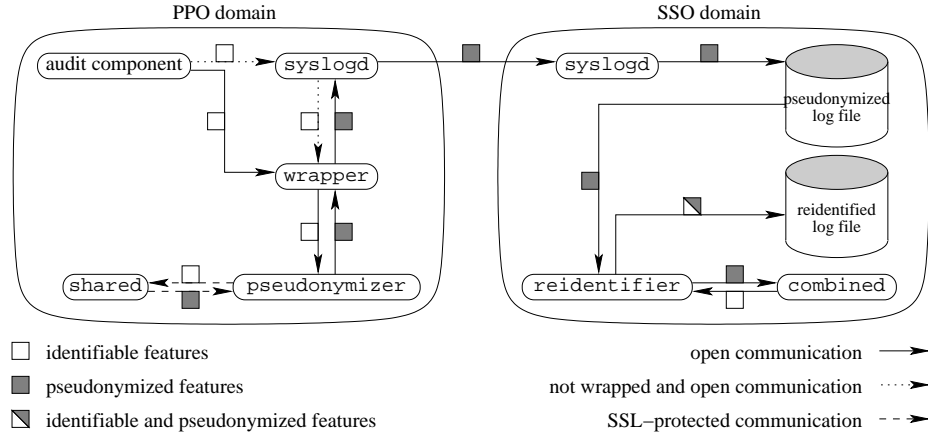


Fig. 1. Interplay with a wrapped syslogd

for that feature. The reidentifier uses Lagrange interpolation to recover the decryption key from a sufficient number of shares belonging to the same feature.

**Architecture:** We supply tools for the PPO and for the SSO, namely **redirector**, **wrapper**, **pseudonymizer**, **shared**, and **reidentifier**, **combined**, respectively. The tools are portable and have been used successfully on Solaris, OpenBSD and Linux.

According to Sect. 3.1 the audit components, **redirector**, **wrapper**, **syslogd**, **pseudonymizer** and **shared** are controlled by the PPO. The audit components are configured to generate only audit data required for sustained service provision, such as audit data indicating proceeding attacks against the site. The audit data is written to `/dev/log` (with the help of **redirectors**), to UDP port 514 and to other locations usually read by **syslogd**, f.i. `/dev/klog` for OpenBSD kernel audit data. In Fig. 1 the **wrapper** intercepts audit data at `/dev/log` and at UDP port 514<sup>2</sup>. Audit data that cannot be intercepted<sup>3</sup> is received by **syslogd** and sent on to the **wrapper** (see the dotted lines in the PPO domain in Fig. 1). The **pseudonymizer** is fed with audit data by the **wrapper**. This solution can be used if **syslogd** shall not be replaced or modified. Optionally we provide a patch for **syslogd** such that incoming audit data is sent directly to the **pseudonymizer** without involving the **wrapper**. For this solution **syslogd** needs to be replaced by a patched version (refer to [1] for details).

The audit data contains identifying features that need to be pseudonymized. The **pseudonymizer** parses each incoming audit record, determines the appro-

<sup>2</sup> We actually use UDP port forwarding to redirect packets sent to port 514 to the port where the **wrapper** listens.

<sup>3</sup> For example OpenBSD kernel audit data written to `/dev/klog` cannot be intercepted because the OpenBSD **syslogd** is hard-coded to read kernel audit data from `/dev/klog`.

priate scenarios and asks **shared** for pseudonyms for the features wrt. these scenarios. **shared** provides the cryptographic primitives, such as symmetric encryption and secret sharing. For encryption **shared** uses the OpenSSL crypto library [17] and the threshold secret sharing scheme is implemented using the GNU Multiple Precision Arithmetic library (GMP) [18]. **shared** may run on the same machine as the **pseudonymizer**, or it may run remotely. This allows several distributed **pseudonymizers** to use a central **shared** with the result that pseudonyms for a given identifying feature are linkable across the corresponding distributed audit components. The protocol used for communication between the **pseudonymizer** and **shared** transfers identifiable features. Since the PPO might not control the network between the **pseudonymizer** and **shared**, the channel is protected using the OpenSSL SSL/TLS library [19] when **shared** runs remotely. The **pseudonymizer** replaces the identifying features with the pseudonyms delivered by **shared**. The pseudonymized audit data is then handed to **syslogd** (via the **wrapper**). In the recommended setup **syslogd** sends the pseudonymized audit records to a remote **syslogd** running under the control of the SSO. Note that using the recommended setup audit data is not written into disk files before it is pseudonymized and received in the SSO domain.

In compliance with Sect. 3.1 the **syslogd** receiving pseudonymized audit records from the **pseudonymizer** as well as the **reidentifier** and **combined** are controlled by the SSO. **syslogd** receives the pseudonymized audit records and handles them according to the SSO's requirements. If the SSO wishes to be able to recover identifying features some time later, **syslogd** needs to be configured to write those audit records to a log file (see Fig. 1). When needed, the **reidentifier** is started manually or automatically on an alarm. It reads the log file, determines which pseudonyms satisfy their scenario and asks **combined** to recover the respective identifying features. **combined** provides cryptographic primitives, such as symmetric decryption and Lagrange interpolation, the reverse operation for the threshold secret sharing scheme we use. For decryption **combined** also uses the OpenSSL crypto library [17] and the Lagrange interpolation again is implemented using the GMP library [18]. **combined** may run on the same machine as the **reidentifier**, or it may run remotely. The protocol used for communication between the **reidentifier** and **combined** transfers identifiable features. Since identifiable features are transferred only if a scenario is satisfied, i.e. the SSO is allowed to see the features, there is no need for a channel providing confidentiality.<sup>4</sup> The **reidentifier** replaces all recoverable pseudonyms with the corresponding identifying features delivered by **combined**.

## 4 Performance

The **pseudonymizer** processes audit data as soon as it has been generated. It is thus required, that it is able to pseudonymize audit records sufficiently fast to avoid a bottleneck, which might impair overall system performance. In contrast,

<sup>4</sup> If the SSO does not control the network between **combined** and the **reidentifier** a confidential channel can be provided.

the `reidentifier` is sparsely used and needs not to satisfy real-time requirements.

The performance measurements were conducted on a single processor Pentium III 650MHz machine with 256MB RAM and a 100Mbps Fast-Ethernet NIC, running OpenBSD 2.7. For all the details about the presented performance measurements and for additional results refer to [1].

#### 4.1 Microbenchmarks for the Cryptographic Components

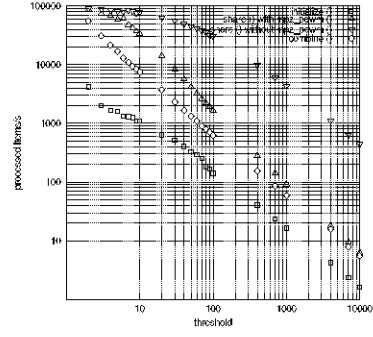
We measured the performance of the cryptographic primitives used in `shared` and `combined`. For secret sharing and Lagrange interpolation they operate over  $GF(P)$  where  $P$  is a prime number of 128 bits. For encryption we also use symmetric 128 bit keys. Thresholds used in the measurements default to 5.

The first time `shared` processes a given feature it uses the `initialize()` routine, which generates a pseudo-random polynomial, encrypts the feature and inserts the results in an AVL tree. We found that the Blowfish encryption in the OpenSSL crypto library operates at about 15370 encryptions per second for features of eight characters. It slows down slightly to about 15150 encryptions per second for features of 32 characters. We expect most features to be pretty short strings. The encryption speed is independent of the number of bits used for the symmetric keys. Depending on the threshold of the scenario associated with a feature, `initialize()` generates a number of pseudo-random coefficients for the corresponding polynomial. The higher the threshold, the more pseudo-random numbers are generated (see `initialize()` in Fig. 2a). `share()` basically evaluates the polynomial `initialize()` provides. Hence, the higher the threshold, the more expensive the exponentiations `share()` calculates using the GMP library call `mpz_powm()` (see `share()` with/without `mpz_powm()` in Fig. 2a).

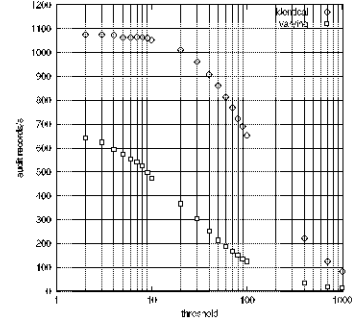
#### 4.2 Macrobenchmarks for the Pseudonymizer

We measured the performance of the `pseudonymizer` for varying parameters using synthetic audit data and configurations. `shared` handles features differently depending on whether they have already been seen in the audit data or not. If a feature is processed for the first time, `initialize()` generates a pseudo-random polynomial and stores it together with the encrypted feature (see Sect. 4.1). In any case a polynomial is used to generate a fresh pseudonym using `share()` (see Sect. 4.1). Owing to these different cases of feature handling we used two kinds of audit data to measure the best and the worst case performance. In the best case all features associated with a given scenario were *identical*. `shared` then calls `initialize()` only once per scenario. In the worst case we had *varying* features associated with a given scenario. As a result `shared` calls `initialize()` for every feature in the audit data.

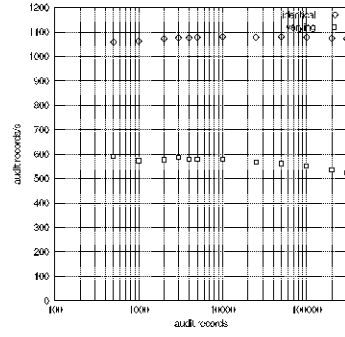
When increasing the number of audit records the `pseudonymizer` processes, we observe for varying features a mild performance penalty due to the growth of the AVL trees storing encrypted features and their corresponding polynomials (see Fig. 2c). A growing number of audit components, event types and scenarios



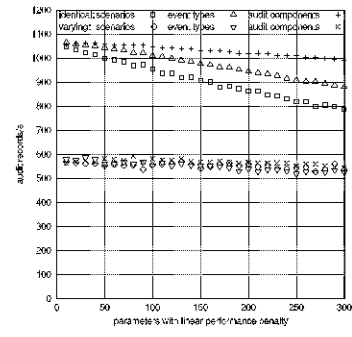
(a) Microbenchmarks



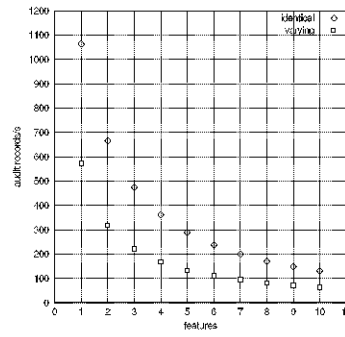
(b) Threshold



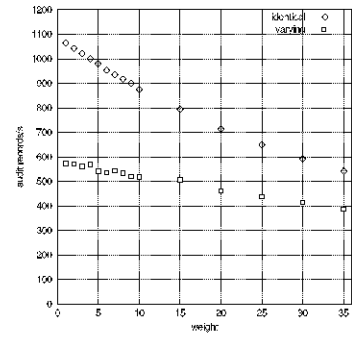
(c) Audit records



(d) Linear influence



(e) Features



(f) Weight

**Fig. 2.** Performance measurements of the pseudonymizer

has a linear influence on the performance since these objects are managed using lists (see Fig. 2d). The more features per audit record are pseudonymized, the more often the `pseudonymizer` asks `shared` for pseudonyms. In addition to the communication overhead more feature type contexts need to be matched by the `pseudonymizer` (see Fig. 2e). The pattern matching is the main cause for the performance loss when many features need to be pseudonymized per audit record. In contrast to this the performance loss caused by generating pseudonyms with larger weights is moderate (see Fig. 2f). There is no additional pattern matching or communication overhead involved. Increasing the threshold does not only slow down the generation of pseudonyms using `share()`, but also the initial generation of pseudo-random numbers for each distinct feature in a scenario using `initialize()` (see `initialize()` and `share()` with `mpz_powm` in Fig. 2a in comparison to Fig. 2b). However, in practice the parameters will only in rare cases reach values as high as shown in Fig. 2. In the next section we show that even in these rare cases the performance of the `pseudonymizer` is sufficient to cope with the audit record volume of a large site.

### 4.3 How Fast Is Fast?

To be able to judge whether our tools perform fast enough in a real world server environment we evaluated *syslog* and *Apache* audit records from a central server at the Center for Communication and Information Processing at the University of Dortmund. The machine hardware consists of a SUN Ultra Enterprise 4000 with 3GB RAM, six Ultra SPARC 168MHz CPUs, three disk arrays totaling 396GB and a 100Mbps full-duplex Fast-Ethernet uplink. The server runs Solaris 7 and about 1050 users are registered, which are employees and students of the University of Dortmund. During working hours an average of 25 users is logged on simultaneously. We took into account the following services provided by the machine: 37 world accessible *Apache* web servers, one FTP server with about 112000 FTP transfers per month with a volume of 12GB, and email services (SMTP, IMAP, POP) with about 45000 emails per month.

We evaluated all *Apache* access audit records collected over a period of 13 days for all 37 web servers. We have also evaluated all *syslog* audit records collected over a period of four weeks. We observed, that the `pseudonymizer` is able to keep up with the number of audit records generated even in situations where some audit component generates an abnormally high number of audit records. See Table 1 for the maximum numbers of audit records we measured. For more details about the evaluation and for statistics refer to [1].

## 5 Related Work

Our tools can be applied to server log data in order to hide information that may directly or indirectly identify the users that accessed the server. There are other well known services available to achieve this at the price of installing some client-side software or requiring the user to perform additional steps in order

**Table 1.** Maximum number of audit records generated per second

audit component	number of audit records	
	per hour	per second
<i>Apache</i>	33506	9.31
<i>syslog</i>	4956	1.38
$\sum$	38462	10.68

to achieve anonymity. More importantly, many of these anonymity tools do not provide for conditional identity recovery.

On the web the P3P framework helps users to match their privacy requirements against the privacy policies of the sites they visit. While this approach allows for fine grained control over what personal data is revealed at the application layer, identifying information from lower layer protocols is not protected. For this purpose there are also many different anonymizing services for accessing the web, such as the *Anonymizer*, *Anonymouse*, *Proxymate* formerly known as *LPWA*, *Crowds*, *Onion Routing* and *JAP*. In addition to filtering the transmission wrt. personal data, these services usually involve one or more intermediaries and aim at decoupling the client from the target server. Thus, identifying data is hidden before it reaches the target server. Similar examples for anonymous mail transfer are *Anonymouse* or remailers such as *Cypherpunk* and *Mixmaster*. Since the techniques employed, f.i. MIXes, are entirely different from the ones used for pseudonymizing log data, we do not go into the details here. These services are surveyed and compared in [20,21,22].

The techniques mentioned above are conducted under the control of the user himself and/or at least one third party trusted by the user wrt. anonymization. Thus, no trust is required in the target service provider. On the other hand these approaches require the user to take various actions to be able to communicate anonymously.

Pertinent legislation in various western countries requires providers of some classes of target services to support anonymous use of their service. To comply with these requirements, service providers need technical means, independent from actions their clients may take to protect their privacy. Existing services based on Unix systems often record personal data in log files. Reconfiguring the system such that it does not record certain kinds of log events is not always an option. The information supplied by some log events may be mission critical and may also contain personal data. One straightforward and easily deployable approach is to anonymize or pseudonymize identifying data after it has been collected, but before it is recorded in log files.

The *Anonymouse log file anonymizer* [20] is implemented as a customizable Perl script that replaces identifying information by default values or more coarse values. Anonymized log files may still be analyzed. In case some event happens, which requires further investigation wrt. the user's identity, it would be useful to be able to recover the original data, which has been anonymized. This feature is not supported by the *Anonymouse log file anonymizer*. The situation is similar

for the aforementioned MIX technologies. Indeed, even in the case of misuse of a service by an anonymous user it is infeasible for the provider or for investigating authorities to establish accountability. Contrary to this our tools provide for recoverability of pseudonymized data under certain conditions.

Due to this property our work is strongly related to privacy enhanced intrusion detection systems. Most of these systems use pseudonymization to hide identifying information after it has been generated and recorded in audit data. Some approaches have been published for the *Intrusion Detection and Avoidance* system (IDA) [23], the *Adaptive Intrusion Detection* system (AID) [23], a Firewall audit analyzer [24] and for the *Aachener Network Intrusion Detection Architecture* (ANIDA) [25]. These approaches have been reviewed and compared to our work in [2]. Intrusion detection systems (IDS) do not only give warnings in case of attacks, they also record evidence for further investigation. In case an attacker is to be prosecuted, it is desirable to extract identifying information from the recorded data. Privacy enhanced IDS support the recovery of pseudonymized information. A common weakness of existing privacy enhanced IDS is that they do not technically bind the recovery of identifying information to a specific purpose such as the investigation of a proceeding attack. Some privacy enhanced IDS require a third party to cooperate in the recovery process. The users have to trust that party to allow recovery only for given legitimate purposes. Other privacy enhanced IDS provide no strong protection against recovery for other purposes [2]. In contrast to this our tools instantly allow for recovery of pseudonymized data if predefined conditions bound to a specific purpose are met, but not otherwise.

Similar mechanisms are found in fair offline electronic payment systems. Such systems allow for anonymous payment as long as no payment unit has been spent more often than permitted. In the case of double spending of electronic coins, the perpetrator can be traced [26]. Some of the recovery techniques used are similar to the scheme our tools employ. Others use group signatures to achieve recoverability [27].

Payment units in fair offline electronic payment systems are also similar to anonymous credentials, which can be used for anonymous authentication. Many schemes for anonymous authentication support identity recovery while employing different techniques. Some schemes have been related to our approach in [4]. Since the resources of the target service are accessed using anonymous and pseudonymous credentials or payment units, the target service does not learn identifying information about the users during authentication. The users do not need to trust the target service provider regarding their anonymity. Also, malformed credentials, which do not exhibit the properties required by the target service can be rejected, avoiding further damage. On the other hand, before they can access any target service anonymously, users need to register with some third party to generate credentials and payment units. Since no such infrastructure has yet been widely adopted, in the meantime service providers can pseudonymize log data to offer privacy enhanced service access.

## 6 Conclusion

We motivated the need for technical enforcement of privacy principles and surveyed related work on privacy enhancing technologies. Though many of the related approaches allow for a strong trust model, they require the user to take additional steps and/or install client-side software. Also many approaches require a widely accepted infrastructure for issuing and managing credentials and payment units. None of these infrastructures is yet in widespread use, i.e. service providers cannot adopt widely accepted standards.

In contrast to this situation service providers are expected to respect and protect the privacy of their clients. We supply tools that help to retrofit privacy protection into existing Unix systems by means of pseudonymization of identifying features in audit data. Different from existing log file anonymizers or privacy enhanced intrusion detection systems, our approach balances requirements regarding anonymity and accountability. That is, pseudonymized identifying features can be revealed only if given definitions of scenarios requiring to do so are satisfied, but not otherwise. A trusted third party ensures that these scenarios meet the requirements regarding anonymity and accountability. This party controls the technical components that enforce the proper pseudonymization, such that later on identifying features can be recovered.

We described the trust model and architecture of the tools, and we presented performance measurements of the pseudonymizer. Finally we demonstrated measurements of the audit data volume generated by a real world system, indicating that the performance of the **pseudonymizer** should be sufficient for application at sites with even larger audit data volume.

**Acknowledgments.** We gratefully acknowledge the efforts of Hans Bornemann, Michael Heyer and Dieter Stolte in providing data about their networks and machines. Most of all Sven Bursch, Kai Grundmann, Stefan Magerstedt and Dennis Real deserve credit for their tireless work on the code as well as the measurements of the tools.

## References

- [1] Ulrich Flegel. Pseudonymizing Unix log files. Technical report, Dept. of Computer Science, Chair VI Information Systems and Security, University of Dortmund, D-44221 Dortmund, May 2002. Extended version of this paper.  
<http://ls6-www.cs.uni-dortmund.de/issi/archive/literature/2002/Flegel:2002a.ps.gz>.
- [2] Joachim Biskup and Ulrich Flegel. On pseudonymization of audit data for intrusion detection. In Hannes Federrath, editor, *Proceedings of the international Workshop on Design Issues in Anonymity and Unobservability*, number 2009 in LNCS, pages 161–180, Berkeley, California, July 2000. ICSI, Springer.



- [3] Joachim Biskup and Ulrich Flegel. Transaction-based pseudonyms in audit data for privacy respecting intrusion detection. In Hervé Debar, Ludovic Mé, and S. Felix Wu, editors, *Proceedings of the Third International Workshop on Recent Advances in Intrusion Detection (RAID 2000)*, number 1907 in LNCS, pages 28–48, Toulouse, France, October 2000. Springer.
- [4] Joachim Biskup and Ulrich Flegel. Threshold-based identity recovery for privacy enhanced applications. In Sushil Jajodia and Pierangela Samarati, editors, *Proceedings of the 7th ACM Conference on Computer and Communications Security*, pages 71–79, Athens, Greece, November 2000. ACM SIGSAC, ACM Press.
- [5] Privacy survey results, January 2002.  
<http://www.cdt.org/privacy/survey/findings/>.
- [6] Louis Harris & Associates Inc. IBM multi-national consumer privacy survey. Technical Report 938568, IBM Global Services, 1999.
- [7] Jarek Rossignac et al. GVU's 10th WWW User Survey, December 1998.  
[http://www.cc.gatech.edu/gvu/user\\_surveys/survey-1998-10/graphs/graphs.html#privacy](http://www.cc.gatech.edu/gvu/user_surveys/survey-1998-10/graphs/graphs.html#privacy).
- [8] Steven R. Johnston. The impact of privacy and data protection legislation on the sharing of intrusion detection information. In Lee et al. [28], pages 150–171.
- [9] National Computer Security Center. US DoD Standard: Department of Defense Trusted Computer System Evaluation Criteria. DOD 5200.28-STD, Supersedes CSC-STD-001-83, dtd 15 Aug 83, Library No. S225,711, December 1985.  
<http://csrc.ncsl.nist.gov/secpubs/rainbow/std001.txt>.
- [10] National Computer Security Center. Audit in trusted systems. NCSC-TG-001, Library No. S-228,470, July 1987.  
<http://csrc.ncsl.nist.gov/secpubs/rainbow/tg001.txt>.
- [11] Common Criteria Implementation Board. *Common Criteria for Information Technology Security Evaluation — Part 2: Security functional requirements, Version 2.1*. Number CCIMB-99-032. National Institute of Standards and Technology, August 1999. <http://csrc.ncsl.nist.gov/cc/ccv20/p2-v21.pdf>.
- [12] C. Lonvick. *RFC 3164: The BSD syslog Protocol*, August 2001.  
<http://www.ietf.org/rfc/rfc3164.txt>.
- [13] *syslog.conf*. Manual Page.
- [14] Martin Roesch. Snort – lightweight intrusion detection for networks. In *Proceedings of LISA'99: 13th Systems Administration Conference*, pages 229–238, Seattle, Washington, November 1999. The Usenix Association, Usenix.
- [15] Giovanni Vigna, Richard A. Kemmerer, and Per Blix. Designing a web of highly-configurable intrusion detection sensors. In Lee et al. [28], pages 69–84.
- [16] A. Shamir. How to share a secret. *Communications of the ACM*, 22:612–613, 1979.
- [17] *OpenSSL cryptographic library*, December 2001.  
<http://www.openssl.org/docs/crypto/crypto.html>.
- [18] Torbjörn Granlund. *The GNU Multiple Precision Arithmetic Library*. GNU, 3.1.1 edition, September 2000. <http://www.gnu.org/manual/gmp/index.html>.
- [19] *OpenSSL SSL/TLS library*, December 2001.  
<http://www.openssl.org/docs/ssl/ssl.html>.
- [20] Claudia Eckert and Alexander Pircher. Internet anonymity: Problems and solutions. In Michel Dupuy and Pierre Paradinas, editors, *Proceedings of the IFIP TC11 16th International Conference on Information Security (IFIP/Sec'01)*, pages 35–50, Paris, France, June 2001. IFIP, Kluwer Academic Publishers.

- [21] Simone Fischer-Hübner. *IT-Security and Privacy: Design and Use of Privacy-Enhancing Security Mechanisms*. Number 1958 in Lecture Notes in Computer Science. Springer, 2001.
- [22] Oliver Berthold, Hannes Federrath, and Marit Köhntopp. Project “Anonymity and unobservability in the internet”. In *Proceedings of the Workshop on Freedom and Privacy by Design / Conference on Freedom and Privacy*, pages 57–65, Toronto, Canada, April 2000. ACM.
- [23] Michael Sobirey, Simone Fischer-Hübner, and Kai Rannenberg. Pseudonymous audit for privacy enhanced intrusion detection. In L. Yngström and J. Carlsen, editors, *Proceedings of the IFIP TC11 13th International Conference on Information Security (SEC’97)*, pages 151–163, Copenhagen, Denmark, May 1997. IFIP, Chapman & Hall, London.
- [24] Emilie Lundin and Erland Jonsson. Anomaly-based intrusion detection: privacy concerns and other problems. *Computer Networks*, 34(4):623–640, October 2000.
- [25] Roland Büschkes and Dogan Kesdogan. Privacy enhanced intrusion detection. In Günter Müller and Kai Rannenberg, editors, *Multilateral Security in Communications*, Information Security, pages 187–204. Addison Wesley, 1999.
- [26] George Davida, Yair Frankel, Yiannis Tsiounis, and Moti Yung. Anonymity control in e-cash systems. In R. Hirschfeld, editor, *Proceedings of the First International Conference on Financial Cryptography (FC’97)*, number 1318 in Lecture Notes in Computer Science, pages 1–16, Anguilla, British West Indies, February 1997. Springer.
- [27] Jaques Traoré. Group signatures and their relevance to privacy-protecting offline electronic cash systems. In J. Pieprzyk, R. Safavi-Naini, and J. Seberry, editors, *Proceedings of the 4th Australasian Conference on Information Security and Privacy (ACISP’99)*, number 1587 in Lecture Notes in Computer Science, pages 228–243, Wollongong, NSW, Australia, April 1999. Springer.
- [28] Wenke Lee, Ludovic Mé, and Andreas Wespi, editors. *Proceedings of the Fourth International Workshop on Recent Advances in Intrusion Detection (RAID 2001)*, number 2212 in LNCS, Davis, California, October 2001. Springer.

## Annex – Example Application

Both, the `pseudonymizer` and the `reidentifier` leverage apriori knowledge provided in configuration files to locate identifying features and pseudonyms in audit data, respectively. Feature types are located using the audit component name (see the `FACILITY` keywords in Fig. 3), the event type context (see the `EVENT` keywords in Fig. 3) and the feature type contexts (see the `LEFTFEATURE` and `RIGHTFEATURE` keywords in Fig. 3). The apriori knowledge associates each feature type and corresponding pseudonyms with one or more scenarios (see the `GROUP` keywords next to the `RIGHTFEATURE` keywords in Fig. 3) as well as (a) corresponding weight(s) (see the `WEIGHT` keywords in Fig. 3), specifying the number of pseudonyms generated for the feature type. Also for each scenario a threshold is defined (see the `THRESHOLD` keyword at the top of Fig. 3), specifying the number of pseudonyms required for associated features to be recoverable from the corresponding pseudonyms.

In the following example we pseudonymize audit records generated by `tcplog`. The `tcplog` program logs TCP flag combinations that should not occur

in regular TCP traffic. Additionally **tcplog** is able to detect **queso** OS fingerprinting scans. **queso** is a reconnaissance tool used to determine the operating system version using the OS fingerprinting technique, offering a subset of the functionality of **nmap**.

We suppose that the PPO of a site considers IP addresses being sensitive information since they have the potential to identify the person behind the associated activities. The PPO thus decides to pseudonymize IP addresses.<sup>5</sup> The SSO knows that for an intruder the information provided by **queso** is useful for determining if a given service provided by a host might be vulnerable to a particular exploit. Hence, the SSO wants to be able to identify the IP addresses originating **queso** scans against his site.

The PPO analyzes, which audit components generate event types containing IP address features, such as the sample audit records in Fig. 4. The SSO provides the PPO with information about scenarios that require feature recovery, such as the **queso** scenario. For the **tcplog** audit component the PPO in cooperation with the SSO extends the apriori knowledge stored for our tools in the configuration files as shown in Fig. 3.

```
GROUP I1 THRESHOLD 6
FACILITY tcplog EVENT 'QUESO'
  LEFTFEATURE ' from ', RIGHTFEATURE ' port' GROUP I1 WEIGHT const(1)
FACILITY tcplog EVENT ' from '
  LEFTFEATURE ' from ', RIGHTFEATURE ' port' GROUP I1 WEIGHT const(1)
```

**Fig. 3.** Sample configuration for pseudonymizing **tcplog** audit records, such as in Fig. 4

```
02:23:37 tcplog[1567]: FIN SYN RST      URG      : port 41067 from 217.82.199.102 port 42312
13:42:06 tcplog[1040]: SYN              RES2     : ftp from 192.168.1.4 port 45691
13:42:06 tcplog[1040]: FIN SYN          PSH      URG      : ftp from 192.168.1.4 port 45693
13:42:06 tcplog[1040]: FIN SYN          PSH      URG      : ftp from 192.168.1.4 port 45693
13:42:06 tcplog[1040]: FIN              PSH      URG      : port 34513 from 192.168.1.4 port 45697
13:42:06 tcplog[1040]: FIN              PSH      URG      : port 34513 from 192.168.1.4 port 45697
13:42:06 tcplog[1040]: QUESO: port 34513 from 192.168.1.4 port 45697
```

**Fig. 4.** Sample unpseudonymized audit records generated by **tcplog**, which contain a **queso** OS fingerprinting scan

When the tools have been configured for the **queso** scenario the audit records in Fig. 4 are pseudonymized by the **pseudonymizer**. The resulting audit records are shown in Fig. 5. For the sake of presentation we moved the additional records generated by the **pseudonymizer** to the bottom of Fig. 5 while retaining the relative order of the **tcplog** records and of the additional records. Note that in the **tcplog** records the IP addresses have been replaced by pointers to the additional

<sup>5</sup> The PPO additionally might want to pseudonymize the port numbers. This also is perfectly possible, but for the sake of the brevity of the example we refrain from doing so.

records. The additional records provide the actual pseudonyms, i.e. the cryptographic material and other information required for the recovery of the replaced IP addresses. Note that the actual pseudonyms for IP address 192.168.1.4 are linkable using the identifier `d2petb50CX0un4CuLPhluM8`. Generally there may occur audit records describing activity that is definitely not related to any scenario, but where features need to be hidden. These audit records can be pseudonymized while omitting the additional records, such that recovery is infeasible later on.

```
02:23:37 tcplog[1567]: FIN SYN RST      URG      : port 41067 from a1 port 42312
13:42:06 tcplog[1040]:      SYN          RES2     : ftp from a2 port 45691
13:42:06 tcplog[1040]: FIN SYN      PSH      URG      : ftp from a3 port 45693
13:42:06 tcplog[1040]: FIN SYN      PSH      URG      : ftp from a4 port 45693
13:42:06 tcplog[1040]: FIN          PSH      URG      : port 34513 from a5 45697
13:42:06 tcplog[1040]: FIN          PSH      URG      : port 34513 from a6 45697
13:42:06 tcplog[1040]: QUESO: port 34513 from a7 port 45697
02:23:37 pseudonymizer: Short=a1 Long=RW4gGpC2dWVL0wXgF20ENg:10:ECnA!_ph6k0hKLZBsMw!H7H2ztc Group=I1
13:42:06 pseudonymizer: Short=a2 Long=5QgfV3!umUu_Fj8MWI2yoA:5b:d2petb50CX0un4CuLPhluM8 Group=I1
13:42:06 pseudonymizer: Short=a3 Long=sW7b167o0iG5eiBpmu7hg:5c:d2petb50CX0un4CuLPhluM8 Group=I1
13:42:06 pseudonymizer: Short=a4 Long=wdEey!pfsM4jGtz91yejZw:5d:d2petb50CX0un4CuLPhluM8 Group=I1
13:42:06 pseudonymizer: Short=a5 Long=Fi7o9GJU!A5SwY0!dAcxsA:5e:d2petb50CX0un4CuLPhluM8 Group=I1
13:42:06 pseudonymizer: Short=a6 Long=rog6E0bLHE3V5moFb1Ar8Q:5f:d2petb50CX0un4CuLPhluM8 Group=I1
13:42:06 pseudonymizer: Short=a7 Long=it0SIYe5EYys1eQNF0hNg:60:d2petb50CX0un4CuLPhluM8 Group=I1
```

**Fig. 5.** Sample pseudonymized audit records from Fig. 4

```
02:23:37 tcplog[1567]: FIN SYN RST      URG      : port 41067 from a1 port 42312
13:42:06 tcplog[1040]:      SYN          RES2     : ftp from 192.168.1.4 port 45691
13:42:06 tcplog[1040]: FIN SYN      PSH      URG      : ftp from 192.168.1.4 port 45693
13:42:06 tcplog[1040]: FIN SYN      PSH      URG      : ftp from 192.168.1.4 port 45693
13:42:06 tcplog[1040]: FIN          PSH      URG      : port 34513 from 192.168.1.4 port 45697
13:42:06 tcplog[1040]: FIN          PSH      URG      : port 34513 from 192.168.1.4 port 45697
13:42:06 tcplog[1040]: QUESO: port 34513 from 192.168.1.4 port 45697
02:23:37 pseudonymizer: Short=a1 Long=RW4gGpC2dWVL0wXgF20ENg:10:ECnA!_ph6k0hKLZBsMw!H7H2ztc Group=I1
```

**Fig. 6.** Sample pseudonymized audit records from Fig. 5 with recovered features

The audit records in Fig. 5 are transferred to the SSO. When the SSO detects the `queso` scan contained in the audit records in Fig. 5 he uses the `reidentifier` to recover the IP address features associated with the satisfied `queso` scenario. The resulting audit records are shown in Fig. 6. Note that the pseudonym in the first audit record in Fig. 6 is not revealed since it corresponds to an IP address not involved in the `queso` scan originated at IP address 192.168.1.4.

# *DPS*: An Architectural Style for Development of Secure Software<sup>\*</sup>

Pascal Fenkam, Harald Gall, Mehdi Jazayeri, and Christopher Kruegel

Technical University of Vienna, Distributed Systems Group  
A-1040 Vienna, Argentinierstrasse 8/184-1  
{p.fenkam, h.gall, m.jazayeri, c.kruegel}@infosys.tuwien.ac.at

**Abstract.** Many claim that software systems must be designed for security. This, however, is far from being an easy task, especially for complex systems. We believe that this difficulty can be alleviated by a set of —preferably rigorous— principles. We propose an architectural style, the *Dual Protection Style (DPS)*, for constructing secure software. This style results from our experience in designing and implementing a distributed, multi-user, medium sized application. We present the applicability and effectiveness of our *DPS* style on the basis of a case study of a distributed software platform for virtual and mobile team collaboration called MOTION. We further elaborate on the description of this architectural style, its formalization and the formal verification of some of its properties.

**Keywords:** Authorization and Access Control, Security Engineering, Formal Methods, Software Architecture, Architectural Style, Alloy.

## 1 Introduction

The dependability of software systems is becoming a foremost concern for the whole society. This is exacerbated by the increasing pervasive use of software. These are no longer simply used in a standalone fashion, but have become highly distributed and embedded in many of our working tools. The pervasiveness of software systems is accompanied by a variety of security risks. For instance, it is not necessary anymore to be physically present in front of a computer to attack its security engine: thanks to the Internet revolution. This frightening reality has (or should have) consequences on the way we build software. It is increasingly argued that software must be designed for security right from the beginning instead of security being added as an afterthought [3,7,9,17,18,22]. Numerous examples are cited in the literature where applications failed to be “upgraded” from non-secure applications to secure applications.

---

<sup>\*</sup> This work is supported by the European Commission in the Framework of the IST Program, Key Action II on New Methods of Work and eCommerce. Project number: IST-1999-11400 MOTION (MOBILE Teamwork Infrastructure for Organizations Networking).

If designing for security seems to be an acceptable and reasonable recommendation, the challenge, however, resides in how to achieve this goal. This is particularly difficult since the concept of security itself is differently understood. Though security is difficult to define, there are well understood security criteria for distributed applications: identification, authentication, authorization and access control, integrity, auditing, etc. At least for these well understood notions, one would expect some principles guiding their introduction into distributed applications. To our knowledge, little work has been done on illustrating how a secure application can be constructed.

In this paper, we propose an architectural style for constructing access controlled applications. This style is a collection of rules that constrain the topology and the behavior of the components of an application that needs to support access control. Our architectural style prescribes ways of controlling access to resources made available by applications. It is based on two principles, user access control and code access control. User access control is the process of verifying that end-customers only perform actions they are allowed to. Code access control on the other hand ensures that applications only invoke methods they are allowed to. Following this architectural style, only one method in each trace of the application must perform user access control. The paper also presents the formal specification of this architectural style in *Alloy* [11], a formal notation for micromodels of software. The main purpose of this formal specification is to provide a strong foundation to the style, and a precise definition of its invariants.

The remainder of the paper is organized as follows. Section 2 is concerned with related work: some principles proposed by various researchers for constructing secure applications are presented. Section 3 gives an overview of the MOTION platform that inspired the development of this architectural style. This case study also helps to better illustrate the problem solved by our work. We present an informal description of the architectural style in Section 4 and its formal description in Section 5. Section 6 concludes the paper.

## 2 Related Work

Building secure software is starting to draw well-deserved attention. Up to now, most computer security articles published in industry magazines focussed on fixing security problems [7]. Among the few that propose principles for building secure software systems, we consider [10,18,23,25,27,28] particularly interesting.

In [18], McGraw and Felten propose twelve rules for writing secure Java programs: 1) programs should not depend on initialization, 2) Use of visibility restrictors, 3) everything should be made final unless there is a good reason not to, 4) programs should not depend on package scope, 5) programmers should avoid using inner classes, 6) programmers should avoid signing code, 7) if signing code is a must, all should be put in one archive file, 8) classes should be made unclonable, 9) classes must be unserializable, 10) classes must be undeserializable, 11) classes should not be compared by name, 12) secrets should not be stored in code.

Similar recommendations are given in [23]. Besides the controversial nature of some of these guidelines, they are oriented towards Java programs. Thus, they can be viewed as low level principles. Referring to the idea that security considerations must be incorporated throughout the entire product lifecycle [22], we argue that there is a need for more high level principles that can be applied at the architecture and design levels.

Viega and McGraw [27] propose a complement to these principles. Ten recommendations are given to be kept in mind throughout the design and the implementation of a secure software system: 1) secure the weakest link, 2) practice defense in depth, 3) fail securely, 4) follow the principle of least privilege, 5) compartmentalize, 6) keep it simple, 7) promote privacy, 8) hiding secrets is hard, 9) be reluctant to trust, 10) use of community resources.

Besides these principles, the Open Group [25] is currently writing a book on security patterns. A first draft of this book is available on the Internet. For the moment, 32 patterns are indexed and 15 of them are documented. These patterns are divided into 5 categories: entity patterns, structural patterns, interaction patterns, behavioral patterns, and available system patterns. For the flavor of these patterns, we give an overview of four of them.

The *Protected System* pattern also called *Reference Monitor* pattern recommends software developers to delegate all access control checks to a guard which enforces security policies. The *Policy Enforcement Point* pattern complements the first by recommending to separate/isolate the code of the reference monitor from the remainder of the application. The *Secure Communication* pattern ensures that security policies of different take-holders still be satisfied when they come to communicate with each other. The *Recoverable Component* pattern requires a system to be structured so that its state can be recovered if the system fails.

An important fact to notice is that although developed separately, most principles provided by the open group are refinements of those proposed by Viega and McGraw in [18]. For instance, the *Checkpointed System* pattern and the *Recoverable Component* obviously contribute to ensure the *Fail Securely* principle. The *DPS* style gives a concrete guideline with which to achieve these principles. After the description of this architectural style in Section 4, we show how it helps in securing the weakest link, keeping things simple, increasing usability, and protecting against malicious code. The next section gives an overview of the MOTION project that inspired the development of the *DPS* architectural style.

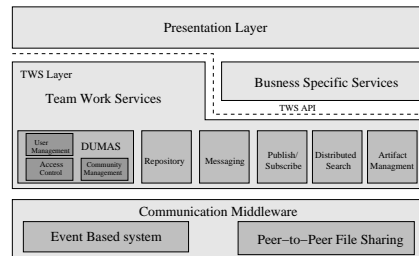
### 3 The MOTION Project

MOTION (MOBILE Teamwork Infrastructure for Organizations Networking) is a platform we designed and prototyped in the MOTION European project [14,21]. This platform addresses the needs of two well known organizations. The first is a manufacturer of mobile phones and the second is a producer of white goods (e.g. refrigerators, washing machines). The platform has a service architecture that supports mobile teamworking by taking into account different connectivity

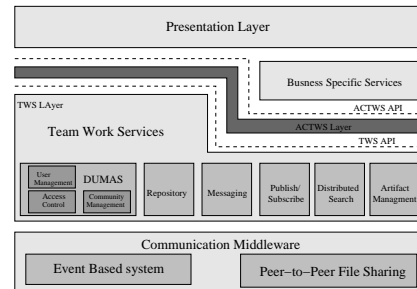
modes of users, provides access support for various devices, supports distributed search of users and artifacts, offers user management facilities in a way where users can be grouped in communities. The MOTION platform also contains an access control mechanism.

### 3.1 The MOTION Architecture

The MOTION system was constructed by assembling different components: PeerWare [19], DUMAS (Dynamic User Management and Access Control System) [6], an XQL engine [8], an XML parser, a repository (comparable to a file system), an artifact manager (comparable to a shell that provides primitives for accessing the file system), etc. Some of these components are commercial off-the shelf while other were built by the project team. A layered view of the MOTION platform is presented in Figure 1.



**Fig. 1.** Overview of the MOTION Architecture



**Fig. 2.** The Access Controlled MOTION

The bottom layer of the architecture provides the communication infrastructure and is realized by PeerWare [19]. The services made available by this component include: peer-to-peer file sharing, an event based system, and mobile code infrastructure.

On top of this layer, we constructed the teamwork services layer (TWS layer). This teamwork services layer is composed of DUMAS, a repository, a messaging system, a user oriented publish- subscribe system (TWS P/S), a component for distributed search, and an artifact management component. DUMAS groups user management, community management, and access control functionalities. The repository is a component responsible for storing different MOTION data, including files, metadata, and access control lists. All methods in the TWS layer are security sensitive and therefore access controlled.

The top-most layer of the MOTION platform consists of the presentation layer and business specific services. This layer includes applications matching the specific requirements of end-customers. They completely rely upon the TWS layer.



### 3.2 Component Integration

In this section, we discuss the integration of the components listed above. The purpose of this section is to illustrate (1) the need for the DPS style, and (2) the benefits we would gain if we had used this style right from the beginning of the development process. We focus on the integration of the TWS layer components with the access control module. This integration essentially consists of inserting the instructions for access control in each public method of the TWS layer. An example is presented in Listing 1. The method **addMember** is shown. The purpose of this method is to add a user to a community. The access control check is performed at line 7. The lines 3 to 6 prepare the arguments for invoking the operation **checkPermission** performing the permission check. This integration had to be performed for approximatively 110 methods made available by the TWS layer. Initially, 100 person-hours were foreseen for this task. At the end we needed about 320 person-hours, more than 200 % time over-run! What went wrong?

```

1      public void addMember(UserID userId,
2                             CommunityID target_community)
3          throws NoSuchEltException, MissingPermissionException,
4                 DuplicatedDefinitionException {
5      UserID actor= motionmanager.getCurrentUserID();
6      Permission permission=new Permission("add-member");
7      authorization.checkPermission(actor, permission,
8                                   target_community );
9
10     motionmanager.getDumasController().getUserManager().
11         linkUser(actor, userId,target_community);
12
13     Element elt=temp.createElement("Action");
14     elt.setAttribute("classname","Community" );
15     elt.setAttribute("methodname","addMember" );
16     elt.setAttribute("argument0",actor.getID() );
17     elt.setAttribute("subjecttype","user");
18     elt.setAttribute("actor",actor.getID());
19     elt.setAttribute("target",target_community.getID());
20
21     TUVEvent evt= new TUVEvent(elt);
22     twsps.publish(evt,peer.getPublicationDocument());
23
24 }
```

**Listing 1.** The Alloy Specification of the DPS Style

The operation **addMember** presented in Listing 1 is performed by requesting the services of three different components: (1) the DUMAS user management module (line 10), (2) the TWS publish- subscribe facility (line 22), and (3) the repository. The repository is not directly invoked by **addMember**, instead **addMember** invokes **linkUser** of DUMAS that further invokes the repository. The invocation of the TWS publish operation is performed for notifying other

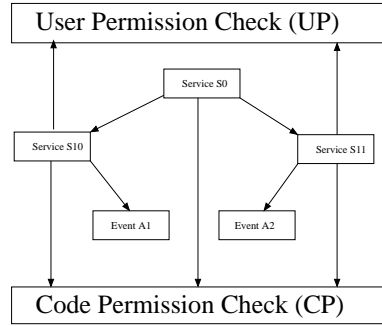
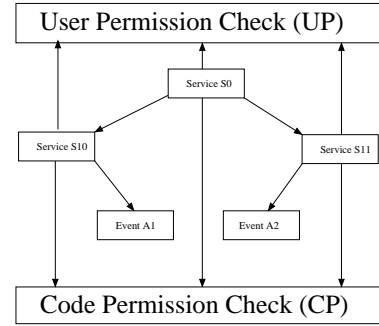
users and components. Each public method of the TWS layer performs access control checks. This is because most of these methods are also directly invoked by the presentation layer or business specific services. The implication is that for invoking the method `addMember`, three different access rights are required:

- the access right `add-member` required by the operation `addMember`. This access right ensures that the user invoking the operation has the permission to insert a user into the target community.
- the access right `write` ensures that the user has the permission to write to the repository.
- the access right `publish` is checked by the TWS `publish/subscribe` for ensuring that the user has the permission to publish the specified event.

We face an embarrassing problem! This multitude of permission-checks does not protect the TWS layer, but makes it unusable. For performing a single operation, too many different access rights are required from the user. We noticed this because the access control component was developed in tandem with a graphical user interface. This user interface allowed users to give us direct feedback. This problem is exacerbated by the fact that these access rights are not directly semantically related to the operation `addMember`. For a customer, there is no obvious reason to require access right for writing to a repository while adding a user to a community. This might even appear as a bug in the system. Besides this usability issue, a second problem, that such an integration of access control into applications faces is the issue of administration. The more complicated an access control system is, the more difficult is its administration and the more likely are security breaches into the system.

To solve this problem, we ensured that users are not asked more than one access right during the execution of a single task. This is achieved by requiring that in each trace of the application, user access rights are checked only once. To improve the TWS layer such that it takes this result into consideration, we had to remove access control checks from all MOTION components first. The next step consisted of building a tiny layer on top of the TWS layer (see Figure 2), that separates the non-access controlled TWS layer from the presentation layer and business specific services. This tiny layer is called access controlled teamwork services layer (ACTWS layer). All methods of the ACTWS layer are access controlled while no method in the TWS layer is access controlled. Further, no operation of the ACTWS layer is allowed to invoke an operation of this layer, be it recursively or not. This would obviously lead to the initial situation.

While the approach described above solves the problem of multiple access rights for a single task, it raises another issue. The layers underlying the ACTWS layer are completely open, without any access control check, thus, they are vulnerable to various attacks. An important kind of attacks is malicious code. First, this attack seems to be à la mode nowadays. Second, it is particularly relevant to the MOTION platform since the underlying middleware uses mobile code for relocation facilities [19]. The solution to this second challenge is to protect components by means of code access control. This is, for each method in the

**Fig. 3.** The Dual Protection Style**Fig. 4.** Example of DPS Violation

MOTION system, there exists a list of components (or a list of methods, depending on the granularity) allowed to invoke it. Thus, if some malicious code wants for instance to modify the access control list stored in the repository, the operation fails since this code is not allowed to interact with the repository. The principle is similar to code access control provided in the Java environment [15].

Given these two techniques, we are able to efficiently protect the MOTION platform. We strongly believe that the experience gained in securing this platform can be useful to other software developers. We therefore propose to capture its quintessential properties through an architectural style that abstracts away from MOTION specific details.

## 4 The DPS Architectural Style

In this section, we give an informal description of the *Dual Protection Style*. The goal of this style is not to guarantee security, but to simplify the construction of access controlled applications. The *DPS* style defines components, connectors and constraints for controlling the access to services provided by an application.

### Vocabulary

In its basic version, the *DPS* recognizes three types of components: a user access control component, a code access control component, and the set of components forming a system and interacting with the user and code access control components. In case of MOTION, such components are, for instance, the repository, the artifact manager, or the community manager.

A component is called *protectee* of a user/code access control component if one of its services interacts with this user/code access control component. This user/code access control component is called a *protector* of the protected component. Further, a component that makes available a service interacting with a user access control component is said to be *user access controlled*. If the component rather interacts with a code access controller, it is said to be *code access controlled*.

For instance, all methods of the TWS layer are code access controlled while all methods of the ACTWS layer are user access controlled.

The connectors between *protectees* and *protectors* are called *access control checks* or *permission checks*. There exist two kinds of *permission check*: *user permission check* and *code permission check*. A *user permission check* is initiated by a *protectee* towards a user access control component. This is a request to the user access control component to check the authorization of a user to access an object. Similarly, a *code permission check* is a request to a code access controller to decide on the authorization of another application to make use of the requested service.

### Invariants

The *DPS* style constrains the behaviors of *protectors* and *protectees* by means of two invariants. The first invariant is best explained in terms of CSP traces [2]. A trace is a sequence of events that a process (or a service) allows. An event is an atomic action, i.e. it cannot be decomposed. In this respect, we consider permission checks as events. A permission check cannot be decomposed into other events. The implication is that any process performing permission checks is not atomic. We use the terms service and process equivalently. An application satisfies the *DPS* style if each of its processes satisfies the *DPS* invariants. Such an application will be said to be *DPS* protected.

The first invariant of the *DPS* style ensures that the number of occurrences of user permission checks in each process is equal to 1. The number of occurrences of an event, say  $a$ , in a trace  $T$  is denoted by  $T \downarrow a$ .

An example of a system  $S$ , consisting of the services  $S_0$ ,  $S_{10}$ ,  $S_{11}$  is illustrated in Figure 3 together with its call graph. Its processes are defined as  $traces(S_{11}) = \langle UP.CP.A_2 \rangle$ ,  $traces(S_{10}) = \langle UP.CP.A_1 \rangle$ ,  $traces(S_0) = \langle CP.UP.CP.A_2, CP.UP.CP.A_1 \rangle$ .  $UP$  and  $CP$  are user and code permission checks respectively and  $A_1$  and  $A_2$  are actions other than permission check.

This system satisfies the first invariant of the *DPS* style. In fact,  $UP.CP.A_1 \downarrow UP = 1$ ,  $UP.CP.A_2 \downarrow UP = 1$ ,  $CP.UP.CP.A_1 \downarrow UP = 1$  and  $CP.UP.CP.A_2 \downarrow UP = 1$ .

Informally, this invariant prescribes that user permissions should be checked only once for each task that a user performs. It does not prevent methods from performing various permission checks, but ensures that all permission checks must be performed in an exclusive manner so that only one of them can be executed each time.

For a program that has no permission checks in **if-else** blocks, the invariant can be expressed in terms of its call graph. In this case, there should be one method performing user permission check in each path of the call graph. An example is depicted in Figure 3. On each of the paths of this call graph, there exists exactly one service that initiates a user permission check.

In general, this invariant cannot be expressed in terms of a call graph since a method whose body includes the instruction  
`if(condition) user-permission-check-1 else user-permission-check-2`

satisfies the first invariant of the *DPS* style but is said to contain two permission checks (following the definition of call graphs [26]).

The second invariant of the *DPS* style ensures that each service performs at least one code permission check. In other terms, all methods must be *protectee* of some code access controller. Considering the example of Figure 3, all services  $S_0$ ,  $S_{11}$ ,  $S_{10}$  perform code permission checks. Unlike the first invariant, the second invariant is easily expressed in terms of methods instead of traces.

Note that the way how the access controller decides on whether users or code have permissions depends on each architecture (thus instance of the style).

The *DPS* style supposes that the identities of users and code can be verified. That is, there exists a reliable mechanism for determining which user/service is requesting a service. Though there exist different mechanisms for authenticating users, this is not the case for code. Considering two services hosted by two different computers and communicating by means of a handshake protocol it is difficult for each of them to verify the identity of the application running at the other end. This problem, however, does not exist when the components are co-located and one is accessing the other through a direct method call. In Java, for instance, the stack inspection algorithm provides the facility of determining which methods are currently being executed.

### Specializations and Instantiations

The *Dual Protection Style* can be specialized in several interesting ways.

We say that a permission check connector is open for a resource  $S$  if the connector authorizes any access to this resource. We say that a permission connector is open if it is open for all resources.

Depending on which connectors are open in a system, the following specializations of the *Dual Protection Style* can be distinguished. In the first case, the *user permission connector* is open, in the second case the *code permission connector* is open, and in the third case both connectors are open.

Strictly speaky, the *DPS* style is violated when one of the above invariants is not satisfied. For instance, a system that has no code access controller can theoretically be said to violate the *DPS* style. On the other hand, this system behaves exactly like a system with the same services but with an open *code permission connector*. A system that theoretically doesn't satisfy the *DPS* style but behaves in practice exactly like a specialization of the *DPS* style is said to have an architecture that is of a degenerate *DPS* style. We do not consider such systems as violation of the *DPS* style.

This fuzzy boundary of architectural styles is not specific to the *DPS* style. It has also been recognized for well known styles such as Pipe-and-Filters. Though *batch sequential* is an architectural style in its own right, it also results from a pipeline (a pipeline is a specialization of pipes-and-filters) where each filter processes all of its input data as a single entity. Another example is provided by the peer-to-peer architectural style that behaves as client-server in some configurations. As recommended by Shaw and Garlan [16], the fact that boundaries of styles overlap should not deter us from identifying the main features of a style

with its central example of use. Thus, in practice, we will not consider systems that practically behave as specializations of the *DPS* style as violation of this style.

The only case that intriguingly violates the *DPS* style is when there exists a trace of the application where the number of occurrences of the user permission checks is greater than 1. Such a case is shown for instance in Figure 4. The branch of the call graph composed of the invocation of  $S_0$ ,  $S_{01}$  contains two user permission checks. In terms of traces, we have

$traces(S_0) = \langle CP.UP.CP.UP.A_2, CP.UP.CP.A_1 \rangle$  and  $CP.UP.CP.UP.A_2 \downarrow UP = 2$ . The `mv` command of the UNIX/LINUX operating system is an example of an application that does not satisfy the *DPS* style (although an equivalent command can be implemented that satisfies *DPS*). For moving a file to a directory, one needs to have the permission to read that file and the permission to write to the target directory. The command `less` on the other hand satisfies the *DPS* style. These commands are applications that are constructed using the system calls of the operating system such as `sys_read` and `sys_write` [13]. A question that can be raised is if the UNIX/LINUX operating system itself satisfies the *DPS* style. We are currently investigating this issue.

Code access control is not specific to Java. The concept of domain type enforcement [1] originally proposed for Unix ensures that code modules are permitted to access only a controlled set of resources (including other modules).

## Advantages

Our style provides a number of advantages. Most of them are oriented towards facilitating the development of access controlled applications. Therefore, we concentrate on these features first and not on the security capabilities of our style. The *DPS* style proposes a clear architecture for easily constructing usable, and secure applications. We are not aware of a similar architectural style for building secure systems.

Starting with the *DPS* style, the developer knows from the beginning of the development process exactly what actions are to be carried out. He can decide exactly where to perform security checks in his components, decide on which protectors to use, and what the implications are in terms of dependability, architecture, and maintenance. In general such an architectural style forces the software developer to take security into consideration very early in the design of its architecture.

Applications based on the *DPS* style are easier to administer. In fact, the administrator knows that for each task, there is at most one access right that must be owned. This is obviously much simpler than administrating an application where tasks require many different permissions. Moreover, the behavior of applications where operations require many access rights might sometimes be unexpected. Considering the case of `addMember` presented above, it is difficult to assign to a user the permission to add users to a community without assigning him the permission to publish events related to the profile of this user such as “user is online”, or “user is expert”. An undesirable side effect! This side effect

takes a particular significance in information flow control. We claim that a system can more easily be proven to satisfy an information flow constraint when its architecture is an instance of the *DPS* style than when it isn't.

The *DPS* style promotes securing the weakest link by minimizing the dependency on customers. The weakest link in an application designates a component, an algorithm, a protocol, or anything else that is likely to be weakly secure. Attackers will obviously look for the weakest link of an application. Social engineering was shown to be one of the most effective means to bypass security restrictions. Security engineers often make assumptions that, though reasonable, do not hold in practice (an example is the famous question asked by most browsers: “do you trust this application?”). It is therefore very probable that users will have difficulties in using complex security mechanisms. The consequence is that either they ignore them, thus being completely vulnerable, or they use them with difficulties. Thus, the less a security mechanism relies on customers, the less it is vulnerable to social engineering efforts and therefore the lower is the probability of being attacked through this link. By reducing the number of access rights required by a user, the developer reduces the difficulties that users might encounter.

In systems where access control is performed at different levels, thus not satisfying the *DPS* style, the software/security engineers will spend significant time in investigating which permissions/access rights must be checked at each stage and in each component. The assumption is thus made that programmers have the necessary background in access control models. This is unfortunately not always the case, as security engineering doesn't even belong in most software engineering curriculums. These difficulties are exacerbated if we take into consideration the iterative nature of the software development process that probably also implies iteration on the access control model(s) and high degree of interference between the software engineers and the security engineers. The *DPS* style allows reducing the interaction between security engineers and software engineers. It finally frees software engineers from dealing with access control by delegating this task to a few people who know exactly where there is a need for access control check in the application. This can be done by constructing a layer as we did in the MOTION case study. Though software engineers need to integrate code permission checks in components, this is a minor issue since there is nothing more than invoking a method that another method can do. The interface to the access control component is thus simple and can easily be specified by security engineers and the API made available to software engineers.

The *DPS* style promotes protecting applications against malicious code. It is not enough for an application *A* to be started by a user to access a resource *r* made available by a *DPS* protected application *C*. The application *A* must be explicitly allowed by the code access controller of *C* to invoke the method leading to *r*.

The *DPS* style achieves a clear decoupling between permission check connectors; one can check for user permissions without checking for code permissions and vice versa. This clear separation is missing in the Java security architecture

[15] where the two connectors are strongly coupled. The separation between these connectors obviously allows security engineers to finer control security mechanisms to match their needs.

### Shortcomings

The *DPS* style relies on the identity of client components. This leads to a subtle limitation in the application of the *DPS* style in distributed environments. This limitation occurs when it is not possible to verify the identity of the requesting service. Let us consider for instance an architecture based on services  $S_{01}$ ,  $S_{02}$ , and  $S_1$  where  $S_{01}$  and  $S_{02}$  both request  $S_1$ . All communication between these services is performed by means of handshake protocols. This means that these services have no methods for verifying the identity of the code with which they communicate. Following the *DPS* style, user permission checks must be performed either by  $S_{01}$  and  $S_{02}$  or by  $S_1$ . In the first case, however,  $S_1$  is totally vulnerable to attacks since there is no way to establish the authenticity of its callers and thus to perform code permission checks.

In the second case, user permission checks must be done by  $S_1$  (following *DPS*). This however is not always feasible.  $S_1$  might be a commercial service made available by a third party while  $S_{01}$ , and  $S_{02}$  are independent services constructed upon  $S_1$ . The recommendation of the *DPS* style would thus mean that  $S_1$  must manage a user access control component and policy for  $S_{01}$  and  $S_{02}$  which is not practicable. A concrete example of this scenario is where  $S_1$  is a search engine such as Google and  $S_{01}$  is a MOTION service. There is no reason why Google should perform access control for the MOTION platform.

We propose two solutions to this limitation. The first solution consists of relaxing the second invariant of the *DPS* style. This relaxation consists of identifying remote applications through the name of the computer hosting them (possibly combined with some other authentication information). The second solution relies on using middleware such as RMI or CORBA. They give applications the possibility to question the identity of a caller. In both solutions, the application of the *DPS* style is risky. The risk in the first solution is obviously bigger than in the second solution. In the first solution, some malicious code can still attack a remote service if it is located on a host authorized to interact with this service. In the second case, one needs to fake the implementation of RMI/CORBA that is used to communicate with the remote service. The fake implementation will thus masquerade the identity of the caller. Obviously one is more protected in the second case, whose dependability can be improved by using further authentication information.

A second shortcoming that can be attributed to the *DPS* style is the lack of defense in depth. The idea behind defense in depth is to provide diverse defensive strategies, so that if one of them fails the other can still prevent a full breach. If we consider the example shown in Figure 3, if a security breach succeeds in  $S_0$ , by means of buffer overflow for instance, the attacker gains control of the system. He can perform everything that the services that  $S_0$  is normally allowed to perform. There exists a solution to this limitation that even produces the



same results as the *DPS* style. This solution consists of reformulating the first invariant of the *DPS*. This reformulation proposes not to limit the number of user permission checks in traces to one, but requires that all user permission checks in a trace be equivalent. We say that two user permission checks are equivalent if the related user access controllers return the same answers given similar input. We, however, prefer the actual version of the invariant since it is much simpler and intuitive enough to be understood and successfully applied by the average programmer.

## 5 An Alloy Specification of the DPS Style

In this section, we present a formal specification of the *DPS* architectural style and verification of some of its properties. As recognized in [12], if it is proven that an architectural style verifies some properties, there is no need to show that its instances verify these properties as well. Formally specifying architectural styles is therefore worthwhile since the cost of specifying and analyzing a style is amortized across instances. Furthermore, a style specification abstracts from many of the details of the instances allowing a style analysis to focus more easily on fundamental properties that might be obscured in the analysis of an instance. This formal specification helped us deepen our understanding of the style. It is only after a formal specification of this style that we could give a clear and simple formulation of the invariants. The *DPS* style is specified in the Alloy specification language. We give a brief overview of this language below. The different *Alloy* constructs used in the specification are explained when they are used.

### 5.1 The Alloy Specification Language

The *Alloy* specification language was designed by Daniel Jackson [11]. It is a first order language that can be viewed as a subset of Z [4]. *Alloy* is a declarative language similar to the formal specification languages Z [4], and VDM [20]. Unlike these languages, *Alloy* is automatically analyzable in the style of model checking. Models described using this specification language are called micromodels. They are intended to be tiny models that focus on the risky parts of a system.

It might be argued that a model should intrinsically abstract from details and thus be an order of magnitude smaller than the system it models. This observation holds! From our experience [5], however, the kind of verification mechanism available for a specification language plays an important role. Considering the example of VDM, the automatic verification facility available up to now is testing (facility provided by the IFAD VDM Tools [24]). To be amenable to testing, however, a VDM-SL specification has to be interpretable and thus written in an executable subset of VDM-SL. Thus, the designer is distracted from the goal of producing a tiny model to the goal of producing an interpretable model. Though interpretable models are an order of magnitude smaller than the systems they model, they are also an order of magnitude bigger than non-interpretable models. With its automatic analyzer, the *Alloy* specification language allows the designer

to indeed concentrate on abstracting from details and thus, on producing tiny models that are easy to understand. Just as important, the analyzer of the *Alloy* specification language gives the designer the kind of benefit of immediate feedback that the execution brings.

```

1  module csp/dps
2
3  sig Event{}
4
5  disj sig UserPermissionCheck, CodePermissionCheck extends Event{}
6
7  sig Trace[value: set Event]{some value}
8
9  fact TraceEquality( all t1, t2 in Trace | t1=t2 iff t1.value =t2.value )
10
11  sig Service{traces:set Trace, actions:set Event}{some traces and all t in traces | actions in t.value}
12
13  sig System{userservices: set Service, internalservices:set Service}{(some userservices+internalservices) and
14    no (userservices & internalservices) }
15
16  fun System::services():set Service{result=this.userservices+this.internalservices}
17
18  sig DPSSystem extends System{}
19
20  fact DPSDefinition( DPSSystem = {x in System | x..inv1() and x..inv2()} )
21
22  fun System::alphabet():set Event{result=(this..services()).traces.value}
23
24  fun System::inv1()(all t in this.userservices.traces | one UserPermissionCheck & t.value )
25
26  fun System::inv2()(all t in this.internalservices +this.userservices | some CodePermissionCheck & t.actions)
27
28  assert DPSSystemExistence{no DPSSystem }
29
30  check DPSSystemExistence for 5
31
32  assert PermissionCheckExistence{all dps in DPSSystem | dps!=none[System] implies
33    some dps..alphabet() & UserPermissionCheck && some dps..alphabet() & CodePermissionCheck}
34
35  check PermissionCheckExistence for 5 but 10 Event
36
37  assert NonDPSSystemExistence{System=DPSSystem}
38
39  check NonDPSSystemExistence for 5
40
41  fun layer1():set Service{result={x in Service | (some UserPermissionCheck & x.traces.value) and
42    (no UserPermissionCheck & x.actions)}}
43
44  fun layer2():set Service{result={x in Service | some UserPermissionCheck & x.actions}}
45
46  fun layer3():set Service {result={x in Service | no UserPermissionCheck & x.traces.value}}
47
48  assert LayersDisjunction{ (no layer1() & layer2()) and (no layer2() & layer3()) and (no layer1() & layer3())}
49
50  check LayersDisjunction for 5
51
52  assert DPSSystemLayeredView{ DPSSystem..services() in layer1() +layer2()+layer3() }
53
54  check DPSSystemLayeredView for 5

```

**Listing 2.** The Alloy Specification of the DPS Style

## 5.2 The DPS Style in Alloy

As the reader must have noticed, the two invariants of the *DPS* styles are based on two different views of applications. The first is based on CSP traces and the second is based on a methods view. The following specification is based on these two views. In this specification, we restrict the notion of trace to the alphabet of this trace. Thus, considering for instance the trace  $\langle a, b, c, d \rangle$  we denote it as  $\{a, b, c, d\}$ . This simplification has no influence on our invariants since they only constrain the number of occurrences of some actions, but not their order.

The line 1 of the specification defines the name of the module including the style. The line 3 declares the type **Event**. An event is decomposable. Two kinds of events are of particular interest, **CodePermissionCheck** and **UserPermissionCheck** declared at line 5. These two kinds of events are disjoint (denoted with **disj**), no event can simultaneously be user permission check and code permission check. A trace is defined by a set of events (line 7). Empty traces are explicitly excluded from our specification. The fact **TraceEquality** ensures that two traces are equal if and only if they have the same alphabet. This is not true in general (see [2]). We define a service using two different views, a method view and a trace view (line 9).

In the first view, a service simply consists of a set of traces. In the second view, a service is defined as a service that further invokes other services and actions. The set of services a service invokes is not of interest to us but the set of actions it invokes is. The set of actions invoked by a service must be a subset of each of its traces.

We further define a system as composed of user services and internal services. User services are services that are available to users of the applications. Internal services are services that are not directly accessible to users but probably to other services. Systems that define service are not of interest to us (line 15). Besides this restriction, we require that no service be simultaneously user service and internal service. A *DPSSystem* is a kind of system (line 22) whose behavior is constrained by invariants *Invariant 1*, and *Invariant 2* (line 24). Invariant 1 (line 25) requires that any user service of a *DPSSystem* performs exactly one user permission check —be it directly or indirectly. That is, given a *DPSSystem*, any trace of any of its user services must contain exactly one user permission check. Invariant 2 (line 27) ensures that each service —be it user service or internal service— be protected by means of code permission checks.

The remainder of the section explains some assertions on the *DPS* style. The *Alloy* automatic analyzer is used to verify these assertions.

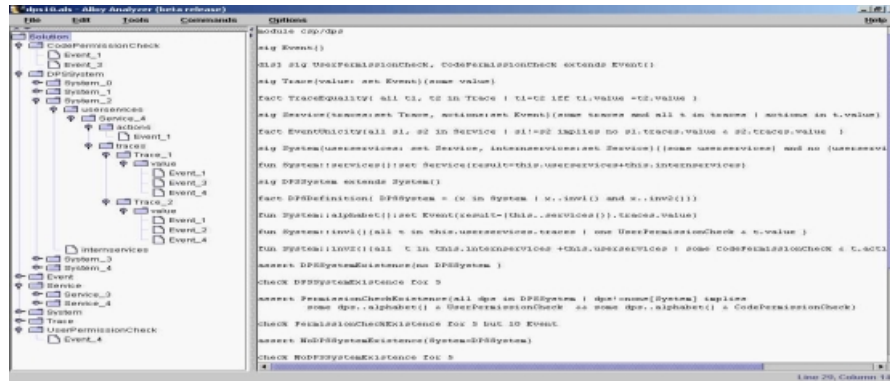


Fig. 5. An Automatic Instantiation of the DPS Style

**Theorem 1 (DPS System Existence).** *There exists a DPS System.*

This theorem is formulated in the specification at line 29. We pretend that there exists no *DPSSystem*. Next, we request the analyzer to verify the assertion (line 31). The number 5 in the check command represents the scope of the analysis. The skolemization of each of the basic types defined in the specification is limited to 5 elements. For instance, the type **Event** is converted by the analyzer to the set  $\{ \text{event\_0}, \dots, \text{event\_4} \}$ .

After checking the assertion, the analyzer replies with a counter-example (à la model checking) that proves the negation of the assertion, thus the existence of some *DPSSystem*. Some instances of *DPSSystem* found by the analyzer are shown in Figure 5. Among the 5 events, **event\_4** is a user permission check while **event\_1** and **event\_3**, are code permission checks. Five systems are also generated by skolemization. All of them are *DPS* systems. Let us analyze **System\_2**. It contains a single user service: **Service\_4** and no internal service. This unique service has two traces composed of the events 1,3,4 and 1,2,4 respectively. One can easily verify that this service indeed has 1) a code permission check in the set of actions it invokes, and 2) each of its traces performs exactly one user permission check.

**Theorem 2 (Permission Check Existence).** *Each DPS System contains at least one user permission controller and one code permission controller.*

Considering the implicit fact that each permission check is oriented towards an access controller the existence of a user/code permission check implies the existence of a user/code access controller. We thus prove that each *DPS* system has a user/code permission check. The formulation of this theorem is given at line 33. It stipulates that the alphabet of any *DPSSystem* includes some user permission checks and some code permission checks. The verification of this theorem is done by instructing the analyzer to check it (line 36). The analyzer fails to find a counter-example.

**Theorem 3 (Non-DPS System Existence).** *Not every system is a DPS System.*

We already showed that there exists some *DPS* system. However, in the counter example given by the analyzer, all systems are *DPS* systems. This raises the question of whether there exists a system that is not a *DPS* system. We pretend that the set of all systems is equal to the set of all *DPS* systems (line 38). The analyzer is able to find a system that is not a *DPS* system, thus falsifying our assertion.

**Theorem 4 (DPS Layered View).** *Any DPS System has a 3 layer architecture.*

We classify the set of services in three categories. The first category consists of services that contain a user permission check in their traces, but not in their actions. In other words, this is the set of services that do not directly invoke

user permission checks, but invoke other services that eventually perform user permission check (line 41). The second category of services consists of services that directly perform user permission checks thus have a user permission check in their set of actions (line 44). The third category of services is the set of services that has no user permission check in their traces (line 46). We show at lines 48-50 that these three layers are disjoint each from the others. We finally show at lines 52-54 that any *DPS* system is composed of these three disjoint layers. This theorem is illustrated by the MOTION layered architecture. The second layer is the ACTWS layer, the first layer is composed of all components below the ACTWS layer, and the third layer is composed of the business specific services and the presentation layer (see Figure 2).

We have formulated and analyzed other theorems on the *DPS* style. These theorems, however, are based on some other views, and thus other specifications and are not presented here.

## 6 Conclusion

We presented the *Dual Protection Style*, an architectural style for building access controlled software systems. This style distinguishes three kinds of components: user access control components, code access control components and protectees. Protectees are components whose services are protected by access controllers. The connectors between protectees and access control components are called permission checks. The *DPS* style constricts the interaction between protectees and protectors by means of two invariants. Besides an informal description of this style, we presented some of its advantages and shortcomings. We finally presented a formal specification of the *DPS* style using the *Alloy* specification language and verification of some of its properties.

A question that can be raised is how are applications constructed without the *DPS* style. We have presented a class of applications whose architectures are said to be degenerate forms of the *DPS* style that behave practically like instances of the *DPS* style. We gave examples of applications in this class. The notion of code permission recently raised attention (due to Java) although it was already investigated in the UNIX environment under the concept of domain type enforcement. We presented a framework of how to combine it with user access control.

The *DPS* style is not a panacea that is applicable in all cases. We presented an example application that does not satisfy the *DPS* style. Most advantages that we presented concerning this style are related to usability and easy applicability. The next step is to investigate how to specify the security properties of this style and how to prove them.

Another issue that needs to be clarified is that of composition. Is *DPS* compatible to software development approaches such as component based development? The answer to this question is yes. Components can still perform different user permission checks and being composed in a system that satisfies *DPS*. For this, the access control system must intentionally be built to support *DPS*. We

are currently constructing such an access control system. The idea behind it is to have a global context variable that records if a user permission check was already performed for a given trace. If so, the permission check is ignored and otherwise performed.

## References

1. Lee Badger, Daniel F. Sterne, David L. Sherman, and Kenneth M. Walker. A domain and type enforcement UNIX prototype. *USENIX Computing Systems*, 9(1):47–83, 1996.
2. C.A.R Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.
3. Premkumar Devanbu and Stuart Stubblebine. Software engineering for security: a roadmap. In *ICSE 2000 Special Volume on The Future of Software Engineering*, 2000.
4. Antoni Diller. *Z: An Introduction To Formal Methods*. Oreilly, Mai 1996.
5. Pascal Fenkam, Harald Gall, and Mehdi Jazayeri. Visual Requirements Validation: Case Study in a Corba-supported environment. In *Proceedings of the 10th IEEE Joint International Requirements Engineering Conference, Essen, Germany*, page to appear, September 2002.
6. Pascal Christian Fenkam. Dynamic user management system for web sites. Master's thesis, Graz University of Technology and Vienna University of Technology, September 2000. Available from <http://www.ist.tu-graz.ac.at/publications>.
7. Anup K Ghosh. Building software component from the ground up. *IEEE Software*, 19(1):14–16, January 2002.
8. GMD. Xql ipsi, <http://xml.darmstadt.gmd.de/xql/>, 2002.
9. Anthony Hall and Roderick Chapman. Correctness by construction: Developing a commercial secure system. *IEEE Software*, pages 18–25, February 2002.
10. Michael Howard and David LeBlanc. *Writing Secure Code*. Microsoft Press, 2001.
11. Daniel Jackson. Alloy: A lightweight object modelling notation. *ACM Transactions on Software Engineering Methodology*, 11(2), April 2002.
12. Daniel Jackson. Automatic analysis of architectural styles. Technical report, MIT Laboratory for Computer Sciences, Software Design Group, Unpublished Manuscript. Available at <http://sdg.lcs.mit.edu/dnj/publications.html>.
13. Kernighan and Pike. *The Unix Programming Environment*. Prentice Hall, April 1984.
14. Engin Kirda, Pascal Fenkam, Gerald Reif, and Harald Gall. A service architecture for mobile teamwork. In *Proceedings of the 14th International Conference on Software Engineering Conference and Knowledge Engineering Ischia, ITALY*, July 2002.
15. Charlie Lai, Li Gong, Larry Koved, Anthony Nadalin, and Roland Schemers. User Authentication and Authorization in The Java Platform. In *Proceedings of the 15th Annual Computer Security Conference, Phoenix, AZ*, December 1999.
16. Marry Shaw and David Garlan. *Software Architecture-Perspectives on an Emerging Discipline*. Prentice Hall, 1996.
17. Gary McGraw. Penetrate and patch is bad. *IEEE Software*, pages 15–16, February 2002.
18. Gary McGraw and Edward W. Felten. *Securing Java, Getting Down to Business with Mobile Code*. John Wiley and Sons, Inc, 1999.

19. Gian Pietro Picco and Gianpaolo Cugola. PeerWare: Core Middleware Support for Peer-To-Peer and Mobile Systems. Technical report, Dipartimento di Electronica e Informazione, Politecnico di Milano, 2001.
20. Nico Plat and Peter Gorm Larsen. An Overview of the ISO/VDM-SL Standard. In *ACM SIGPLAN Notices*. ACM SIGPLAN, September 1992.
21. Gerald Reif, Engin Kirda, Harald Gall, Gian Pietro Picco, Gianpaola Cugola, and Pascal Fenkam. A web-based peer-to-peer architecture for collaborative nomadic working. In *10th IEEE Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE)*, Boston, MA, USA. IEEE Computer Society Press, June 2001.
22. Michael P. Ressler. Security sensitive software development. In *IEEE International Carnahan Conference on Security Technology (ICCST)*, 1989.
23. Sun Microsystem. Security code guidelines. Technical report, Sun Microsystem, February 2000. Available at <http://java.sun.com/security/seccodeguide.html>.
24. The Institute of Applied Computer Science, IFAD. *The IFAD VDM Toolbox*. IFAD Denmark, 1999. Available from [www.ifad.dk](http://www.ifad.dk).
25. The Open Group. *Guide to Security Patterns, Draft 1*. The Open Group, April 2002. Available at [www.opengroup.org](http://www.opengroup.org).
26. Frank Tip and Jens Palsberg. Scalable Propagation-based Call Graph Construction Algorithms. In *Proceedings of the ACM Conference on Object Oriented Programming Systems, Languages and Applications (OOPSLA 2000)*. ACM Press, October 2000.
27. John Viega and Gary McGraw. *Building Secure Software, How to Avoid Security Problems the Right Way*. Addison Wesley Professional Computing Series, 2002.
28. Joseph Yoder and Jeffrey Barcalow. Architectural patterns for enabling application security. In *Proceedings of the Pattern Languages of Programming (PLoP) Workshop*, September 1997.

# A New Infrastructure for User Tracking Prevention and Privacy Protection in Internet Shopping

Matthias Enzmann, Thomas Kunz, and Markus Schneider

Fraunhofer Gesellschaft (FhG), Institute for Secure Telecooperation (SIT)  
Dolivostr. 15, D-64293 Darmstadt, Germany `firstname.lastname@sit.fraunhofer.de`

**Abstract.** Web technologies provide several means to infringe user privacy. This is especially true when customers buying tangible goods submit orders that contain their real identity and physical address. Then, in practice, the vendor can link this information with all information gathered about the customer, obtained through various means. In this paper, we present a solution that is based on mobile agents and a new infrastructure consisting of a *mobile agent base station* and a *cardinality observer*. This infrastructure can be used to prevent the vendor from directly linking information gathered about the customer with identifying information usually contained in the customer's order. The vendor can only assign customers to their correct profiles with some probability which depends on the number of candidate profiles. The new infrastructure allows the customer to decrease this probability in several ways. The usage of both the cardinality observer and the mobile agent base station deterministically guarantees to the customer that an agent only starts its journey when a desired threshold for the linking probability has been reached. In a second variant using only the mobile agent base station, the linking probability is decreased in a probabilistic manner by introducing a fixed delay before mobile agent release.

## 1 Introduction

Privacy threats and problems in the context of electronic commerce are extensively discussed in literature since the discovery of the economical importance of the Internet. In academic work done so far, solutions were mostly proposed for those cases in which the trade objects are restricted to intangible goods, e.g., see [2,11]. There, all phases of a typical business process consisting of *search*, *order*, *pay*, and *deliver* can be performed electronically. As a consequence, technical means developed so far for communication networks can be used to protect a customer's privacy, e.g., anonymity networks [9,12].

When dealing with tangible goods —e.g., books, CDs—, these techniques can also be used. In the *search* phase, when the buyer browses through the product catalog, anonymity networks can be used to prevent re-identification and to protect the buyer against some threats, e.g., price discrimination. But unfortunately, these techniques cannot be used to prevent *linkability* between the phases which allows the vendor to learn much more about the buyer than necessary for carrying out the business process.

In practice, in order to receive a tangible good a buyer has to reveal his identity and address to the vendor for reasons of delivery. It is not realistic to assume that either all buyers get their ordered goods in a *poste restante* manner where the identity of a buyer



can remain hidden, or that an additional third party receives the package on behalf of the buyer in order to hide his identity. Thus, the vendor learns at least who is buying what. But presently, the vendor can learn much more. Since the vendor can link the data given to him during the *order* phase to the buyer's activities during the *search* phase, the vendor gets much deeper insight into the buyer's interests than necessary for the following phases of the business process. This situation can be compared to physical world scenarios, where one is being completely traced while walking through a shop before going to the cash desk. The vendor's ability to link these two phases is achieved by using several tracking mechanisms.

The goal of this work is to present a convenient solution that prevents the vendor from linking data collected during the *search* phase with data obtained in the *order* phase of the same business process. In this work, we propose a mobile agent system and a specific infrastructure to solve the problems described above. Instead of filling goods into a virtual trolley during the *search* phase, the customer inserts the order information into a mobile agent. This agent is sent to a central *mobile agent base station*, a component of our infrastructure, which is permanently online. From this base station, the agent starts its journey. After having finished its tasks it returns to its base station.

The solution provides some additional useful features. By using another infrastructure component, the *cardinality observer*, it can be guaranteed that the agent is only sent when a desired number of candidates has been reached. As a direct consequence, this results in a linking probability that is below a specific value. In another weaker approach, the buyer can define a certain delay between the transfer of the agent and the start time of the agent's journey. Thus, there is the possibility to increase the cardinality of the group of potential candidates to be associated to a product, and thereby reducing the linking probability for the vendor in a probabilistic way. Another advantage of our solution results from mobile agent functionality. Due to its capability to actively make decisions with regard to results that were created on its journey, it is possible to define orders depending on the availability of products from previous orders of the same shopping tour.

## 2 Tracking Users

The vendor's ability for linking buyer activities can be based on several technical possibilities for tracking them in HTTP communication. Here, *tracking* means the re-identification of a user subsequently sending requests to a server. We can distinguish between the re-identification of users in distinct sessions and the re-identification of users within one session, possibly in different phases.

### 2.1 Profiles

As long as customers do not prevent the vendor from tracking them the vendor is able to record profiles. Such profiles can be understood as a sequence of requests for resources — like web pages — that can be associated with one session, and thereby with one customer, using the methods presented below. Eventually, the vendor's database contains a variety of profiles  $prof_1, \dots, prof_m$  that could be exploited afterwards. When a customer clicks

on products  $p_1, \dots, p_\nu$  while browsing through the catalog, then  $prof_i = (p_1, \dots, p_\nu)$  can be created. In the following, we show several possibilities for user tracking, suitable for creating profiles and associating profiles with customer identities. In general, user tracking can be achieved by exploiting specifics of the underlying transport protocol(s), or by specially created content. In both cases, certain protocol and/or content elements can be used to introduce a so called hidden channel.

## 2.2 Tracking by Protocol

*IP Addresses.* The first means to link the business process' phases is given by IP addresses. One possibility to cope with this problem from the customer's perspective is to disconnect in order to conduct the *order* phase with a new IP address obtained from his ISP after re-connecting —a solution which is not very convenient. Another option for the buyer to solve the problem could be the use of an anonymity network based on mixes [8,12] in order to hide his own IP address from the vendor. But if all requests are routed via the same sequence of mixes the probability for correct linking on the vendor side can be very high. They can be correlated with the IP address of the last mix in the chain. Another possibility for the buyer to hide his own IP address could be the use of the *crowds* approach where routes of subsequent messages can be different with high probability [9]. But all these anonymization countermeasures can be circumvented if a vendor uses cookies or dynamically generates URLs that allow session binding since anonymity networks do not protect against tracking through higher layer protocols.

*Cookies.* Another means to track the buyer and to link the phases is given by cookies [6]. As a countermeasure, a buyer may refuse cookies. But for shopping applications, they are often required. In a more laborious way, the buyer could first browse through the product catalog, then delete his cookies, and afterwards come back to the desired products and fill them in the virtual trolley without any further detours. Beside the inconvenience of this, the vendor can still track the buyer with the method presented below.

*Dynamic URLs.* The concept of a session has been introduced to HTTP through dynamic URLs. There, a part of the URL which is returned to a customer is unique in a sense that distinct requestors of the same web resource can be distinguished via this reference. This allows a web server to track a customer. A countermeasure for this would be to shut down the browser after the *search* phase and restart it in order to go directly to the *order* phase.

The previous considerations show, that presently there are only inconvenient solutions for the buyer to prevent a vendor from undesired linking.

## 2.3 Tracking by Content

A less obvious and more subtle method for tracking users is using the content information that a user requests. Using content tracking, a vendor serves all of its customers slightly different content in one phase in order to use this content in a subsequent phase (for re-identification of the same customer), when the content, or a part of it, is resent to the vendor. For instance, it is possible to associate a profile created in the *search* phase with a specific buyer from the *order* phase, if such content is presented in the order, no matter how carefully the buyer avoided all protocol tracking.

*Product identifiers.* One way of identifying a buyer's search phase *a posteriori* is using specially encoded product identifiers (PIDs). Such an identifier could contain a static part that refers to the product and a dynamic part that identifies a certain *search* phase. Since customers are used to numeric PIDs, they will not notice that they are given information leaking PIDs.

*Prices.* By giving different customers different prices, prices can also be used for the purpose of linking phases when contained in an order. In contrast to PIDs, prices only allow unique re-identification in theory. In practice, a vendor only has a finite range of prices which customers are willing to pay for a given product. If this range is depleted it is necessary to re-use some of the prices, and thereby uniqueness is lost.

*Product ordering.* Another method for content tracking is correlating the sequence in which products were clicked in the *search* phase with the product sequence submitted in a buyer's order. This is possible, since the ordered goods normally appear in the vendor's collected profile —possibly interleaved by other products that have been viewed but not ordered— in exactly the same sequence as in the buyer's order. As with prices, using product ordering for mapping profiles to buyers is probabilistic, since there can be several candidates.

### 3 Threats to Privacy

In today's web practice, we can identify a lack of privacy enhancing technologies [3]. It can be assumed that this lack of adequate privacy enhancing technologies is an additional barrier for the diffusion of e-commerce applications [4,13]. Thus, there is a need to change the present situation by the introduction of new technical solutions that allow to avoid, or to reduce the invasion of privacy.

In practice, a customer that decides to buy a tangible good electronically, usually has to reveal his identity and address to the vendor in order to allow delivery. Thus, a vendor clearly learns who is buying what. But by tracking the user through the whole business process, the vendor learns much more, e.g., what other products the customer is interested in. This data can be used to enrich the customer's profile stored in the vendor's database, and thereby, to obtain a more profound basis for customer categorization to be used for data mining.

### 4 Mobile Agents

Mobile agents are autonomous programs, which migrate through a network of sites to accomplish tasks or take orders on behalf of their owners. The owner of an agent can instruct it to visit many hosts in a network, and thereby execute some desired tasks for him. After having carried out all instructions, the agent returns to its base station and delivers the results it collected during its journey to its owner. One of the advantages for using mobile agent technology is that transaction costs for the agent owner are remarkably reduced since after leaving its owner it migrates from one host to the next autonomously. Thus, during this period the owner is not required to maintain his online

connection to the network. In the past years, lots of work has been done in the area of mobile agents and there is a variety of mobile agent systems, e.g., Aglets [7].

In the following, we will not focus on a specific mobile agent system. We will consider mobile agents in a rather abstract way. This means that exclusively those components of mobile agents will be considered which are of special relevance for the solution presented in this paper. In our level of abstraction a mobile agent  $a$  consists of the following components:  $a = (bc, r, d, \delta)$ . The component  $bc$  denotes the *binary code* of the agent to be executed. Furthermore,  $r$  describes the mobile agent's *route* as an  $(n+1)$ -tuple (with  $n \geq 1$ ) consisting of host addresses  $ad(s_i)$  that have to be visited on the agent's journey:  $r = (ad(s_1), \dots, ad(s_n), ad(bs))$ . This route is given by the agent owner. The agent starts its journey at a base station  $bs$  where it returns to when it has visited the stations contained in the route. Since the first migration is  $bs \rightarrow s_1$ , the first route entry is given by  $ad(s_1)$ . The component  $d$  denotes the *data* given by the agent owner. This data will be used as input for the computations at the hosts  $s_1, \dots, s_n$  on the agent's journey. Thus, we can think of it as  $d = (d_1, \dots, d_n)$  where  $d_i$  means the input data for  $s_i$  with  $1 \leq i \leq n$ . The data obtained as an output of the computations are contained in  $\delta$ . Similarly, we have here  $\delta = (\delta_1, \dots, \delta_n)$ .

## 5 Adaption of Agent Components

In the following, we will adapt the previously introduced components of a mobile agent according to the requirements of our solution. Thereby, the main protection goal we have in mind is *privacy*. Additionally, we also focus on *data origin authentication* which includes *data integrity* of the mobile agent, and *non-repudiation*.

In order to do this, we introduce some expressions. Let  $E_{e_i}(d_i)$  denote a ciphertext obtained via an asymmetric algorithm  $E$ , e.g., *RSA* [10], for the encryption of the data  $d_i$  by using  $s_i$ 's public key  $e_i$ . Furthermore,  $Sig_x(y_1, \dots, y_\nu)$  denotes a digital signature of party  $x$  on some contents  $y_1, \dots, y_\nu$ . This allows us to introduce an element  $\tilde{d}$  consisting of encrypted input data for  $s_1, \dots, s_n$  and a signature of the agent owner  $b$  which will be the buyer:  $\tilde{d} = (E_{e_1}(d_1), \dots, E_{e_n}(d_n), Sig_b(E_{e_1}(d_1), \dots, E_{e_n}(d_n), bc))$ . Furthermore, we protect the agent route in a similar way as it was done in [14] by an onion-like encryption concept. In each layer of the onion structure we have a host address and a signature by the agent owner  $b$  on this address combined with other agent components.

$$\begin{aligned} \tilde{r} = & ((ad(s_1), Sig_b(ad(s_1), bc, \tilde{d}), \\ & E_{e_1}(ad(s_2), Sig_b(ad(s_2), bc, \tilde{d}), \\ & E_{e_2}(ad(s_3), Sig_b(ad(s_3), bc, \tilde{d}), \\ & \vdots \\ & E_{e_{n-1}}(ad(s_n), Sig_b(ad(s_n), bc, \tilde{d})) \dots), \\ & ad(bs)) \end{aligned} \quad (1)$$

The principle of how to deal with  $\tilde{r}$  is as follows. The base station learns through the first entry of  $\tilde{r}$  where to dispatch the agent. Before the agent is sent to  $s_1$ ,  $bs$  deletes this entry from  $\tilde{r}$ . When  $s_1$  receives the agent with the new  $\tilde{r}$ , it decrypts the ciphertext contained in

$\tilde{r}$  and obtains the successor address  $ad(s_2)$ , a signature, and a further ciphertext. Before sending the agent to  $s_2$ , the address and the signature are deleted from  $\tilde{r}$ . This procedure will be repeated until the agent arrives at  $s_{n-1}$ . Here the last decryption is necessary, i.e.,  $s_{n-1}$  gets the last address  $ad(s_n)$  and signature contained in the onion. After deletion of these parameters from  $\tilde{r}$ , the agent is sent to  $bs$  as defined by the last entry  $ad(bs)$ . The idea of the route protection is that visited hosts do not learn to which hosts the agent migrates to except their respective predecessor and successor. If the agent owner has the interest that visited hosts do not get information about other hosts contained in the route, he can introduce dummy hosts in the route that just forward the agent.

The signatures contained in  $\tilde{r}$  and  $\tilde{d}$  provide data integrity, and allow that modifications can be detected. The signatures depend on several agent components that prevent attackers from the exchange of components taken from different agents belonging to the same agent owner  $b$ .

After having produced the computation results  $\delta_i$  at host  $s_i$ , they will be signed by  $s_i$ . Thus, we define  $\tilde{\delta}_i = E_b(\delta_i, Sig_{s_i}(\delta_i, bc, \tilde{d}))$ . At the end of the journey, the agent contains all computation results, i.e.,  $\tilde{\delta} = (\tilde{\delta}_1, \dots, \tilde{\delta}_n)$ . Initially, portions  $\delta_i$  are empty. In the following, we assume that an agent  $\tilde{a}$  consists of the following components:  $\tilde{a} = (bc, \tilde{r}, \tilde{d}, \tilde{\delta})$ .

## 6 Achieving Privacy via Agents

In the following, we show how to prevent a vendor from linking gathered information about a buyer obtained by tracking him during the *search* phase with his real identity when he submits an order. In 6.1, we show how to deliver an order with an agent. In 6.2 and 6.3, we explain how different kinds of tracking are prevented. In 6.4, we point out how the linking probability for the vendor can be decreased by using functionalities of the new infrastructure components. Routes containing several vendor addresses are discussed in 6.5.

### 6.1 Order Delivery via Agents

For simplicity, we assume that a buyer  $b$  decides to buy at just one vendor, say  $s_1$ . While searching through  $s_1$ 's catalog,  $b$  has a look at various products, and finally decides to buy a subset  $p_1, \dots, p_k$  of these products. During this time,  $s_1$  can track  $b$ 's activities and store them in a new profile, say  $prof_i$ , which belongs to a single identity, but he is unable to map these activities to  $b$ 's identity. Then,  $b$  creates the order information including identifiers of these products, his name and address, to be contained in  $d_1$ . After that,  $b$  creates  $\tilde{d} = E_{e_1}(d_1, Sig_b(E_{e_1}(d_1), bc))$  and  $\tilde{r} = (ad(s_1), ad(bs))$ , and finally  $\tilde{a} = (bc, \tilde{r}, \tilde{d}, \tilde{\delta})$ , with  $\tilde{\delta} = \emptyset$ .

In a next step,  $b$  will transfer  $\tilde{a}$  to the base station  $bs$  and instruct it to dispatch  $\tilde{a}$ . This base station is maintained by a specialized mobile agent base provider. This provider does not play the role of a trusted third party in the usual sense. In fact, the buyer's trust in the provider is rather minimal. Since the provider does not get any information about the *search* phase, he cannot exploit this information for himself nor pass it to others. This is due to the fact that the agents do not contain information like IP address,

cookie information, or dynamic URLs that can be exploited by the vendor for linking. The provider is trusted that he does not give the buyer's IP address to the vendor, that he does not delete agents after having received them, and that he does not release the agents earlier as declared (to become clear later).

Now,  $\tilde{a}$  can migrate to  $s_1$  and deliver its data  $d_1$ . Since  $d_1$  is encrypted asymmetrically, it can only be opened by  $s_1$ , i.e.,  $bs$  does not learn which products were ordered by  $b$ . Furthermore,  $s_1$  can check by verifying the signature if the order was undeniably created by  $b$  and if it was modified. When the data contained in the agent does not seem to be corrupted,  $s_1$  creates the output  $\delta_1$  and  $\tilde{\delta}_1 = E_b(\delta_1, \text{Sig}_{s_1}(\delta_1, bc, \tilde{d}))$  to be handed over to the agent. E.g., such a computation output could be  $s_1$ 's confirmation of having received the order and a declaration of carrying it out immediately. For integrity reasons, all outputs should be signed by the vendor. Afterwards, the agent is sent to  $bs$  according to the last route entry  $ad(bs)$ . There,  $\tilde{a}$  waits for  $b$  until he connects the next time to  $bs$ .

## 6.2 Tracking by Protocol

Using our mobile agent approach, the vendor cannot use IP addresses, cookies, or dynamic URLs for linking the received order to *search* phase activities, since he is no more directly receiving the order from the buyer  $b$ . In order to get a deeper insight into  $b$ 's interests,  $s_1$  can try to link available profile data  $prof_1, prof_2, \dots$  with the data obtained from the agent to find out  $b$ 's real profile  $Profile(b)$ . But this means for him to carry out a random experiment, assuming there is no other information that allows linking. Since  $b$  has ordered  $p_1, \dots, p_k$ , the vendor searches in his database for profiles that include  $p_1, \dots, p_k$  and that have been recorded in a relevant time interval. Certainly, the order does not belong to a profile that has been recorded one year ago. If we assume that  $s_1$  finds  $\mu \geq 1$  profiles  $prof_1, \dots, prof_\mu$  each containing at least  $p_1, \dots, p_k$ , then for  $s_1$  the probability for correct linking of a stored profile to  $b$  is given by  $P(Profile(b) = prof_i) = \frac{1}{\mu}$ , provided that there is no other hidden information that makes linking easier. The probability depends on the number  $\mu$  of profile candidates. Obviously,  $\mu$  decreases monotonically when  $k$  increases, i.e., including a higher number of products in the order can increase the linking probability.

## 6.3 Tracking by Content

Since content tracking is not easily detectable *per se*, we are introducing certain rules and procedures that must be followed in order to prevent content tracking, or at least make it detectable by a buyer.

*Product identifiers.* A malicious vendor could offer such an agent-based privacy protecting service and still try to link the *search* and *order* activities via leaking product IDs (PIDs). In such a case, the linking probability for the vendor would be  $P(Profile(b) = prof_i) = 1$ . In order to prevent such attacks, PIDs must be verifiable by the buyer in a sense that hidden channels become obvious. E.g., this could be achieved by some natural language encoding of PIDs. In such a scheme, the set of potential candidates for PIDs is rather small, e.g., for a music CD the PID could be (*artist name, title*) instead of some ID *B0000262WI* and therefore, a customer can detect with high probability if there is some hidden information.

*Prices.* Another possibility to increase the linking probability could be achieved via variable pricing strategies. E.g., an unfair vendor could offer the same product at slightly differing prices. When he later receives an order that contains the price presented in the offer then the vendor can easily exclude all those profiles from the potential set of candidates in which he offered the specific product for a different price. If the buyer does not send the prices the vendor might sell him the ordered goods at (theoretically) any price. In order to prevent this, we propose that the vendor gives a temporary price guarantee for any product he sells, i.e., he commits himself to charge a fixed price for a given product for some period of time. This could be realised, e.g., by creating and publishing a vendor-signed document that contains the products, their respective prices, and the validity period. This guarantee can be downloaded by the buyer and be kept for later reference. Of course, the concept using guarantees only works in cases where prices do not change too frequently. If a vendor serves customers with distinct price guarantees in order to track them, then he does not know the exact prices at which he offered his products to a specific customer. Thus, if he is trying to identify a buyer by his offered prices, the vendor can only guess the price a customer expects according to his downloaded guarantee. Should the vendor make a wrong guess to the buyers disadvantage, he could present his price guarantee, proving that the vendor tried to deceive him. In this case, since the price guarantee is disclosed, the vendor would be able to link a profile to the customer. However, disputes are unattractive for the vendor because if they occur frequently the vendor's reputation will degrade. In addition, if the vendor makes a wrong guess to the buyer's advantage, i.e., billing him for a lower price than he expected, then he will surely accept and the vendor lowers his profit and also would link to a wrong profile.

*Product ordering.* In order to counter the effects of correlating the click sequence  $prof_i$  from the *search* phase with the product ordering in the order, the buyer should have the possibility to arrange the PIDs in an arbitrary sequence. This sequence can be randomly created, or obtained by sorting the PIDs, e.g., in lexicographic order. This is sufficient as long as there are other profiles stored in  $s_1$ 's database produced by other customers that also viewed at least  $p_1, \dots, p_k$  in one session and therefore help to decrease the vendor's linking probability.

#### 6.4 Decreasing Probability for Linking

In the following, we will show how a buyer can decrease the linking probability. Since this probability depends on the number  $\mu$  of candidates it is the goal to have large  $\mu$ . One solution would be to motivate a large community to visit the vendor's catalog. But this is not realistic. In the following, we provide two solutions that allow buyers to hide their profiles in larger groups.

**Start of Journey Dependent on Time.** We propose that the base station should provide a service for the buyer to define a delay between receiving the agent from the buyer and dispatching it, eventually. Since this delay is unknown to the vendor, he does not know whether to link the order with profiles stored in the past minutes, hours, or even days. In order to have the correct profile with high probability in the set of candidates, the time

interval considered by the vendor should be large. But then,  $\mu$  can also be assumed to have increased. Of course this depends on the specific statistics of the vendor's web site. The dispatching time can be specified by the buyer by instructing the base station not to release the agent before a delay  $\Delta\tau$ , or not before a time  $MM:dd:hh:mm$ . When price guarantees are used, the delay must not exceed the validity period of the price guarantee.

This solution only requires the existence of the base station. But on the other hand, there is no guarantee that the number of profile candidates is high enough. This guarantee is achieved by using the cardinality observer.

**Start of Journey Driven by Cardinality.** A delay still bears the risk of re-identification if the number of candidates  $\mu$  is still relatively small or even one. In that case, a buyer  $b$  might not want to submit his order until  $\mu$  reaches a certain threshold  $t$ , i.e., there is a group of  $\mu \geq t$  customers that also viewed the products  $p_1, \dots, p_k$ —maybe beside others—to be bought by  $b$ . Thus, there is a need to measure  $\mu$  in order to control the linking probability. For that, we propose two approaches. The first one, described as *perfect counting*, is accurate on the size of  $\mu$ , however, possibly requires higher transaction costs. The second approach, called *partial counting*, is based on an estimate  $\mu' \leq \mu$  by counting only profiles of those customers that ordered some products. Thus, customers that only viewed the products are not counted. This approach yields almost no additional transaction costs. Both approaches require a new infrastructure component, the *cardinality observer co*. The *co* counts the number of times specific sets of products from some vendor have been viewed within distinct sessions. This also ensures that products are not double counted, for instance, when a customer views a product again within the same session. Beside its counting facilities, *co* offers a notification service in order to inform *bs* that the desired threshold  $t$  for a specific agent has been reached. The *co* is trusted to send only honest notifications, i.e., it will not send a notification before  $t$  has been reached. In the following, we assume that both parties playing the roles of *bs* and *co* are distinct and that they are trusted not to collude.

*Perfect Counting.* In our first approach, a customer  $b$  sends messages containing his profile data—products  $p_1, \dots, p_\nu$  he viewed at some vendor  $v$ —to *bs* and forwards some non-identifying part of this data to *co*. This forwarded data is protected in a way that the customer identity remains hidden to *co* while the content remains hidden to *bs*. This can be achieved by a mechanism in which the buyer encrypts this data with *co*'s public key in order to hide the products from *bs*. The sequence of messages takes the form  $E_{co}(rand_1, v, p_1), \dots, E_{co}(rand_1, v, p_\nu)$ , where  $rand_1$  is a random number for linking messages from one customer's session, and  $p_1, \dots, p_j$  are the product IDs contained in web pages viewed by  $b$ . These subsequent messages are sent automatically in parallel to  $b$ 's search activities. They are sent regardless of  $b$ 's decision to buy at  $v$  or not. When  $b$  buys some of the products  $p_1, \dots, p_k$  out of the products  $p_1, \dots, p_\nu$  he viewed, he must send a final message  $E_{co}(rand_2, v, p_1, \dots, p_k, t)$  to *co* that contains a random number  $rand_2$ , the vendor  $v$ , the products to be bought  $p_1, \dots, p_k$ , and  $b$ 's preferences for the minimal size  $t$  of candidates regarding these products. Here,  $rand_2$  serves as a temporary agent ID for that agent which waits at *bs* to be released when the number of candidates has reached  $t$ . Of course, the set of candidates is not always growing. If candidate profiles become too old, they should not be considered anymore.



In this approach, *co* essentially learns the same information from the *search* phase as the vendor, hence, it has perfect knowledge regarding the vendor's linking probability. However, in contrast to the vendor, *co* is never given *b*'s name and therefore it cannot link the obtained profile to *b*'s identity. Thus, *co* cannot exploit this data in order to infringe *b*'s privacy. Furthermore, *bs* does not get aware of *b*'s profile data or *b*'s order data. This approach clearly increases transaction costs since IDs of products viewed by *c* are transferred to *co* in order to allow accurate counting.

*Partial Counting.* In this approach, *b* does not send his profile data  $p_1, \dots, p_\nu$  to *co* via *bs* while he is searching through the vendor site. Instead, he stores his profile data locally and transfers them to *bs* together with the agent in case he decides to order some products  $p_1, \dots, p_k$ . Of course, both the order data and the profile data are protected so that *bs* does not become aware of them. This is achieved by embedding a ciphertext  $E_{co}(v, (p_1, \dots, p_k), (p_1, \dots, p_\nu), t)$  into the agent. After the agent has been transferred to *bs*, *bs* extracts this message and forwards it to *co* together with a *bs*-chosen agent ID. This agent ID is necessary in order to notify *bs* which agent should be released. Also here, *b* hides his identity from *co*.

Using this method, *bs* must only send one additional message compared to the approach with fixed delay where we do not have a *co*. Compared to the approach for perfect counting, here the costs for sending messages can be drastically reduced. However, this approach is possibly less accurate than the former one, since here only customers that finally ordered some products are considered by *co* instead of the possibly larger set of customers which viewed these products. Hence, when a customer orders  $p_1, \dots, p_k$ , then *co* will notify the corresponding agent to start its journey when the condition  $\mu' \geq t$  becomes true, where  $\mu'$  is the number of customers that also ordered  $p_1, \dots, p_k$  — potentially beside other products. Clearly,  $\mu' \leq \mu$ , where  $\mu$  is the number of profiles containing  $p_1, \dots, p_k$  which were recorded by the vendor. This means that agents will be delayed longer than necessary because the threshold  $t$  might have already been crossed since non-buying customers are ignored. Also here, the number of relevant candidates  $\mu'$  may be reduced when profiles should not be considered anymore.

**Start of Journey Dependent on Combination of Delay and Cardinality.** The ideas of the previous considerations can be combined in order to trigger the start of the agent journey. With this, it is possible that an agent can be released when at least one of both conditions is satisfied —time or cardinality. E.g., *co* can notify *bs* to release the agent when the desired cardinality threshold has been reached even though the delay given to *bs* is not reached. In this case, it is not necessary for the agent to wait longer, and the buyer may get the ordered goods earlier. In the opposite case, when the desired delay is reached without satisfying the cardinality condition, the agent starts its journey even if the linking probability is still too large. By this combination, the buyer can make sure that the delivery of the ordered products will not be delayed indefinitely.

## 6.5 Routes with Several Shops

So far, we have considered only cases in which a mobile agent exclusively visits one vendor for delivering some order information. Shopping tours with several vendors can

be advantageous, e.g., in cases when an order for vendor  $s_i$  should only be ordered if products at  $s_j$ , to be visited before, are available. Furthermore, this possibility reduces the number of mobile agents migrating through the network. Concerning the order information transferred by the agent, there arise no privacy problems since this information can only be opened by the desired vendor because of asymmetric encryption.

One could argue that with longer routes vendors learn which other hosts have to be visited on the shopping tour which gives some further insight into the buyers behaviour or interests. But this threat can be tackled with the route protection scheme presented in expression (1). With this solution, a vendor only learns about his predecessor and successor in the shopping tour. If a buyer does not like that a vendor gets aware of other shops that could be predecessor or successor then the buyer could create the agent route  $\tilde{r}$  with intermediate  $bs$ . This increases the number of migration steps on the agent's journey but allows the buyer to enhance his privacy.

In the case of dependent agent computations —e.g., when order delivery at  $s_i$  depends on computation results at  $s_j$ — results have to be communicated from one vendor to another. If one is immediately following the other then the exchange of the required results can be done directly —provided that the buyer is willing to reveal these shopping tour stations to the vendors. If the buyer decides to hide the vendor identity from other vendors by agent exchange via  $bs$ , then the results can be encrypted for  $bs$  which decrypts and re-encrypts them for the vendor that requires these results. By applying such concepts, it can be achieved that a vendor will not get aware of what  $b$  is ordering at other vendors as far as the vendors are not colluding and exchange their trading information.

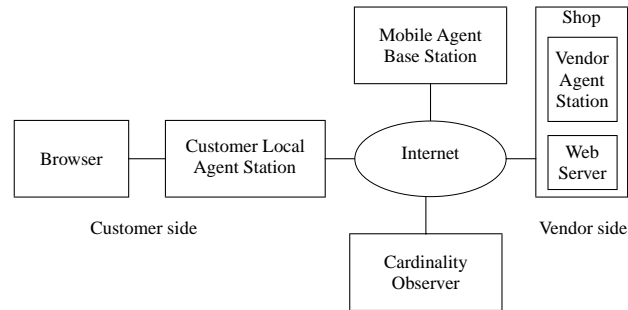
## 7 Architecture

In the following, we will sketch the architecture including the infrastructure components that are needed for our solution. Figure 1 depicts the basic components which are required in our system. The customer uses a simple web browser and the *Customer Local Agent Station* (CLAS) which is a specific component of our solution. The vendor runs a usual web server and a *Vendor Agent Station* (VAS). Additionally, there is the *Mobile Agent Base Station* (MABS) which is provided by a third party that offers mobile agent services to customers. And finally, we have the *Cardinality Observer* (CO) operated by another third party.

### 7.1 Customer Components

The customer is running two applications: a browser and a CLAS. The CLAS component is independent of a specific vendor. The CLAS software is provided by a party that is trusted to not have embedded a Trojan horse in it. The CLAS comprises several functionalities: (1) a client-side proxy, (2) a *Product Selector* (PS) from which products can be put into a trolley-like agent, (3) a *Mobile Agent Generation* (MAG) component like an agent workbench, and (4) a client to use the services of the MABS.

When the customer browses through the vendor catalog all requests are first sent to the CLAS (figure 2, message 1) which forwards them to the vendor system (figure 2, message 2). The corresponding vendor response is received by the CLAS (figure 2,

**Fig. 1.** Architecture overview

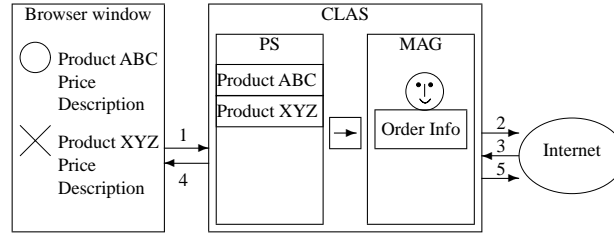
message 3). Message 3 consists of a part destined to the CLAS and another part destined to the browser. After having extracted his own part, the CLAS forwards the remaining message part to the browser (figure 2, message 4). The browser message part consists of normal web page content, e.g., displaying products with some additional information. The CLAS message part consists of the product identifiers for the same products which are displayed at the same time by the browser. These identifiers are displayed by the PS subcomponent. In contrast to normal scenarios, when the customer decides to buy a product, he does not put it in some virtual trolley by clicking in the browser window. The browser window is just used to get product information and to navigate through the catalog. Instead, he takes the product ID from the PS and transfers it to the MAG. On this agent workbench, the customer composes the agent in the desired manner, i.e., he has to select the appropriate agent from the set of potential candidates. Then, he has to create the input data while respecting the constraints which exist on the shopping tour, e.g., buy at shop *B* only if the order at shop *A* can be fulfilled. Such constraints influence the order of hosts to be visited, and thus, have to be considered when the agent route is composed. After having selected an appropriate order of shops, the CLAS re-arranges the sequence of the selected products as described in subsection 6.3, and then the MAG automatically creates a route according to equation (1).

The CLAS is also responsible for the creation of profile messages which are destined for the CO. In case of *perfect counting*, it also sends them to the MABS. In case of *partial counting*, it embeds them into the agent.

After having finished the shopping tour, the buyer transfers the agent to the MABS (figure 2, message 5). Then, he can also give some additional instructions, such as the agent release delay, and the threshold.

## 7.2 Mobile Agent Base Station

The MABS provides a docking station for an agent, as well as an alarm clock and relaying/sending/receiving facilities. The alarm clock is used to wake up the agent after a pre-determined period of time in the case when the agent's start is controlled by a delay. Message relay is needed if the MABS supports cardinality driven schemes with perfect counting, sending is required if a MABS supports partial counting. In both cases



**Fig. 2.** CLAS and browser interaction

the MABS receives the notification message from the CO, and the MABS has to map the notification to the correct agent. When any condition for the agent release is satisfied, the MABS lets the agent start its journey. When the agent returns to the MABS after its journey, it is stored in a personal inbox which is only accessible by the corresponding buyer. After the receipt of the agent, the MABS can inform the buyer that the agent has returned and can now be picked up. Then, the buyer can connect to the MABS and download the agent in order to verify the results of its journey. Therefore, the MAG also comprises the necessary functionality for decrypting the data which were encrypted for the customer by the visited vendors.

### 7.3 Cardinality Observer

The main goal of the CO is to send notifications for agent releases when desired thresholds are reached. This means that the CO has to keep track of all relevant profiles obtained from customers via the MABS. Beside a profile  $prof_i$ , the CO has to store the time  $\tau_i$  at which the corresponding profile was received—as will become clear soon. Therefore, all pairs  $(prof_i, \tau_i)$  belonging to a vendor  $v$  are stored in some vendor-specific set  $S_v = \{(prof_1, \tau_1), \dots, (prof_m, \tau_m)\}$ . If a buyer orders  $p_1, \dots, p_k$  from  $v$ , then the CO has to check if  $p_1, \dots, p_k$  are contained as a whole at least  $t$  times in  $prof_1, \dots, prof_m$ . As soon as the check result is positive, then the release of the corresponding agent containing the order for  $p_1, \dots, p_k$  will be initiated at the MABS.

It is important that cardinality checks should not be based on profiles which are too old, since such checks could lead to undesired consequences. E.g., if a buyer sends an order for which  $t - 1$  matching profiles are stored at the CO all of which are older than a year, this new profile would trigger the agent's release. But in such a case, the vendor would be able to link the profile correctly, since he would probably not use such old profiles for linking. Thus, at time  $\tau$  an old profile  $prof_i$  with time  $\tau_i < \tau - \Delta T$  should not be considered in the cardinality check, i.e., these entries should be deleted from  $S_v$ . There,  $\Delta T$  is a parameter that has to be defined by the CO.

### 7.4 Vendor Components

On the vendor side, the solution requires a *Vendor Agent Station* (VAS) beside the usual vendor infrastructure. This VAS provides all the required functionality that is necessary

for the interaction with the customer agents generated by the customer, i.e., basic agent execution facilities and required security mechanisms as previously explained. Furthermore, the vendor has to provide messages that consist of a CLAS part and a browser part. Whenever a customer requests product information of potentially ordered goods by using mobile agents, the vendor's web server responds with adequate messages which can be processed by the CLAS.

By using the CLAS for selecting the product to be ordered in the proposed way, we can be sure that no hidden channel is established to the vendor. Hidden channels could be possible when a buyer selects a product directly from a browser. Such a hidden channel would increase the linking probability by reducing the set of candidates to those parties that really selected the corresponding products. But our solution deals with a set of potential candidates containing all those parties that only viewed or finally decided to buy the product.

## 8 Related Work

Other work done in the area of privacy protection also focusing on the prevention of exploiting server log files can be found in anonymity literature, e.g., see [8,9,12]. Typically, in anonymity the goal is on *sender anonymity*, *receiver anonymity*, and *unlinkability of sender and receiver*. There, the focus is not on the question on how to avoid linkability of subsequent messages exchanged between a customer and a web server. Since we are dealing with applications in which the buyer reveals his identity in some phase of the business relationship, we require this unlinkability of messages exchanged in distinct phases. Most of the work done so far (e.g., [2,11]) was dealing with online selling. There, the focus for privacy protection has exclusively been on the trade of intangible goods where anonymity networks can be used without problems. This means that no real identity and physical address have to be revealed.

Other work dealing with privacy protection concerning gathering information on customer activities in business processes was presented in [1,5]. There, they deal with privacy protection in customization and targeting.

## 9 Conclusion and Future Work

We have presented a solution which allows a buyer of tangible goods to enhance his privacy. Our solution applies mobile agents and a new infrastructure in order to prevent a vendor from linking information gathered in the *search* and *order* phase of the business process. For the vendor, the linking of these activities is a random experiment. The success probability of this experiment depends on the number of customers that have viewed the corresponding products. The proposed system and the new infrastructure components allow the introduction of distinct variants like agent delay and cardinality control which both have advantageous effects on the linking probability. Furthermore, we have considered the case of shopping tours in which an agent delivers orders to several vendors. The proposed solution can be implemented in existing electronic shops where tangible goods are sold in order to enhance the privacy of the buyer.

Potential problems regarding attacks on the cardinality observer need to be resolved in future work.

## References

1. Robert M. Arlein, Ben Jai, Markus Jakobsson, Fabian Monrose, and Michael K. Reiter. Privacy-preserving global customization (extended abstract). In *Proceedings of the 2nd ACM conference on Electronic Commerce (EC'00)*, October 2000.
2. Feng Bao and Robert Deng. Privacy protection for transactions of digital goods. In *Information and Communications Security (ICICS 2001), Third International Conference, Proceedings*, number 2229 in LNCS. Springer Verlag, November 2001.
3. Roger Clarke. Internet privacy concerns confirm the case for intervention. *Communications of the ACM*, 42(2), February 1999.
4. Donna L. Hoffman, Thomas P. Novak, and Marcos Peralta. Building consumer trust online. *Communications of the ACM*, 42(4), April 1999.
5. Ari Juels. Targeted advertising ... and privacy too. In *Progress in Cryptology — CT-RSA 2001, The Cryptographers' Track at RSA Conference 2001 San Francisco, Proceedings*, number 2020 in LNCS. Springer Verlag, 2001.
6. D. Kristol and L. Montulli. HTTP State Management Mechanism. RFC 2109, February 1997.
7. Danny B. Lange and Mitsuru Oshima. *Programming and Deploying Java Mobile Agents with Aglets*. Addison-Wesley, 1998.
8. Michael G. Reed, Paul F. Syverson, and David M. Goldschlag. Proxies for anonymous routing. In *Proceedings of 12th Annual Computer Security Applications Conference (ACSAC'96)*. IEEE Press, December 1996.
9. Michael K. Reiter and Aviel D. Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1), 1998.
10. Ron L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2), February 1978.
11. Stuart G. Stubblebine, Paul F. Syverson, and David M. Goldschlag. Unlinkable serial transactions: Protocols and applications. *ACM Transactions on Information and System Security*, 2(4), 1999.
12. Paul F. Syverson, Michael G. Reed, and David M. Goldschlag. Private web browsing. *Journal of Computer Security — Special Issue on Web Security*, 5(3), 1997.
13. Huaiqing Wang, Matthew K.O. Lee, and Chen Wang. Consumer privacy concerns about internet marketing. *Communications of the ACM*, 41(3), March 1998.
14. Dirk Westhoff, Markus Schneider, Claus Unger, and Firoz Kaderali. Protecting a mobile agent's route against collusions. In *Selected Areas in Cryptography, 6th Annual International Workshop (SAC'99)*, number 1758 in LNCS. Springer Verlag, 2000.

# Different Smartcard-Based Approaches to Physical Access Control\*

Óscar Cánovas<sup>1</sup>, Antonio F. Gómez<sup>2</sup>, Humberto Martínez<sup>2</sup>, and  
Gregorio Martínez<sup>2</sup>

<sup>1</sup> Department of Computer Engineering

<sup>2</sup> Department of Information and Communications Engineering  
University of Murcia, Spain

ocanovas@ditec.um.es, {skarmeta, humberto, gregorio}@dif.um.es

**Abstract.** Although the basic function of physical access control can be considered simple, that is, to allow authorized personnel to gain access to buildings and deny unauthorized entry, the problem should not be so drastically simplified. In this work, we present two solutions to this problem that make use of special devices named TICA and smartcards. These proposals follow different approaches to manage permissions and authorizations. On the one hand, we present a centralized and on-line solution that makes use of one central database containing the privileges assigned to every user. On the other hand, we describe a system where the authorization-related information is managed in a distributed way. It is an off-line solution based on a RBAC (Role Based Access Control) model that has been designed using authorization certificates.

## 1 Introduction

Physical security involves the provision of mechanisms to prevent unauthorized physical access to specific facilities, such as laboratories or offices. Moreover, physical access control to computing equipment is especially significant since disabling the servers, routers, or other telecommunications equipment can have a great impact on a large community of people.

Physical access control is a classic problem. In general, a common set of issues exists that must be addressed. First is the problem of key distribution, that is, how to provide the appropriate keys to the authorized users who might need them. In the same way, the keys must be canceled or revoked after some special conditions in order to avoid unauthorized accesses. Then, it is necessary to specify how the user information, i.e. personal information and access privileges, will be managed. Additionally, people use to act on some role during their everyday tasks, and it is important to reflect those roles in the access control management process.

---

\* Partially supported by TIC2000-0198-P4-04 project (ISAIAS), and by 2I02SIU0005 project (m-PISCIS)

Traditionally, the main proposals have been based on a centralized database containing information about valid users. For example, we can imagine a user having some piece of identifying digital data, most probably unique, which is presented to a special device located at the entrance of a particular building or room. The device does not know which keys are valid ones hence it must perform a query to the central database asking about the user privileges. That piece of digital data can be a public key, a distinguished name, an identity certificate, a fingerprint, or simply a login name.

On the other hand, in recent years, public key cryptography has also been proposed as a tool for solving the problems related to authorization and access control. Once the questions about identity in distributed systems have been solved by the X.509 [HFS99] or OpenPGP [CDFR98] standards, determining what the identities should be allowed to do is becoming an interesting research topic. From the *Trust Management* [BFIK99] approach to specifications like SPKI/SDSI (Simple Public Key Infrastructure / Simple Distributed Security Infrastructure) [EFL<sup>+</sup>99], the main intention is to capture security-relevant information and to bind that information to public keys. Moreover, these proposals might be used to implement some of the RBAC (Role Based Access Control) models proposed by Sandhu [SCFY96]. As we will see, those RBAC models are suitable in order to manage authorization-related information in physical access control systems.

The systems we present in this paper make use of a special device called Intelligent Access Control Device (TICA) [oM01]. TICAs contain smart cards readers and are able to exchange information with an application server. They also carry out authorization decisions regarding information concerning access control policies and digital certificates. It is, currently, one of the main elements of the two systems that allow the establishment of physical access and working time control policies for the personnel in our university. On the other hand, smartcards are used as information repositories, and also as special devices performing cryptographic functions. In general, the authorization information contained in these devices is used by the users in order to demonstrate their rights. As we will see, that information can range from unique identifiers to digital certificates.

In this paper, we propose two systems for physical access control. These solutions are based on a centralized and a distributed access control management system respectively. As we will see, our RBAC and decentralized system provides some additional mechanisms which are not present in our centralized proposal, such as non-repudiation, scalability or dependence of network connectivity.

This paper is organized as follows. Section 2 provides a description of the main elements composing a TICA device. Section 3 outlines the main features of the smartcards we have used to implement our systems. Section 4 presents the design and the implementation of the centralized system. Section 5 contains the details about the distributed authorization management system. It explains how some of the drawbacks related to the first solution can be overcome, and depicts some implementation issues. Finally, Section 6 makes some concluding remarks.



## 2 Description of the TICA

TICA devices have been developed by our university for physical access control purposes. They are located at the entrances of the different buildings and/or departments. A TICA device is composed of different elements to interact with the environment, the users, and any possible remote application. It uses Linux as operating system, and it includes a Java virtual machine. The processing module is based on a i486DX single board computer (SBC) running at 133 MHz and an interface board based on the PIC-16F877 microcontroller running at 20 MHz. The following elements are connected to the TICA:

- Ethernet port to access the local area network.
- Smartcard (ISO/IEC 7816) compatible reader.
- 4x4 matrix keyboard to introduce commands and personal identification number (PIN).
- 4x20 LCD display to show application messages.
- Beeper for user operations feed-back.
- I/O port to connect sensors (temperature, switches, etc) and actuators (relays).
- Additional ports are available for future or client-specific expansions.

It is worth noting some examples of the functionality of the TICAs. They can open doors, check if the door is opened or closed, check if the box of the device itself is being opened or forced, check the internal temperature, etc. They can also perform additional operations like lighting control, alarm management and presence control.

## 3 Smart Cards

As stated in the introduction of this paper, one of the most important components we have used to design our system are TICAs –presented in the previous section– which use smart cards readers to access the information contained in the smart cards [FPMY98] owned by final users.

Regarding smart cards, we have designed and implemented the system with Java Cards [mic00b,mic00c] and full-RSA (with a 1024 bits RSA on-board implemented algorithm) smart cards, acting as RSA cryptographic devices and as user private information repositories, containing private keys, certificates, personal identification numbers, and authorization information.

Smart cards are, by definition, electronic devices similar to credit cards except they contain a microprocessor chip that can run programs and store data with a high level of security. This protection is mainly based on the physical protection provided by the smart card owner (who normally stores it in his own wallet) and certain security levels managed by the microprocessor.

This security is defined to protect the information itself and to protect the communication with the outside world. For this last case, the RSA (normal and Java Card) microprocessor cards we have used to develop this architecture are

able, using on-board software stored in ROM memory, to run symmetric (shared keys between sender and receiver) cryptographic algorithms such as DES and 3-DES and thus authenticate its communication with any device sharing the same keys, as it can be the case of external SAM (Secure Access Module) devices [FPMY98]. This security level is complementary to the PIN-based security access level normally used to protect the specific information contained in the smart card, as the user private key or the personal identification number.

It is also very important to remark that this kind of RSA smart cards we are using has an on-board key generation system and then the user is able to generate and use his private keys (to sign or decrypt messages), but unable to export these private keys to the outside world in any way. This is quite important, to avoid the creation of any weak point in the whole centralized and distributed physical access control systems we have designed and implemented.

In the case of Java Cards, that is, smart cards that can manage and run Java applications stored in its own memory, the security level remains the same, with symmetric algorithms for external communications, PIN codes to protect the internal data, and on-board private key generation (unable to be exported in any case). The advantage of this kind of smart cards is based on the possibility of using different Java Card applets [mic00a] to manage properly the SPKI credentials associated to different application environments.

## 4 A Centralized Solution

In this section, we are going to present the centralized system which has been used as the starting point to design our decentralized solution. This system is similar to other commercial products existing today, and the point of describing this widely and deployed solution is to present some drawbacks which can be overcome using a different approach. However, it is being successfully used in many scenarios (including our University, where about one hundred TICAs are being used for access control purposes to laboratories, buildings, and car parks), and it addresses the main issues concerning the access control problem.

*Key Distribution* is performed using unique identifiers. Each user has a unique identifier which is stored into his smartcard. The central database contains one table for every installed TICA. These tables are formed by different records which specify the unique identifiers of the authorized users, the set of permissions assigned to the identifier, and some additional data. When a new user is authorized to perform a specific action with a particular TICA, a new record has to be inserted in the related table.

*Key Revocation* is very straightforward. If the authorization manager wants to revoke some permissions previously granted to a specific user for a particular TICA, all he has to do is to delete the records of the identifier involved from the database tables.

*Roles* are not considered. This solution binds permissions to users directly. The introduction of roles might provide the ability of establishing independent relations between users and roles, and between roles and permissions.

The principal guidelines of this type of access control management are shown in Figure 1. During an initial registration phase, the user obtains her smartcard and his unique identifier. He can also request to have access to some specific building or department. Depending on the particular access control policy, the manager might include a record in the database table related with the TICA controlling the requested access. Then, the user inserts her smartcard into the TICA and he requests an specific action (open the door, start working, etc.). Finally, the TICA makes a query to the central database asking whether or not to grant the action.

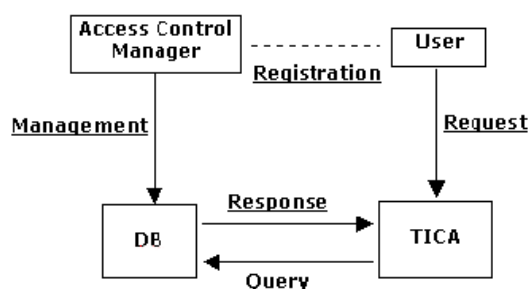


Fig. 1. Centralized access control system

The client application that has been placed in the TICA devices has been fully developed in Java. It runs in the SBC board and communicates with the interface board by way of a serial port. Although the application requires being on-line in order to query the application servers, it is designed to be fault tolerant, including the possibility of maintaining a local copy of the authorization-related information in case of a network failure in order to keep on working. Furthermore, it keeps a transactions cache and can be synchronized with a NTP (Network Time Protocol) server, which is important to obtain time-consistent records. It makes use of the following technologies:

- *JavaComm*. Java technology that allows communications with other hardware components by means of serial and parallel ports. In this way, a Java application can communicate with different devices that are not connected to the LAN.
- *OCF (Open Card Framework)*. Java framework that allows a structured access to smartcards in a device independent way [For01]. Thus, TICA devices are able to use both ISO/IEC 7816 smartcards and JavaCards. Moreover OCF allows the use of different smartcard readers with the only requisite of having an OCF driver.
- *SSL (Secure Sockets Layer)*. Security protocol [AFK96] used in all transactions between the TICAs and the application servers accessing the central database. These communications are client and server authenticated. Each

time a new TICA device is connected to the network, its manager has to generate a private key for the device and to request its corresponding certificate to the certification authority of the existing PKI.

Although the above presented solution has been successfully used, and it solves most of the common problems about access control, it has some drawbacks that can be fixed following a different approach.

## 5 Distributed Management of Physical Access Control

### 5.1 Motivation for a RBAC and Decentralized System

Our centralized system requires permanent connectivity to the application server providing access to the database. When the connection to the database is broken, or the database is down, the whole system continues providing service, but it is based on local copies of the authorization data. Therefore, further modifications of the database will only be seen by the TICAs remaining connected. In this way, the system becomes unstable since some TICAs might deny some operations that are authorized by others, and vice versa. Besides this, there are some environments where it is not possible to have network connectivity. The access control system might be truly distributed without the need for a central point or permanent network connectivity since it is possible to perform access control decisions without any queries to external elements. In this distributed solution, the access control terminals can be off-line, and they perform their tasks in an independent and decentralized way.

If the environment in which the access control system is used grows large enough, the task of managing each TICA, the list of authorized users, or the local copies of the authorization information become too big. A better solution is defining user groups in order to assign permissions to the group name instead of trying to manage each user individually. However, if group membership is defined by the database, all problems derived by the network dependence will be present here again.

Finally, the design of the current centralized system does not provide a basic security service: non-repudiation. The database and the TICAs store log files containing information about the actions that have been requested by the users (both authorized and unauthorized). However, those log files cannot be used as proof for non-repudiation. We can imagine a user being denied a particular access to a specific building or department. The log files do not demonstrate that the access was indeed performed since they are susceptible to be modified or altered by anyone gaining access to the database. Although there are well known solutions, based on encryption, to store confidential data on insecure platforms, what we really need is a piece of data generated by the user, not by the TICAs or the database itself, which could be used as irrefutable evidence, that is, a digitally-signed access control request. This set of digitally-signed requests, together with the authorization decisions, can be used later for auditing or for forensic purposes following an incident.

## 5.2 Architectural Design of the System

**Background.** This decentralized system is based on a RBAC (Role Based Access Control) model. The central concept of RBAC is that permissions are associated with roles, and users are assigned to appropriate roles. This greatly simplifies management of permissions since the two relations are considered completely independent. Roles can be seen as various job functions in an organization, and users can be assigned to one role depending on their responsibilities. As is commented in [SCFY96], *"the particular collection of users and permissions brought together by a role is transitory. The role is more stable because an organization's activities or functions usually change less frequently"*. We find RBAC a valuable choice in order to design a physical access control system. As we have previously noted, centralized systems become complex when the number of users is high. Complexity can be reduced first assigning users to specific roles, and then determining role permissions. Roles provide several advantages beyond indirection. For example, as we will see, a particular user acting as role researcher has a role membership certificate stored in the smart card, and that certificate does not contain any information about role permissions. Modification of permissions related to the role does not imply modification of the certificate stored in the smart card, and therefore management is greatly simplified.

However, RBAC models can be implemented in different ways. For example, it is possible to design a RBAC system where relations among users and roles (and roles and permissions) are defined as database tables, which are stored in a central server. While problems derived from scalability are reduced, some of them still remain related to central-server dependence and network connectivity.

We have designed a system where each TICA is the beginning of the authorization path, and not only the enforcement point. The device is able to make the security decision regarding the authorization data presented by the user requesting the access. Role membership and the assignment of permissions to roles are encoded in digital certificates, which are distributed among the related users and TICAs. Revocation of such documents is managed using two different approaches. First, it is possible to use short lived certificates [Mye98]. In absence of network connectivity, an off-line device cannot retrieve a regularly-updated CRL, or perform an OCSP (Online Certificate Status Protocol) [MAM<sup>+</sup>99] query. However, certificates with a short validity interval can naturally bound the effect of revocation, and they constitute a good alternative for this type of scenarios. On the other hand, TICAs are able to retrieve information from external entities since they have an Ethernet port to access the local area network. If network connectivity is possible, TICAs might periodically check a CRL or perform OCSP queries.

Our solution has been designed using the SPKI/SDSI public key infrastructure. The crucial component to a PKI is the key pair, and in SPKI/SDSI the public and private keys are central. In contrast to classic PKI theory, SPKI/SDSI emphasizes that the public key is the principal. Of course, the notion of an owner of a public key is allowed, but is not necessary. In fact, in order to determine

whether an individual has access to a particular building it might not be mandatory to authenticate the user identity.

SPKI/SDSI defines three forms of credential: ID, Attribute and Authorization. An ID credential binds a name (an identity or a group name) to a public key, an attribute credential binds an authorization to an ID, and an authorization credential binds an authorization to a public key. Some RBAC models can be implemented using these types of credentials. ID credentials are useful to implement role membership (two individuals follow the same role if they have ID certificates sharing the same ID), and to bind an identity to a public key (however, we use them only for group membership purposes). Attribute certificates can be used to specify the permission set assigned to a particular role. Finally, authorization certificates are useful in order to establish a direct relation between a public key and an authorization (names are not used).

**Architectural elements.** In this section we are going to give a brief description about the core entities of the distributed management system. We introduce why they are necessary and how they interoperate in a typical scenario.

- *Users.* A user is a person with a smart card containing a RSA or DSA key pair. The public key might be included in a X.509 identity certificate issued by a particular certification authority, but it is not mandatory (i.e. the smartcard might contain no information about the user's identity). The smartcard must store additional authorization, attribute or ID certificates. Users are able to use some of the available TICA to gain access to a particular building, or in order to perform requests associated with presence control mechanisms. They submit a digitally signed request to the TICA specifying the action being demanded and all certificates related to that action (which are stored into the smartcard).
- *Naming Authorities (NA).* They are responsible for issuing ID certificates for the users. This type of certificates can be used to define group (or role) membership, or simply in order to assign a name to a particular public key. The way a NA is able to determine whether a user is a member of a particular group is application-specific. When a role membership request is granted by a NA, the related certificate is stored in the user smartcard, and it can be published in a public data repository.
- *Authorization Authorities (AA).* Authorization and attribute certificates are issued by AAs. There are two possible ways to generate these type of certificates. On the one hand, users can directly request authorization certificates for a particular set of TICAs. If the request is granted, certificates are issued and stored in their smartcards. On the other hand, some special users (system managers) can also request attribute certificates, that is, authorization certificates where the subject is not a particular public key, but a role name defined by a specific NA. Those attribute certificates are normally stored in the TICAs since they are supposed to be long term certificates (role functions do not usually change very often).

- *TICAs*. Our system is primarily based on delegation chains [Aur98], and TICAs are the beginning of the trust path. TICAs can establish their own access control conditions, trusted entities, and authorization mechanisms. Each TICA issues an authorization certificate for a set of specific AAs. These certificates basically give the AAs total authority over the device, and also the permission to further delegate the access control is granted. TICAs can also delegate the authority by means of ACL (Access Control List) entries containing the same information included in those certificates. The main difference is that ACLs are local information not digitally signed, and cannot be published.

Generation of the certificates involved is performed using a distributed credential management system [CG02]. This system addresses some problems related to scalability, certificate distribution, and interoperability. We define how certification requests can be expressed, how different security policies can be enforced using this system, and which are the entities involved in a certification scenario.

Figure 2 shows how the system entities are related. In this scenario, the TICA issues several authorization certificates to different AAs. Those certificates give the AAs a set of privileges over the device and the permission to delegate them. Then, authorization authorities create new attribute certificates giving a subset of such permissions to the roles defined by any of the existing naming authorities. Normally, those certificates have the delegation flag deactivated (we have designed a  $RBAC_0$  system, but it might be extended in order to support role hierarchies). Roles are managed by NAs. They issue ID certificates in order to state that a particular user has been assigned to a specific role. Finally, users digitally sign access requests, which are presented to the TICAs together with the digital certificates stored in their smartcards. The request contains a time stamp reflecting the moment when it was formulated, and an authorization tag representing the action being demanded. The path displayed by the figure can be verified by the TICA because it originates from the TICA itself (this kind of situation is called an authorization loop).

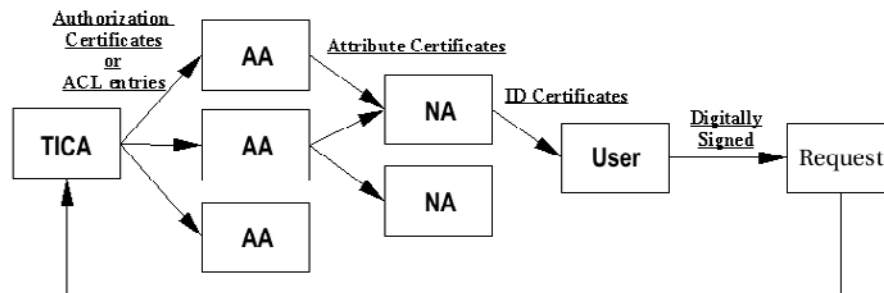


Fig. 2. Delegation path

The certificate chain can be also used as non-repudiation proof. As we mentioned previously, our centralized system did not provide irrefutable evidence about access requests. However, in this distributed approach, the authorization sequence is composed by digitally signed statements. Digitally-signed requests can be used to demonstrate that a particular request was indeed formulated, and results obtained from the certificate chain discovery method can determine whether the request was granted. Obviously, existence of an accepting certification path does not prove a door was opened. The important part is that the TICA behavior should be verifiable by a neutral third party. As we will see, the software controlling the TICA is digitally signed and it has been developed using CDSA (Common Data Security Architecture). CDSA provides services which can be used to ensure component integrity and trusted identification of the component's source.

**Authorization tags.** In SPKI/SDSI, credentials are expressed as tuples. Attribute and authorization certificates are specified as 5-tuples (Issuer, Subject, Propagation, Tag, Validity). *Issuer* identifies the entity that issues the authorization. *Subject* is the entity acquiring the authorization (a public key or a group of principals). *Propagation* is a boolean indicating whether the subject is allowed to further propagate the authorization. *Tag* is a free-style S-expression [RL] specifying the permission being provided. Finally, *Validity* specifies the validity period for this credential. ID credentials are defined by 4-tuples (Issuer, Name, Subject, Validity). *Issuer* identifies the entity defining the ID in its private name space. *Name* is the ID. *Subject* is the public key, or the name that will be bound to the ID. *Validity* is the validity period for this credential.

Because the SPKI certificate specification does not give any description of how the authorization information should be presented, we can choose the representation form freely. Moreover, the fact that the exact definition of the contents of the authority field is left application specific is not a problem since the source of a certificate chain and the final verifier are always the same principal.

S-expressions that we have used as authorization tags have the following format:

```
(tag (d-tica (section/tica (permissions (time-interval))))))
```

- *d-tica*. This identifies the tag as a TICA-related authorization.
- *section*. TICAs can be grouped into sections. For example, a section might be the identifier of a particular building or department. In this way, we can specify a set of TICAs using prefixes.
- *tica*. This is the identifier of a particular TICA, as for instance `library/main-door`.
- *permissions*. Permissions being granted, as for instance `open-door`, `start-working-day`, `end-working-day`, etc
- *time-interval*. Some permissions can be restricted to specific time intervals.



**Example application.** In this section, we are going to show the elements involved in a particular delegation chain. First we have the following authorization certificate issued by *tica1*. This certificate gives full authority to *AA1* over the *tica1* located in the *section1*, and the permission to further delegate it.

```
(tica1, AA1, true, (tag (d-tica (section1/tica1 (*)))), forever)
```

Then, *AA1* has two alternatives. First, it might issue another (short lived) authorization certificate to a specific final user *U1*.

```
(AA1, U1, false, (tag (d-tica (section1/tica1 (open-door)))),  
12/1/01..12/31/01)
```

The second option is issuing an attribute certificate for the *R1* role members defined by *NA1*.

```
(AA1, R1 in NA1, false, (tag (d-tica (section1/tica1 (open-door)))),  
forever)
```

*NA1* issues (short lived) ID certificates for the different members of roles defined by itself. One of those certificates is the following one.

```
(NA1, R1, U1, 12/1/01..12/31/01)
```

When the user *U1* inserts his smartcard into the *tica1*, and he requests the door to be opened, the following signed statement is generated (where time-stamp represents the current instant).

```
(U1, (tag (d-tica (section1/tica1 (open-door (time-stamp))))),  
time-stamp)
```

Now, *tica1* can use some of the available certificate chain discovery methods [Eli98] in order to determine whether a delegation path authorizing such a request exists. In this particular example, the certificates above presented will authorize the action if it is performed during the specified validity ranges.

### 5.3 Some Implementation Details

There are two main applications that have been implemented. First, we have an application that is able to generate SPKI certificates and to store them in smartcards. The other application is the software installed in the TICA devices.

Both implementations are based on CDSA (Common Data Security Architecture) [Cor01]. CDSA is a set of layered security services that provide the infrastructure for scalable, extendible and interoperable security solutions. These security services are performed by add-in security modules. The five basic service categories are: Cryptographic Service Providers (CSPs), Trust Policy Modules (TPs), Certificate Library Modules (CLs), Data Storage Library Modules (DLs), and Authorization Computation Modules (ACs). We used the Intel version 3.14

of that architecture in order to develop a new add-in multi-module which provides CSP and DL services (version 3.14 kindly added a SPKI CL module which supports attribute and ID certificates upon our request).

Our CSP module adds support for smartcards. It performs digital signature operations that make use of the private key stored in the smartcard. In this way, our applications can use the CSSM (Common Security Services Manager) API in order to, transparently, create digital signatures generated by the stored private key. We have also developed a CSP module for smart cards with cryptographic capabilities (digital signature, encryption/decryption). Using this type of smart cards, private keys are protected from external software, which increases system security.

On the other hand, SPKI credentials are inserted and loaded from the smartcard using our own DL module. This module defines the particular database containing the authorization information, and the way data is managed.

## 6 Conclusions

In this paper, we have presented two different physical access control systems making use of special devices (TICAs) and RSA smart cards. As we have explained, those devices and this kind of smartcards can be used to implement access control using different approaches, ranging from secure queries to a central database, to a distributed offline system making use of authorization certificates. We think that our decentralized solution provides some additional mechanisms in relation to the centralized one, such as non-repudiation, better scalability, and suitability for offline environments. To the best of our knowledge, there is no similar RBAC and SPKI-based proposal for physical access control environments.

## References

- [AFK96] A. O. Alan, P. Freier, and P. C. Kocher. *The SSL Protocol Version 3.0*, 1996. Internet Draft.
- [Aur98] T. Aura. On the structure of delegation networks. In *Proc. 11th IEEE Computer Security Foundations Workshop*, pages 14–26, MA USA, June 1998. IEEE Computer Society Press.
- [BFIK99] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. Keromytis. The role of trust management in distributed systems security. In *Secure Internet Programming: Security Issues for Distributed and Mobile Objects*, volume 1603 of *LNCS*, pages 185–210. Springer, 1999.
- [CDFR98] J. Callas, L. Donnerhake, H. Finney, and R. Thayer. *OpenPGP Message Format*, 1998. Request For Comments (RFC) 2440.
- [CG02] O. Canovas and A. F. Gomez. A Distributed Credential Management System for SPKI-Based Delegation Systems. In *Proceedings of 1st Annual PKI Research Workshop*, Gaithersburg MD, USA, April 2002.
- [Cor01] Intel Corporation. *Common Data Security Architecture (CDSA)*. World Wide Web, <http://developer.intel.com/ial/security>, 2001.

- [EFL<sup>+</sup>99] C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen. *SPKI certificate theory*, September 1999. Request For Comments (RFC) 2693.
- [Eli98] J.E. Elien. Certificate discovery using spki/sdsi 2.0 certificates. Master's thesis, Massachusetts Institute of Technology, May 1998.
- [For01] Open Card Forum. *Open Card Framework*. World Wide Web, <http://www.opencard.org>, 2001.
- [FPMY98] J. Ferrari, S. Poh, R. Mackinnon, and L. Yatawara. *Smart Cards: A Case Study*. IBM RedBooks, <http://www.redbooks.ibm.com/pubs/pdfs/redbooks/sg245239.pdf>, October 1998.
- [HFS99] R. Housley, W. Ford, and D. Solo. *Internet Public Key Infrastructure, Part I: X.509 Certificate and CRL Profile*, January 1999. Request for Comments (RFC) 2459.
- [MAM<sup>+</sup>99] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams. *OCSP: Online Certificate Status Protocol*, June 1999. Request For Comments (RFC) 2560.
- [mic00a] Sun microsystems. *Java Card 2.1.1 Application Programming Interface*. World Wide Web, <http://java.sun.com/products/javacard/javacard21.html>, May 2000.
- [mic00b] Sun microsystems. *Java Card 2.1.1 Runtime Environment (JCRE) Specification*. World Wide Web, <http://java.sun.com/products/javacard/javacard21.html>, May 2000.
- [mic00c] Sun microsystems. *Java Card 2.1.1 Virtual Machine Specification*. World Wide Web, <http://java.sun.com/products/javacard/javacard21.html>, May 2000.
- [Mye98] M. Myers. Revocation: Options and challenges. In *Proceedings of Financial Cryptography 98*, volume 1465 of *LNCS*, pages 165–171. Springer-Verlag, 1998.
- [oM01] University of Murcia. *KRONOS Project*. World Wide Web, <http://ants.dif.um.es/kronos>, 2001.
- [RL] R. Rivest and B. Lampson. *SDSI: A simple distributed security infrastructure*.
- [SCFY96] R. Sandhu, E. Coyne, H. Feinstein, and C. Youman. Role-based access control models. *IEEE Computer*, 29(2), February 1996.

# Authenticity and Provability – A Formal Framework<sup>\*</sup>

Sigrid Gürgens, Peter Ochsenschläger, and Carsten Rudolph

Fraunhofer – Institute for Secure Telecooperation SIT  
Rheinstrasse 75, D-64295 Darmstadt, Germany  
{guergens, ochsen Schlaeger, rudolphc}@sit.fraunhofer.de

**Abstract.** This paper presents a new formalisation of authenticity and proof of authenticity. These security properties constitute essential requirements for secure electronic commerce and other types of binding telecooperation. Based on the notions of formal language theory, authenticity and proof of authenticity are defined relative to the agents' knowledge about the system. Abstraction by language homomorphisms satisfying particular properties preserves the respective security properties from a higher to a lower level of abstraction. Thus, the new formalisation is suitable for a top-down security engineering method. The approach is demonstrated by a typical e-commerce example, a price-offer transaction. We present specifications of this example on two different abstraction levels. On the lower of these abstraction levels, Asynchronous Product Automata (APA) are used to model cryptographic protocols, and properties of cryptographic algorithms are formally described by abstract secure channels.

## 1 Introduction

Authentication and non-repudiation are essential security requirements for electronic commerce applications and other types of binding telecooperation. Symmetric and asymmetric encryption techniques and different types of digital signatures can be used to provide these security services. However, cryptographic algorithms can only provide isolated functionality that has to be integrated into more or less complex cryptographic protocols. Numerous examples exist where flaws in cryptographic protocols have been very hard to detect. Consequently, the need for formal verification of cryptographic protocols is widely accepted and various different techniques including logics of authentication and formal model-checking have been proposed and successfully used to find security flaws.

Most of the work in the field of cryptographic protocols has concentrated on the analysis of authentication and key establishment protocols. Therefore, formalisation of security properties is usually based on special representations of protocols on a fixed level of abstraction and restricted to authentication, freshness and confidentiality of keys. Our work is related to a range of work concerning

---

<sup>\*</sup> Part of this work is based on results of a project funded by Siemens AG.

general specification of security properties. Meadows and Syverson [13] define a formal language for the specification of requirements on cryptographic protocols. Authenticity is defined as the property that a message is only accepted if at some point in the past it was sent by a specific agent and before that requested from the recipient. Schneider [12] uses a more general approach to define authenticity of events or sets of events on globally viewed traces. Both approaches do not cover provability or non-repudiation of authenticated events towards other agents. In the context of analysing security protocols with respect to accountability, Kailar [7] introduces the notion of strong and weak proofs that can be transferable or non-transferable. He then uses inference rules to reason about accountability. Another framework to reason about properties of electronic commerce protocols was proposed by Clarke, Jha and Marrero [3]. However, both approaches do not provide the formalization of a proof itself. All these papers are exclusively concerned with the analysis of security protocols. To the best of our knowledge, there is no approach to a formalisation of security properties of electronic commerce protocols that supports a design methodology.

Our approach for the systematic design of a binding telecooperation is to specify the application process and the security requirements on a high level of abstraction independently of any implementations such as cryptographic protocols, and then to verify that a cryptographic protocol implementing the application satisfies the required security properties, provided that the underlying cryptographic algorithms are secure. This approach, formulated in terms of formal language theory, is based on a model for key establishment protocols [11] and is related to a formal framework for binding cooperations [5], where those states of a binding cooperation are determined in which proofs of authenticity are necessary to reach the goal of the cooperation.

In this paper, we introduce formal specifications of the security properties authentication and proof of authentication. The satisfaction of these properties is relative to the agents' view of what has happened in the system and which global system behaviours they consider possible according to their knowledge of the system. The definitions can be used on different levels of abstraction. Suitable language homomorphisms map from lower to higher levels of abstraction. We present sufficient conditions on language homomorphism to preserve authenticity and proof of authenticity from a higher to a lower level of abstraction. This means that if the properties are satisfied by an abstract specification one can conclude that respective properties are satisfied by the refined specification.

To illustrate our approach, we give the abstract specification of a simple e-commerce transaction and show that this specification provides the required authenticity and proof of authenticity under reasonable system assumptions. We then present a concrete cryptographic protocol on a lower abstraction level using Asynchronous Product Automata (APA) [10]. The system assumptions of the abstract specification correspond to properties of cryptographic algorithms modelled as abstract secure channels in the APA specification of the cryptographic protocol. We finally define a homomorphism that maps the protocol specification to the abstract specification and show that it satisfies the neces-

sary properties, thus proving that the protocol itself provides authenticity and proof of authenticity.

## 2 Formal Specification of Security Properties

### 2.1 System Behaviour and Abstraction by Language Homomorphisms

We start this section with a short summary of the necessary concepts of formal languages. The behaviour  $S$  of a discrete system can be formally described by the set of its possible sequences of actions. Therefore  $S \subseteq \Sigma^*$  holds where  $\Sigma$  is the set of all actions of the system and  $\Sigma^*$  the set of all finite sequences of elements of  $\Sigma$ , including the empty sequence denoted by  $\varepsilon$ . This terminology originates from the theory of formal languages [4], where  $\Sigma$  is called the alphabet, the elements of  $\Sigma$  are called letters, the elements of  $\Sigma^*$  are referred to as words and the subsets of  $\Sigma^*$  as formal languages. Words can be composed: if  $u$  and  $v$  are words, then  $uv$  is also a word. This operation is called the *concatenation*; especially  $\varepsilon u = u\varepsilon = u$ . A word  $u$  is called a *prefix* of a word  $v$  if there is a word  $x$  such that  $v = ux$ . The set of all prefixes of a word  $u$  is denoted by  $\text{pre}(u)$ ;  $\varepsilon \in \text{pre}(u)$  holds for every word  $u$ . We denote the set of letters in a word  $u$  by  $\text{alph}(u)$ .

Formal languages which describe system behaviour have the characteristic that  $\text{pre}(u) \subseteq S$  holds for every word  $u \in S$ . Such languages are called *prefix closed*. System behaviour is thus described by prefix closed formal languages.

The set of all possible continuations of a word  $u \in S$  is formally expressed by the *left quotient*  $u^{-1}(S) = \{y \in \Sigma^* \mid uy \in S\}$ .

Different formal models of the same application/system are partially ordered with respect to different levels of abstraction. Formally, abstractions are described by so called alphabetic language homomorphisms. These are mappings  $h^* : \Sigma^* \rightarrow \Sigma'^*$  with  $h^*(xy) = h^*(x)h^*(y)$ ,  $h^*(\varepsilon) = \varepsilon$  and  $h^*(\Sigma) \subseteq \Sigma' \cup \{\varepsilon\}$ . So they are uniquely defined by corresponding mappings  $h : \Sigma \rightarrow \Sigma' \cup \{\varepsilon\}$ . In the following we denote both the mapping  $h$  and the homomorphism  $h^*$  by  $h$ . These homomorphisms map action sequences of a finer abstraction level to action sequences of a more abstract level.

Consider for example an application consisting of four actions: a price for a certain service is requested, the request is received, and then an offer for this service is sent and received. On a high level of abstraction this may be formalized by only specifying the names of the actions (e.g. PRICEREQ-S, PRICEREQ-R, OFFER-S, OFFER-R) and system assumptions such as an offer can only be received if at some point before it was sent. On a lower level of abstraction more information about the system may be specified: The agents' actions may be split into internal and external ones, their internal memory may be specified, etc. A homomorphism can be defined that maps the internal actions of the agents onto  $\varepsilon$ , all send actions of price requests onto PRICEREQ-S, all price request receive actions onto PRICEREQ-R, etc. This homomorphism serves as an abstraction of the finer system specification.

## 2.2 Malicious Behaviour

For the security analysis of protocols malicious behaviour is often explicitly specified and included in the system behaviour. However, in general malicious behaviour is not previously known and one may not be able to adequately specify all possible actions of dishonest agents. Therefore, in this paper we use a different approach. Malicious behaviour is not explicitly specified but restricted by system assumptions and by assumptions about the underlying security mechanisms (for example the application of cryptography).

Let  $\Sigma$  be an arbitrary set of actions (that may contain malicious actions) and  $S_C \subseteq \Sigma^*$  a correct system behaviour without malicious actions. A behaviour containing malicious actions is denoted by  $S$ . We assume that  $S_C \subseteq S \subseteq \Sigma^*$ . Let  $\mathbb{P}$  be a set of agents. For each  $P \in \mathbb{P}$  we denote by  $W_P \subseteq \Sigma^*$  the set of those sequences agent  $P$  considers to be possible in  $S$ .  $W_P$  formalizes  $P$ 's knowledge about a system  $S_C$ . The set  $\Sigma^*$  as well as the sets  $W_P$  may contain malicious behaviour. Both sets are not completely specified and are in general infinite, but all sets  $W_P$  are restricted by the following assumptions:

- $W_P$  satisfies the properties of underlying security mechanisms (see Section 4.3 for formal definitions of security mechanisms of a system  $S$ ).
- System assumptions cannot be violated in  $W_P$ . (A system assumption is, for example, that access to an agent's internal memory is protected).

We assume  $S_C \subseteq W_P$ , i.e. every agent considers the correct system behaviour to be possible. Security properties can now be defined relative to  $W_P$ . A system  $S$  may satisfy a security property with respect to  $W_P$ , but may fail to satisfy the same property with respect to a different  $W_P$ , if  $P$  considers other actions to be possible on account of weaker system assumptions and security mechanisms.

While  $W_P$  formalizes what agent  $P$  considers possible in general, the *local view* of  $P$  determines what  $P$  considers possible after a sequence of actions  $\omega \in S$  has happened. In general, agents may not be able to monitor the complete system, thus their local view of  $\omega$  will differ from  $\omega$ . We denote the local view of agent  $P$  on  $\Sigma^*$  by  $\lambda_{P,\Sigma} : \Sigma^* \rightarrow \Sigma_P^*$ .

In order to determine what is the local view of an agent of the system  $S$ , we first observe that the actions  $\Sigma$  of the system can be separated into those that are relevant for a security property and those that are not, the latter set of actions being denoted by  $\Sigma_{/\#}$ . In the price request/offer example introduced in Section 2.1, we may want the offer to provide certain security properties, while the sending and receiving of a price-request are not relevant for any security property and therefore belong to  $\Sigma_{/\#}$ . All actions relevant for a security property (sending and receiving of the offer) are performed by and can be assigned to exactly one agent. Thus  $\Sigma = \bigcup_{P \in \mathbb{P} \cup \{\#\}} \Sigma_{/P}$  (where  $\Sigma_{/P}$  denotes all actions performed by agent  $P$ ). The homomorphism  $\pi_{P,\Sigma} : \Sigma^* \rightarrow \Sigma_{/P}^*$  defined by  $\pi_{P,\Sigma}(x) = x$  if  $x \in \Sigma_{/P}$  and  $\pi_{P,\Sigma}(x) = \varepsilon$  if  $x \in \Sigma \setminus \Sigma_{/P}$  formalizes the assignment of actions to agents and is called the *projection* on  $P$ .

Different levels of abstraction induce different definitions of the respective local views of agents. On a high level of abstraction where the behaviour of a

system  $S' \subseteq (\Sigma')^*$  is described by sequences of actions containing essentially the action name and the agent performing the action (if any), all information about an action  $x \in \Sigma'_{/P}$  is available for agent  $P$ . Thus  $P$ 's local view of the system  $S'$  is defined by  $\lambda_{P,\Sigma'} = \pi_{P,\Sigma'}$ . But in a system  $S$  on a lower level of abstraction additional information such as all agents' memory may be available, the elements of  $\Sigma$  may contain information about the global system state (e.g. all agents' memory). However, an agent  $P$  generally cannot “see” the complete global state (he cannot see, for example, other agents' memory). Therefore, the projection  $\pi_{P,\Sigma}$  may not be adequate to define the local view of an agent  $P \in \mathbb{P}$ . Thus, on a lower abstraction level we may need a different definition for the local view of the agents. See Section 5.2 for the definition of the local view on a lower abstraction level. On each abstraction level, the local views of all agents together with the behaviour not assigned to any agent contain all information about the respective system behaviour.

For a sequence of actions  $\omega \in S$  and agent  $P \in \mathbb{P}$ ,  $\lambda_{P,\Sigma}^{-1}(\lambda_{P,\Sigma}(\omega)) \subseteq \Sigma^*$  is the set of all sequences that look exactly the same from  $P$ 's local view after  $\omega$  has happened. But depending on his knowledge about the system  $S$  and underlying security mechanisms and system assumptions,  $P$  does not consider all sequences in this set possible. Thus he can use his knowledge to reduce this set:  $\lambda_{P,\Sigma}^{-1}(\lambda_{P,\Sigma}(\omega)) \cap W_P$  describes all sequences of actions  $P$  considers to be possible when  $\omega$  has happened.

In this paper we use systems  $S \subseteq \Sigma^*$  and  $S' \subseteq (\Sigma')^*$  on two specific abstraction levels to illustrate our approach. We will refer to these levels as to “the high level” and “the low level” of abstraction. Accordingly, we use  $W_P$  and  $\lambda_P$  whenever we are referring to a general system  $S$  and  $W'_P$  and  $\lambda'_P$  when we want to differentiate between agents' knowledge about the respective systems and their local views on the systems on the two different abstraction levels.

### 2.3 Authenticity and Proof of Authenticity

In this section we formally define the security properties authenticity and proof of authenticity and then give sufficient conditions on language homomorphism to preserve these properties from higher to lower levels of abstraction.

A set of actions  $\Gamma$  is authentic for agent  $P$  if in all sequences that  $P$  considers possible after a sequence of actions has happened, some time in the past an action in  $\Gamma$  must have happened. One (or more) of the actions of  $\omega$  performed by  $P$  is responsible for  $\Gamma$  being authentic for  $P$ . This can be, for example, that  $P$  receives an offer. If then all sequences of actions he considers possible contain a send offer action performed by  $Q$ , then the set of actions where  $Q$  sends an offer is authentic for  $P$ . Our notion of authenticity defers from others in that we define authenticity relative to what an agent considers possible, while usually authenticity is defined taking a global view of the system (see e.g. [12]).

In the following let  $S \subseteq \Sigma^*$  be a prefix closed language describing the behaviour of a system and let  $\mathbb{P}$  be a set of agents.



**Definition 1 (Authenticity)** A set of actions  $\Gamma \subseteq \Sigma$  is authentic for  $P \in \mathbb{P}$  after a sequence of actions  $\omega \in S$  with respect to  $W_P$  if  $\text{alph}(x) \cap \Gamma \neq \emptyset$  for all  $x \in \lambda_P^{-1}(\lambda_P(\omega)) \cap W_P$ .

Some actions do not only require authenticity but also need to provide a proof of authenticity. If agent  $P$  owns a proof of authenticity for a set  $\Gamma$  of actions, he can send this proof to other agents, which in turn can receive the proof and then own it. In the following definition the set  $\Gamma P$  denotes actions that provide agents with proofs about the authenticity of  $\Gamma$ . If agent  $P$  has executed an action from  $\Gamma P$  then  $\Gamma$  is authentic for  $P$  and  $P$  can forward the proof to any other agent using actions in  $\Gamma S$ .

**Definition 2 (Proof of authenticity)** A pair  $(\Gamma S, \Gamma P)$  with  $\Gamma S \subseteq \Sigma$  and  $\Gamma P \subseteq \Sigma$  is a pair of sets of proof actions of authenticity for a set  $\Gamma \subseteq \Sigma$  with respect to  $(W_P)_{P \in \mathbb{P}}$  if for all  $\omega \in S$  and for all  $P \in \mathbb{P}$  with  $\text{alph}(\pi_P(\omega)) \cap \Gamma P \neq \emptyset$  the following holds:

1. For  $P$  the set  $\Gamma$  is authentic after  $\omega$  and
2. for each  $R \in \mathbb{P}$  there exists actions  $a \in \Sigma_{/P} \cap \Gamma S$  and  $b \in \Sigma_{/R} \cap \Gamma P$  with  $\omega ab \in S$ .

Agent  $P \in \mathbb{P}$  can give proof of authenticity of  $\Gamma \subseteq \Sigma$  after a sequence of actions  $\omega \in S$  if 1 and 2 hold.

In the following, we shortly call  $(\Gamma S, \Gamma P)$  a *proof action pair* for  $\Gamma$ .

This definition represents one specific type of proofs. Kailar has classified proofs as strong or weak and transferable or non-transferable [7]. In terms of this classification our definition provides strong transferable proofs with the additional property that the possibility of the proof transfer is reliable. These proofs require the assumption that no agent disposes of his proofs. From a technical point of view this is the most simple formal definition of this property. However, other types of proofs can be formalized in a similar way. For example, by introducing an additional class of actions representing the loss of proofs, Definition 2 can be modified: Agent  $P$  can give proof of authenticity only as long as no corresponding loss action has occurred. Corresponding modifications of Definition 2, Definition 4, and Theorem 2 are given in [6].

As already explained, we describe an application on different levels of abstraction. On a lower level we may describe the system behaviour by using tuples (global state, action, global successor state) that contain information about agents' memory etc. These tuples can easily be mapped to the respective actions of the higher abstraction level by use of an appropriate homomorphism. However, properties that hold on the high abstraction level need not necessarily hold on the low level and vice versa. The following two definitions give sufficient conditions of homomorphisms such that these "transport" certain security properties from a high to a low abstraction level.

**Definition 3** Let  $h : \Sigma^* \rightarrow \Sigma'^*$  be an alphabetic language homomorphism and for  $P \in \mathbb{P}$  let  $\lambda_P : \Sigma^* \rightarrow \Sigma_P^*$  and  $\lambda'_P : \Sigma'^* \rightarrow \Sigma_P'^*$  be the homomorphisms

describing the local views of  $P$  on  $\Sigma$  and  $\Sigma'$ , respectively. The language homomorphism  $h$  preserves authenticity on  $S$  if for each  $P \in \mathbb{P}$  exists a mapping  $h'_P : \lambda_P(S) \rightarrow \lambda'_P(S')$  with  $\lambda'_P \circ h = h'_P \circ \lambda_P$  on  $S$ .

$f \circ g$  denotes the composition of functions  $f$  and  $g$ . The above property captures the fact that the homomorphism  $h$  has to be consistent with the local views of  $P$  on both abstraction levels in order to preserve authenticity. This means that the actions relevant for  $\Gamma'$  being authentic for  $P$  and their inverse images under  $h$  must be equally visible by  $P$ . In particular, the inverse image of the action being responsible for the authenticity of  $\Gamma'$  must not be mapped to  $\varepsilon$  by  $P$ 's local view  $\lambda_P$  on the lower abstraction level.

**Theorem 1** *If  $\Gamma' \subseteq \Sigma'$  is authentic for  $P \in \mathbb{P}$  after  $\omega' \in h(S)$  with respect to  $W'_P$ , and if  $h$  preserves authenticity on  $S$ , then  $\Gamma = h^{-1}(\Gamma') \cap \Sigma$  is authentic for  $P \in \mathbb{P}$  after each  $\omega \in h^{-1}(\omega') \cap S$  with respect to each  $W_P$  with  $W_P \subseteq h^{-1}(W'_P)$ .*

**Proof:** Let  $x \in \lambda_P^{-1}(\lambda_P(\omega)) \cap W_P$ . Then  $\lambda_P(x) = \lambda_P(\omega)$  and  $x \in W_P$ . Hence  $h'_P(\lambda_P(x)) = h'_P(\lambda_P(\omega))$ , which implies  $\lambda'_P(h(x)) = \lambda'_P(h(\omega))$  by Definition 3. Hence  $h(x) \in \lambda'^{-1}_P(\lambda'_P(h(\omega)))$ . Since by assumption  $W_P \subseteq h^{-1}(W'_P)$  and  $x \in W_P$ ,  $h(x) \in W'_P$  and therefore  $h(x) \in \lambda'^{-1}_P(\lambda'_P(h(\omega))) \cap W'_P$ . Now, by the definition of authenticity of  $\Gamma'$ ,  $\text{alph}(h(x)) \cap \Gamma' \neq \emptyset$ , which implies  $\text{alph}(x) \cap \Gamma \neq \emptyset$ .  $\square$

We now define properties for a homomorphism in order to preserve proofs. The first condition essentially states that whenever the homomorphic image of a sequence of actions  $\omega \in S$  can in the abstract system be continued with a proof send action in  $\Gamma S'$  and a proof receive action in  $\Gamma P'$ , there must be appropriate proof send and receive actions in the inverse image of  $\Gamma S'$  and  $\Gamma P'$ , respectively, to continue the sequence of actions  $\omega$  in  $S$ .

The second condition states that  $h$  must not map actions performed by and assigned to agent  $P$  on the low abstraction level onto actions assigned to a different agent on the high abstraction level.

Note that  $((\Gamma S' \cap \Sigma'_{/P})(\Gamma P' \cap \Sigma'_{/R}))$  is the set of all words of length 2 with the first letter in  $(\Gamma S' \cap \Sigma'_{/P})$  and the second letter in  $(\Gamma P' \cap \Sigma'_{/R})$ .

**Definition 4** *Let  $\Gamma S' \subseteq \Sigma'$  and  $\Gamma P' \subseteq \Sigma'$ .  $h : \Sigma^* \rightarrow \Sigma'^*$  preserves  $(\Gamma S', \Gamma P')$ -proofs on  $S$  if  $h$  preserves authenticity on  $S$  and if the following holds:*

1.  $h(\omega)^{-1}(h(S)) \cap ((\Gamma S' \cap \Sigma'_{/P})(\Gamma P' \cap \Sigma'_{/R})) \neq \emptyset$  implies  $\omega^{-1}(S) \cap ((h^{-1}(\Gamma S') \cap \Sigma_{/P})(h^{-1}(\Gamma P' \setminus \Sigma'_{/\#}) \cap \Sigma_{/R})) \neq \emptyset$  for each  $P, R \in \mathbb{P}$  and  $\omega \in S$ .
2.  $h(\Sigma_{/P}) \cap (\Sigma' \setminus \Sigma'_{/\#}) \subseteq \Sigma'_{/P}$  for each  $P \in \mathbb{P}$  and  $h(\Sigma_{/\#}) \subseteq \Sigma'_{/\#}$ .

**Theorem 2** *Let  $(\Gamma S', \Gamma P')$  be a proof action pair for  $\Gamma' \subseteq \Sigma'$ . Let  $h : \Sigma^* \rightarrow (\Sigma')^*$  be a homomorphism that preserves  $(\Gamma S', \Gamma P')$ -proofs on  $S$ , and let  $\Gamma S = h^{-1}(\Gamma S') \cap \Sigma$ ,  $\Gamma P = h^{-1}(\Gamma P' \setminus \Sigma'_{/\#}) \cap \Sigma$  and  $\Gamma = h^{-1}(\Gamma') \cap \Sigma$ . Then:*

1.  $(\Gamma S, \Gamma P)$  is a proof action pair for  $\Gamma \subseteq \Sigma$ .

2. If agent  $P \in \mathbb{P}$  can give proof of authenticity of  $\Gamma'$  after  $\omega' \in h(S)$  then  $P$  can give proof of authenticity of  $\Gamma$  after each  $\omega \in h^{-1}(\omega') \cap S$ .

**Proof:** If  $\omega \in S$  and  $\text{alph}(\pi_P(\omega)) \cap \Gamma P \neq \emptyset$  then  $h(\omega) \in h(S)$  and  $\text{alph}(\pi'_P(h(\omega))) \cap \Gamma P' \neq \emptyset$  by the second condition of Definition 4. Now authenticity of  $\Gamma$  for  $P$  after  $\omega$  follows from authenticity of  $\Gamma'$  for  $P$  after  $h(\omega)$  by Theorem 1. By assumption, for each  $R \in \mathbb{P}$  exist actions  $a' \in \Sigma'_{/P} \cap \Gamma S'$  and  $b' \in \Sigma'_{/R} \cap \Gamma P'$  with  $h(\omega)a'b' \in h(S)$ . This implies  $h(\omega)^{-1}(h(S)) \cap ((\Gamma S' \cap \Sigma'_{/P})(\Gamma P' \cap \Sigma'_{/R})) \neq \emptyset$ . Hence by Definition 4 there exist actions  $a \in \Gamma S \cap \Sigma_{/P}$  and  $b \in \Gamma P \cap \Sigma_{/R}$  with  $\omega ab \in S$ . This completes the proof of 1. Proposition 2 is shown similarly.  $\square$

### 3 Abstract Specification of a Price Offer

In this section the security properties defined above are demonstrated in the first step of a typical e-commerce situation, namely the price offer/request protocol already introduced in Section 2.1. As sending and receiving of the price request may not be relevant for the security of the transaction, we only consider the situation in which service providers make offers to clients. We assume that every agent  $P \in \mathbb{P}$  can act as client and receive offers, while service providers are in the set  $\mathbb{SP} \subseteq \mathbb{P}$ . For simplicity we do not explicitly specify prices and services. We write PRO to denote the price offer example. Since we specify the system on a high abstraction level, alphabet, system, etc., will be denoted by  $\Sigma'$ ,  $S'_{PRO}$ , etc., in contrast to alphabet  $\Sigma$ , system  $S_{PRO}$  etc. on the low abstraction level.

To formalize the security property that each client can prove the authenticity of an offer received on a most abstract level, we have to consider the following actions:

- (1) service provider SP makes an offer: OFFER- $S_{SP}$ ,
- (2) client P receives a proof of (1): PROOF- $R_P(SP)$ ,
- (3) client P sends a proof of (1): PROOF- $S_P(SP)$ .

On this abstract level we make no distinction between receiving an offer from SP and receiving a proof that SP made an offer. These three types of actions exactly fit to the sets  $\Gamma$ ,  $\Gamma P$ ,  $\Gamma S$  defined in the previous chapter.

So the action set of our abstract system is given by

$$\Sigma' = \bigcup_{SP \in \mathbb{SP}, P \in \mathbb{P}} \{\text{OFFER-}S_{SP}, \text{PROOF-}R_P(SP), \text{PROOF-}S_P(SP)\}.$$

Now, in our framework we have to consider each agent's knowledge  $W'_P$  about the possible sequences of actions. To formalize proof of authenticity of offers, each agent has to know that a proof can only be received if a corresponding offer has been sent before, and that a proof can only be sent if it has been received before.

So for  $P \in \mathbb{P}$ , let

$$W'_P = \Sigma'^* \setminus \left( \bigcup_{SP \in \mathbb{SP}, Q \in \mathbb{P}} (\Sigma' \setminus \{\text{OFFER-S}_{\mathbb{SP}}\})^* \{\text{PROOF-R}_Q(SP)\} \Sigma'^* \cup (\Sigma' \setminus \{\text{PROOF-R}_Q\})^* \{\text{PROOF-S}_Q(SP)\} \Sigma'^* \right)$$

By this definition all  $W'_P$  are equal and we consider the abstract system behaviour  $S'_{PRO} = W'_P$  for each  $P \in \mathbb{P}$ .

As mentioned in Section 2.2, at this level of abstraction we can define an agent's local view by  $\lambda'_P = \pi'_P$  and by assigning the actions to agents in the following way:

For  $P \in \mathbb{P}$ , let

$$\Sigma'_{/P} = \begin{cases} \bigcup_{SP \in \mathbb{S}} \{\text{OFFER-S}_P, \text{PROOF-R}_P(SP), \text{PROOF-S}_P(SP)\} & \text{if } P \in \mathbb{SP} \\ \bigcup_{SP \in \mathbb{S}} \{\text{PROOF-R}_P(SP), \text{PROOF-S}_P(SP)\} & \text{if } P \in \mathbb{P} \setminus \mathbb{SP} \end{cases}$$

We will now show that in the system  $S'_{PRO}$  an offer is authentic for a client  $P$  whenever he receives a corresponding proof (condition 1 of Definition 2) and that, whenever  $P$  received a proof, he is able to forward this proof, i.e. to give proof of authenticity of the offer (condition 2 of Definition 2).

**Property 1** *For each  $SP \in \mathbb{SP}$ ,  $(\{\text{PROOF-S}_P(SP) | P \in \mathbb{P}\}, \{\text{PROOF-R}_P(SP) | P \in \mathbb{P}\})$  is a proof action pair of authenticity for  $\{\text{OFFER-S}_{\mathbb{SP}}\}$  with respect to  $(W'_P)_{P \in \mathbb{P}}$ .*

**Proof:**

Condition 1: Let  $\omega \in S'_{PRO}$  with  $\text{alph}(\pi'_P(\omega)) \cap \{\text{PROOF-R}_P(SP) | P \in \mathbb{P}\} \neq \emptyset$ . Hence we have  $\text{PROOF-R}_P(SP) \in \text{alph}(\omega)$  which implies  $\text{PROOF-R}_P(SP) \in \text{alph}(x)$  for each  $x \in \lambda'^{-1}_P(\lambda'_P(\omega))$ . By the definition of  $W'_P$ , it follows that  $\text{OFFER-S}_P(SP) \in \text{alph}(x)$  for each  $x \in \lambda'^{-1}_P(\lambda'_P(\omega)) \cap W'_P$ .

Condition 2: In particular, for  $x = \omega$ , we have  $\text{PROOF-R}_P(SP) \in \text{alph}(\omega)$  and  $\text{OFFER-S}_P(SP) \in \text{alph}(\omega)$ . Hence  $\omega \text{PROOF-S}_P(SP) \text{PROOF-R}_R(SP) \in S'_{PRO}$  for each  $R \in \mathbb{P}$  because none of these action sequences violates the restrictions defining  $S'_{PRO}$ .

## 4 Asynchronous Product Automata

On the lower abstraction level, we model a system of protocol agents using Asynchronous Product Automata (APA). APA are a universal and very flexible operational description concept for cooperating systems [10]. It “naturally” emerges from formal language theory [9]. APA are supported by the SH-verification tool that provides components for the complete cycle from formal specification to exhaustive verification [10].

An APA can be seen as a family of elementary automata. The set of all possible states of the whole APA is structured as a product set; each state is divided into state components. In the following the set of all possible states is

called state set. The state sets of elementary automata consist of components of the state set of the APA. Different elementary automata are “glued” by shared components of their state sets. Elementary automata can “communicate” by changing shared state components.

Figure 1 (see Section 4.2) shows a graphical representation of the APA for a system of two protocol agents A and B. The figure shows the structure of the automaton. The circles represent state components and a box corresponds to one elementary automaton. The full specification of the APA includes the transition relations of the elementary automata and the initial state. The neighbourhood relation  $N$  (indicated by arcs) determines which state components are included in the state of an elementary automaton and may be accessed and changed by one of its state transitions. For example, a state transition of automaton  $\text{Send}_A$  may change the content of  $\text{Network}$  and  $\text{State}_A$ , while  $\text{Internal}_A$  may change  $\text{State}_A$  but cannot access  $\text{Network}$ .

#### 4.1 The Formal Definition

An *Asynchronous Product Automaton* consists of a family of *State Sets*  $Z_S, S \in \mathbb{S}$ , a family of *Elementary Automata*  $(\Phi_e, \Delta_e), e \in \mathbb{E}$ , and a *Neighbourhood Relation*  $N : \mathbb{E} \rightarrow \mathcal{P}(\mathbb{S})$ ;  $\mathcal{P}(X)$  is the power set of  $X$  and  $\mathbb{S}$  and  $\mathbb{E}$  are index sets with the names of state components and elementary automata, respectively. For each Elementary Automaton  $(\Phi_e, \Delta_e)$ ,  $\Phi_e$  is its *Alphabet* and  $\Delta_e \subseteq \times_{S \in N(e)}(Z_S) \times \Phi_e \times \times_{S \in N(e)}(Z_S)$  is its *State Transition Relation*.

An APA’s (global) *States* are elements of  $\times_{S \in \mathbb{S}}(Z_S)$ . Each APA has one *Initial State*  $s_0$ .

The behaviour of an APA is represented by all possible sequences of state transitions starting with initial state  $s_0$ . In the following, in a sequence of state transitions  $\omega$ , the successor state of state  $s$  is denoted by  $\bar{s}$ , state  $s_i$  occurring before state  $s_j$  is denoted by  $s_i < s_j$ . A state transition sequence  $\omega$  ending in state  $s$  is denoted by  $\omega_s$ .

#### 4.2 APA Model for Protocols

Each protocol participant  $P$  is modelled by four elementary automata  $\text{Send}_P, \text{Rec}_P, \text{App}_P$  and  $\text{Internal}_P$  and four state components  $\text{Channel}_P, \text{Internal}_P, \text{State}_P$  and  $\text{ApplMem}_P$ . There is a further state component  $\text{Network}$  that is used for the communication between the agents. Figure 1 shows a graphical representation of the APA for a system of two protocol agents A and B.

No elementary automaton of an agent  $P$  has access to a different agent’s state components. The only state component shared between all agents is the component  $\text{Network}$ , which is used for communication. The general idea of how to use such an APA for modelling a protocol is the following: The automaton  $\text{App}_P$  serves as the interface between protocol and application. It specifies how applications request security services and access the result. The automata  $\text{Send}_P$  and  $\text{Rec}_P$  perform the communication between different agents. Messages are marked with a tag indicating the particular abstract channel they are sent on

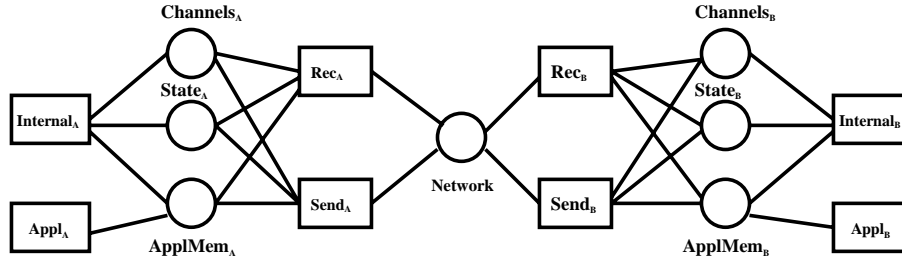


Fig. 1. APA model with two protocol agents A and B

(see below).  $\text{Internal}_P$  is used for  $P$ 's internal actions. The state component  $\text{State}_P$  serves as the agent's internal memory which can be used to perform checks during the run of the protocol and  $\text{Channel}_P$  holds the channels  $P$  is able to send on and to receive from, respectively. For the sake of brevity, we omit the formal definition of the state sets and refer the reader to [6].

In addition to the graphical representation specifying elementary automata, state components and neighbourhood relationship, for the specification of an APA the following parts have to be specified.

*State sets.* For each state component  $C \in \mathbb{S}$  its state set  $Z_C$  has to be defined. In the protocol model, the state of a state component is a multiset. Therefore, state sets are sets of multisets. For each state component  $C \in \mathbb{S}$  a set  $M_C$  has to be specified such that the state set  $Z_C$  is defined as the set of all multisets of  $M_C$ .

*State transition relation.* For the definition of the state transition relation we use so-called *state transition patterns* (because of space restrictions we omit the formal definition). In Section 5 we show how they are employed.

### 4.3 Abstract Secure Channels

Security requirements on e-commerce protocols are usually realised using cryptographic algorithms. In contrast to computational models for cryptography where properties of these algorithms are usually expressed in terms of computational complexity and probabilities, abstract formal models for cryptographic protocols require abstract representations of cryptographic algorithms.

In the APA protocol model presented in this paper, we model requirements on underlying cryptographic algorithms in terms of abstract secure channels. The idea of modelling communication with abstract channels was first introduced in [2,8] and used to determine which combination of secure channels is required to realise a specific security goal in a cryptographic protocol and to compare various approaches to establish secure channels in open networks. In [1] they are used in a framework for design of secure protocols.

The use of abstract channels allows to distinguish between different properties of cryptographic algorithms while at the same time abstracting from implicit assumptions on keys and on possible implementation details. This in turn reduces the complexity of protocol specification and relocates implementation issues to a lower level of abstraction.

Our model includes abstract channels with various different properties, such as providing authenticity and proof of authenticity, indistinguishability of data, time related properties etc. However, in this paper we restrict ourselves to introducing abstract channels for authenticity and proof of authenticity and a broadcast channel (to model unprotected communication). For the formal definitions of other abstract secure channels see [11].

In the following,  $S$  denotes the prefix closed language representing a behaviour of a system structured as described in Section 4.2. We first define send and receive events on channels. The data structure of state component *Network* defines that each message is tagged with the name of the channel it is sent on. So every element of the state of *Network* has the form  $(message, channelname)$ .

**Definition 5**  $(s_i, e_i, \bar{s}_i)$  is a send event with message  $m_i$  on channel *ChannelName* iff exists  $P \in \mathbb{P}$  with  $e_i = Send_P$  and  $(m_i, ChannelName)$  being added to  $Network(s_i)$ .

**Definition 6**  $(s_i, e_i, \bar{s}_i)$  is a receive event with message  $m_i$  on channel *ChannelName* iff exists  $P \in \mathbb{P} : e_i = Rec_P$  and  $(m_i, ChannelName)$  being read or removed from  $Network(s_i)$ .

An authentication channel has the property that only one agent can send and all agents can receive messages on this channel (provided they dispose of it). If agent  $P$  receives a message on  $Q$ 's authentication channel, a matching send event must have happened before in all sequences that  $P$  considers to be possible.

**Definition 7** For a system  $S$ , a channel  $(channel, Q)$  is called an Authentication Channel of agent  $Q$  if for all  $P \in \mathbb{P}$  holds that for all  $\omega \in S$  with the property that  $(s_i, Rec_P, \bar{s}_i) \in alph(\omega)$  is a receive event on  $(channel, Q)$  with message  $m_i$ , for all  $x \in \lambda_P^{-1}(\lambda_P(\omega)) \cap W_P$  exists a send event  $(s_j, e_j, \bar{s}_j) \in alph(x)$  on  $(channel, Q)$  with  $s_j < s_i$ ,  $e_j = Send_Q$  and message  $m_j = m_i$ . We denote the authentication channel of agent  $Q$  with  $(Auth, Q)$ .

A proof channel is an authentication channel which additionally provides a proof of authenticity on receipt of a message on the channel. The proof is considered to be a message as well and can therefore be forwarded to other agents. A proof channel provides strong, transferable proofs as defined by Kailar [7].

**Definition 8** For a system  $S$ , a channel  $(channel, Q)$  is called a Proof Channel of agent  $Q$  if

1.  $(channel, Q)$  is an authentication channel of  $Q$ ,

2. for all  $P \in \mathbb{P}$ ,  $\mathcal{N} \subseteq \text{Messages}$  holds:  
 $(\text{proof}, Q, \mathcal{N}) \in \text{State}_P(\bar{s}_i)$  implies that for all  $\omega \in S$  with  $(s_i, e_i, \bar{s}_i) \in \text{alph}(\omega)$ , the set of send events on  $(\text{channel}, Q)$  with message  $m \in \mathcal{N}$  is authentic for  $P$  after  $\omega$  and there exists a receive event  $(s_k, \text{Rec}_P, \bar{s}_k) \in \text{alph}(\omega)$  with  $s_k < s_i$  and  $(\text{proof}, Q, \mathcal{N}) \hookrightarrow \text{State}_P(s_k)$ .
3. for all  $P \in \mathbb{P}$  and  $\mathcal{N} \in \text{Messages}$  holds:  
 $(\text{proof}, Q, \mathcal{N}) \in \text{State}_P(s_i)$  for all send events  $(s_i, \text{Send}_P, \bar{s}_i)$  with message  $m = (\text{proof}, Q, \mathcal{N})$ .

We denote the proof channel of agent  $Q$  with  $(\text{Proof}, Q)$ .

## 5 A Concrete Price Request Protocol

In this section we present a concrete protocol implementing the price request/offer example introduced in Section 3 and show that it satisfies adequate concrete representations of Property 1. To implement these properties, we use a proof channel of  $SP$ . The messages that do not require any security property are sent on a broadcast channel. We assume that every agent can send and receive on the broadcast channel.

### 5.1 Specification

*Initial state.* For agents  $P, SP \in \mathbb{P}$  the initial state of the system is defined by  $\text{Channels}_P(s_0) = \{((\text{Proof}, SP), \text{rec})\}$  and  $\text{Channels}_{SP}(s_0) = \{((\text{Proof}, SP), \text{send})\}$ . If  $P = SP$ , then  $\text{Channels}_P$  includes both channel tuples. The state component Network and all other state components  $\text{State}_P$  and  $\text{ApplMem}_P$  of agents  $P \in \mathbb{P}$  are empty.

*Protocol steps.* We use so-called transition patterns to specify the protocol steps. For the definition of the state transition relation of an elementary automaton  $e \in \mathbb{E}$ , we need to specify the properties of all states of components  $C \in N(e)$  where  $e$  is active, i.e. can perform a state transition, and the changes of the states caused by the state transition. In the transition pattern notation the statements before  $\xrightarrow{e}$  constitute the prerequisites of  $e$  to perform a state transition and the statements behind  $\xrightarrow{e}$  describe the changes of the state.

- Step 1 ()

$\text{Appl}_P$   
 $\xrightarrow{\quad}$

$(\text{send}, \text{price\_req}) \hookrightarrow \text{ApplMem}_P$

The name of this pattern is *Step 1* and there are no variables.

Elementary automaton  $\text{Appl}_P$  starts the protocol by adding  $(\text{send}, \text{price\_req})$  to the state component  $\text{ApplMem}_P$  (denoted by  $\hookrightarrow$ ).

- Step 2 ()

$(\text{send}, \text{price\_req}) \in \text{ApplMem}_P$   
 $\text{Send}_P$   
 $\xrightarrow{\quad}$

If  $(\text{send}, \text{price\_req})$  is contained in the state of  $\text{ApplMem}_P$ , elementary automaton  $\text{Send}_P$  may send a price request.



$(send, price\_req) \leftrightarrow \text{ApplMem}_P$	$(send, price\_req)$ is removed from $\text{ApplMem}_P$ ,
$(sent, price\_req) \hookrightarrow \text{State}_P$	$(sent, price\_req)$ is added to $\text{State}_P$ and
$((price\_req), broadcast) \hookrightarrow \text{Network}$	the message is added to $\text{Network}$ as broadcast.

We assume that the user does not request the price from a particular service provider, so that the message does not include any address.

• Step 3 ( $m$ )

$(m, broadcast) \in \text{Network}$   
 $m = (price\_req)$   
 $\xrightarrow{\text{Rec}_{SP}}$   
 $(rec, price\_req) \hookrightarrow \text{ApplMem}_{SP}$

In Step 3 we have the variable  $m$ . Variable  $m$  is bound to a broadcast message in  $\text{Network}$ .  $SP$  checks that  $m$  equals  $(price\_req)$ .

The price request is added to the application interface.

• Step 4 ( $P$ )

$(rec, P, price\_req) \in \text{ApplMem}_{SP}$   
 $\xrightarrow{\text{Appl}_{SP}}$   
 $(rec, P, price\_req) \leftrightarrow \text{ApplMem}_{SP}$   
 $(price, offer) \hookrightarrow \text{ApplMem}_{SP}$

• Step 5 ( $price$ )

$(price, offer) \in \text{ApplMem}_{SP}$   
 $((Proof, SP), send) \in \text{Channels}_{SP}$   
 $\xrightarrow{\text{Send}_{SP}}$   
 $(price, offer) \leftrightarrow \text{ApplMem}_{SP}$   
 $((price, offer), (Proof, SP)) \hookrightarrow \text{Network}$

• Step 6 ( $SP, m$ )

$(sent, price\_req) \in \text{State}_P$   
 $((Proof, SP), rec) \in \text{Channels}_P$   
 $(m, (Proof, SP)) \in \text{Network}$   
 $elem(2, m) = offer$   
 $\xrightarrow{\text{Rec}_P}$   
 $(sent, price\_req) \leftrightarrow \text{State}_P$   
 $(rec, SP, m) \hookrightarrow \text{ApplMem}_P$   
 $(proof, SP, m) \hookrightarrow \text{State}_P$

• Step 7 ( $SP, m$ )

$(rec, SP, m) \in \text{ApplMem}_P$   
 $\xrightarrow{\text{Appl}_P}$   
 $(rec, SP, m) \leftrightarrow \text{ApplMem}_P$

• Proof Send ( $SP, m$ )

$(proof, SP, m) \in \text{State}_P$   
 $\xrightarrow{\text{Send}_P}$   
 $((proof, SP, m), Broadcast) \hookrightarrow \text{Network}$

• Proof Rec ( $SP, m$ )

$(m, broadcast) \in \text{Network}$   
 $elem(1, m) = proof$   
 $SP := elem(2, m)$   
 $\xrightarrow{\text{Rec}_P}$   
 $(proof, SP, elem(3, m)) \hookrightarrow \text{State}_P$

The transition patterns Step 1, Step 2, ..., Step 7, Proof Send and Proof Rec define a set of state transitions which constitute a system  $S_{PRO} \subseteq \Sigma^*$  without any malicious behaviour. Here  $\Sigma$  is the set of all defined state transitions in the APA protocol model described in Section 4.2. We now consider sets  $S \subseteq \Sigma^*$  including not explicitly specified malicious behaviour. It is assumed that  $S_{PRO} \subseteq S$ . The structure of the APA model implies implicit assumptions on what agents cannot do. Most important is the following assumption: The neighborhood relationship between elementary automata and state components implies that no agent can read or change the state of other agents' internal state components  $\text{Channels}_P$ ,  $\text{State}_P$  and  $\text{ApplMem}_P$ .

## 5.2 Relation between Different Levels of Abstraction

As the next step we want to establish a relation between the abstract system  $S'_{PRO}$  specified in Section 3 and all systems  $S$  with  $S_{PRO} \subseteq S \subseteq \Sigma^*$ . Therefore we specify a homomorphism from  $\Sigma^*$  to  $\Sigma'^*$  and show that it preserves authenticity and proofs in accordance with Definition 3 and Definition 4, respectively.  $S$  includes malicious behaviour and is restricted by the knowledge of agents about the security mechanisms used in the system. For each agent  $P$ , this knowledge is described by the set  $W_P$ . Therefore we assume  $S \subseteq W_P$  for all  $P \in \mathbb{P}$ .

The first assumption about the knowledge of  $P \in \mathbb{P}$  concerns the proof channel of  $SP \in \mathbb{SP}$ .

**Assumption 1** *For each  $SP \in \mathbb{SP}$  the channel  $(\text{Proof}, SP)$  is a proof channel in accordance with Definition 8, i.e. for all  $P \in \mathbb{P}$ ,  $W_P$  satisfies the properties of a proof channel.*

The property that every agent  $P \in \mathbb{P}$  can give proof of authenticity requires that all agents keep the proofs in their respective State component. If an agent deletes proofs he cannot give proof anymore. Notice that no other agent but  $P$  itself can change  $\text{State}_P$ .

**Assumption 2** *For all  $P \in \mathbb{P}$ ,  $SP \in \mathbb{SP}$  and messages  $m$  there is no state transition in  $S$  with  $(\text{proof}, SP, m) \leftrightarrow \text{State}_P$ .*

We now define a homomorphism  $h$  that maps  $S$  onto the abstract system  $S'_{PRO}$ . We then show that this homomorphism preserves proof pairs.

**Definition 9** *Let  $S \subseteq \Sigma^*$  be the protocol system defined by the state transition patterns in Section 5.1 enriched by malicious behaviour and let  $S'_{PRO} \subseteq \Sigma'^*$  be the abstract system defined in Section 3. We define a homomorphism  $h : \Sigma^* \rightarrow \Sigma'^*$  for  $P \in \mathbb{P}$  and  $SP \in \mathbb{SP}$  by*

$$h(s, (e, a, var), \bar{s}) = \begin{cases} \text{OFFER-S}_{SP} & \text{if } e = \text{Send}_{SP} \text{ and} \\ & (m, (\text{Proof}, SP)) \hookrightarrow \text{Network} \\ & \text{with } \text{elem}(2, m) = \text{offer} \\ \text{PROOF-R}_P(SP) & \text{if } e = \text{Rec}_P \text{ and} \\ & (\text{proof}, SP, m) \hookrightarrow \text{State}_P \\ & \text{with } \text{elem}(2, m) = \text{offer} \\ \text{PROOF-S}_P(SP) & \text{if } e = \text{Send}_P \text{ and} \\ & ((\text{proof}, SP, m), \text{broadcast}) \hookrightarrow \\ & \text{Network with} \\ & \text{elem}(2, m) = \text{offer} \\ \varepsilon & \text{else} \end{cases}$$

$(s, (e, a, var), \bar{s})$  denotes a state transition of elementary automaton  $e$  with  $a \in \phi_e$  (the alphabet of  $e$ ) and the list of interpretations of variables  $var$ .

Homomorphism  $h$  is now used to show that the concrete protocol system  $S_{PRO}$  enriched with malicious behaviour satisfies an adequate representation of Property 1 of the abstract specification  $S'_{PRO}$  in Section 3. First, it is shown that the homomorphic image of  $S$  under homomorphism  $h$  is the abstract specification  $S'_{PRO}$  assuming the use of secure proof channels. Then,  $h$  is proven to preserve authenticity and proofs. This section concludes with a proof that the system  $S_{PRO}$  enriched with malicious behaviour satisfies a property analogous to Property 1, as it provides the respective proof pairs.

**Lemma 1** *If Assumption 1 holds then  $h(W_P) \subseteq W'_P$  for  $P \in \mathbb{P}$ .*

**Proof:** We show that for every sequence  $\omega \in (\Sigma')^*$  that is not in  $W'_P$  all sequences in the inverse images  $h^{-1}(\omega)$  violate the properties of the proof channel and consequently are not in  $W_P$ .  $\omega \notin W'_P$  implies

$$\omega \in \bigcup_{SP \in \mathbb{SP}, Q \in \mathbb{P}} (\Sigma' \setminus \{\text{OFFER-S}_{SP}\})^* \{\text{PROOF-R}_Q(SP)\} \Sigma'^* \\ \cup (\Sigma' \setminus \{\text{PROOF-R}_Q\})^* \{\text{PROOF-S}_Q(SP)\} \Sigma'^*$$

By definition of  $h$  and Definition 8 follows that each state transition sequence in  $h^{-1}(\omega)$  violates the properties of (Proof, SP), thus is not element of  $W_P$ . This is a contradiction to Assumption 1 and it follows that  $h(W_P) \subseteq W'_P$ .  $\square$

By induction over the sequences of state transitions in  $S_{PRO}$  it can be shown that

**Lemma 2** *For  $S_{PRO} \subseteq \Sigma^*$  defined by the transition patterns Step 1, Step2, ..., Step 7, Proof Send and Proof Rec holds:  $h(S_{PRO}) = S'_{PRO}$ .*

**Lemma 3**  *$h(S) = S'_{PRO}$  if  $S_{PRO} \subseteq S \subseteq W_P$  for all  $P \in \mathbb{P}$ .*

**Proof:** By  $S \subseteq W_P$  and Lemma 1 holds  $h(S) \subseteq h(W_P) \subseteq W'_P = S'_{PRO}$ . With  $h(S_{PRO}) = S'_{PRO}$  and  $S \supseteq S_{PRO}$  follows  $h(S) = S'_{PRO}$ .  $\square$

In order to show that  $h$  preserves authenticity and proofs we need to specify the local views of the agents on both levels of abstraction. As already explained

in Section 2.2, the agents' local view of the abstract system is given by  $\pi'_P : \Sigma'^* \rightarrow \Sigma'_{/P}$  with  $\pi'_P(x) = x$  if  $x \in \Sigma'_{/P}$  and  $\pi'_P(x) = \varepsilon$  if  $x \in \Sigma' \setminus \Sigma'_{/P}$ , i.e.  $\lambda'_P = \pi'_P$ .

The following gives a general definition of the agents' local view in a system modelled by APA.

**Definition 10** Let  $P \in \mathbb{P}$  be an agent,  $E_P$  the set of elementary automata of  $P$  and  $s$  be a global system state of a state transition system  $S \subseteq \Sigma^*$ . Then  $s_P = \times_{C \in N(E_P)} (C(s))$  defines  $P$ 's view of the state  $s$ . The agent's view of the system  $S$  is given by the homomorphism  $\lambda_P : \Sigma^* \rightarrow \Sigma_P^*$  with

$$\lambda_P((s, e, \bar{s})) = \begin{cases} (s_P, e, \bar{s}_P) & \text{if } e \in E_P \\ \varepsilon & \text{else} \end{cases}$$

**Lemma 4**  $h$  preserves authenticity on  $S$ .

**Proof:** According to Definition 3, for each  $P \in \mathbb{P}$  we need to find a homomorphism  $h'_P : \lambda_P(S) \rightarrow \lambda'_P(S'_{PRO})$  such that  $\lambda'_P \circ h = h'_P \circ \lambda_P$ . This homomorphism is defined by

$$h'_P(s_P, (e, a, var), \bar{s}_P) = \begin{cases} \text{PROOF-R}_P(SP) & \text{if } e = \text{Rec}_P \text{ and} \\ & (proof, SP, m) \hookrightarrow \text{State}_P \\ & \text{with } elem(2, m) = offer \\ \text{PROOF-S}_P(SP) & \text{if } e = \text{Send}_P \text{ and} \\ & ((proof, SP, m), \\ & \text{broadcast}) \hookrightarrow \text{Network} \\ & \text{with } elem(2, m) = offer \\ \varepsilon & \text{else} \end{cases}$$

Obviously, this homomorphism provides the desired property.  $\square$

**Lemma 5** For  $SP \in \mathbb{SP}$  let  $\Gamma S'_{SP} = \{\text{PROOF-S}_P(SP) | P \in \mathbb{P}\}$  and  $\Gamma P'_{SP} = \{\text{PROOF-R}_P(SP) | P \in \mathbb{P}\}$ . Then for each  $SP \in \mathbb{SP}$  the homomorphism  $h : \Sigma^* \rightarrow \Sigma'^*$  defined in Definition 9 preserves  $(\Gamma S'_{SP}, \Gamma P'_{SP})$ -proofs on  $S$ .

**Proof:** By Lemma 4 homomorphism  $h$  preserves authenticity on  $S$ . Thus it remains to show conditions 1 and 2 of Definition 4.

Condition 1 We show the implication in two steps.

(i) We first show that  $h(\omega)^{-1}(h(S)) \cap (\Gamma S'_{SP} \cap \Sigma'_{/P}) \neq \emptyset$  implies  $\omega^{-1}(S) \cap (h^{-1}(\Gamma S'_{SP}) \cap \Sigma_{/P}) \neq \emptyset$ :

In  $S'_{PRO}$ , an action  $\text{PROOF-S}_P(SP)$  cannot occur without a previous action  $\text{PROOF-R}_P(SP)$ . By Lemma 3 holds  $h(S) = S'_{PRO}$ . Therefore,  $h(\omega)^{-1}(h(S)) \cap (\Gamma S'_{SP} \cap \Sigma'_{/P}) \neq \emptyset$  implies  $alph(h(\omega)) \cap (\Gamma P'_{SP} \cap \Sigma'_{/P}) \neq \emptyset$ . By the definition of  $h$  follows  $alph(\omega) \cap (h^{-1}(\Gamma P'_{SP} \cap \Sigma'_{/P})) \neq \emptyset$ . Hence, in  $\omega$  there exists a receive action by  $P$  with  $(proof, SP, m) \hookrightarrow \text{State}_P$ . Now, with Assumption 2 follows that this proof is not removed, i.e.  $(proof, SP, m) \in \text{State}_P$  after  $\omega$  has happened. Therefore, a state transition  $a \in \Sigma$  as defined with transition pattern Proof Send is possible, hence  $\omega^{-1}(S) \cap (h^{-1}(\Gamma S'_{SP}) \cap \Sigma_{/P}) \neq \emptyset$ .

- (ii)  $a \in \Gamma S'_{SP} \cap \Sigma_{/P}$  implies that  $((proof, SP, m), broadcast) \in \text{Network}$  after  $\omega a$  has happened. Therefore, a state transition  $b \in \Sigma$  in accordance with transition pattern Proof Rec is possible, hence  $\omega^{-1}(S) \cap (h^{-1}(\Gamma P'_{SP}) \cap \Sigma_{/R}) \neq \emptyset$ .

This shows that  $\omega$  can be continued in  $S$  with appropriate Proof Send and Proof Receive actions.

Condition 2 Holds by definition of  $h$ .  $\square$

**Property 2** For  $SP \in \mathbb{SP}$  let  $\Gamma S_{SP} = h^{-1}(\Gamma S'_{SP}) \cap \Sigma$ ,  $\Gamma P_{SP} = h^{-1}(\Gamma P'_{SP}) \cap \Sigma$  and  $\Gamma_{SP} = h^{-1}(\text{OFFER-}S_{SP}) \cap \Sigma$ . Then for each  $SP \in \mathbb{SP}$ ,  $(\Gamma S_{SP}, \Gamma P_{SP})$  is a proof action pair of authenticity for  $\Gamma_{SP}$  with respect to  $(W_P)_{P \in \mathbb{P}}$ .

**Proof:** By Theorem 2, the assertion holds if the following conditions hold:

- (1)  $(\Gamma S'_{SP}, \Gamma P'_{SP})$  is a proof action pair of authenticity for  $\{\text{OFFER-}S_{SP}\}$ .  
This holds by Property 1 in Section 3.
- (2) Homomorphism  $h$  preserves  $(\Gamma S'_{SP}, \Gamma P'_{SP})$ -Proofs on  $S$ .  
This condition holds by Lemma 5.  $\square$

## 6 Conclusions

In this paper we have presented a methodology for the specification of e-commerce transactions on different levels of abstraction, providing certain security properties. Based on a formal framework for binding cooperations we have defined the concepts of authenticity and proof of authenticity, using the notions of formal languages and language homomorphisms. The universality of the definitions allows to apply them to any specification language with a semantics based on labeled transition systems. We have formulated conditions on homomorphisms under which they preserve these properties from a higher to a lower abstraction level, thus serving as a means of refinement. This is in line with our general approach for verification of communicating systems where arbitrary safety and liveness properties of abstract specifications are transferred to more concrete ones by means of so-called simple language homomorphisms [10, 9]. For the specification on a lower abstraction level, we have used Asynchronous Product Automata (APA), a general class of communicating automata. Suitable homomorphisms map an APA specification onto a more abstract specification and transfer authenticity and proofs from the higher to the lower abstraction level.

Our approach is related to a range of work concerning authentication and proof (see, for example, [12], [13] and [7]). However, these papers are concerned with the analysis of security protocols. To the best of our knowledge, there is no approach that formally defines a proof of authenticity and provides a design methodology. Moreover, our approach is equally useful for the continuation of the design process where the protocol specification is transferred to the next level of refinement by introducing cryptographic mechanisms. This is subject of future work.

In a forthcoming paper it will be shown that our approach is adequate to formalize the concept of confidentiality.

## References

1. C. Boyd. A Framework for Design of Key Establishment Protocols. *Lecture Notes in Computer Science*, 1172:146–157, 1996.
2. C. Boyd and W. Mao. Design and analysis of key exchange protocols via secure channel identification. In J. Pieprzyk and R. Safavi-Naini, editors, *ASIACRYPT '94*, volume 917 of *Lecture Notes in Computer Science*, pages 171–181. Springer, 1994.
3. E.M. Clarke, S. Jha, and W. Marrero. A machine checkable logic of knowledge for specifying security properties of electronic commerce protocols. In *LICS Security Workshop*, 1998.
4. S. Eilenberg. *Automata, Languages and Machines*. Academic Press, New York, 1974.
5. R. Grimm and P. Ochsenschläger. Binding Cooperation, A Formal Model for Electronic Commerce. *Computer Networks*, 37:171–193, 2001.
6. S. Gürgens, P. Ochsenschläger, and C. Rudolph. Authenticity and Provability - a Formal Framework. GMD Report 150, GMD – Forschungszentrum Informationstechnik GmbH, 2001.
7. R. Kailar. Accountability in Electronic Commerce Protocols. *IEEE Transactions on Software Engineering*, 22(5):313–328, 1996.
8. U. M. Maurer and P. E. Schmid. A Calculus for Secure Channel Establishment in Open Networks. In D. Gollmann, editor, *Computer Security – ESORICS 94*, volume 875 of *LNCS*, pages 175 – 192. Springer, 1994.
9. P. Ochsenschläger, J. Repp, and R. Rieke. Abstraction and composition – a verification method for co-operating systems. *Journal of Experimental and Theoretical Artificial Intelligence*, 12:447–459, 2000.
10. P. Ochsenschläger, J. Repp, R. Rieke, and U. Nitsche. The SH-Verification Tool – Abstraction-Based Verification of Co-operating Systems. *Formal Aspects of Computing, The Int. Journal of Formal Methods*, 11:1–24, 1999.
11. C. Rudolph. *A Model for Secure Protocols and its Application to Systematic Design of Cryptographic Protocols*. PhD thesis, Queensland University of Technology, 2001.
12. S. Schneider. Security Properties and CSP. In *Symposium on Security and Privacy*. IEEE, 1996.
13. P. Syverson and C. Meadows. A Logical Language for Specifying Cryptographic Protocol Requirements. In *Proceedings of the 1993 IEEE Computer Society Symposium on Security and Privacy*, pages 165–177. IEEE Computer Society Press, New York, 1993.

# Protocol Engineering Applied to Formal Analysis of Security Systems

Javier Lopez, Juan J. Ortega, and Jose M. Troya

Computer Science Department, University of Malaga, Spain  
{jlm, juanjose, troya}@lcc.uma.es

**Abstract.** Every communication system requiring security properties is certainly critical. In order to study the security of communication systems, we have developed a methodology for the application of the formal analysis techniques of communication protocols to the analysis of cryptographic ones. We have extended the design and analysis phases with security properties. Our methodology uses a specification technique based on the HMSC/MSC requirement languages, and translates it into a generic schema for the SDL specification language, which is used for the analysis. Thus, the technique allows the specification of security protocols using a standard formal language and uses Object-Oriented for reusability purposes. The final goal is not only the formal specification of a security system, but to examine the possible attacks, and later use the specification in more complex systems.

## 1 Introduction

Nowadays, it is widely accepted that critical systems have to be analyzed formally in order to achieve well-known formal method benefits [10]. These methods characterize the behavior of a system in a precise way and can verify its formal specification. Design and analysis of security systems must benefit from advantages of formal methods because of the evident critical nature of such type of systems.

During last years, the cryptographic protocol analysis research area [14] has seen an explosive growth, with numerous formalisms being developed. We can divide this research into three main categories: logic-based [2], model checking [1,13,15] and theorem proving [5]. Only recently there has been a tendency to try to combine them.

We believe that the results obtained in the analysis phase of cryptographic protocols have a direct application on the design phase of a secure communication system. The reason is that there is no strong relation between security analysis tools and formal methods techniques of communication protocols.

Therefore, we have developed a new methodology [11] to specify secure systems and to check that they are not vulnerable against well-known attacks. Our approach uses a requirement language to describe security protocols, as well as a generic formal language, together with its associate verification methods and tools. In our method a simple and powerful intruder process is explicitly added to the specification, so that the verification of the security properties guarantees the robustness of the protocol against attacks of such an intruder. The intruder controls the transmission medium and

can perform the attacks [6]. His possible actions are killing, sniffing, intercepting, redirect, delaying, delivering, reordering, replaying, and faking of messages.

Currently, secrecy and authentication [17] are the security properties more widely analyzed. By analyzing secrecy we prevent the intruder from being able to derive the plaintext of messages passing between honest nodes. Our analysis consists of checking if the secret item can be deduced from the protocol messages and the intruder's database knowledge.

For an authentication protocol to be correct, it is required that a user Bob does not finish the protocol believing that it has been running with a user Alice unless Alice also believes that she has been running the protocol with Bob. Our analysis consists of checking if there is a reachable state where Bob has finished correctly and Alice will never reach her final state.

The paper is structured in the following way. In section 2 we summarize SDL features and tools. In section 3 we explain how SDL can be used to specify security protocols and cryptographic operations. At the same time we show how security protocols can be modeled as safety properties and checked automatically by a model-based verification tool. Section 4 shows an example of how our methodology is applied to EKE protocol. The last section contains conclusions and future work.

## 2 SDL Language

Specification and Description Language (SDL) [9] is a standard language for specifying and describing systems. It has been developed and standardized by ITU-T in the recommendation Z.100. An SDL specification/design (a system) consists of a number of interconnected modules (blocks). A block can recursively be divided into further blocks forming a hierarchy of blocks. Channels define the communication paths through which the blocks communicate with each other or with the environment. Each channel usually contains an unbounded FIFO queue that contains the signals that are transported on it. One or more communicating processes describe the behavior of the leaf blocks, and extended finite state machines describe the processes.

In addition, SDL supports object-oriented design [18] by a type concept that allows specialization and inheritance to be used for most of the SDL concepts, like blocks, processes, data types, etc. The obvious advantage is the possibility to design compact systems and to reuse components, which in turn reduces the required effort to maintain a system. SDL has adopted the term “type”, which corresponds to the term “class” used in many of the object-oriented notations and programming languages.

Telelogic's Tau SDL Suite provides an environment for developing SDL systems and implementations. The SDL Suite comprises a set of highly integrated tools that automate the transition from specification to real-time execution. With SDL's graphical language, SDL Suite describes, analyses, simulates, and supports generations of C/C++ applications. Thus SDL Suite simplifies testing and verifying the application by virtue of the formal semantics of the SDL language that makes tool support in early phases possible. The engineer composes diagrams, supported by a formal, well-defined graphical syntax that defines the program functionality, eliminating the need to manually write whole sections of code. We use the SDL



Validator tool for verification purposes. Indeed, we are going to take advantage of its exploration algorithms and its check mechanism.

## 2.1 SDL Validator

The SDL Validator [7] is based on state space exploration. State space exploration is based on automatic generation of reachable states of systems. The reachable state space of a system or application comprises all possible states it can assume, and all possible ways it can be executed. A reachability graph is one way to conceptually view reachable state space, though one rarely computes it since it is too large for realistic applications.

In addition, a reachability graph represents the complete behavior of an application. The nodes of the graph represent SDL system states, and it contains all necessary information to describe the application state. For an SDL system, the complete description of the application states includes the flow state of all concurrent processes, the value of all the variables, procedure call stacks, activated timers, signals in transmission and their parameter values, and so on.

The edges of the reachability graph represent SDL events that can take the SDL system from one system state to the next system state. The edges define the atomic events of the SDL system. These can be SDL statements like assignments, inputs and outputs, or complete SDL transitions depending on how the state space exploration is configured.

## 2.2 Exploration Algorithms

The state space can be explored using following algorithms: random walks, exhaustive exploration, bit-state exploration, and interactive simulation. The random walk algorithm randomly traverses the state space. Each time when several possible transitions are available, the SDL Validator chooses one of them and executes it. The random walk algorithm is useful as an initial attempt for robustness testing of an application and if the state space is too large even for a partitioned bit state search.

The exhaustive exploration algorithm is a straightforward search through the reachability graph. Each system state encountered is stored in RAM. Whenever a new system state is generated, the algorithm compares it with previously generated states to check if the new one was reached already during the search. If so, the search continues with the successor state. If the new state is the same as a previously generated state in RAM, the current path is pruned, and the search backs up to try other alternatives. The exhaustive exploration algorithm requires a lot of RAM, which limits its practical application.

The algorithm called bit state exploration can be used to efficiently validate reasonably large SDL systems. It uses a data structure called “hash table” to represent the system states that are generated during the exploration. When we want to analyze a particular situation, we use the interactive simulation. We guide state exploration to the goal scenario, and then we check for analysis results.

### 2.3 Checking Method

The Validator has several ways to check an SDL specification. These are essentially scenario verification and observer process. In order to obtain scenario specifications we use the Message Sequence Chart (MSC) [8] language (Recommendation ITU-T Z.120). We can verify a MSC by checking if there is a possible execution path for the SDL system that satisfies the MSC, or by checking MSC violation. This is achieved by loading MSC and performing a state space exploration set up in a way suitable for verifying MSCs. The MSC verification algorithm is a bit state exploration that is adapted to suit the needs of MSC verification.

The more powerful way to check a SDL specification is the observer process mechanism. The purpose of an observer process is to make it possible to check more complex requirements on the SDL system than can be expressed using MSCs. The basic idea is to use SDL processes (called “observer processes”) to describe the requirements that are to be tested and then include these processes in the SDL system. Typical application areas include feature interaction analysis and safety-critical systems.

By defining processes to be observer processes, the Validator will start to execute in a two-step fashion. First, the rest of the SDL system will execute one transition, and then all observer processes will execute one transition and check the new system state.

The assert mechanism enables the observer processes to generate reports during state space exploration. These reports will be included in the list of generated reports in the Report Viewer.

## 3 Analysis Mechanism

Our approach (depicted in figure 1) performs the design and analysis of a security protocol in the same way as the design and analysis of a traditional communication protocol. Firstly, we define system requirements. These are specified in the Security Requirements Specification Language (SRS�), which has been designed to define the security features and analysis strategy. Next we translate the system requirements into an SDL system in a semi-automatically manner, and analyze it. Furthermore it can be applied for code generation and testing, such as traditional communication protocol.

The aim of the SRS� is to define a high level language in order to specify cryptographic protocols and secure systems. This language must be modular to achieve reusability, it must be easy to learn, and has to use security concepts. The SRS� is divided into three parts; the first of which is the specification of protocol elements, the second one is message exchange flow, and the third comprises the security analysis strategies.

The essential elements required to define a security protocol can be divided into several categories. These are explained as follows (keywords in cursive):

- Entities: Agent (*Initiator/Responder*), principal identification; *Server\_Key*, provides a key; *Server\_Time*, provides time token; *Notary*, registers the transaction; *Server Certification Authority* (SCA), validates a certificate.

- Message: *Text*, clear text; *Random Number*, for freshness purposes; *Timestamp*, actual time; *Sequence*, count number.
- Keys: *Public\_key*, for instance, in PKCS#12 format; *Certificate*, public key signed by CA; *Private\_key*, used to sign; *Shared\_key*, secret key shared by more than one entity; *Session\_key*, it is a secret key used to encrypt a transmission.

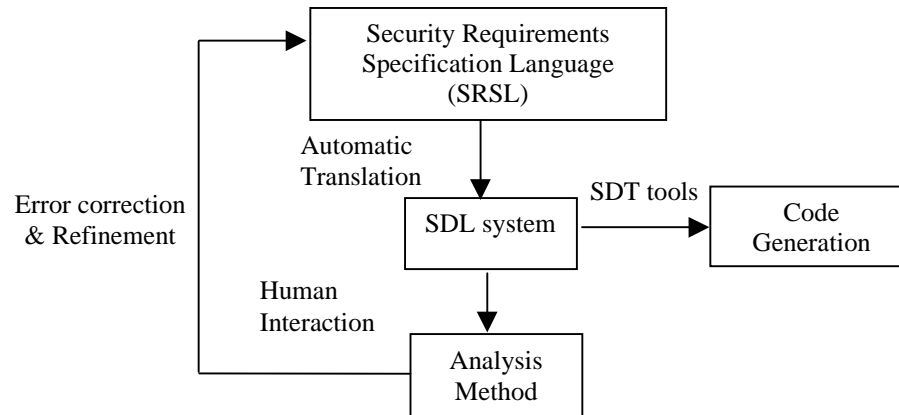


Fig.1. Methodology structure

In addition, SRSL may operate with data types previously defined. These operations are: Concatenate, composition of complex data (operator ','), it can be identified as a name; Cipher/Decipher, provides a cipher data and cleartext data, respectively (for instance, RSAcipher/RSADecipher PKCS#1 format); Hash, the result of a one-way algorithm; Sign, message hash encrypted with signer's private key (for instance, RSASign PKCS#7).

The message exchange is defined in the requirements language most widely utilized in telecommunications area, the Message Sequence Chart (MSC) and its extension High MSC (HMSC). We can specify elementary scenarios (MSC), and compose them into a complex protocol (HMSC).

We may describe security systems in a multilayer way. The first layer is the communication medium, besides it is used to apply an attack strategy (intruder behavior). The other layers depend on security mechanisms employed in system development.

For instance, we consider a system that uses SSL security mechanism to achieve server authentication and secret communication. Furthermore, it has to be design in order to achieve that the user's credit card number and the product code data are sent to the application server ( responder ), and the server gets an evidence of this. This means that the server wants to check non-repudiation of origin property. The module specification is depicted in figure 2.

The SSL layer can be described in a standard security communications package. Therefore, part of Medium layer is generated automatically. Consequently, we only have to specify the Initiator-Responder protocol. It is composed of simple scenarios described in MSC. This is depicted in figure 3.

As we can see in the MSC description of user protocol ,the security analysis section describes at least the check property. This is called "check" and it has three possible conditions: Authentication(A,B), for analyzing the authentication between agents A and B; secret(X), to evaluate if X can be deduced (also called confidentiality); Nonrepudiation(A,X), checks if data X ( the evidence ) can be produced only by A. Therefore, it is possible to define a specific attack scenario using "session\_instance" and "intruder\_behavior" sections, in order to prune the exploration space.

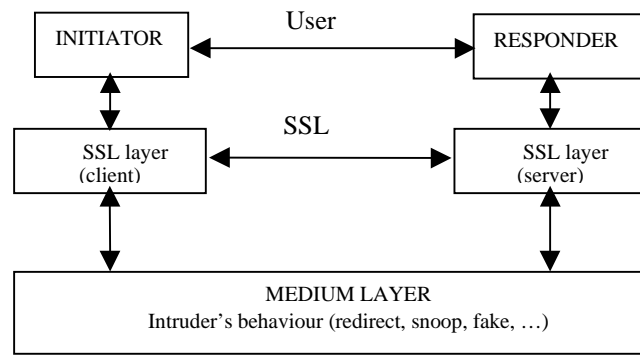


Fig. 2. Modules schema of security system specification

An Automatic translator program is used to achieve the SDL system from SRSL. The SDL system is composed of a package where data types are defined, and another package where one type process for each protocol agent, and a group of type process observer and process medium are specified.

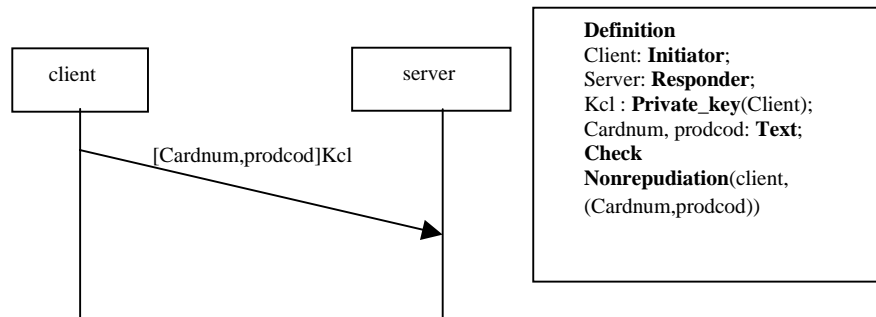


Fig. 3. MSC description of User Protocol

In order to analyze security properties, we evaluate the SDL system behavior when different kinds of attacks are applied by way of medium processes. Observer

processes check if a specific state is reached which results in an information report. This report corresponds to a failure scenario.

The analyzer creates the medium and observer processes for any kind of vulnerability that we want to examine. We have implemented generic medium and observer processes, but they must be extended depending on the given system environment. A more detailed description of every part of the SDL system is explained in the following.

### 3.1 Data Types Package

The messages, which are sent by protocol agents, are constructed by a concatenation of elemental data types and cryptographic operations. These data types can be divided into agent identification, number (random, time-stamping, etc...), symmetric key and asymmetric key pair. Thus, the operations used are cipher, decipher, sign and hash.

To be more precise, we use an abstract object certificate to model certificates, i.e. we abstract from the public key included in a certificate, a simplification which is accepted for analysis purposes. Of course, for code generation we need to use the ASN.1 notation, because this obtains an unambiguous data management.

The package “anacryptlib” defines data types used by the SDL specification. The SDL data types do not support recursive definition, so we make use of enumerated and structured data types. The elemental data types defined are: (a) agent Identification, it is an enumerated sort with all possible agents names; (b) number, it is a fresh and/or random value; (c) Secret key, this represents symmetric keys; (d) public key, this is composed by a key pair (private and public key); (e) encrypted message, which is implemented with a structured sort composed by an item message and an item symmetric or asymmetric cipher; (f) signed message, it is defined as a structured sort with a message and the private key of the signer.

Freshness or temporary secrets are implemented appending an item that has the process instance values, in particular we use the SDL sort PID for this purpose. Furthermore, we define a type set of knowledge for each data type. The intruder utilizes these types to store message knowledge.

### 3.2 Agents Package

The generic model identifies each protocol agent with a process type SDL. All process types are stored in a package in order to be used in other specifications. An agent specification is absolutely independent of the rest of the system, so they are generated in separated modules. Furthermore, this specification permits concurrent instances so that we can evaluate this behavior in the analysis stage.

The generic state transition of an agent process is triggered when it receives a message which is correct (i.e. accepted by the agent). Then, either the next message is composed to be sent to the receiver agent or it stops if the final state of the protocol for this process is reached. If the message is not correct, it will return to waiting message state.

The process in SDL is a finite state machine, so it finishes when executing a stop statement or it provides a deadlock if no signal arrives. Our model has to explore all possibilities; thus, we need to develop a mechanism to ensure that all signals sent

must be processed. Consequently, we have appended a state called “final” to notice the end of the protocol execution, and a general transition composed on a common “save” statement and a continuous signal, with less priority than the input statement, that checks if there are some signals waiting to be processed. By means of this structure we transform a finite state machine in an infinite one, only for analyzing purposes.

At this point, we have specified a security protocol in the same way as we might specify a traditional communication protocol, therefore we can examine the classical liveness properties. The specification must be well formed, but it is not the main aim of a secure system. In the next subsection, we are going to explain how to check the security properties.

### 3.3 Model Medium-Observer Processes

The intruder's behavior is divided into two aspects, exploration algorithm and check mechanism. The exploration algorithm is provided by a medium process, and observer processes perform the check mechanisms.

We consider two types of medium process model. The first is characterized by an exploration mechanism that searches all possibilities. It begins examining all combinations of different initial knowledge for each agent. Afterwards, it checks concurrent agents' execution, we try combinations of two concurrent sessions, and so on. Our algorithm finishes when an “out of memory” is produced or when it detects that the significant intruder knowledge is not incremented. In the general case the completeness problem [12,16] is undecidable, thus if the algorithms terminates without having found a flaw we do not have a proof that there is no flaw.

The second type of medium process is developed with an intruder specialized in finding a specific flaw. If we typify a kind of attack, we can evaluate the protocol trying to find a specific flaw. Perhaps this is not the best solution but it is very useful for a protocol designer to be sure that with respect to this kind of attack the protocol is not vulnerable. The only result we get is that a specific vulnerability does not occur in the cases we have examined.

The state transition of the process medium is triggered when it receives any message. After receipt, the message is stored in the intruder's knowledge database. The intruder then decides which operation to perform next and proceeds to the next routing state. We have defined three different operations: eavesdrop, redirect, and impersonate. For an eavesdrop operation the intruder intercept the message and it is not sent to any agent. Divert operation means that the intruder intercepts the message but does not forward into the original receiver. For an impersonate operation the intruder sends a faked message to the original receiver.

The observer process carries out the check mechanism. This is a special SDL type of process that is evaluated in each transition of the protocol specification. It has access to all variables and states of the complete process instances, so we can test it automatically.

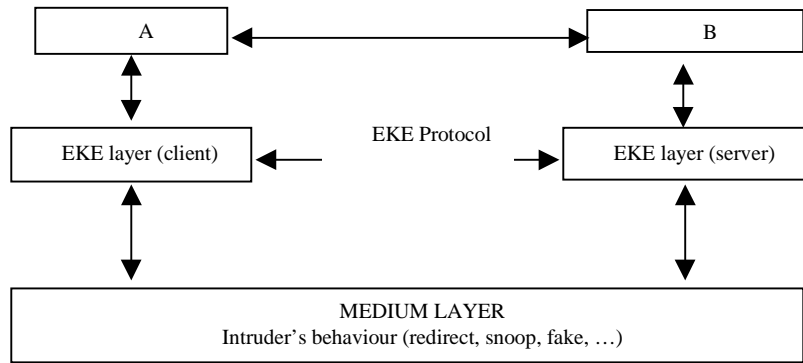
The security properties are proved using condition rules. These rules check different situations where it is possible that there exists protocol vulnerability. These elements are agent's states, variable value, and sent signals.

Currently we analyze secrecy and authentication properties. When we want to check secrecy properties, we examine if from the intruder knowledge a specific value

that we consider secret can be deduced. Authentication is examined by checking that all the principals finish at the expected protocol step. Some authors [4,5] call this the correspondence or precedence property.

#### 4 Example of Protocol Analysis

As an example to illustrate our proposal we explain the specification and analysis of a security system that makes use of the EKE (Encrypted Key Exchange) protocol [3], in order to gain access to a host, and cipher communications (figure 4).



**Fig. 4.** Multilayer module structure of EKE environment

We will evaluate EKE protocol. It is a key exchange authentication protocol that resists dictionary attacks by giving passive attacker insufficient information to verify a guessed password. As stated, it performs key exchange as well, so both parties can encrypt their transmissions once authentication is established. In the most general form of EKE, the two communicating parties encrypt short-lived public keys with a symmetric cipher, using their shared secret password as a key. Since it was designed, EKE has been developed into a family of protocols, many of which are stronger than the original or add new desirable properties. The basic EKE protocol is specified in SRSL (Figure 5). This represents two agents "A" and "B", A is the initiator and B is the responder. "P" is a shared key (a symmetric key shared by A and B). "Ka" is A's public key. "Re" is a session key (a fresh symmetric key) generated by B, and "Na" and "Nb" are fresh and random numbers of A and B, respectively.

Firstly, we produce the SDL specification of the EKE protocol, similar to an ordinary communications protocol. Then we create the medium and observer processes, in order to analyze the correspondence flaw found with the CASRUL analysis tool [4]. This flaw is produced when we execute two sessions concurrently. During the first session, "A" and "B" are instanced to "b" and "a" principals identification, respectively, while during the second session, "A" and "B" are instanced to "a" and "b" principals identification, respectively. The observer process checks if agents of sessions 1 and 2 reach a final state while their corresponding parties do not reach it.

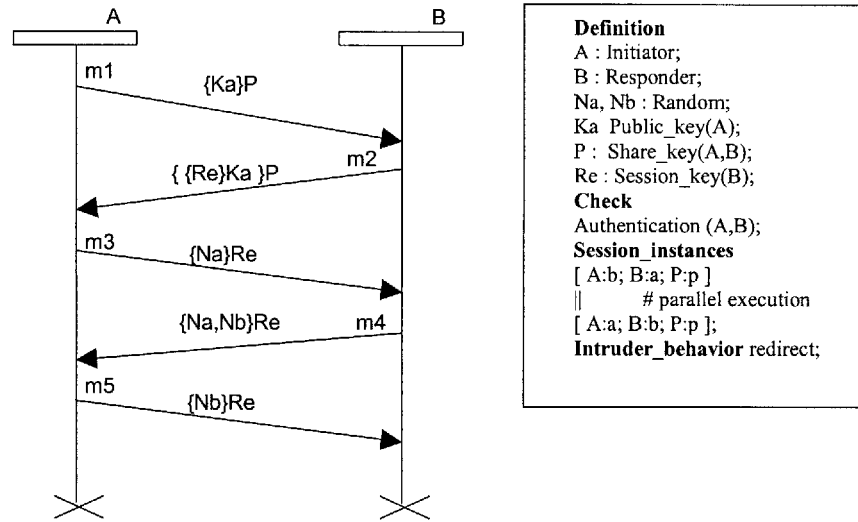


Fig. 5. EKE specification in SRSL

#### 4.1 Data Type Definition

The package "anacryptlib" includes all message definitions for analysis purposes. Messages are defined as SDL struct sort. These belong to a generic choice sort called "TMESSAGE". Additionally a generic struct sort called "TENCMESS" is defined that results from applying security operations to messages. The respective operators are "enc" to encipher, "denc" to decipher, "sign" to perform a digital signature, and "hash" to apply a hash function, as we explained above.

#### 4.2 Agent Definition

Agents are defined as a process type included in the SDL package called "agents". We specify the agent initiator process type ("agentA"), and the agent responder process type ("agentB"). Both process types have states called "mess" plus a number of the message. Each state has an input signal that is triggered when its related message is received. This message is checked before being accepted, and the process stops if it has reached the final state, or it composes the corresponding message, sends it, and proceeds to next state.

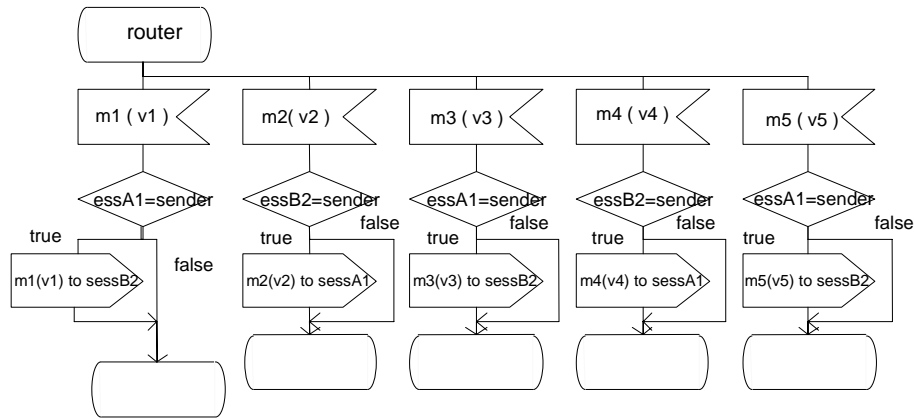
In order to treat all messages that are received, we have defined an asterisk state that saves all unprocessed signals, and in a lower priority level it checks if there are any queued messages. If this is the case, it changes its state to the one related to that signal. Fresh data types are provided adding to their definition a process identification item (PID SDL sort). This is used to differentiate every concurrent session. All these process types can be used in a more complex system where the EKE protocol is the authentication procedure.



### 4.3 Intruder Model

The Intruder model is divided into the exploration algorithm and the check mechanism. These depend on the analysis strategy that the analyzer must evaluate. There are a number of attack types that have been developed, which can be implemented in our analysis mechanism.

The process type called "CorresAttack" provides the exploration algorithm. This consists of a state that is triggered by any input message and executes the intruder's operation. Then, the divert operation is applied sending a message from the first session to the second session, and vice versa.(figure 6). This kind of attack is called the man-in-the-middle attack.



**Fig. 6.** Main state of "CorresAttack" process

In order to check the correspondence flow we create the process type observer called "obvcorresattack". The main state ( checking ) has two possible alternatives which we can see in the following SDL program code:

```

STATE checking
if ((GetState(A2)='final') AND (GetState(B1)='final'))
  AND(((GetState(B2)!='final')AND ((Getstate(A1)!='final'))))
then
  REPORT "authentication error"
  STOP
else
  if((GetState(A1)='final')AND
    GetState(B2)='final')AND(((GetState(B1)!='final')AND
    (Getstate(A2)!='final'))))
  then
    REPORT "authentication error"
    STOP
  else
    NEXTSTATE checking
  
```

The first condition is true when agent A of session two (A2) and agent B of session one (B1) reach a "final" state, and the other agents do not reach it. The second condition checks the inverse situation. When the checked condition is true, a report action is executed, and the observer process scan stop searching or continue exploring for a new failure scenario. The report is provided in MSC language.

#### 4.4 Verification Procedure

In order to explore the correspondence failure we have defined the processes distribution that is depicted in figure 7. The system is specified connecting through the instance of the process type "Corresattack" called "medattack", an instance of process type "agentA" and another of process type "agentB". Those instances are called "A" and "B", respectively.

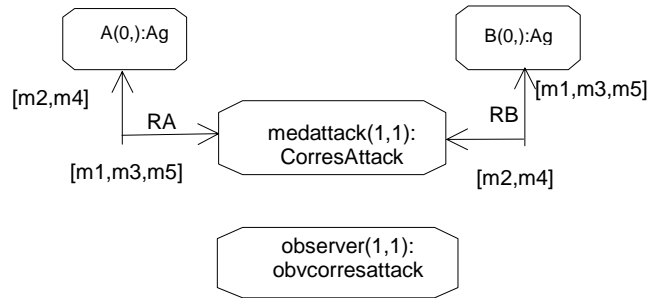


Fig. 7. SDL processes distribution

This processes distribution enforces all messages that are sent between A and B to pass through the medium instance, where they are processed by the intruder's operations, i.e. the intruder's operations provide redirect operations.

#### 4.5 Analysis Result

We then load the SDL specification into the SDL Validator tool. Firstly we configure the Validator in order to evaluate the system correctly. Then we execute the exhaustive exploration option. It finishes quickly and it generates an MSC report (figure 8). The following code shows how two sessions start at the same time. When the medium process intercepts the messages from one session and sends to the other session, agent A of the first session and agent B of the second session reach a "final" state. This means that agent A of the first session believes that in this session it is communicating with agent B, but in fact it is connected with agent B of the second session.

First session instance  
A\_1=b, B\_1=a

Second session instance  
A\_2=a, B\_2=b

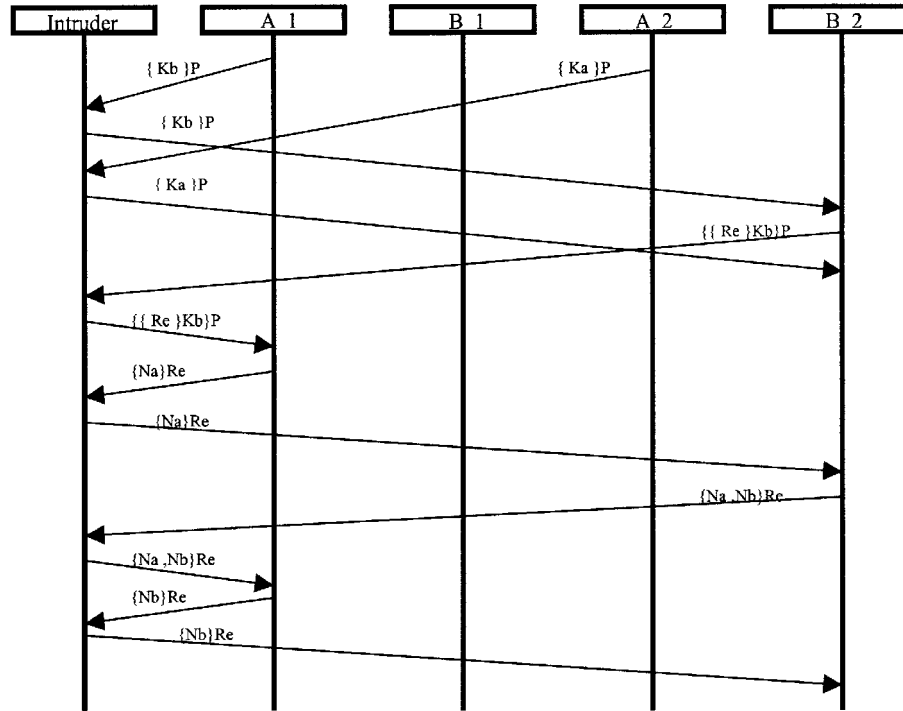


Fig. 8. MSC of correspondence attack of EKE protocol

## 5 Conclusions and Future Work

We have presented a new mechanism to specify security protocols and their possible attacks. The security protocol is specified in SRSL language and translated into the SDL system, and attacks are implemented as SDL processes that develop the intruder's behavior and check safety properties. A protocol specification is independent of the analysis procedures, so it can be used in other environments as well.

Several kinds of security attacks are analyzed using our method. It is essential to study how they can be produced in a real environment. We examine the result scenario provided in an analysis procedure, and redesign the security protocol if was necessary.

Currently we are extending SRSL in order to represent more complex protocols and to analyze other properties, such as anonymity. In order to check several kinds of attacks, we are gathering a set of generic attacks. Furthermore, we are studying how to implement these attacks in the Internet environment.

## References

1. R. Alur, T. Henzinger, F. Mang, S. Qadeer, S. Rajamani, and S. Tasiran. Mocha: modularity in model checking. CAV 98: Computer-aided Verification, Lecture Notes in Computer Science 1427, pages 521–525. Springer-Verlag, 1998
2. M. Burrows, M. Abadi, and R. Needham. A logic of authentication. In Proceedings of the Royal Society, Series A, 426(1871):233–271, 1989
3. S.M. Bellovin and M. Merrit. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In Proceedings of IEEE Symposium on Research in Security and Privacy, pages 72–84, 1992
4. CASRUL. <http://www.loria.fr/equipes/protheo/SOFTWARES/CASRUL/>.
5. G. Denker and J. Millen. CAPSL intermediate language. In Formal Methods and Security Protocols, 1999. FLOC '99 Workshop
6. D. Dolev and A. Yao. On the security of public key protocols. IEEE Transactions on Information Theory, IT-29:198–208, 1983
7. D. Hogrefe. Validation of SDL systems. Computer Networks and ISDN Systems, 28 (12):1659–1667, June 1996
8. ITU-T, Geneva, Switzerland. Message Sequence Charts, 1999. ITU-T Recommendation Z.120
9. ITU-T, Geneva, Switzerland. Specification and Description Language (SDL), 1999. ITU-T Recommendation Z.100
10. G. Holzmann. Design and Validation of Computer Protocols. Prentice-Hall, Englewood Cliffs, 1991
11. J. Lopez, J.J. Ortega, J. M. Troya. Verification of authentication protocols using SDL-Method. First International Workshop on Security in Information Systems (SIS 2002), April 2002, pp. 61–71
12. G. Lowe. Towards a Completeness Result for Model Checking of Security Protocols. 11th IEEE Computer Security Foundations Workshop, pages 96–105. IEEE Computer Society, 1998
13. W. Marrero, E. Clarke, and S. Jha. Model checking for security protocols. DIMACS Workshop on Design and Formal Verification of Security Protocols, 1997
14. C. Meadows. Open issues in formal methods for cryptographic protocol analysis. Proceedings of DISCEX 2000, pages 237–250. IEEE Comp. Society Press, 2000.
15. J. C. Mitchell, M. Mitchell, and U. Stern. Automated analysis of cryptographic protocols using Murphi. In Proceedings of IEEE Symposium on Security and Privacy, pages 141–151. IEEE Computer Society Press, 1997.
16. M. Rusinowich, M. Turuani. Protocol Insecurity with Finite Number of Sessions is NP-complete. 14th IEEE Computer Security Foundations Workshop June 11–13, 2001
17. P. Ryan and Scheneider. The Modelling and Analysis of Security Protocols: the CSP Approach. Addison-Wesley, 2001
18. A. Sarma J. Ellsberger, D. Hogrefe. SDL–Formal object-oriented language for communication systems. Prentice-Hall, 1997.

# Applications of Multiple Trust Authorities in Pairing Based Cryptosystems

L. Chen<sup>1</sup>, K. Harrison<sup>1</sup>, D. Soldera<sup>1</sup>, and N.P. Smart<sup>2</sup>

<sup>1</sup> Hewlett-Packard Laboratories,  
Filton Road,  
Stoke Gifford,  
Bristol, BS34 8QZ,  
United Kingdom.

{liqun.chen, keith.harrison, david.soldera}@hpl.hp.com

<sup>2</sup> Computer Science Department,  
Woodland Road,  
University of Bristol,  
Bristol, BS8 1UB,  
United Kingdom.  
nigel@cs.bris.ac.uk

**Abstract.** We investigate a number of issues related to the use of multiple trust authorities and multiple identities in the type of identifier based cryptography enabled by the Weil and Tate pairings. An example of such a system is the Boneh and Franklin encryption scheme. We present various applications of multiple trust authorities. In particular we focus on how one can equate a trust authority with a way to add contextual information to an identity.

## 1 Introduction

In 1984 Shamir [11] introduced the concept of identity based cryptography and proposed a signature scheme based on the RSA assumption. Over the years a number of researchers tried to propose secure and efficient identity based encryption algorithms, but with little success. This state of affairs changed in 2001 when two identity based encryption algorithms were proposed, one by Cocks [6] based on the quadratic residuosity assumption and another by Boneh and Franklin based on the Weil pairing, although using a variant based on the Tate pairing is more efficient. A number of identity based signature algorithms based on the Tate pairing also exist, e.g. [5], [9] and [10].

In this paper we investigate the applications of the Boneh and Franklin scheme in more detail. In particular we examine how the key structure can lead to interesting applications. Due to our later applications we prefer to talk about *identifier* based cryptography rather than *identity* based cryptography. This is because the word identity seems to imply the name of someone (such as their email address), whilst the schemes actually associate keys with strings.

Each string being able to represent anything, such as legal terms and conditions, and not just a persons identity.

We shall use the additive property of keys of the pairing based systems to define a way of expressing groupings in a clear and concise manner. We then use this to define policies which are then enforced via the means of possession of certain keys. This results in a kind of “key calculus”.

The “key calculus” we present is different from, but inspired by, the ideas of Desmedt [7] and Boyd [3] [4] who presented the notion of group oriented cryptography. Usually, one sees group oriented cryptography in the context of group signatures, we shall be concerned in this paper with applications to encryption. Much of our discussion will, however, also apply to identifier based signature schemes as well.

The paper is organised as follows, first we recap on the Boneh-Franklin encryption scheme. Then we go through a number of applications which are enabled due to the additive property of the keys in the scheme. Finally we give the semantics of the resulting key calculus which will describe precisely what sets of keys can be singled out in any given application.

## 2 Notation and Pairing Based Schemes

### 2.1 The Tate Pairing

Let  $G_1$  and  $G_2$  denote two groups of prime order  $q$  in which the discrete logarithm problem is believed to be hard and for which there exists a computable bilinear map

$$t : G_1 \times G_1 \longrightarrow G_2.$$

We shall write  $G_1$  with an additive notation and  $G_2$  with a multiplicative notation, since in real life  $G_1$  will be the group of points on an elliptic curve and  $G_2$  will denote a subgroup of the multiplicative group of a finite field.

Since the mapping is bilinear, we can move exponents/multipliers around at will. For example if  $a, b, c \in \mathbb{F}_q$  and  $P, Q \in G_1$  then we have

$$\begin{aligned} t(aP, bQ)^c &= t(aP, cQ)^b = t(bP, cQ)^a = t(bP, aQ)^c = t(cP, aQ)^b = t(cP, bQ)^a \\ &= t(abP, Q)^c = t(abP, cQ) = t(P, abQ)^c = t(cP, abQ) \\ &= \dots \\ &= t(abcP, Q) = t(P, abcQ) = t(P, Q)^{abc} \end{aligned}$$

These tricks will be used repeatedly throughout this document.

We define the following cryptographic hash functions

$$\begin{aligned} H_1 : \{0, 1\}^* &\longrightarrow G_1, \\ H_2 : G_2 &\longrightarrow \{0, 1\}^*. \end{aligned}$$

## 2.2 Types of Public/Private Key Pairs

We require the following two types of keys:

- A standard public/private key pair is a pair  $(R, s)$  where  $R \in G_1$  and  $s \in \mathbb{F}_q$  with

$$R = sP$$

for some given fixed point  $P \in G_1$ .

- An identifier based key pair is a pair  $(Q_{\text{ID}}, S_{\text{ID}})$  where  $Q_{\text{ID}}, S_{\text{ID}} \in G_1$  and there is some trust authority (TA) with a standard public/private key pair given by  $(R_{\text{TA}}, s)$ , such that the key pair of the trust authority and the key pair of the identifier are linked via

$$S_{\text{ID}} = sQ_{\text{ID}} \text{ and } Q_{\text{ID}} = H_1(\text{ID}),$$

where ID is the identifier string.

## 2.3 Identifier Based Encryption

We recap on the Boneh and Franklin encryption algorithm. Although much of what we will discuss also applies to other pairing based schemes such as the short signature scheme of Boneh, Lynn and Shacham [2] or the identity based signature schemes to be found in [5], [9] and [10].

The scheme of Boneh and Franklin [1], allows the holder of the private part  $S_{\text{ID}}$  of an identifier based key pair to decrypt a message sent to her under the public part  $Q_{\text{ID}}$ . We present only the simple scheme which is only ID-OWE, for a ID-CCA scheme one applies the Fujisaki-Okamoto transformation [8].

Let  $m$  denote the message to be encrypted

- **Encryption :**  
Compute  $U = rP$  where  $r$  is a random element of  $\mathbb{F}_q$ . Then compute

$$V = m \oplus H_2(t(R_{\text{TA}}, rQ_{\text{ID}}))$$

Output the ciphertext  $(U, V)$ .

- **Decryption :**  
Decryption is performed by computing

$$\begin{aligned} V \oplus H_2(t(U, S_{\text{ID}})) &= V \oplus H_2(t(rP, sQ_{\text{ID}})) \\ &= V \oplus H_2(t(P, Q_{\text{ID}})^{rs}) \\ &= V \oplus H_2(t(sP, rQ_{\text{ID}})) \\ &= V \oplus H_2(t(R_{\text{TA}}, rQ_{\text{ID}})) \\ &= m. \end{aligned}$$

### 3 Applications of the Addition of Keys

The main observation of this section is that we can add trust authorities public keys or users identifier based public keys together. We in this way obtain corresponding “virtual” secret keys to obtain an exponential number of keys. We shall go on to show a number of applications of this ability to add keys, a property which does not hold in standard cryptographic systems. We consider only the simplest case, other cases are possible but lead to more complicated protocols. In a later section we shall extend the notion and explain precisely which virtual keys one can obtain.

We first present the case where there are  $n$  trust authorities each with their own standard public/private key pair

$$R_{TA_i} = s_i P.$$

We assume that all the public keys  $R_{TA_i}$  are trusted by all parties in the system. In addition, suppose we have a fixed identifier  $ID$  and we obtain the  $n$  private keys corresponding to this identifier from the relevant trust authorities, i.e. we have

$$S_{ID_i} = s_i Q_{ID}$$

where

$$Q_{ID} = H_1(ID).$$

Given an  $n$  bit string  $\mathbf{b} = (b_1, \dots, b_n)$  we can then form the private key

$$S_{ID, \mathbf{b}} = \sum_{i=1}^n b_i S_{ID_i}$$

corresponding to the “virtual” trust authority with public key

$$R_{TA, \mathbf{b}} = \sum_{i=1}^n b_i R_{TA_i}$$

and private key

$$s_{\mathbf{b}} = \sum_{i=1}^n b_i s_i.$$

In this way we can produce  $2^n$  different public/private key pairs given only  $n$  trust authorities.

As a second case assume we have one fixed trust authority with its standard public/private key pair

$$R_{TA} = sP.$$

Now assume we have  $n$  identifier's  $ID_i$ , with private keys given by

$$S_{ID_i} = sQ_{ID_i}$$

where



$$Q_{ID_i} = H_1(ID_i).$$

Given an  $n$  bit string  $\mathbf{b} = (b_1, \dots, b_n)$  we can then form the private key

$$S_{ID, \mathbf{b}} = \sum_{i=1}^n b_i S_{ID_i}$$

corresponding to the “virtual” identifier

$$Q_{ID, \mathbf{b}} = \sum_{i=1}^n b_i Q_{ID_i}.$$

This two situations open up a large number of possible application areas which we shall now explore.

### 3.1 Escrowing of Keys

The first, obvious, application of key addition is that one does not need to have a single point of failure with a single trust authority. With identifier based cryptography the trust authority is always able to either decrypt messages (with an encryption scheme) or to sign messages (with a signature scheme). By using the key addition property one can distribute this key amongst a number of trust authorities, such that no single trust authority ever obtains more than their own portion. Indeed users, both encryptors and decryptors, can perform this operation themselves on the fly by selecting which subsets of trust authorities they will use in any given message. This property of identifier based systems in that the encryptor can choose, at the time of encryption, which trust authorities it is willing to trust we feel gives a significant advantage to using multiple trust authorities.

### 3.2 Contexts for Identities

A major problem when using an identity as a means to deduce a public key is that “names” for identities are not unique. There may be many “Ron Rivest”s in the world, so which one do I mean when I send an encrypted message to the person with identity “Ron Rivest”? We seem to have the same global name hierarchy problems that traditional PKI technologies come with.

Using multiple trust authorities we can add some context to each identity. For example suppose the International Association of Cryptologic Research (IACR) was a trust authority with public/private key pair

$$R_{IACR} = rP.$$

They could issue all cryptographers with the name “Ron Rivest” the same public/private key pair

$$S_{\text{Ron Rivest}, IACR} = rQ_{\text{Ron Rivest}},$$

where

$$Q_{\text{Ron Rivest}} = H_1(\text{Ron Rivest}).$$

Now, suppose there was also a trust authority for the US government with public/private key pair given by

$$R_{\text{US}} = sP.$$

This trust authority would issue all US citizens with a public/private key pair corresponding to their name, i.e. Ron Rivest would be given the key pair

$$S_{\text{Ron Rivest,US}} = sQ_{\text{Ron Rivest}}.$$

In addition suppose there is a trust authority for the Massachusetts Institute of Technology with public/private key pair

$$R_{\text{MIT}} = tP.$$

They would issue all employees with the public/private key corresponding to their name as

$$S_{\text{Ron Rivest,MIT}} = tQ_{\text{Ron Rivest}}.$$

Suppose some third party wished to send an encrypted message to Ron Rivest. They may know that Ron Rivest is an US national and a cryptographer who works at MIT. They therefore create the “virtual” trust authority public key corresponding to what context they wish to put the identity of Ron Rivest in, i.e. they can create

$$R_{\text{virtual}} = R_{\text{IACR}} + R_{\text{US}} + R_{\text{MIT}}.$$

Note, the corresponding secret key for the virtual trust authority is

$$r + s + t,$$

but no party knows this key (and no two colluding parties can determine this key). The third party now encrypts the data for Ron Rivest using the virtual trust authorities public key and the public key

$$Q_{\text{Ron Rivest}}.$$

The encrypted data is then sent, along with some information to encode which contexts the identity is being considered in.

Ron Rivest can then decrypt the information using the private key

$$S_{\text{Ron Rivest,IACR}} + S_{\text{Ron Rivest,US}} + S_{\text{Ron Rivest,MIT}}.$$

Clearly this use of contexts to narrow down the possibilities for sending data to the wrong Ron Rivest also comes with the added benefit that no single trust authority can decrypt the encrypted message.

A similar technique can also be applied to any form of group membership, for example in access control systems where the rights to do some task may depend on whether the principal is a member of a number of different groups.

### 3.3 Legal Hoop Jumping

One can use a similar idea to create legal hoops through which people must jump before a certain action can be performed. We give an example, which may be peculiar to the United Kingdom, although others can be easily created.

In the United Kingdom every car needs to display a tax disk. This is purchased each year for a nominal fee, and essentially proves that at a given point in the year the owner of the car had car insurance and a certificate of road worthiness for the car. We describe a possible online car tax disk dispenser.

We note the three obvious trust authorities

- The ownership of the car is recorded by the Driver and Vehicle Licensing Agency (DVLA).
- The insurance certificate is produced by an insurance company, say AXA.
- The certificate of road worthiness is produced by an accredited garage, say Joes Garage.

The three trust authority public keys we denote by

$$R_{\text{DVLA}}, R_{\text{AXA}}, R_{\text{Joes}}.$$

Suppose the owner of the car with registration number **X 675 AHO** wished to obtain a new tax disk from the government. They could then log into some web site and claim that they owned the car, that they had insured it through AXA and that Joes Garage had issued them with a certificate of road worthiness. The government could then email the user an encrypted version of the tax disk, upon payment of some fee, where the encryption is under the virtual trust authority

$$R_{\text{DVLA}} + R_{\text{AXA}} + R_{\text{Joes}},$$

and the identifier is

$$Q_{\text{X 675 AHO}}.$$

The owner would need to obtain from each trust authority the corresponding private key (clearly date/year information needs to be added but we ignore that issue here for simplicity),

$$S_{\text{X 675 AHO,DVLA}}, S_{\text{X 675 AHO,AXA}}, S_{\text{X 675 AHO,Joes}}.$$

The owner now adds these private keys together to form another private key,

$$S_{\text{X 675 AHO,virtual}},$$

with which they can decrypt the electronic form of the tax disk and print it on their printer.

### 3.4 Concurrency Issues

One can imagine a situation where there is a trust authority which reveals the secret key corresponding to the identifier given by the time and date, at given time intervals. The secret key is broadcast to all, anyone who then uses the secret key in combination with others is therefore proving that they did so after a certain time. One should think of this trust authority acting like a Greenwich Pips signal. In this way issues of concurrency can be brought into applications.

One application would be that of press releases. Suppose a press release related to a companies accounts needs to be released simultaneously at 12.00 on June 1st to a group of financial journalists. To do this one takes the two public keys

$$\begin{aligned} Q_{\text{journalists}} &= H_1(\text{journalists}), \\ Q_{1200:01:06} &= H_1(1200:01:06), \end{aligned}$$

and forms the sum

$$Q = Q_{\text{journalists}} + Q_{1200:01:06}.$$

If the trust authority for both identifies is the same  $R_{\text{TA}}$ , then using  $R_{\text{TA}}$  and  $Q$  one can encrypt the press release and distribute it before the event, knowing that only after 12.00 on June 1st a given group of people (namely journalists) will be able to read the press release. Later we shall see a more general solution to this problem, which does not require the trust authorities to be the same.

### 3.5 Advantages over Traditional Solutions

Whilst in standard discrete logarithm based schemes one can still add keys together to form a “virtual” private key, the difference with identifier based schemes is that the public keys need to exist before the addition takes place. This is because the only way traditional keys are trusted is via a digital certificate. Hence, the trust authority (or CA for traditional schemes) needs to certify the public key before it is added to another one.

With identifier based schemes one knows the public key by simply knowing the identifier. Hence, one can add the public keys or TA keys together and encrypt the message before the entity singled out by the identifier has gone to the trust authority to obtain their private key.

## 4 A “Key Calculus”

In this section we formally define our key calculus which formed the basis of the examples above. The following formalism is richer than the examples above. We assume a set of “atomic” identifiers  $I$ , these are identifiers which correspond to true identifier based public keys,

$$Q_{\text{ID}_i} = H_1(\text{ID}_i),$$

where  $\text{ID}_i \in I$ .

We also assume a set of “atomic” trust authorities  $T$ . These are trust authorities which possess public/private key pairs,

$$R_{\mathbf{TA}_i} = s_i P,$$

where  $\mathbf{TA}_i \in T$ , for which the value of  $s_i$  is explicitly known by at least one party (i.e. the trust authority). We shall assume all parties in the system have trusted copies of the TA’s public keys  $R_{\mathbf{TA}_i}$ .

To each pair  $(\mathbf{ID}, \mathbf{TA})$  of atomic identifier/trust authority keys, called an atomic pair, there is a corresponding secret key denoted by

$$S_{\mathbf{ID}, \mathbf{TA}} = s_{\mathbf{TA}} Q_{\mathbf{ID}},$$

although it may be the case that no one in the system, bar the trust authority, knows the value of  $S_{\mathbf{ID}, \mathbf{TA}}$ .

The goal is to encrypt a message  $m$  to a set of people who know a given subset of keys. We would like to do this in as short a message as possible and be able to describe completely exactly which set of keys are needed to decrypt the message.

#### 4.1 Naïve Conjunction of Sets

We let  $\mathbb{S}$  denote a set of atomic pairs  $(\mathbf{ID}_i, \mathbf{TA}_j)$ . We first wish to encrypt a message so that it can be decrypted only by people who possess, for every pair in  $\mathbb{S}$ , the equivalent atomic secret key. This can be trivially done using an onion form of encryption where each encryption is applied in turn to obtain the ciphertext. Not only is this trivial technique wasteful of computing resources it also implies that the decryptor needs to apply the necessary decryptions in the same order that the encryptor applied the encryptions.

We will try and be more efficient, but this will come at the price of not being able to deal with all possible sets  $\mathbb{S}$ . First we define an operation of sets of atomic pairs  $\mathbb{S}_1, \mathbb{S}_2$

$$\mathbb{S}_1 \otimes \mathbb{S}_2 = \{(\mathbf{ID}_i, \mathbf{TA}_j) : (\mathbf{ID}_i, \mathbf{TA}') \in \mathbb{S}_1 \cup \mathbb{S}_2, (\mathbf{ID}', \mathbf{TA}_j) \in \mathbb{S}_1 \cup \mathbb{S}_2\}.$$

A set  $\mathbb{S}$  will be called admissible if

- $\mathbb{S} = \{(\mathbf{ID}, \mathbf{TA})\}$  consists of a single atomic pair.
- or  $\mathbb{S}$  is built up from two admissible sets  $\mathbb{S}_1$  and  $\mathbb{S}_2$  using the operation

$$\mathbb{S} = \mathbb{S}_1 \otimes \mathbb{S}_2.$$

We let  $\mathbf{ID}(\mathbb{S})$  denote the set of identifiers and  $\mathbf{TA}(\mathbb{S})$  denote the set of trust authorities contained in an admissible set  $\mathbb{S}$ . The “virtual” trust authority corresponding to  $\mathbb{S}$  is given by

$$R_{\mathbb{S}} = \sum_{\mathbf{TA} \in \mathbf{TA}(\mathbb{S})} R_{\mathbf{TA}},$$

and the “virtual” identifier is given by

$$Q_S = \sum_{ID \in ID(S)} Q_{ID}.$$

Encrypting using these two public keys, namely  $R_S$  and  $Q_S$ , means that only recipients who possess **every** private key corresponding to every pair in  $S$  will be able to decrypt the message. Equivalently, the decryptor needs to know

$$S_S = \sum_{ID \in ID(S)} \left( \sum_{TA \in TA(S)} S_{ID,TA} \right).$$

Hence, we call this secret key the virtual secret key corresponding to the admissible set  $S$ .

**Example.** To see the use of the special conjunction operation  $\otimes$  consider the case of two trust authorities and two identifiers. We form the admissible atomic sets

$$S_1 = \{(ID_1, TA_1)\}, \quad S_2 = \{(ID_2, TA_2)\},$$

where  $R_{TA_j} = s_j P$ . Then

$$S = S_1 \otimes S_2 = \{(ID_1, TA_1), (ID_2, TA_1), (ID_1, TA_2), (ID_2, TA_2)\}.$$

Suppose a cryptogram  $(U, V)$  is sent, in the Boneh-Franklin scheme, to the virtual trust authority and identifier corresponding to the admissible set  $S$ . This cryptogram corresponds to the public key of the trust authority

$$R = R_{TA_1} + R_{TA_2},$$

and the “identifier” given by

$$Q = Q_{ID_1} + Q_{ID_2}.$$

To decrypt the cryptogram we need to compute

$$\begin{aligned} t(R, rQ) &= \prod_{i,j=1}^2 t(R_{TA_j}, rQ_{ID_i}) = \prod_{i,j=1}^2 t(s_j P, rQ_{ID_i}), \\ &= \prod_{i,j=1}^2 t(rP, s_j Q_{ID_i}) = \prod_{i,j=1}^2 t(U, S_{ID_i, TA_j}), \\ &= t(U, \sum_{i,j=1}^2 S_{ID_i, TA_j}). \end{aligned}$$

Hence, we need to know the four secret keys  $S_{ID_i, TA_j}$  for  $1 \leq i, j \leq 2$ , or in other words we need to know the “virtual” secret key

$$S_S = \sum_{i,j=1}^2 S_{ID_i, TA_j}.$$

We have seen in Section 3 examples where either  $ID_i = ID_{i'}$  for all  $i, i'$  or  $TA_j = ID_{j'}$  for all  $j, j'$ . We leave it to the reader to check all our previous examples correspond to admissible sets.

#### 4.2 General Conjunction of Sets

We have seen that encryption to the virtual trust authorities and virtual identifier represented by an admissible set  $\mathbb{S}$  is equivalent to encryption to an entity which possesses the virtual secret key  $S_{\mathbb{S}}$ . From now on we will drop the usage of the word “virtual” when dealing with the trust authority and identifier based keys represented by an admissible set  $\mathbb{S}$ .

The above form of conjunction of keys is too restrictive in that it does not allow us to encrypt to someone who holds the secret key corresponding to the pairs  $(ID_1, TA_1)$  and  $(ID_2, TA_2)$ , without knowing also knowing the secret keys corresponding to  $(ID_2, TA_1)$  and  $(ID_2, TA_1)$ .

To enable this we change the encryption algorithm slightly. Suppose we have admissible sets  $\mathbb{S}_i$ , corresponding to the pairs  $(ID_i, TA_i)$ . To encrypt to the conjunction  $\mathbb{S}_1 \vee \mathbb{S}_2 \vee \dots \vee \mathbb{S}_n$  we transmit  $U = rP$  and

$$V = m \oplus H_2 \left( \prod_{i=1}^n t(R_{TA_i}, rQ_{ID_i}) \right).$$

Decryption is performed by computing

$$m = V \oplus H_2 \left( t(U, \sum_{i=1}^n S_{ID_i, TA_i}) \right).$$

So for decryption we still add private keys together, but for encryption we need to multiply ephemeral keys together, after they have been passed through the pairing.

We can now return to our press release example. Using an authority for time, which simply issues the private key according to the current time at given time intervals, and another (different) authority to issue private keys to all journalists, we can encrypt a press release so that the intended recipients can only read it at some point in the future. Notice, how this is simple using identifier based systems since the encryptor does not need to know in advance what the time authority will do. The encryptor need only know that the authority will issue the private key corresponding to a given identifier.

With a traditional public key system one could obtain the same effect but only by knowing the public key corresponding to the private key at the encryption stage, but this would involve the encryptor asking the time authority for precisely which public key should be used.

#### 4.3 Naive Disjunction of Sets

The use of variable identifiers has little use unless one is able to define a disjunction operation. For example one may wish to send an encrypted email to either

the president or CEO of the ACME company. Hence, using the trust authority ACME one would want to encrypt to either

$$(\text{PRESIDENT}, \text{ACME}) \text{ or } (\text{CEO}, \text{ACME}).$$

We now show how to modify the Boneh-Franklin encryption scheme so that we can encrypt to a disjunction of admissible sets, in a way which is akin to the naive conjunction above, in the next subsection we shall deal with a general form of disjunction which only works for two admissible sets. The following idea of a naive disjunction will work for an arbitrary collection of trust authorities and/or identifiers but we present, for ease of exposition, the case where we have two trust authorities and two identifiers.

Suppose the two trust authorities have public keys

$$R_{T_1} = s_1P \text{ and } R_{T_2} = s_2P.$$

Let the two identifiers be

$$Q_A \text{ and } Q_B.$$

Hence, there are a total of four secret keys

$$\begin{aligned} S_{A,1} &= s_1Q_A, & S_{A,2} &= s_2Q_A, \\ S_{B,1} &= s_1Q_B, & S_{B,2} &= s_2Q_B. \end{aligned}$$

We would like to encrypt a message so that a party who has any one of these secret keys is able to decrypt the message. We make the assumption that there are four fixed integers

$$x_1, x_2, y_1, y_2 \in \mathbb{F}_q^*$$

known to all parties, such that  $x_1 \neq x_2$  and  $y_1 \neq y_2$ . We then form the virtual public keys given by

$$R_V = R_{T_1} + y_1R_{T_2} \text{ and } Q_V = Q_A + x_1Q_B.$$

We also form the “auxiliary” keys,

$$T_R = R_{T_1} + y_2R_{T_2} \text{ and } T_Q = Q_A + x_2Q_B.$$

The cryptogram is then formed by computing, for some random  $r \in \mathbb{F}_q^*$ ,



$$\begin{aligned} U_1 &= rP, & U_2 &= rT_Q, & U_3 &= rT_R, \\ V &= m \oplus H_2(t(R_V, rQ_V)). \end{aligned}$$

To decrypt the message  $(U_1, U_2, U_3, V)$  we will make use of the identities,

$$\begin{aligned} R_V &= \alpha R_{T_1} + \beta T_R \quad \text{or} \quad R_V = \gamma R_{T_2} + T_R, \\ Q_V &= \delta Q_A + \epsilon T_Q \quad \text{or} \quad Q_V = \zeta Q_B + T_Q, \end{aligned}$$

where

$$\begin{aligned} \alpha &= (y_2 - y_1)/y_2, & \beta &= y_1/y_2, & \gamma &= y_1 - y_2, \\ \delta &= (x_2 - x_1)/x_2, & \epsilon &= x_1/x_2, & \zeta &= x_1 - x_2. \end{aligned}$$

The precise identities one uses will depend on which of the secret keys the decryptor has access to.

**Decryptor knows  $S_{A,1}$ .** We first show how to decrypt assuming the recipient has the private key  $S_{A,1}$ . The decryptor needs to compute  $t(R_V, rQ_V)$ , which with knowledge of  $S_{A,1}$  they can do via:

$$\begin{aligned} t(R_V, rQ_V) &= t(\alpha R_{T_1} + \beta T_R, r(\delta Q_A + \epsilon T_Q)), \\ &= t(\alpha R_{T_1}, r\delta Q_A) \cdot t(\alpha R_{T_1} + \beta T_R, r\epsilon T_Q) \cdot t(\beta T_R, r\delta Q_A), \\ &= t(rs_1\alpha P, \delta Q_A) \cdot t(\alpha R_{T_1} + \beta T_R, \epsilon U_2) \cdot t(r\beta T_R, \delta Q_A), \\ &= t(r\alpha P, s_1\delta Q_A) \cdot t(\alpha R_{T_1} + \beta T_R, \epsilon U_2) \cdot t(\beta U_3, \delta Q_A), \\ &= t(\alpha U_1, \delta S_{A,1}) \cdot t(\alpha R_{T_1} + \beta T_R, \epsilon U_2) \cdot t(\beta U_3, \delta Q_A). \end{aligned}$$

**Decryptor knows  $S_{A,2}$ .** We first show how to decrypt assuming the recipient has the private key  $S_{A,2}$ . The decryptor needs to compute  $t(R_V, rQ_V)$ , which with knowledge of  $S_{A,2}$  they can do via:

$$\begin{aligned} t(R_V, rQ_V) &= t(\gamma R_{T_2} + T_R, r(\delta Q_A + \epsilon T_Q)), \\ &= t(\gamma R_{T_2}, r(\delta Q_A + \epsilon T_Q)) \cdot t(rT_R, \delta Q_A + \epsilon T_Q), \\ &= t(\gamma rs_2 P, \delta Q_A) \cdot t(\gamma R_{T_2}, r\epsilon T_Q) \cdot t(U_3, \delta Q_A + \epsilon T_Q), \\ &= t(\gamma U_1, \delta S_{A,2}) \cdot t(\gamma R_{T_2}, \epsilon U_2) \cdot t(U_3, \delta Q_A + \epsilon T_Q). \end{aligned}$$

The other two cases we leave for the reader, since they are computed in a similar way.

#### 4.4 General Disjunction of Sets

Just as with our earlier naive conjunction, which led to the construction of the admissible sets, the above naive disjunction has a number of problems. The main being that when encrypting to the disjunction of the two pairs  $(ID_1, TA_1)$  and  $(ID_2, TA_2)$ , the above construction will allow anyone with the secret key corresponding to the pair  $(ID_1, TA_2)$  to decrypt. This may not be the effect we are after.

In this subsection we show how a proper disjunction can be created between two identifier/authority pairs, such that only the desired recipients can decrypt the resulting ciphertext. Once again we are looking for a more efficient solution than simply encrypting the message twice to different possible recipients. Our solution however does not extend to forming a disjunction of more than two identifier/authority pair, unlike the naive form of disjunction above.

Suppose I have the two pairs  $(ID_1, TA_1)$  and  $(ID_2, TA_2)$ . To encrypt to either of these pairs one forms the virtual identifier/trust authority keys given by

$$R_V = Q_B + x_1 R_{T_1} \text{ and } Q_V = Q_A + y_1 R_{T_2},$$

and the auxiliary keys given by

$$T_R = Q_B + x_2 R_{T_1} \text{ and } T_Q = Q_A + y_2 R_{T_2},$$

again assuming some globally fixed integers  $x_1, x_2, y_1, y_2 \in \mathbb{F}_q^*$ .

The cryptogram is then formed by computing, for some random  $r \in \mathbb{F}_q^*$ ,

$$\begin{aligned} U_1 &= rP, & U_2 &= rT_Q, & U_3 &= rT_R, \\ V &= m \oplus H_2(t(R_V, rQ_V)). \end{aligned}$$

To decrypt the message  $(U_1, U_2, U_3, V)$  we will make use of the identities,

$$\begin{aligned} R_V &= \alpha R_{T_1} + T_R \text{ or } R_V = \beta Q_B + \gamma T_R, \\ Q_V &= \delta Q_A + \epsilon T_Q \text{ or } Q_V = \zeta R_{T_2} + T_Q, \end{aligned}$$

where

$$\begin{aligned} \alpha &= x_1 - x_2, & \beta &= (x_2 - x_1)/x_2, & \gamma &= x_1/x_2, \\ \delta &= (y_2 - y_1)/y_2, & \epsilon &= y_1/y_2, & \zeta &= y_1 - y_2. \end{aligned}$$

The precise identities one uses will depend on which of the secret keys the decryptor has access to.

If the decryptor has knowledge of  $S_{A,1}$  they use the identities

$$R_V = \alpha R_{T_1} + T_R \text{ and } Q_V = \delta Q_A + \epsilon T_Q$$

to decrypt using

$$\begin{aligned} t(R_V, rQ_V) &= t(\alpha R_{T_1} + T_R, r\delta Q_A + r\epsilon T_Q), \\ &= t(\alpha rP, \delta S_{A,1}) \cdot t(\alpha R_{T_1}, r\epsilon T_Q) \cdot t(rT_R, Q_V), \\ &= t(\alpha U_1, \delta S_{A,1}) \cdot t(\alpha R_{T_1}, \epsilon U_2) \cdot t(U_3, Q_V). \end{aligned}$$

If the decryptor has knowledge of  $S_{B,2}$  they use the identities

$$R_V = \beta Q_B + \gamma T_R \text{ and } Q_V = \zeta R_{T_2} + T_Q$$

to decrypt using

$$\begin{aligned} t(R_V, rQ_V) &= t(\beta Q_B + \gamma T_R, r\zeta R_{T_2} + rT_Q), \\ &= t(\beta S_{B,2}, \zeta rP) \cdot t(\beta Q_B, rT_Q) \cdot t(\gamma rT_R, Q_V) \\ &= t(\beta S_{B,2}, \zeta U_1) \cdot t(\beta Q_B, U_2) \cdot t(\gamma U_3, Q_V). \end{aligned}$$

Finally notice if the decryptor uses the other two combinations of the identities then they recover no information since, for example, using

$$R_V = \beta Q_B + \gamma T_R \text{ and } Q_V = \delta Q_A + \epsilon T_Q$$

we obtain

$$\begin{aligned} t(R_V, rQ_V) &= t(\beta Q_B + \gamma T_R, \delta rQ_A + \epsilon rT_Q), \\ &= t(\beta Q_B, \delta rQ_A) \cdot t(\gamma U_3, \delta Q_A) \cdot t(R_V, \epsilon U_2) \end{aligned}$$

Hence, to decrypt, one would need knowledge of either  $rQ_A$  or  $rQ_B$  neither of which is available to the decryptor.

## 5 Conclusion

We have seen how multiple trust authorities combined with conjunctions of keys can be used in a variety of applications. We have explained how various conjunctions of keys (or essentially access rights) can be created in an efficient manner. The use of identifier based systems with multiple trust authorities and multiple identifiers we believe opens up simplified ways of specifying access rights, a number of possible real life examples we have given in the text.

We have shown how to build up so called ‘admissible sets’ of identifier/trust authority pairs, via a form of specialised conjunction. These admissible sets can then be used to form general conjunctions, and or a special form of naive disjunction. The general form of disjunction is only possible for two such sets. A general form of disjunction for arbitrary numbers of admissible sets we leave as an open research problem.

## References

1. D. Boneh and M. Franklin. Identity based encryption from the Weil pairing. *Advanced in Cryptology – CRYPTO 2001*, Springer-Verlag LNCS 2139, 213–229, 2001.
2. D. Boneh, B. Lynn and H. Shacham. Short signatures from the Weil pairing. *Advances in Cryptology – ASIACRYPT 2001*, Springer-Verlag LNCS 2248, 514–532, 2001.
3. C. Boyd. Some applications of multiple key ciphers. *Advances in Cryptology – EUROCRYPT ’88*, Springer-Verlag LNCS 330, 455–467, 1988.
4. C. Boyd. A new multiple key cipher and an improved voting scheme. *Advances in Cryptology – EUROCRYPT ’89*, Springer-Verlag LNCS 434, 617–625, 1989.
5. J. C. Cha and J. H. Cheon. An Identity-Based Signature from Gap Diffie-Hellman Groups. Preprint 2002.
6. C. Cocks. An identity based encryption scheme based on quadratic residues. *Cryptography and Coding*, Springer-Verlag LNCS 2260, 360–363, 2001.
7. Y. Desmedt. Society and group oriented cryptography: A new concept. *Advances in Cryptology – CRYPTO ’87*, Springer-Verlag LNCS 293, 120–127, 1987.

8. E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Advances in Cryptology – CRYPTO '99*, Springer-Verlag LNCS 1666, 537–554, 1999.
9. F. Hess. Efficient identity based signature schemes based on pairings. To appear *Selected Areas in Cryptography – 2002*.
10. K. Paterson. ID-based Signatures from Pairings on Elliptic Curves. Preprint 2002.
11. A. Shamir. Identity based cryptosystems and signature schemes. *Advanced in Cryptology - CRYPTO '84*, Springer-Verlag LNCS 196, 47–53, 1985.

# Plausible Deniability Using Automated Linguistic Stegonagraphy

Mark Chapman<sup>1</sup> and George Davida<sup>2</sup>

<sup>1</sup> Omni Tech Corporation  
N27 W23676 Paul Road  
Pewaukee, WI 53072, U.S.A.  
Tel.: (262) 523-3300, Fax: (262) 523-2233  
e-mail: mark.chapman@omnitechcorp.com  
www: <http://www.omnitechcorp.com/>

<sup>2</sup> Department of EE & CS  
University of Wisconsin–Milwaukee  
Milwaukee, WI 53201, U.S.A.  
Tel.: (414) 229-5192 , Fax:(414) 229-6958  
e-mail: [davida@cccns.uwm.edu](mailto:davida@cccns.uwm.edu)

**Abstract.** Information hiding has several applications, one of which is to hide the use of cryptography. The Nicetext [5,6] system introduced a method for hiding cryptographic information by converting cryptographic strings (random-looking) into “nice text” (namely innocuous looking). The system retains the ability to recover the original ciphertext from the generated text. Nicetext can hide both plaintext and cryptographic text.

The purpose of such transformations are to mask ciphertext from anyone who wants to detect or censor encrypted communication, such as a corporation that may monitor, or censor, its employee private mail. Even if the message is identified as the output of Nicetext, the sender might claim that the input was simply a pseudo-random number source rather than ciphertext.

This paper extends the Nicetext protocol to enable deniable cryptography/messaging using the concepts of plausible deniability [2,7]. Deniability is derived from the fact that even if one is forced to reveal a key to the random string that “nice text” reverts to, the real cryptographic/plaintext messages may be stored within additional required sources of “randomness” in the extended protocol.

## 1 Introduction

There are many methods to maintain secrecy. Cryptography attempts to make information unintelligible [10]. Information hiding and covert-channel techniques attempt to transmit information within existing protocols [4]. Steganography attempts to transmit information within existing information [8]. The Deniable Nicetext protocol combines these techniques in an attempt to maintain secrecy

through deniability. In this case, secrecy may be for both the information itself and the use of particular cryptographic techniques.

A *Survey of Information Hiding* [8] introduces different methods of hiding information. These include steganography, water marking, covert channels and other methods of hiding. The concept of *Chaffing and Winnowing*, [9] and [3], introduces a method to hide messages in network traffic.

The purpose of this paper is to propose an enhanced automated system to enable someone to deny that a cryptographic, or random, string has meaning. The concept of *Plausible Deniability* in this context implies that it is difficult for an adversary to prove the meaning of a particular message. The authors do not provide any ability to deny the transmission or receipt of a message. The emphasis is on the ability to deny meaning. Another benefit may be the ability to deny the use of a specific cryptographic algorithm.

The term, *Plausible deniability* has distinct meanings in the legal profession [11]. There are also political contexts of plausible deniability where a person is deliberately kept in the dark to render their denial of an event “plausible” or even believable. What is important to understand is that any such scheme, automated or not, requires a certain level of deceit from some human being who is interested in denying the claim. The deceit, in the case of the system discussed in this paper, must come from the sender and receiver of the messages.

The results of the Deniable Nicetext transformations may contain two or more messages. It should be difficult for an observer to prove the existence of more than one messages. Even with the suspicion of multiple messages, the external encryption routines should make it difficult for an adversary to prove the meaning of a particular message. With the mutual cooperation from the sender and receiver it should be possible to plausibly deny the meaning of the entire message.

This paper, introduces a system to achieve deniability of cryptograms as well as plaintext. The system introduced extends the Nicetext suite of applications [5,6]. After briefly discussing the Nicetext system for completeness, the authors introduce Recursive Nicetext which leads to the new *Deniable Nicetext*.

## 2 Basic Nicetext

The basic idea of the Nicetext protocol is to transform a ciphertext,  $c$ , into harmless looking text through the following algorithm:

1. Independent of the input ciphertext, choose a contextual template, if current one is exhausted.
2. Read the next *word type* from the template.
3. Read enough bits from input ciphertext to select a particular word of the proper type from the dictionary.
4. Output the word.
5. Repeat until end of input ciphertext.

Given a code dictionary, such as that found in Table 1, and a *contextual template*, Nicetext converts the bits of ciphertext into an “interesting” stream of words. Table 2 demonstrates how different templates may generate different nice-text from the same input ciphertext.

**Table 1.** Basic Dictionary Table with Multiple Types.

Type	Code	Word
name_male	0	$\longleftrightarrow$ ned
name_male	1	$\longleftrightarrow$ tom
name_female	0	$\longleftrightarrow$ jody
name_female	1	$\longleftrightarrow$ tracy

**Table 2.** Different Contextual Templates Produce Different Nicetext from the same input.

Contextual template $s$	Ciphertext $c$	$NT_{d,s}(c)$
name_male name_male name_male	011	$\longrightarrow$ “ned tom tom”
name_male name_male name_female	011	$\longrightarrow$ “ned tom tracy”
name_male name_female name_male	011	$\longrightarrow$ “ned tracy tom”
name_male name_female name_female	011	$\longrightarrow$ “ned tracy tracy”
name_female name_male name_male	011	$\longrightarrow$ “jody tom tom”
name_female name_male name_female	011	$\longrightarrow$ “jody tom tracy”
name_female name_female name_male	011	$\longrightarrow$ “jody tracy tom”
name_female name_female name_female	011	$\longrightarrow$ “jody tracy tracy”

The term to describe the output of a steganography transformation is stego-text or, more generally, stego-object [1]. In this paper, stego-text and nice-text are synonymous.

The reverse transformation is a simple substitution cipher where appropriate bits from the dictionary replace the words parsed from the nice-text. The contextual templates have nothing to do with the ciphertext recovery process.

Much of the prior work on Nicetext focused on creating large, sophisticated dictionaries. These dictionary may contain hundreds of thousands of words carefully categorized into thousands of word types. The encoding scheme for the dictionary must follow the properties required for forward and reverse transformation.

Armed with sophisticated dictionaries, the Nicetext system uses multiple techniques to automatically generate contextual-templates from example texts. For example, it is trivial for the Nicetext system to create a contextual template for this paragraph or, for that matter, this entire paper <sup>1</sup>.

<sup>1</sup> Contextual templates also provide information about capitalization, punctuation and whitespace. The Nicetext system does not currently store any ciphertext in

It is then possible to convert any ciphertext into nice-text which has structure and to some extent meaning, *the ultimate goal of linguistic steganography*, similar to this paper.

To demonstrate the quality of the various transformation methods, the authors start with the popular children's story, *The Wizard of Oz*, which is available from the Project Gutenberg web page at [gutenberg.org](http://gutenberg.org). The truncated examples only show the first paragraph for brevity. The following is an actual excerpt from L. Frank Baum's *The Wizard of Oz*:

#### THE WONDERFUL WIZARD OF OZ

##### 1. The Cyclone

Dorothy lived in the midst of the great Kansas prairies, with Uncle Henry, who was a farmer, and Aunt Em, who was the farmer's wife. Their house was small, for the lumber to build it had to be carried by wagon many miles. There were four walls, a floor and a roof, which made one room; and this room contained a rusty looking cookstove, a cupboard for the dishes, a table, three or four chairs, and the beds. Uncle Henry and Aunt Em had a big bed in one corner, and Dorothy a little bed in another corner. There was no garret at all, and no cellar—except a small hole dug in the ground, called a cyclone cellar, where the family could go in case one of those great whirlwinds arose, mighty enough to crush any building in its path. It was reached by a trap door in the middle of the floor, from which a ladder led down into the small, dark hole. . . .

Using a very large dictionary with 189,991 words put into 56,429 categories, the system automatically generated several different contextual templates. This paper does not describe the contextual template generation process.

The Nicetext process uses these advanced dictionaries and templates to convert ciphertext into new stories which, to varying degrees, share similar characteristics with the original document. Rather than showing the contextual templates, this paper emphasizes the results of transformations of the following ciphertext, shown in hexadecimal notation:

145727 030276 120333 134506 163625 001617 050723 126027

The first example shows the results of transforming the above ciphertext using a series of contextual templates based on the structure of the sentences in the original story:

So the soldier was summoned and entered the Throne Room doggedly, for wherewithal Oz was alive he never was allowed to come farther than the door. When the Guardian of the Gate saw them again he wrapped greatly that they should leave the beautiful City to get into new trouble.

---

whitespace, etc. to provide maximum flexibility and robustness against countermeasures designed to thwart other programs which exclusively use whitespace to store information.



I am never hungry, he said, and it is a lucky thing I am not, before my mouth is only commented, and if I could cut a hole in it so I could eat, the straw I am stuffed neath would come out, and that would spoil the shape of my head. The sky was blackened, and a low rumbling sound was overheard in the air. Surely no wild beast should wish a babylonier home. . . .

The primary benefit of sentence-level contextual templates is the wide variety of text that the system can generate from the same input ciphertext. The second example transforms the same ciphertext using the same sentence-level contextual template library to create a very different story:

The wild crows flew in one great flock toward Dorothy and her companions. You have plenty of courage, I am sure, answered Oz. No; there were some, but the monster has eaten them all. I was afraid I could have to stay in the river forever, he said, but the kind Stork saved me, and if I ever beget any brains I shall find the Stork again and do her some kindness in return. She wiggled at the rapid flight of the Monkeys, but was glad the journey was over. There! All you need is confidence in yourself. . . .

The reason for the variation is that contextual templates are chosen independently from the input ciphertext. The above two examples used different sequences of sentence-level templates. The Nicetext system can recover the input ciphertext, in this case, 145727. . . , from either generated text.

The main disadvantage of sentence-level contextual templates is that, although each sentence appears to properly mimic the style of the original document, the sequence of sentences appears to be random, as it should.

An alternative is to use contextual templates based on larger sections of a document. The next example uses templates based on the structure and format of the entire original document:

## THE SUPEREXCELLENT WIZARD OF OZ

### 10. The Cyclone

Dorothy lived in the midst of the great Kansas prairies, versus Uncle Henry, whosoever was a farmer, and Aunt Em, whoso was the farmer's wife. Their house was small, before the lumber to build it had to be carried by cockle many miles. There were four walls, a floor and a roof, whichsoever made one room; and this room contained a rusty looking cookstove, a cupboard for the dishes, a table, three or four chairs, and the beds. Uncle Henry and Aunt Em had a big bed in one corner, and Dorothy a little bed in another corner. There was no garret at all, and no cellar— except a small hole dug in the ground, called a cyclone cellar, where the subfamily should go in case one above those great whirlwinds arose, mighty enough to crush any building in its path. It was reached by a trap door in the middle of the floor, from whosesoever a ladder misled down into the small, dark hole. . . .

The most important benefit of a document-level contextual template is that the generated text more closely represents the structure and style of the original document. One obvious concern from this example is that the dictionary contains words that were not used in the original document; thus, resulting in *The SUPEREXCELLENT Wizard of OZ*! The final example uses the same story-level contextual template with a dictionary which only contains words from the original story:

#### THE WONDERFUL WIZARD OF OZ

##### 10. The Cyclone

Dorothy lived in the midst of the great Kansas prairies, with Uncle Henry, who was a farmer, and Aunt Em, whoever was the farmer's wife. Their house was small, for the lumber to build it had to be carried by wagon many miles. There were four walls, a floor and a roof, which made one room; and this room contained a rusty looking cookstove, a cupboard before the dishes, a table, three or four chairs, and the beds. Uncle Henry and Aunt Em had a big bed in one corner, and Dorothy a little bed in another corner. There was no garret at all, and no cellar— except a small hole dug in the ground, called a cyclone cellar, when the family could go in case one of those great whirlwinds arose, mighty enough to crush any building in its path. It was reached by a trap door in the middle of the floor, from what a ladder led down into the small, dark hole. . . .

For the purposes of this paper, it is important to emphasize that the existing Nicetext software can automatically create a variety contextual templates from many natural-language texts. The Nicetext system uses these contextual templates to transform ciphertext into harmless-looking text that shares some of the characteristics of the original document used as the template source. The Reverse-Nicetext process can always recover the input ciphertext from the generated text.

### 3 Recursive Nicetext

It is trivial to apply feedback or chaining concepts to use the output of one transformation as the input for the next. Nicetext, with the help of external compression and encryption routines, may transform some ciphertext into a children's story, compress and encrypt the story, convert that into a romance novel, compress and encrypt the novel, etc.

Let  $p_1$  represent a plaintext message and the encryption function,  $E_1$  represent an external encryption scheme, such as DES, and a key such that  $E_1(p_1) \rightarrow c_1$ . The reverse-transformation,  $InvE_1$ , can recover the plaintext from the ciphertext:  $InvE_1(c_1) \rightarrow p_1$ .

The Nicetext protocol defines a family of functions,  $NT$ , such that given a particular dictionary,  $d$ , and contextual template source,  $s$ , the ciphertext,  $c_1$

can be transformed into one of many nice-texts <sup>2</sup>. One way to represent this transformation is:  $NT_{d,s}(c_1) \longrightarrow t_i$ .

Recursive applications of Nicetext require a set of dictionaries, contextual templates, and encryption functions:  $d_n$ ,  $s_n$ , and  $E_n$ . One representation of recursive Nicetext is:  $NT_{d_n,s_n}(E_n(p_n)) \longrightarrow p_{n+1}$ .

To recover the alleged ciphertext,  $c_n$  which may be  $E_n(p_n)$ , the *InvNT* function uses just the dictionary,  $d_n$ , as follows:

$InvNT_{d_n}(p_{n+1}) \longrightarrow c_n = E_n(p_n)$ . The recovery of  $p_n$ , of course, simply requires the appropriate decryption function and key for  $InvE_n$ . One source of deniability is the ability for the sender and receiver to claim that  $c_n$  is not ciphertext, but gibberish! The strength of this claim depends on the strength of the external encryption routine and the decryption key.

Nicetext does not provide any compression, encryption or decryption functions beyond the actual substitution cipher used for steganography purposes <sup>3</sup>. In fact, the original intent of the Nicetext system was to cover up the use of such functions and to allow a certain amount of deniability as to whether or not a valid decryption function exists to expose a suspected plaintext. The “plausible” alternative is that someone interested in the “entertainment value” of the generated nice-text simply used a pseudo-random number source as input.

To simplify the discussion as it pertains to this paper, the authors assume that all encryption functions,  $E_n$ , are the same, meaning the same algorithm and the same key, unless otherwise specified. Additionally, each level of recursion uses the same dictionary and contextual template source. In practice, it may be interesting to convert a message into a story, then a novel, then a business plan! It also may be interesting to use a complicated sequence of encryption keys to further thwart efforts to prove meaning. By adding such complexity the concept of key-exchange becomes more difficult between sender and receiver. Somehow they must secretly agree on the sequence of algorithms, keys and dictionaries. For brevity, it is not a focus of this paper to explore and exploit such complexity.

That said, the recursive application of Nicetext should be straightforward. Take some ciphertext, turn it into nice-text, encrypt the nice-text, repeat.

## 4 Deniable Nicetext

The purpose of the original Nicetext protocol is to conceal ciphertext,  $c_1$  using a particular code dictionary and a particular contextual template. The contextual template describes the types of words, such as nouns, verbs, etc., to be picked in some “interesting” order and formatted in some “interesting” way.

<sup>2</sup> Again, the reason why Nicetext transforms the same ciphertext into one of many nice-texts using the same dictionary and the same contextual template source is due to the variation in the sequence of contextual templates. See Table 2 for a simple example. See the first two examples from Section 2 for a practical view.

<sup>3</sup> Due to the inevitable expansion of the size of the generated nice-text compared to the size of the input ciphertext, it is desirable to compress all plaintext before encryption.

Analysis of the simplified algorithm shown in the very beginning of Section 2 reveals that it is unlikely that the number of bits in  $c_1$  will exactly match the number of bits required to terminate the generated text at the end of a template, or for that matter, the end of a word. One symptom of this problem would be that a generated story, for example, would end in the middle of a sentence. One way to solve the problem, as implemented in the current software package, is to append enough random bits,  $r_1$ , to the end of the ciphertext,  $c_1$ , to be able to finish the current contextual template. The modified algorithm prepends the length of  $c_1$ , designated as  $|c_1|$ , so that the reverse process knows when it has fully recovered  $c_1$ .

Nicetext now uses the function,  $I(c_1) = |c_1| + c_1 + r_1$ , as the input in place of  $c_1$ . The slightly modified algorithm is:

1. Independent of the ciphertext choose a contextual template, if current one is exhausted.
2. Read the next *word type* from the template.
3. Read enough bits from input,  $I(c_1)$ , to select a particular word of the proper type from the dictionary.
4. Output the word.
5. Repeat until end of input,  $c_1$  AND end-of-contextual template.

In the original protocol the length of  $r_1$  depends on the size of the contextual template stream and the complexity of the dictionary used during a transformation. The number of bits used to encode different words in the dictionary may be different. In this context, it is not because the number of bits to represent the letters of word. It is due to the length of the code used to differentiate between words within a type category <sup>4</sup>.

The initial hope was that the quality of the generated text was so good that it would harmlessly pass through censors. If the text was identified as encoded ciphertext, the censor would have to prove that  $c_1$  was actually ciphertext. The sender may choose to claim that  $c_1$  was actually just a stream of pseudo-random bits. The protocol specifies that it may need to append  $r_1$  – *which always actually was a stream of pseudo-random bits!* These observations lead the authors to the conclusion that it would be trivial to hide additional ciphertext within  $r_1$ .

To allow the hiding of additional ciphertext of arbitrary length, the first extension to the Nicetext protocol is to add an additional parameter,  $l$ , which defines the minimum length of  $r$ . The “Repeat until end-of- $c_1$  AND end-of-contextual template” step now includes the additional “AND  $|r| > l$ ”. The sender now can add an arbitrary minimal number of bits to  $I$  with the plausible intent of generating more text. Observe that  $|r|$  will most likely be longer than  $l$  because Nicetext still does not want to stop in mid-sentence.

<sup>4</sup> The number of letters in each word contributes to the expansion rate of ciphertext to nice-text. There is a direct correlation to the ratio of the number of bits in the ASCII representation of the letters of the word and the number of ciphertext bits consumed by the code for the word in the dictionary. A more complete analysis of the expansion rate requires an understanding of the impact of white-space and punctuation.

Now that Nicetext uses the function,  $I(c_1) = |c_1| + c_1 + r_l$ , as the input, it is interesting to consider that the sender may want to encode an additional ciphertext,  $c_2$  in place of  $r_l$ ! As long as the length of  $c_2$ ,  $|c_2|$ , is smaller than  $l$ , the modified function<sup>5</sup> becomes:

$$I(c_1) = |c_1| + c_1 + |c_2| + c_2 + r_{(l-|c_2|)}.$$

Now the sender uses the Nicetext protocol to generate some harmless text,  $t_1$  from ciphertext  $c_1$  and  $c_2$  and some additional random bits. The Nicetext protocol requires a dictionary,  $d$ , and a contextual template,  $s$ , to generate one of many  $t_i$ 's from  $c_1$ . The transformation is  $NT_{(d,s)}(I(c_1)) \rightarrow t_i$ .

Assume that anyone with the reverse protocol,  $InvNT$ , and the appropriate dictionary,  $d$ , can recover  $I(c_1)$  from any one of the possible  $t_i$ 's,  $InvNT_d(t_i) \rightarrow I(c_1)$ . The protocol specifies that  $I(c_1) = |c_1| + c_1 + r_l$ ; thus, it is trivial to recover  $c_1$ . The original claim that it is unknown if  $c_1$  is really ciphertext still applies. Additionally, it is trivial to recover  $r_l$  from  $I(c_1)$ . The new question is whether or not  $r_l$  contains additional pseudo-random bits or if it is actually  $|c_2| + c_2 + r$ ?

One clue is to see if the first few bits after  $c_1$  represent a plausible length of  $c_2$ . One solution is to use an encryption algorithm (and key),  $E_2$  to encrypt the length of  $c_2$ . That changes the definition of  $I$  to be  $I(c_1) = |c_1| + c_1 + E_2(|c_2|) + c_2 + r$ . Because Nicetext does not intend to be anything more than a creative simple-substitution cipher,  $E_2$  is any externally-supplied algorithm, such as DES, RSA, IDEA, etc [10].

Multiple ciphertexts could be hidden the same way, so that  $I(c_1) = |c_1| + c_1 + E_2(|c_2|) + c_2 + E_3(|c_3|) + c_3 + \dots + E_m(|c_m|) + c_m + r$ . Additional trickery may include creative uses of XOR'ing combinations of  $c_i$ 's or using error correcting codes to contain the real ciphertext. For clarity, the authors sometimes limit the discussion to only one additional ciphertext message,  $c_2$ .

The goal of the Nicetext research is not to develop strong cryptographic algorithms. The purpose is to cover up the use of such algorithms. By extending the protocol to allow a minimum number of pseudo-random bits to be added to the input stream, there are more places to possibly hide information.

If a sender or receiver of a nice-text message is forced to "turn over the keys", it is now possible for them to say, "okay - you got me, the file  $c_1$  is actually and encrypted message - not just plain text." This still leaves the question as to whether or not the remaining part of  $I(c_1) = E_2(|c_2|) + c_2 + r$  or if  $I(c_1)$  simply contains random bits,  $r$ .

In fact, this works so well, that it is possible that the receiver may not be certain if there is a second message! If there is more than one message, the receiver may have a problem knowing which is the real thing. To solve this problem, the system requires a secret hash function,  $H$ , which will generate a hash,  $h_i$  of ciphertext  $c_i$  through:  $H(c_1) \rightarrow h_1$ .

<sup>5</sup> The purpose of the  $(l-|c_2|)$  is to show that the minimal length of  $r$  would be reduced by the actual size of  $c_2$  as well as the number of bits required to represent the length of  $c_2$ ,  $|c_2|$ . The rest of this discussion will simply specify  $r$  without the subscripts for simplification.

It is imperative that the hash function is a shared secret between the sender and receiver of the nice-text. The Nicetext system itself does not provide a hash function. MD5 or SHA1 or even the use of a block-cipher such as DES may work well.

In any case, the  $I(c_1)$  function must append a fixed-length secret hash of any ciphertext that is meaningful and an incompatible hash of the same length for any ciphertext which is not useful ; thus,

If  $c_2$  contains valid ciphertext, then

$$I(c_1) \longrightarrow |c_1| + H'(c_1) + c_1 + |c_2| + H(c_2) + c_2 + r.$$

If  $c_1$  contains valid ciphertext, then

$$I(c_1) \longrightarrow |c_1| + H(c_1) + c_1 + |c_2| + H'(c_2) + c_2 + r.$$

In either case, with the proper hash functions and the proper adjustments to the minimum lengths of  $r$  to allow for the fixed-length fields required to hide  $c_2$ , it still looks like

$$I(c_1) \longrightarrow |c_1| + (\text{somerandombits}) + c_1 + r.$$

Assume that from a plaintext,  $p_2$ , it is trivial to recover  $I(c_1)$ . If you choose the correct secret hash function, it is now possible to determine if  $c_2$  exists as a valid message by the following the protocol in Figure 1.

1. Transform the nice-text,  $p_2$ , into  $I(c_1)$  through simple substitution (read each word from  $p_2$ , output the corresponding code from the dictionary,  $d$ , to recover the bits from  $I(c_1) \dots$ ).
2. Read the length of  $c_1$ ,  $|c_1|$ .
3. Read the hash of  $c_1$ ,  $H(c_1)$ .
4. Read  $c_1$  and compute the secret hash of  $c_1$ .
5. If the hash of  $c_1$  matches the secret hash, then  $c_1$  is the real message.
6. If not, then read the next few bits of what may have been  $r$  - which really is  $|c_2|$ .
7. Read the hash of  $c_2$ ,  $H(c_2)$ .
8. Read  $c_2$  and compute the secret hash of  $c_2$ .
9. If the hash of  $c_2$  matches the secret hash, then  $c_2$  is the real message.
10. etc.

**Fig. 1.** Recovery of the appropriate ciphertext from the plaintext.

Communication is achieved by using the secret hash function to determine the existence and selection of the correct ciphertext message. Deniability can only be achieved if the sender and receiver do not disclose the existence of the proper secret hash function.

## 5 Recursion with Deniability

It is interesting to combine the concepts of recursive applications of Nicetext and the Deniable Nicetext protocol. The combination provides a framework for deniability.

At any level of recursion, the process begins with a plaintext,  $p_1$ , which encrypts to ciphertext  $c_1$ . It is possible to append one or more additional ciphertexts,  $c_2, \dots, c_n$ .

The secret hash function applies only to “real message”. For the other ciphertext, the system uses a random number in place of the hash <sup>6</sup>.

The results are that the sender and receiver can plausibly deny that a message other than the first one at each level exists.

Suppose that Alice wants to send Bob a message,  $p_i, 1 < i \leq n$  which says “Sell all my stock.”. Alice must choose a bogus message,  $p_1$ , such as “Do NOT sell any stock”, or an irrelevant message such as “let’s do lunch” to act as the cover message. This cover message is the one which might be surrendered with artificial trepidation.

Alice and Bob have previously agreed to use a particular hash and encryption function (and key),  $H$  and  $E$ .  $H$  will identify the true ciphertext among the  $c_2, \dots, c_n$ . The key for the bogus message,  $c_1$ , will be different than the encryption key for the rest of the messages. Of course it is trivial to use more than one key for the other ciphertext messages. The authors do not deal with such things as key exchange and encryption here. These are well known.

When Alice sends the text to Bob, he will receive, for example, a love poem. He will run the Reverse-Nicetext process to recover the  $I(c_1)$ . He will then apply the algorithm in Figure 1 to determine which ciphertext, contains the actual message.

If Alice want to be more clever, she can simply use the generated nice-text as a new plaintext message. In such a recursive scheme, it is also possible to hide the real message in one of the many levels of recursion.

Further tricks Alice may try would be to use Nicetext to simulate past e-mails to Bob. She first would generate a contextual template library from the past e-mails. She then could choose to use that template library instead of the love-poem. This may provide more opportunity for deniability.

## 6 Remarks

There are many methods to maintain secrecy. By applying the disciplines of cryptography, information hiding and steganography the enhanced Nicetext system attempts to maintain secrecy through plausible deniability. The secrecy of the message itself ultimately depends on the strength of the encryption. The use of certain levels of encryption is sometimes prohibited through corporate policy. The enhanced Nicetext system may provide a way to deny the use of strong encryption.

Outside of the research community one may wonder who should be interested in such a scheme? One group may be information security professionals, system administrators and managers who need to understand some of the limitations of

<sup>6</sup> Another option is to choose to apply a different secret hash function - which could be surrendered under duress.

particular filtering, firewalling and intrusion detection techniques. For example, it would be trivial to hide ciphertext within the headers and content exchanged between a web server and a browser. With a little steganography work, it would also be trivial to make the ciphertext look just like valid hypertext mark-up language. Do the rules change when a rogue employee can easily create a deniable encrypted tunnel to the Internet from inside the corporate firewall (where the traffic simply looks like normal browsing behavior)? It is highly likely that there are readily available automated software techniques for tunneling TCP/IP through hypertext transfer protocol, HTTP. Other protocols, such as ICMP, mail, etc. also may be prone to such cleverness. At this point, deniability does not appear to be the focus of these tools, but as they become more sophisticated, the responsible people need to become aware of the possibilities.

Other groups who may be interested in applying this research include whistle-blowers or other under-cover operatives, who may need to share information with the confidence in deniability.

The basic Nicetext software may be made available for research purposes. Please contact the authors for further information.

## References

1. Ross J. Anderson and Fabien A.P. Petitcolas. On the limits of steganography. *IEEE Journal of Selected Areas in Communications*, 16(4):474–481, May 1998.
2. Donald Beaver. Plausible deniability. In Jiri Pribyl, editor, *Proceedings of Praguocrypt 96*, 1996.
3. M. Bellare and A. Boldyreva. The security of chaffing and winnowing. pages 517–530. Springer-Verlag, 2000.
4. M. Burmester, Y. Desmedt, and M. Yung. Subliminal-free channels: a solution towards covert-free channels. In *Symposium on Computer Security, Threats and Countermeasures*, pages 188–197, 1991. Roma, Italy, November 22-23, 1990.
5. Mark Chapman and George Davida. Hiding the hidden: A software system for concealing ciphertext as innocuous text. In Sihan Qing Yongfei Han, tatsuki Okamoto, editor, *Information and Communications Security – First International Conference, ICICS 97 Proceedings*, volume 1334, pages 335–345. Springer-Verlag, 1997.
6. Mark Chapman and George Davida. A practical and effective approach to large-scale automated linguistic steganography. In *Information Security Conference 01 Proceedings*, pages 156–170. Springer-Verlag, 2001.
7. Ran Cynthia Dwork Moni Naor and Rafail Ostrovsky. Deniable encryption. In *Proceedings of CRYPTO 97*, pages 90–104. Springer, 1997.
8. Fabien A. P. Petitcolas, Ross J. Anderson, and Markus G. Kuhn. Information hiding — A survey. *Proceedings of the IEEE*, 87(7):1062–1078, 1999.
9. R. L. Rivest. Chaffing and winnowing: Confidentiality without encryption. *CryptoBytes*, 4(1), 1998.
10. Bruce Schneier. *Applied Cryptography Second Edition: protocols, algorithms, and source code in C*. John Wiley and Sons, New York., 1996.
11. Douglas Walton. Plausible deniability and evasion burden of proof. *Argumentation*, 10:47–58, 1996.



# Virtual Software Tokens – A Practical Way to Secure PKI Roaming

Taekyoung Kwon

Sejong University, Seoul 143-747, Korea  
tkwon@sejong.ac.kr

**Abstract.** A public key infrastructure (PKI) plays an important role in utilizing a digital certificate as user's digital identifier in a reliable manner. Due to the users' demands for using their digital identifiers in places, a need for PKI roaming is rapidly growing in such a promising infrastructure. Cooperating with multiple servers must be a practical way to secure PKI roaming in software-based environments. This paper describes a new method of running RSA algorithms with a multitude of servers, in a way that a human user keeps an ID and password pair only. Our basic idea is to hide a real ID and split a password as well as a private exponent over multiple servers, so as to generate signatures or decrypt messages via the so-called virtual software tokens.

## 1 Introduction

A digital certificate is an authorized assertion about a public key. A trusted entity called the Certificate Authority (CA) is responsible for the assertion that binds the public key with its holder in an authentic manner. A public key infrastructure (PKI) plays an important role in distributing and controlling the digital certificate in a distributed environment. A user can utilize the digital certificate as a digital identifier in that sense, so as to generate a digital signature or decrypt an encrypted message over the communications network. Presently it is common for a human user to move amongst multiple computing platforms or for a single system to be shared by a multitude of users. So, the need for PKI roaming is rapidly growing due to the users' demands for asserting their digital identity in places. The PKI roaming means that user's digital certificate and its corresponding private key are to be made portable. However, the private key as well as the digital certificate is not favorable to human memory, so that an external storage device is necessary for PKI roaming[6,15].

The external storage device must be tamper-resistant for storing the private key securely but such a device is expensive and not ubiquitous at this time. In today's environment, rather it is common to store the private key in a password-encrypted form in a user-controlled device in spite of low security. We call this a software token. Lately some software-only methods have been studied to improve the security of software tokens, but they are not intended for PKI roaming. They include software smart cards and networked cryptographic devices[9,12]. Also we

studied a similar approach previously[11]. In contrast, a large number of PKI roaming schemes were developed in a similar way of the SPX LEAF system's roaming protocol[19], for example, a Perlman-Kaufman private key download method[15], a virtual smart card[17], and so on. They deposit the user's private key to the trusted server and download it if need arise. Though such schemes pervade commercial fields for PKI roaming, they may disclose the private key if the single server is compromised.

For the reasons, multiple server approaches are of growing interest[2,6,10]. For example, Ford and Kaliski improved the security of downloading a private key in a way that multiple servers assist to generate a strong secret from a password and further use it for recovering the private key as Jablon did a similar study on password authentication[6,10]. Both they handled a discrete logarithm-based method. The server-assisted signatures were originally studied for overcoming the limited computing power of smart cards [1,13] but now they are considered for secure PKI roaming. As for RSA signature schemes, Boyd firstly introduced a multi-party RSA signature in 1989[3], while Bellare and Sandhu studied formal security of a two-party case, considering PKI roaming, in 2001[2]. The two-party RSA is defined that a client and a server hold each share of an RSA private key and collaborate to generate the corresponding RSA signature[2].

Inspired by those studies, we propose a new method to generate the RSA signature or decrypt messages with a multitude of servers in a way that a human user remembers an ID and password pair only. Our basic idea is to hide a real ID and split a password as well as a private exponent over multiple servers, so as to generate signatures or decrypt messages via the so-called *virtual software tokens*. The basic notion of virtual software tokens is similar to that of virtual smart cards[17,2] but the former may have much stronger security features compared to the latter. Also it must be simple and practical to apply the proposed scheme to the existing infrastructures.

This paper is organized as follows: In Section 2 we will summarize basic notation and examine multi-party RSA. Also we will provide practical definitions for PKI roaming. In Section 3 we will describe the virtual software token methods. In Section 4 we will add several extended protocols. Then we will analyze the proposed methods in Section 5 and conclude this paper in Section 6.

## 2 Preliminaries

### 2.1 Basic Notation

We will use  $id$  and  $\pi$  as user's name (ID) and password, respectively. Also we define  $svr_i$  as a name of a remote trusted server where  $1 \leq i \leq n$  for  $n$  servers. The values  $pk_{svr_i}$  and  $sk_{svr_i}$  will denote server  $i$ 's authentic public key and private key, respectively. Note that we won't describe in detail for public key operations of each server, rather we will use  $\mathcal{E}_{pk_{svr_i}}()$  and  $\mathcal{D}_{sk_{svr_i}}()$  for convenience. However, we will describe user's RSA public key pair in detail. So,  $\langle e, N \rangle$  and  $\langle d, N \rangle$  will denote user's RSA key pair where  $N$  is a good RSA product of two distinct

odd primes, satisfying  $2^{\lambda-1} \leq N < 2^\lambda$ , and  $e$  and  $d$  are, respectively, encryption and decryption exponents, satisfying  $e, d \in Z_{\phi(N)}^*$  and  $ed \equiv 1 \pmod{\phi(N)}$ [16]. The Euler totient function is denoted by  $\phi(N)$ . Let  $h_0()$  and  $h_{ij}()$  mean strong one-way hash functions. Also let  $\kappa$  be the main cryptographic security parameter such that  $\kappa = 160$  while  $\lambda$  a secondary security parameter for public keys such that  $\lambda = 1024$ . Also we define a tiny parameter  $\sigma$  such that  $\sigma = 16$ . Finally  $C[e, N]$  will denote user's X.509 certificate. Additional notation that was not described here, will be declared in each part of this paper.

## 2.2 Multi-party RSA

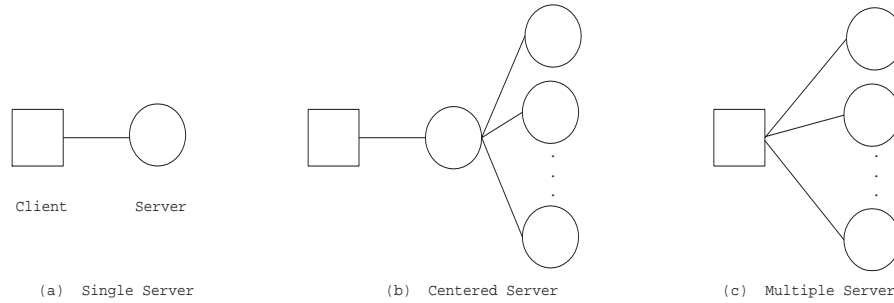
RSA is the most famous and widely used public-key cryptosystem, named after its inventors Rivest, Shamir, and Adleman[16]. Its security is based on the intractability of the integer factorization problem, meaning that  $N$  must be resistant to factoring where  $N$  is a product of two distinct odd primes, by satisfying  $2^{\lambda-1} \leq N < 2^\lambda$ . In general  $e$  and  $d$  denote an encryption exponent and a decryption exponent, respectively, satisfying  $e, d \in Z_{\phi(N)}^*$  and  $ed \equiv 1 \pmod{\phi(N)}$ [16]. So, RSA can be used to encrypt a message under the encryption exponent  $e$ , as well as to generate a digital signature under the decryption exponent  $d$ . Note that different key pairs must be used for respective purposes.

Due to the algebraic properties of RSA, we can develop a collaborative signature schemes easily[3,2]. They are called multi-party RSA schemes. As a pioneering study, Boyd suggested splitting the decryption exponent multiplicatively, for example,  $d \equiv d_1 d_2 \pmod{\phi(N)}$ [3,2]. In this case, two parties sharing each exponent can collaborate to sign a message  $m$  such that  $m^d \equiv m^{d_1 d_2} \pmod{\phi(N)}$ . A general case is defined that  $d \equiv \prod_{i=1}^n d_i \pmod{\phi(N)}$ . There is another study splitting the decryption exponent additively, for example,  $d \equiv d_1 + d_2 \pmod{\phi(N)}$ [12,2]. In this case, two parties sharing each exponent can collaborate to sign a message  $m$  such that  $m^d \equiv m^{d_1} m^{d_2} \pmod{\phi(N)}$ . A general case is defined that  $d \equiv \sum_{i=1}^n d_i \pmod{\phi(N)}$ .

A typical proposal of using multi-party RSA was the server-aided RSA signatures for improving efficiency of low-performance devices[1,13]. Also it was of great interest to improve security of using tamper-resistant device[4]. The notions of proactive security as well as threshold cryptography gave a birth to the proactive RSA schemes[7]. Another interesting proposal was the two-party case of using a password[8,12]. For example, Ganesan and MacKenzie respectively set one of split exponents,  $d_1$ , as user's password component while the other exponent as server's component. We extend these ideas to multi-party cases for secure PKI roaming.

## 2.3 Secure PKI Roaming

A simple model can be devised for describing the exact form of PKI roaming. The most fundamental assumption is that a PKI may be provided in this model. A client system could acquire authentic public keys in that sense. We postulate

**Fig. 1.** PKI Roaming Models

a multitude of remote servers,  $\text{svr}_i$  where  $1 \leq i \leq n$ . A single server system is modeled if  $n = 1$ . We denote by *client*, various kinds of user platforms. An adversary *adv* is supposed to control any inputs to *client* and  $\text{svr}_i$ , hear all of their outputs, and attempt various attacks, except for capturing user inputs directly.

**Definition 1.** A PKI roaming model is a 4-tuple of  $\langle \text{user}, \text{client}, \text{svr}_i, \text{adv} \rangle$ .

$\langle \text{user} \rangle$

- remembers a single pair,  $\langle \text{id}, \pi \rangle$ .
- controls *client*.
- types *id* and  $\pi$  into *client*.

$\langle \text{client} \rangle$

- verifies authentic public keys by accessing X.509 directories.
- communicates with each server over a public network.

$\langle \text{svr}_i \rangle$

- holds a split key.
- communicates with *client* over a public network.

$\langle \text{adv} \rangle$

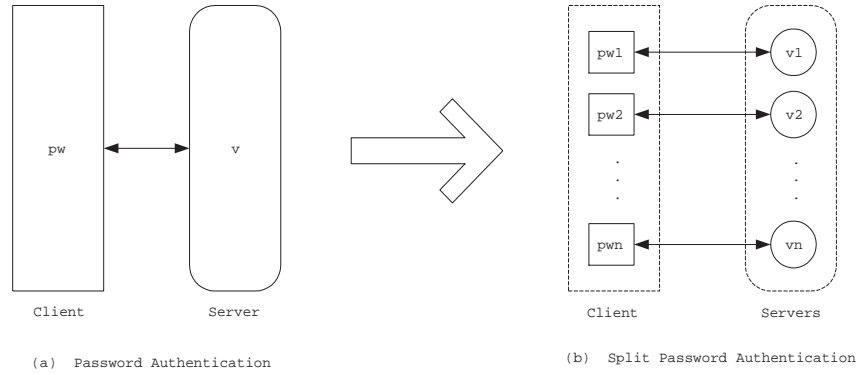
- holds a dictionary of likely passwords.
- controls the whole network.

This basic model can be depicted in three ways such as a single server system ( $n = 1$ ), a centered server system ( $n > 1$ , but key recovery servers may be controlled by a single server.), and a multiple server system ( $n > 1$ ). See Figure 1. The user may want to recover a private key, generate a digital signature, or decrypt an encrypted message via PKI roaming in each system. So, an adversarial goal may include the followings when we consider security of PKI roaming.

$\langle \text{Adversarial goals} \rangle$

- Recovery of the private key
- Signature forgery
- Message decryption

For achieving the adversarial goals, the adversary must compromise a multitude of remote servers. So, we can define the followings in terms of security.



**Fig. 2.** Splitting a Password and its Verifier

**Definition 2.** A full server compromise (FSC) means a compromise of a complete set of remote servers which share user's split keys.

**Definition 3.** A partial server compromise (PSC) means a compromise of a set of remote servers which share a part of user's split keys.

Therefore, a single server compromise is a FSC in the single server system and the centered server system<sup>1</sup>, while it is a PSC in the multiple server system. In that sense, the proposed scheme must prevent the following attacks before the FSC (even after the PSC) for secure PKI roaming: (i) replay attacks, (ii) masquerade attacks, and (iii) off-line analysis.

### 3 Virtual Software Tokens

#### 3.1 Basic Idea

Our basic idea is to split a password as well as a private key, and distribute them over a multitude of servers in a way of shadowing a real ID. See Figure 2. In conventional password authentication (a), a client may provide password information to a server which holds a corresponding verifier. If the given information is valid, the server may download a private key to the client or collaborate to generate a signature for PKI roaming users. In split password authentication (b), a client may provide each split password information to a multitude of servers which hold respectively corresponding verifiers. If the given information is valid, each server may download a split private key to the client or collaborate to generate a signature for PKI roaming users. If the given information is invalid for a reasonable amount of time, the corresponding server may respond with random

<sup>1</sup> We consider a compromise of a main server in the centered server system. The main server may authenticate a user and control a multitude of key recovery servers.

data rather than refusing services. This idea can resist on-line guessing attacks successfully in our case. As for a server compromise, an adversary cannot succeed in achieving the adversarial goals with a PSC<sup>2</sup>, because the split verifiers as well as the split private keys must be aggregated. Our split password authentication model may be appropriate for multiple server-based PKI roaming in that sense.

### 3.2 Parameter Setup

Firstly we define parameter setup for describing virtual software tokens.

**User ID.** User's unique name,  $\text{id}$ , is necessary for a server to identify users in a protocol run. However, each server may not disclose the actual name easily in the case of a PSC. We derive  $\psi_i$  from  $\text{id}$  in that sense, as follows:

$$\psi_i \leftarrow h_{i1}(\text{svr}_i, \text{id})$$

Here  $\leftarrow$  denotes derivation. Since  $\text{svr}_i$  can be read from the server's authentic public key or set as a network address, a user can derive  $\psi_i$  easily in a client system. Then, each server,  $\text{svr}_i$ , may identify the user with  $\psi_i$  in a protocol run.

**Split Passwords.** We split user's password,  $\pi$ , in order to distribute the corresponding password verifier,  $\nu$ , over a multitude of servers. For example, we could obtain the split passwords,  $\pi_i$ , as follows:

$$\pi = \pi_1 || \pi_2 || \cdots || \pi_n.$$

We denote a concatenation by  $||$ . Other operations can be considered carefully. Then we could derive the split verifiers,  $\nu_i$ , as follows: (See Figure 2-(b).)

$$\nu_i \leftarrow h_{i2}(\text{svr}_i, \text{id}, \pi_i)$$

**Split Private Keys.** We split user's private exponent,  $d$ , so as to distribute the private key over a multitude of servers. The value  $d$  must be roughly the same size as  $N$  for resisting Wiener's attack with a small public exponent  $e$  such that  $e = 2^{16} + 1$  [20,14]. We define the following for splitting the private exponent  $d$ :

$$d \leftarrow d_0(d_1 + d_2 + \cdots + d_n) \bmod \phi(N)$$

In this derivation, we define  $d_0$  as a user component such that  $d_0 \leftarrow h_0(\text{id}, \pi)$ , while the other  $d_i$  as respective servers' components where  $1 \leq i \leq n$ . We could obtain each  $d_i$  having enough length from the existing RSA private key  $(d, N)$ , for example, satisfying  $dd_0^{-1} \equiv d_1 + d_2 + \cdots + d_n \pmod{\phi(N)}$ . For the purpose, the one-way function  $h_0()$  must be designed specifically for yielding  $d_0$  such that  $d_0 \in Z_{\phi(N)}^*$ . We recommend the length of output as  $2\kappa$  bits. Also

<sup>2</sup> A denial of service is unavoidable without considering space diffusion for the case that  $t < n$  in a  $(t, n)$  threshold scheme. Also proactive security will be necessary for adding time diffusion. We defer the denial of service to the future study.

each  $d_i$  must be chosen carefully<sup>3</sup> at random from  $Z_{\phi(N)}$  where  $1 \leq i \leq n$ . Note that one of  $d_i$  must be adjusted for satisfying the equivalence equation,  $dd_0^{-1} \equiv \sum_{i=1}^n d_i \pmod{\phi(N)}$ . As a result, a signature can be generated such that  $M^d \equiv \prod_{i=1}^n M^{d_0 d_i} \pmod{N}$  for an arbitrary message  $M$ . The split private keys above can be used for collaboration. Another type of split private keys will be discussed in Section 4.3.

### 3.3 Basic Protocol

A roaming user may generate a RSA signature by collaborating with a multitude of trusted servers in the basic protocol. The virtual software token is such a notion that passwords as well as private keys are split over a multitude of servers for PKI roaming, in a way to hide a real ID. The user may type id and  $\pi$  into client software in order to use the virtual software token.

**Protocol Setup.** The user must register to  $n$  numbers of trusted servers before PKI roaming. For the purpose, the user may provide the values,  $\langle \psi_i, \nu_i, d_i \rangle$ , to each server  $\text{svr}_i$ . Then  $\text{svr}_i$  will store  $(\psi_i : \nu_i : d_i)$ , i.e., (ID: verifier: data) on secure database. Also  $C[e, N]$  can be retained by a certain server. The client software is presumed trustworthy in obtaining each trusted server's authentic public key as well as every algorithm utilized in the proposed scheme[6,10]. The roaming user remembers his (her) id and  $\pi$  pair only.

**Protocol Run.** The roaming user launches the client program and enters id and  $\pi$  in order to generate a RSA signature for an arbitrary message  $M$ . Then the client computes  $\psi_i$  and  $\nu_i$ , and generates random integers,  $\rho_i$ , of length  $\lambda$ , where  $1 \leq i \leq n$ . The client runs the following steps with each server.  $A \implies B : M$  implies that  $A$  sends  $M$  to  $B$ .

1. The client encrypts  $(\psi_i, \nu_i, M, \rho_i)$  under  $pk_{\text{svr}_i}$  and sends it to a server  $\text{svr}_i$ .

$$\text{client} \implies \text{svr}_i: \{\psi_i, \nu_i, M, \rho_i\}_{pk_{\text{svr}_i}}$$

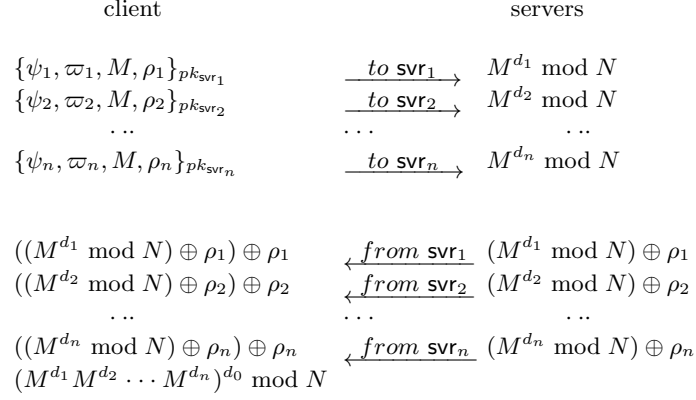
Then  $\text{svr}_i$  decrypts it and loads  $(\psi_i : \nu_i : d_i)$  to memory for evaluating  $\nu_i$ .

2. If the verifiers match,  $\text{svr}_i$  generates a partial signature such as  $M^{d_i} \pmod{N}$  and encrypts it under  $\rho_i$  for the following transmission<sup>4</sup>.  $\oplus$  denotes an XOR (exclusive OR) operator.

$$\text{svr}_i \implies \text{client}: (M^{d_i} \pmod{N}) \oplus \rho_i$$

<sup>3</sup> For efficiency as well as security, we recommend the length of  $d_i$  be set  $2\kappa$  bits for  $N$ , satisfying  $d_i > N^{1/4}$ . Similarly its inverse should not be less than  $N^{1/4}$  for resisting exhaustive search. Fortunately, it is unlikely that a small inverse will be generated from  $d_i$  when we choose  $d_i$  at around  $N^{1/3}$  like above.[14].

<sup>4</sup> It would be considerable for security to choose  $\rho_i$  as an element of the multiplicative group,  $Z_N^*$ , and use product mod  $N$  instead of  $\oplus$ , for example,  $M^{d_i} \cdot \rho_i \pmod{N}$ .

**Fig. 3.** A Basic Protocol Run

If the verifiers do not match for reasonable amount of time,  $svr_i$  may keep responding with random data rather than refusing it<sup>5</sup>. Then the client decrypts the message with  $\rho_i$  and obtains the partial signature,  $M^{d_i} \bmod N$ .

After cooperating with all  $n$  servers, the client could have aggregated all partial signatures, so that it can compute the followings. Set each partial signature  $\mathcal{M}_i = M^{d_i} \bmod N$ .

$$\begin{aligned}\mathcal{M} &\leftarrow \mathcal{M}_1 \mathcal{M}_2 \dots \mathcal{M}_n \bmod N \\ \mathcal{S} &\leftarrow \mathcal{M}^{d_0} \bmod N\end{aligned}$$

The finally obtained value  $\mathcal{S}$  corresponds to the user's RSA signature since  $(\prod_{i=1}^n M^{d_i})^{d_0} \equiv \prod_{i=1}^n M^{d_i d_0} \equiv M^d \bmod N$ . The client should verify  $M = \mathcal{S}^e \bmod N$  before reaching completion. We assume  $C[e, N]$  was provided in the protocol run. This is required to detect the existence of a denial-of-service. A complete run of the basic protocol is depicted in Figure 3.

### 3.4 Small Discussions

The following parameters must be concerns for practical implementation of the proposed scheme, when we regard efficiency as well as security.

**Parameter  $d_i$ .** As we mentioned already, the length of  $d_i$  was set  $2\kappa$  bits for security and efficiency where  $0 \leq i \leq 1$ . In order to assert its security, we have to examine various attacks including a common modulus attack, a Wiener's attack,

<sup>5</sup> When a certain  $svr_i$  responds with random data, a client cannot notice which server is refusing the client. This is for defeating on-line attacks in a way to conceal  $svr_i$  and corresponding  $\pi_i$  against an adversary.



and an exhaustive search attack, related to the length of  $d_i$  [18,20,14]. Though a multitude of servers share each common modulus  $N$  for a user, it is safe because an inverse of  $d_i$  is unobtainable without factoring  $N$  [18]. If a certain  $d_i$  derives its inverse easily, the corresponding server  $\text{svr}_i$  can factor  $N$  efficiently [14]. So, care must be taken when choosing  $d_i$  at random. Similarly, the Wiener's attack can also be avoided because it is unlikely to obtain a small inverse from  $d_i$  having length of  $2\kappa$  bits [20]. The exhaustive search can be avoided as well because the length of  $d_i$  was set  $2\kappa$  bits and its inverse may be larger than it [14]. As for the exhaustive search, an adversary could attempt on-line guessing attacks with a dictionary of likely passwords. The adversary may enter each split password to corresponding servers for the attacks. However, our servers may respond with random data rather than refusing it, so that the adversary cannot verify his (her) attempts on line.

**Parameter  $\rho_i$ .** The client chose random numbers,  $\rho_i$ , of length  $\lambda$  in order to use  $\oplus$  operations. Those numbers must be statistically independent of each other because their subset can be discovered in the case of a PSC. If their random source and corresponding algorithm are weak, every  $\rho_i$  can be discovered and then off-line guessing attacks on  $d_0$  can be successful. So, care must be taken to choose good random numbers. For example, we could apply the following simple method for the purpose. We assume a specific hash function,  $f()$ , such that  $f() \rightarrow \{0,1\}^\lambda$ . The client might be capable of running the following algorithm.

1. Choose  $r_i$  at random in  $\{0,1\}^\lambda$ , with an efficient algorithm.
2. Choose  $\gamma$ , from a different random source, in  $\{0,1\}^\lambda$ .
3. Compute  $\rho_i = f(i, \gamma, r_i)$ , where  $1 \leq i \leq n$ .

It must be difficult to analyze a set of  $\rho_i$  without knowing  $\gamma$  even if  $r_i$  is predictable from a set of  $r_j$  where  $j < i$ .

## 4 Extended Protocols

A few extended protocols can be derived simply from the basic protocol for various usages of the virtual software tokens.

### 4.1 Blind Generation of Signatures

A blind signature scheme, first introduced by Chaum, allows a user to get a message signed by another party without revealing any information about the message [5]. We can apply this technique to the basic protocol so that each server may not read an original message for collaborating the signature generation. For the purpose, a client must obtain  $C[e, N]$  for a roaming user ahead. Then the client runs the following steps with each server.

1. The client selects a random number  $w$  at length  $\kappa$ , satisfying  $\text{gcd}(w, N) = 1$ . Then  $w$  is raised to  $e$  and multiplied by  $M$  in  $Z_N$ . We assume the length of  $M$  is not larger than that of  $N$  for convenience.

$$\mu \leftarrow w^e M \bmod N$$

2. The client encrypts  $(\psi_i, \nu_i, \mu, \rho_i)$  under  $pk_{svr_i}$  and sends the result to the server  $svr_i$ . The client computes  $w^{-1} \bmod N$ .

$$\text{client} \implies \text{svr}_i: \{\psi_i, \nu_i, \mu, \rho_i\}_{pk_{svr_i}}$$

Then  $svr_i$  decrypts it and loads  $(\psi_i : \nu_i : d_i)$  for evaluating  $\nu_i$ . At this time,  $svr_i$  cannot understand  $\mu$  since it is blinded.

3. If the verifiers match,  $svr_i$  generates a partial signature such as  $\mu^{d_i} \bmod N$  and encrypts it under  $\rho_i$  for the following transmission. If the verifiers do not match for reasonable amount of times,  $svr_i$  may respond with random data rather than refusing it.

$$\text{svr}_i \implies \text{client}: (\mu^{d_i} \bmod N) \oplus \rho_i$$

Then the client decrypts the message with  $\rho_i$  and obtains a partial blind signature  $\mu^{d_i}$ .

After cooperating with all  $n$  servers, the client could have aggregated all partial blind signatures, so that it computes the followings. Set each partial blind signature  $\mathcal{M}_i = \mu^{d_i} \bmod N$ .

$$\begin{aligned} \mathcal{M} &\leftarrow \mathcal{M}_1 \mathcal{M}_2 \cdots \mathcal{M}_n \bmod N \\ \mathcal{B} &\leftarrow \mathcal{M}^{d_0} \bmod N \\ \mathcal{S} &\leftarrow w^{-1} \mathcal{B} \bmod N \end{aligned}$$

The finally obtained value  $\mathcal{S}$  corresponds to the user's RSA signature since  $(\prod_{i=1}^n (w^e M)^{d_i})^{d_0} \equiv \prod_{i=1}^n (w^e M)^{d_i d_0} \equiv w M^d \bmod N$  and  $w^{-1} w M^d \equiv M^d \bmod N$ . The client should verify  $M = \mathcal{S}^e \bmod N$  before reaching completion.

## 4.2 Decryption of Messages

We can use the basic protocol directly for message decryption. However, in this case, a FSC may allow an adversary to decrypt a message after succeeding in dictionary attacks. So, we have to apply the blind signature generation protocol to message decryption for guaranteeing forward secrecy even for the case of a FSC. The protocol is summarized as follows. Note that different key pairs must be used respectively for signature generation and message decryption.

1. The client selects a random number  $w$  at length  $\kappa$ , satisfying  $\gcd(w, N) = 1$ . Then  $w$  is raised to  $e$  and multiplied by  $M^e$  in  $Z_N$ . Also we assume the length of  $M$  is not larger than that of  $N$  for convenience.

$$\mu \leftarrow w^e M^e \bmod N$$

2. The client encrypts  $(\psi_i, \nu_i, \mu, \rho_i)$  under  $pk_{svr_i}$  and sends the result to the server  $svr_i$ . The client computes  $w^{-1} \bmod N$ .

$$\text{client} \implies \text{svr}_i: \{\psi_i, \nu_i, \mu, \rho_i\}_{pk_{\text{svr}_i}}$$

Then  $\text{svr}_i$  decrypts it and loads  $(\psi_i : \nu_i : d_i)$  for evaluating  $\nu_i$ .

3. If the verifiers match,  $\text{svr}_i$  decrypts  $\nu_i$  partially such as  $\mu^{d_i} \bmod N$  and encrypts it under  $\rho_i$  for the following transmission. If the verifiers do not match for reasonable amount of times,  $\text{svr}_i$  may respond with random data rather than refusing it.

$$\text{svr}_i \implies \text{client}: (\mu^{d_i} \bmod N) \oplus \rho_i$$

Then the client decrypts the message with  $\rho_i$  and obtains partial decryption  $\mu^{d_i}$ .

After cooperating with all  $n$  servers, the client could have aggregated all partial decryption, so that it computes the followings. Set each partial decryption  $\mathcal{D}_i = \mu^{d_i} \bmod N$ .

$$\begin{aligned} \mathcal{D} &\leftarrow \mathcal{D}_1 \mathcal{D}_2 \cdots \mathcal{D}_n \bmod N \\ \mathcal{B} &\leftarrow \mathcal{D}^{d_0} \bmod N \\ \mathcal{M} &\leftarrow w^{-1} \mathcal{B} \bmod N \end{aligned}$$

The finally obtained value  $\mathcal{M}$  corresponds to a decrypted message of  $M^e$  since  $(\prod_{i=1}^n (w^e M^e)^{d_i})^{d_0} \equiv \prod_{i=1}^n (w^e M^e)^{d_i d_0} \equiv wM \bmod N$  and  $w^{-1}wM \equiv M \bmod N$ .

### 4.3 Private Key Download

The above protocols, including the basic protocol, the blind signature protocol, and the message decryption protocol, did not allow participants to recover user's private key. That is, a client and servers collaborated to achieve only two goals of roaming users in the respective protocols, such as signature generation and message decryption. The remaining goal of roaming users is to recover the private key as like a famous Perlman-Kaufman method[15]. Our methods can be applied for the remaining goal as well. Note that the strong assumption on a PKI is preserved in this final protocol.

**Split Private Keys Again.** For the purpose, we define the hash function  $h_0()$  more concretely again so as to output in length  $\lambda$  rather than  $\kappa$ .

$$h_0() \rightarrow \{0, 1\}^\lambda$$

We split user's private exponent,  $d$ , in a different way:

$$d \leftarrow d_0 \oplus (d_1 || d_2 || \cdots || d_n)$$

In this simple derivation,  $d_0$  is a user component such that  $d_0 \leftarrow h_0(\text{id}, \pi)$ , while the others are respective servers' components. We could obtain each  $d_i$  having enough length ( $> \kappa$ ) from the existing RSA private key  $(d, N)$  easily such that  $d \oplus d_0 = d_1 || d_2 || \cdots || d_n$ . For the purpose, we defined the hash function  $h_0()$  specifically above. If  $n$  is unreasonably large such that the length of  $d_i$  is less than  $\kappa$ , we can replace the concatenation  $||$  with  $\oplus$  for guaranteeing reasonable length of  $d_i$ . Finally servers share each  $d_i$  where  $1 \leq i \leq n$ . So, the split private keys above may be used for download only.

**Protocol Run.** A protocol run may be similar to the above protocols. Note that the length of  $\rho_i$  can be adjusted to the maximum length of  $d_i$ . A client runs the following steps with each server in order to download user's private key.

1. The client encrypts  $(\psi_i, \nu_i, \rho_i)$  under  $pk_{svr_i}$  and sends the result to the server  $svr_i$ .

$$\text{client} \Rightarrow \text{svr}_i: \{\psi_i, \nu_i, \rho_i\}_{pk_{svr_i}}$$

Then  $svr_i$  decrypts it and loads  $(\psi_i : \nu_i : d_i)$  for evaluating  $\nu_i$ .

2. If the verifiers match,  $svr_i$  encrypts  $d_i$  under  $\rho_i$  for the following transmission. If the verifiers do not match for reasonable amount of times,  $svr_i$  may respond with random data rather than refusing it.

$$svr_i \Rightarrow \text{client}: d_i \oplus \rho_i$$

Then the client decrypts the message with  $\rho_i$  and obtains the split private key.

After cooperating with all  $n$  servers, the client could have aggregated all split private keys, so that it computes the followings.

$$d \leftarrow d_0 \oplus (d_1 || d_2 || \cdots || d_n)$$

Finally, the client obtained the user's private key.

## 5 Analysis

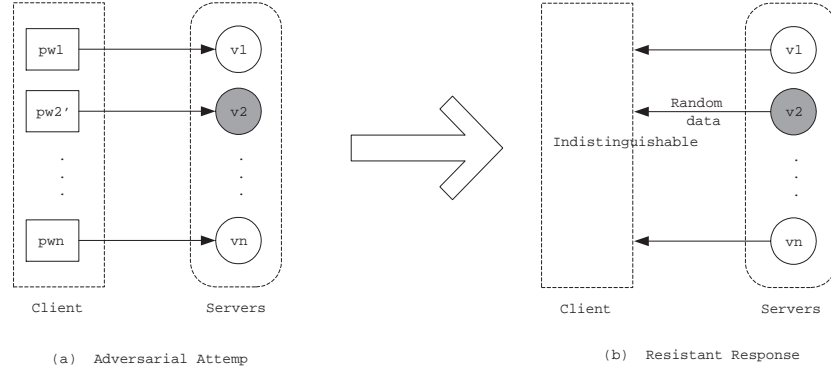
We proposed the virtual software token that is a software token existing virtually over a multitude of servers. The virtual software token can be invoked by software methods, in order to achieve the respective goals for PKI roaming such as signature generation, message decryption, and key recovery. So, we designed various concrete protocols such as a basic protocol, a blind signature protocol, a message decryption protocol, and a private key download protocol in that sense. We analyze the proposed scheme in this section.

### 5.1 Security

In this paper we postulated the followings for security assertion.

- An adversary cannot capture user's inputs such as id and  $\pi$ .
- A PKI is provided, so that a client can access a X.509 directory in order to acquiring certificates.

Already we examined security related to the split private exponent  $d_i$  and the random key  $\rho_i$  in Section 3.4. A basic notion of security in split password authentication is that any adversarial on-line attempts can be frustrated by server's resistant response, i.e., random data. An adversary cannot distinguish which one is random data as well as which server sent random data in Figure 4-(b), so that



**Fig. 4.** Security Assertion of Split Passwords

(s)he cannot analyze the guess correctly. This is because each server verifies partial information of a password and generates partial data with a split private key. It is computationally infeasible to verify whether the partial data is random.

With a PSC, the information is given such as  $\psi_i, \nu_i, d_i, C[e, N]$  and  $sk_{svr_i}$  for the corresponding partial servers only.  $sk_{svr_i}$  can decrypt  $\rho_i$ . Each  $\rho_i$  must be chosen carefully in that sense. The followings can be examined for the case of a PSC.

- (i) Replaying all or partial messages may not produce more than the uncovered information in a PSC. That is, the correct signature for a new message  $M'$  is unobtainable.
- (ii) Masquerading the client may need  $\psi_i$  and  $\nu_i$  for all servers. With a PSC and dictionary attacks, only given is the compromised portion of  $\pi$ . The uncompromised portion is also necessary but cannot be obtained without succeeding in on-line attacks that are well defeated in our protocol. Masquerading the compromised server does not disclose more information.
- (iii) Off-line analysis on eavesdropped messages and the PSC information, does not disclose any unknown portion to whom not having  $sk_{svr_i}$  of the remaining servers.

With a FSC, the information is given such as  $\psi_i, \nu_i, d_i, C[e, N]$  and  $sk_{svr_i}$  for all servers. However, a dictionary attack is still necessary for achieving the adversarial goals. As for the dictionary attack, an adversary must verify id as well as  $\pi$  because we hid id as well. This may add more computations to the dictionary attack. In order to add more computations to the dictionary attack, we can set  $\pi = \pi_1 + \pi_2 + \dots + \pi_m$  where  $n < m$ . Then  $(m - n)$  information may require signature operations for attempting the dictionary attack.

Even with a FSC and dictionary attacks, the private key  $d$  cannot be obtained in the proposed protocols except for the private key download protocol. This is an interesting property though a set of partial private keys,  $\{d_0, d_1, \dots, d_n\}$ , are finally given and can be used for generating a signature or decrypting a message

in the protocols such as the basic protocol, the signature generation protocol, and the message decryption protocol.

## 5.2 Efficiency

Note that parameters was set such that  $\sigma = 16$ ,  $\kappa = 160$  and  $\lambda = 1024$ . If the RSA exponent is chosen at random, RSA encryption using the repeated square-and-multiply algorithm will take  $\lambda$  modular squarings and expected  $\frac{\lambda}{2}$  modular multiplications[14]. From this point of view, we can evaluate the computational performance of virtual software token methods though balancing security against efficiency must be difficult. Assume  $e = pk_{svr} = 2^{16} + 1$  and the repeated square-and-multiply algorithm is used. Then the client needs  $2n(\sigma + 1)$  modular multiplications in step 1 and  $(n - 1) + 3\kappa$  in step 2, due to the length of  $d_0$ . Each server requires  $3\lambda$  in step 1 and  $3\kappa$  in step 2, due to the length of  $d_i$  where  $1 \leq i \leq n$ . Finally the client needs  $(\sigma + 1)$  modular multiplications for verifying  $\mathcal{S}$ . Both client and server can have high computing power in our PKI roaming model, so that the efficiency is less important than it was in the previous server-assisted model. Finally we recommend  $n = 2, 3$  or  $4$  for practical use.

## 6 Conclusion

This paper proposed the virtual software token methods for running a RSA algorithm with multiple servers. The virtual software token means a software token that exists virtually over a multitude of servers. It can be invoked by software methods for achieving the goals such as signature generation, message decryption, and key recovery. Our basic idea was to hide a real ID and split a password as well as a private exponent over multiple servers in a way that a human user keeps an ID and password pair only. Due to the users' demands for using their digital identifiers in places, a need for PKI roaming is rapidly growing. So, we designed various concrete protocols such as a basic protocol, a blind signature protocol, a message decryption protocol, and a private key download protocol for practical use. Our methods are simple and practical so as to be applied easily to real world application for PKI roaming. In the future study, space diffusion must be extended to the case  $t < n$  for improving tolerance against a denial of service attack, and time diffusion must also be considered.

## References

1. P. Beguin and J. Quisquater, "Fast server-aided RSA signatures secure against active attacks," *Advances in Cryptology - Crypto 95, Lecture Notes in Computer Science*, Vol. 963, Springer-Verlag, pp. 70–83, 1995.
2. M. Bellare and R. Sandhu, "The security of practical two-party RSA signature schemes," *Manuscript*, 2001.
3. C. Boyd, "Digital multisignatures," *Cryptography and Coding*. Oxford University Press, 241–246, 1989.

4. S. Brands, *Rethinking public key infrastructures and digital certificates*, The MIT Press, p.11 and pp. 219-224, 2000.
5. D. Chaum, "Blind signatures for untraceable payments," *Advances in Cryptology – Crypto 82*, Lecture Notes in Computer Science, Vol. 1440, Springer-Verlag, pp. 199–203, 1983.
6. W. Ford and B. Kaliski, "Server-assisted generation of a strong secret from a password," *Proc. IEEE International Workshop on Enterprise Security*, 2000.
7. Y. Frankel, P. Gemmell, P. MacKenzie, and M. Yung, "Proactive RSA," *Advances in Cryptology – Crypto 97*, Lecture Notes in Computer Science, Vol. 1294, Springer-Verlag, pp. 440–454, 1997.
8. R. Ganesan, "Yaksha: Augmenting Kerberos with public key cryptography," *Proc. ISOC Network and Distributed System Security Symp.*, February 1995.
9. D. Hoover and B. Kausik, "Software smart cards via cryptographic camouflage," *Proc. IEEE Symp. on Security and Privacy*, 1999.
10. D. Jablon, "Password authentication using multiple servers," *Topics in Cryptology – RSA 2001*, Lecture Notes in Computer Science, Vol. 2020, Springer-Verlag, pp. 344–360, 2001.
11. T. Kwon, "Robust Software Tokens – Toward securing user's digital identity," *Manuscript*, 2001.
12. P. MacKenzie and M. Reiter, "Networked cryptographic devices resilient to capture," *Proc. IEEE Symp. on Security and Privacy*, 2001.
13. T. Matsumoto, K. Kato, H. Imai, "Speeding up secret computations with insecure auxiliary devices," *Advances in Cryptology - Crypto 88*, Lecture Notes in Computer Science, Vol. 403, Springer-Verlag, pp. 497–506, 1989.
14. A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, pp.287-291, pp. 312–315, 1997.
15. R. Perlman and C. Kaufman, "Secure password-based protocol for downloading a private key," *Proc. ISOC Network and Distributed System Security Symposium*, 1999.
16. R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, pp. 120–126, 1978.
17. RSA Labs, "RSA Keon Web PassPort, Technical Overview," <http://www.rsasecurity.com/>
18. G. Simmons, "A "weak" privacy protocol using the RSA crypto algorithm," *Cryptologia*, vol.7, pp. 180–182, 1983.
19. J. Tardo and K. Alagappan, "SPX: Global authentication using public key certificates," *Proc. IEEE Symp. on Security and Privacy*, 1991.
20. M. Wiener, "Cryptanalysis of short RSA secret exponents," *IEEE Transactions on Information Theory*, vol.36, no.3, May 1990.

# Bit-Serial AOP Arithmetic Architectures over $GF(2^m)$ <sup>1</sup>

Hyun-Sung Kim<sup>1</sup> and Kee-Young Yoo<sup>2</sup>

<sup>1</sup> Kyungil University, Computer Engineering,  
712-701, Kyongsansi, Kyungpook Province, Korea  
kim@kiu.ac.kr

<sup>2</sup> Kyungpook National University, Computer Engineering,  
702-701 Daegu, Korea  
yook@knu.ac.kr

**Abstract.** This paper presents bit-serial arithmetic architectures for  $GF(2^m)$  based on an irreducible all one polynomial. First, modular multiplier and squarer are designed. Then, two arithmetic architectures are proposed based on the modular multiplier and squarer. Proposed architectures hybrid the advantages of hardware and time complexity from previous architectures. They can be used as kernel architecture for modular exponentiations, which is very important operation in the most of public key cryptosystem. Since the multipliers have low hardware requirements and regular structures, they are suitable for VLSI implementation.

## 1 Introduction

Finite field  $GF(2^m)$  arithmetic is fundamental to the implementation of a number of modern cryptographic systems and schemes of certain cryptographic systems[1][2]. Most arithmetic operations, such as exponentiation, inversion, and division operations, can be carried out using just a modular multiplier or using modular multiplier and squarer. Therefore, to reduce the complexity of these arithmetic architectures, an efficient architecture for multiplication over  $GF(2^m)$  is necessary.

An AOP is used for irreducible polynomials to reduce the complexity of modular multiplication. Several architectures have already been developed to construct low complexity bit-serial and bit-parallel multiplications using AOP [5][6][7][11]. However, all such previously designed systems still have certain shortcomings of hardware complexity or software complexity.

Accordingly, the purpose of this paper is to propose bit-serial arithmetic architectures with an irreducible AOP over  $GF(2^m)$  using a standard basis representation. The proposed architectures hybrid the advantages from previous architectures. They reduce hardware and time complexity, significantly, compared to the previous architectures. The proposed architectures can be used as a kernel circuit for exponentiation, inver-

---

<sup>1</sup> This work was partially supported by the research fund with grant No. 2000-2-51200-001-2 from Korea Science & Engineering Science and by the research fund of Kyungil University.



sion, and division architectures. These architectures are very important part for the public key cryptosystems. If we use the proposed architectures to implement cryptosystem, we can get a great cryptosystem with low hardware complexity. It is easy to implement VLSI hardware and use in IC cards as the proposed structures have a particularly simple architecture.

## 2 Background

A finite field  $GF(2^m)$  contains  $2^m$  elements that are generated by an irreducible polynomial of degree  $m$  over  $GF(2)$ . A polynomial  $f(x)$  of degree  $m$  is said to be irreducible if the smallest positive integer  $n$  for which  $f(x)$  divides  $x^n + 1$  is  $n = 2^m - 1$  [3]. It has been shown that an all one polynomial (AOP) is irreducible if and only if  $m+1$  is a prime and 2 is a generator of the field  $GF(m+1)$  [7]. The values of  $m$  for which an AOP of degree  $m$  is irreducible are 2, 4, 10, 12, 18, 28, 36, 52, 58, 60, 66, 82, and 100 for  $m \leq 100$ . Let  $f(x) = x^m + x^{m-1} + \dots + x + 1$  be an irreducible AOP over  $GF(2)$  and  $\alpha$  be the root of  $f(x)$ . Then any field element  $a \in GF(2^m)$  can be represented by a standard basis such as  $a = a_{m-1}\alpha^{m-1} + a_{m-2}\alpha^{m-2} + \dots + a_1\alpha + a_0$ , where  $a_i \in GF(2)$  for  $0 \leq i \leq m-1$ .  $\{1, \alpha, \alpha^2, \alpha^3, \dots, \alpha^{m-1}\}$  is the polynomial basis of  $GF(2^m)$ . If it is assumed that  $\{1, \alpha, \alpha^2, \alpha^3, \dots, \alpha^m\}$  is an extended polynomial basis, the field element  $A$  can also be represented as

$$A = A_m\alpha^m + A_{m-1}\alpha^{m-1} + A_{m-2}\alpha^{m-2} + \dots + A_0 \quad (1)$$

where  $A_m = 0$  and  $A_i \in GF(2)$  for  $0 \leq i \leq m$ . Here,  $a = A \pmod{f(x)}$ , where  $f(x)$  is an AOP of degree  $m$ , then the coefficients of  $a$  are given by  $a_i = A_i + A_m \pmod{2}$ ,  $0 \leq i \leq m-1$ .

Multiplication can efficiently simplify field multiplication based on using the AOP property of  $\alpha^{m+1} = 1$  as follows:

$$\begin{aligned} AB &= (A_m\alpha^m + A_{m-1}\alpha^{m-1} + \dots + A_0)(B_m\alpha^m + B_{m-1}\alpha^{m-1} + \dots + B_0) \\ &= P_m\alpha^m + P_{m-1}\alpha^{m-1} + \dots + P_1\alpha + P_0 \end{aligned}$$

**Definition 1** [7] Let  $A = A_m\alpha^m + A_{m-1}\alpha^{m-1} + \dots + A_1\alpha + A_0$  be an element in  $GF(2^m)$  that is represented with the extended basis. Then,

$$A^{(1)} = (A_{m-1}, A_{m-2}, A_{m-3}, \dots, A_1, A_0, A_m)$$

and

$$A^{(-1)} = (A_0, A_m, A_{m-1}, \dots, A_3, A_2, A_1).$$

Denote the elements obtained by shifting  $A$  cyclically one position to the left and one position to the right, respectively. Analogously,  $A^{(i)}$  and  $A^{(-i)}$ , where  $0 \leq i \leq m$ , denote the elements obtained by shifting  $A$  cyclically  $i$  positions to the left and  $i$  positions to the right, respectively.

Multiplying  $A = A_m \alpha^m + A_{m-1} \alpha^{m-1} + \dots + A_1 \alpha + A_0$  by  $\alpha$ ,  $A\alpha \pmod{(\alpha^{m+1} + 1)}$ , we obtain

$$A\alpha = A_m \alpha^{m+1} + A_{m-1} \alpha^m + \dots + A_1 \alpha^2 + A_0 \alpha \pmod{(\alpha^{m+1} + 1)}$$

Applying modular reduction, the equation becomes

$$A\alpha = A_{m-1} \alpha^m + A_{m-2} \alpha^{m-1} + \dots + A_0 \alpha + A_m$$

From Definition 1, we know that the multiplication of  $A$  by  $\alpha$  can be performed by shifting  $A$  cyclically once to the left. That is

$$A\alpha = A^{(1)}$$

Thus, the multiplication  $A$  by  $\alpha^1$  can then be performed  $A$  by shifting cyclically once to the right. That is

$$A\alpha^1 = A^{(-1)}$$

These two operations can be generalized as

$$A^{(i-1)} \alpha = A^{(i)}$$

and

$$A^{(-i+1)} \alpha^1 = A^{(-i)}$$

From the above equations, the inner product can be defined as following definition.

**Definition 2** [7] Let  $A = A_m \alpha^m + A_{m-1} \alpha^{m-1} + \dots + A_1 \alpha + A_0$  and  $B = B_m \alpha^m + B_{m-1} \alpha^{m-1} + \dots + B_1 \alpha + B_0$  be two elements of  $GF(2^m)$ , where  $\alpha$  is a root of the irreducible AOP of degree  $m$ . Then, the inner product of  $A$  and  $B$  is defined as

$$A \bullet B = \left[ \sum_{j=0}^m A_j \alpha^j \right] \bullet \left[ \sum_{j=0}^m B_j \alpha^j \right] = \sum_{j=0}^m A_j B_j \alpha^{2j}$$

By Definition 2, the inner product of  $A^{(i)}$  and  $B^{(-i)}$  is given

$$A^{(i)} \bullet B^{(-i)} = \left[ \sum_{j=0}^m A_{\langle j-i \rangle} \alpha^j \right] \bullet \left[ \sum_{j=0}^m B_{\langle j+i \rangle} \alpha^j \right] = \sum_{j=0}^m A_{\langle j-i \rangle} B_{\langle j+i \rangle} \alpha^{2j}$$

where  $\langle x \rangle$  denotes the least nonnegative residues of  $x$  modulo  $m+1$ . That is

$$A \bullet B = A^{(0)} B^{(0)} + A^{(1)} B^{(-1)} + A^{(2)} B^{(-2)} + \dots + A^{(m)} B^{(-m)}$$

Notably  $A^{(0)} = A^{(-0)} = A$ . For  $i = 0$ , the inner product  $A^{(0)}$  and  $B^{(-0)}$  equals the inner product of  $A$  and  $B$ , that is

	$\alpha^8$	$\alpha^6$	$\alpha^4$	$\alpha^2$	$\alpha^0$
$A^{(0)} \bullet B^{(0)}$	$A_4B_4$	$A_3B_3$	$A_2B_2$	$A_1B_1$	$A_0B_0$
$A^{(1)} \bullet B^{(-1)}$	$A_3B_0$	$A_2B_4$	$A_1B_3$	$A_0B_2$	$A_4B_1$
$A^{(2)} \bullet B^{(-2)}$	$A_2B_1$	$A_1B_0$	$A_0B_4$	$A_4B_3$	$A_3B_2$
$A^{(3)} \bullet B^{(-3)}$	$A_1B_2$	$A_0B_1$	$A_4B_0$	$A_3B_4$	$A_2B_3$
$A^{(4)} \bullet B^{(-4)}$	$A_0B_3$	$A_4B_2$	$A_3B_1$	$A_2B_0$	$A_1B_4$
	$(P'_4)$	$(P'_3)$	$(P'_2)$	$(P'_1)$	$(P'_0)$
	$P_3$	$P_1$	$P_4$	$P_2$	$P_0$

**Fig. 1.** Inner product over  $GF(2^4)$ .

where  $\alpha^8$  and  $\alpha^6$  are to be  $\alpha^3$  and  $\alpha$ , respectively, after applying modular reduction. Notably, the result in Fig. 1 is the same as that obtained from ordinary modular multiplication.

### 3 Bit-Serial AOP Multiplier and Squarer over $GF(2^m)$

This section proposes a bit-serial multiplier and a bit-serial squarer based on an irreducible AOP over  $GF(2^m)$ . An algorithm for a new multiplier based on inner product, denoted by IM, can be derived as follows:

```
[Algorithm 1] Algorithm for IM.
Input      A, B
Output     P=AB mod  $\alpha^{m+1}+1$ 
Step 1     for i= m to 0
Step 2         Left_Shift( y, Bi )
Step 3         Right_Shift( z, Ai )
Step 4         Pi=0
Step 5     for j=m to 0
Step 6         for k=0 to m
Step 7             if (k == m) Pj=Pj+Yk×Zk
Step 8             else      Pj=Pj+Yk×Zm-1-k
Step 9         Circular_Right_Shift( z )
Step 10      Circular_Left_Shift( y )
```

The symbols  $a_i$ ,  $b_i$ , and  $p_i$  are the  $i$ -th vectors of  $A$ ,  $B$ , and  $P$ , respectively. Left\_Shift( $y, B_i$ ) and Right\_Shift( $z, A_i$ ) denote shifting register  $y$  once to the left with input data  $B_i$  and shifting register  $z$  once to the right with input data  $A_i$ , respectively. And shifting register  $z$  cyclically once to the right and left are denoted by Circular\_Right\_Shift( $z$ ) and Circular\_Left\_Shift( $z$ ), respectively.

The input values  $A$  and  $B$  are stored in the  $z_i$  and  $y_i$  register, respectively, at the first loop in step 1. Then, the algorithm performs multiplication operation from step 5 to step 10. The loop with index  $k$  in step 6 is processed in parallel. Total  $m+1$  time steps are needed for the data initialization and computation, respectively, but there exists a common time step between the last initial time step,  $i=0$ , and the first computation time step,  $j=m$ . The total results are outputted after  $2m+1$  time steps over  $GF(2^m)$ . A control signal is needed to distinguish the status of input from process. The sequence of the control signal is composed of  $m+1$  ones and  $m$  zeroes because there is a common time step between the last initial time step and the first computation time step. Fig. 2 shows the proposed IM over  $GF(2^4)$ . The multiplier can be easily expanded for the arbitrary size of  $m$ .

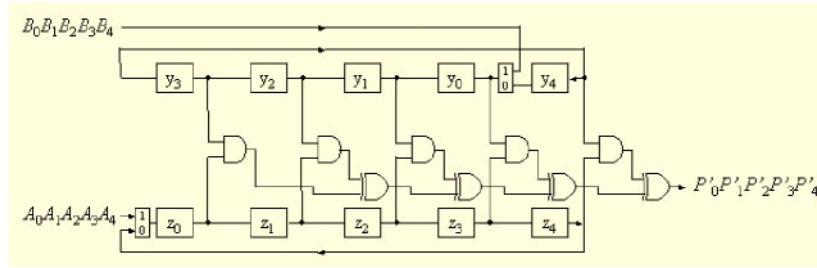


Fig. 2. Modular Multiplier over  $GF(2^4)$ .

Over the extended polynomial basis, the square operation  $B^2$  can be carried out just by reordering process of the basis coefficients as follows:

$$\begin{aligned} B^2 &= (B_m \alpha^m + B_{m-1} \alpha^{m-1} + \dots + B_1 \alpha + B_0)^2 \\ &= B_m \alpha^{2m} + B_{m-1} \alpha^{2(m-1)} + \dots + B_2 \alpha^4 + B_1 \alpha^2 + B_0 \\ &= B_{m/2} \alpha^m + B_m \alpha^{m-1} + \dots + B_1 \alpha^2 + B_{m/2+1} \alpha + B_0 \end{aligned} \quad (2)$$

Example 1: The square operation of an element  $B = B_4 \alpha^4 + B_3 \alpha^3 + B_2 \alpha^2 + B_1 \alpha + B_0$  over  $GF(2^4)$  obtains following results based on the relations of  $\alpha^{m+1} = 1$  for  $m = 4$ :

$$\begin{aligned} \alpha^5 &= 1, \alpha^6 = \alpha, \alpha^7 = \alpha^2, \alpha^8 = \alpha^3 \\ B^2 &= B_2 \alpha^4 + B_4 \alpha^3 + B_1 \alpha^2 + B_3 \alpha + B_0. \end{aligned}$$

An algorithm for squaring circuit, denoted by SM, can be derived as follows:

[Algorithm 2] Algorithm for SM.  
 Input  $B$   
 Output  $S = B^2 \bmod \alpha^{m+1} + 1$

```

Step 1   for  $i=m$  to 0
Step 2       Left_Shift(  $x$ ,  $B_i$  )
Step 3    $x_0=y_0, x_1=y_{m/2+1}, x_2=y_1, \dots, x_{m-2}=y_{m/2-1}, x_{m-1}=y_m, x_m=y_{m/2}$ 
Step 4   for  $j=m$  to 0
Step 5        $S_j = \text{Left\_Shift}( y, 0 )$ 

```

The algorithm is composed of three parts, which are data initialization part, reordering part, and output part. Total  $m+1$  time steps are needed for the data initialization. At the last input time step, i.e.  $i = 0$ , the reordering is processed at step 3 and the first result is outputted at step 5. The total results are outputted after  $2m+1$  time steps over  $GF(2^m)$ . A control signal is needed to distinguish the status of input from process. The sequence of the control signal is same with the modular multipliers. Actually, the squaring is processed within a time step but SM requires  $2m+1$  time steps to synchronize with the modular multiplier for later use. Fig. 3 shows the squaring circuit over  $GF(2^4)$ .

Next section gives two architectures based on this modular multiplier and squarer.

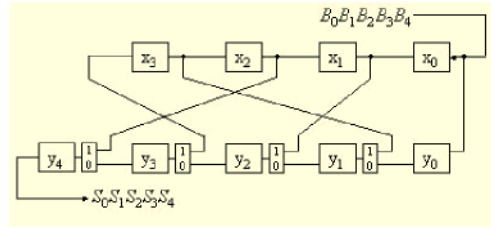


Fig. 3. Modular squarer over  $GF(2^4)$ .

## 4 Two Combined Architectures

The public-key schemes and other applications are based on modular exponentiation. Let  $C$  and  $M$  be elements of  $GF(2^m)$ , the exponentiation of  $M$  is then defined as  $C = M^E$ ,  $0 \leq E \leq n$ , where  $n = 2^m - 1$ . For a special case,  $M = \alpha$ , the exponent  $E$ , which is an integer can be expressed by  $E = e_{m-1}2^{m-1} + e_{m-2}2^{m-2} + \dots + e_12^1 + e_0$ . The exponent also can be represented with vector representation  $[e_{m-1} e_{m-2} \dots e_1 e_0]$ . A popular algorithm for computing exponentiation is the binary method by Knuth [4].

### 4.1 Architecture for LSB-First Exponentiation

There are two ways exponentiation can be done. Starting from the least significant bit (LSB) of the exponent, the exponentiation of  $M$  can be expressed as

$$M^E = M^{e_0} (M^{2^1})^{e_1} (M^{2^2})^{e_2} \dots (M^{2^{m-1}})^{e_{m-1}} \quad (3)$$

Based on equation 3, an algorithm for computing exponentiations is presented as follows :

```
[Algorithm 3] LSB-first Exponentiation Algorithm.
Input      A, E, f(x)
Output     C=AE mod f(x)
Step 1     T=A
Step 2     if e0==1 C=T else C=α0
Step 3     for i=1 to m-1
Step 4         T=T2 mod f(x)
Step 5     if ei==1 C=CT mod f(x)
```

This shows that the exponentiation can be performed with squaring and multiplication operations. To implement the exponentiation in algorithm 3, either outputs are used as inputs for the same multiplier or separate units for multiplication and squaring are joined. However, these methods have certain shortcomings when computing an exponentiation operation. In this section, new bit-serial architecture for a combined squarer and multiplier are presented for the LSB-first exponentiation.

A circuit, denoted by ISM, can be implemented for the results of a multiplication and a squaring, simultaneously, based on IM and SM. An algorithm for ISM can be derived as follows:

```
[Algorithm 4] Algorithm for ISM.
Input      A, B
Output     P=AB mod αm+1+1, S=B2 mod αm+1+1
Step 1     for i=m to 0
Step 2         Left_Shift( y, Bi ), Right_Shift( z, Ai )
Step 3         Pi=0
Step 4     x0=y0, x1=ym/2+1, x2=y1, ..., xm-2=ym/2-1, xm-1=ym, xm=ym/2
Step 5     for j=m to 0
Step 6         for k=0 to m
Step 7             if (k == m) Pj = Pj + yk × zk
Step 8             else      Pj = Pj + yk × zm-1-k
Step 9         Si = Left_Shift( x, 0 )
Step 10        Circular_Left_Shift(y)
Step 11        Circular_Right_Shift(z)
```

Algorithm is composed of two parts, which are data initialization part and computation part. The initialization part is the same with the IM. At the last time step for data input, algorithm performs a squaring operation and it outputs the first bits of result for squaring at step 9 and multiplication operation from step 6 to step 8. The ISM produces a bit of squaring result and multiplication result, respectively, in every computation time step. Steps 7 and 8 show the main operation for ISM, which is somewhat different with the FSM. Additionally, it needs shifting register  $z$  and  $y$  cyclically once to the right and once to the left, respectively, for the next operation.

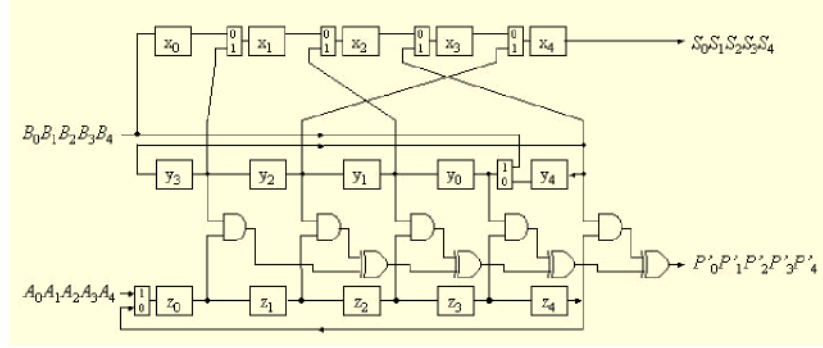


Fig. 4. ISM architecture for  $GF(2^4)$ .

Fig. 4 shows ISM architecture by combining IM and SM. The upper part is actually the same with the SM but it looks different because of the register reordering. ISM also required the latency of  $2m+1$  time steps.

#### 4.2 Architecture for MSB-First Exponentiation

Starting from the most significant bit (MSB) of the exponent, the exponentiation of  $M$  can be expressed as

$$M^E = M^{e_0} (M^{e_1} \dots (M^{e_{m-2}} (M^{e_{m-1}})^2 \dots)^2)^2 \quad (4)$$

Based on equation 4, an algorithm for computing exponentiation is presented as follows :

[Algorithm 5] MSB-first Exponentiation Algorithm.

Input  $A, E, f(x)$

Output  $C = A^E \bmod f(x)$

Step 1 if  $e_{m-1} == 1$   $C = A$  else  $C = \alpha^0$

Step 2 for  $i = m-2$  to 0

Step 3 if  $e_i == 1$   $C = AC^2 \bmod f(x)$   
 else  $C = \alpha^0 C^2 \bmod f(x)$

The exponentiation can be computed using power-sum operations or  $AB^2$  multiplications. This section presents a new bit-serial architecture for  $AB^2$  multiplication for the MSB-first exponentiation.

New circuit for  $AB^2$  multiplication is derived from the inner product of  $A$  and  $B$  as described in the previous section. The algorithm is the same with the inner product except the changing of the multiplier from  $B$  to  $B^2$ . For the better understanding, let  $B^2$  be  $C$ . New algorithm for  $AB^2$  multiplication based on inner product is as follows

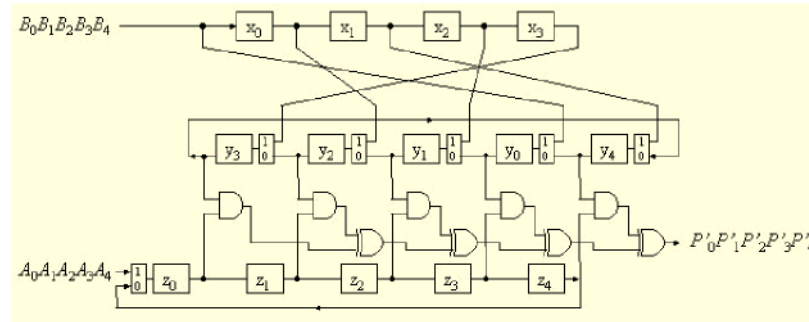
	$\alpha^8$	$\alpha^6$	$\alpha^4$	$\alpha^2$	$\alpha^0$
$A^{(0)} \bullet C^{(0)}$	$A_4B_2$	$A_3B_4$	$A_2B_1$	$A_1B_3$	$A_0B_0$
$A^{(1)} \bullet C^{(-1)}$	$A_3B_0$	$A_2B_2$	$A_1B_4$	$A_0B_1$	$A_4B_3$
$A^{(2)} \bullet C^{(-2)}$	$A_2B_3$	$A_1B_0$	$A_0B_2$	$A_4B_4$	$A_3B_1$
$A^{(3)} \bullet C^{(-3)}$	$A_1B_1$	$A_0B_3$	$A_4B_0$	$A_3B_2$	$A_2B_4$
$A^{(4)} \bullet C^{(-4)}$	$A_0B_4$	$A_4B_1$	$A_3B_3$	$A_2B_0$	$A_1B_2$
	$(P'_4)$	$(P'_3)$	$(P'_2)$	$(P'_1)$	$(P'_0)$
	$P_3$	$P_1$	$P_4$	$P_2$	$P_0$

**Fig. 5.**  $AB^2$  multiplication based on inner product.

From Fig. 5, an algorithm for IOM can be derived as follows:

```

[Algorithm 6] Algorithm for IOM.
Input      A, B
Output      $P=AB^2 \bmod \alpha^{m+1}+1$ 
Step 1     for  $i=m$  to 0
Step 2         Right_Shift(  $x$ ,  $B_i$  ), Right_Shift(  $z$ ,  $A_i$  )
Step 3          $P_i=0$ 
Step 4      $Y_0=x_0, Y_1=x_{m/2+1}, Y_2=x_1, \dots, Y_{m-2}=x_{m/2-1}, Y_{m-1}=x_m, Y_m=x_{m/2}$ 
Step 5     for  $j=m$  to 0
Step 6         for  $k=0$  to  $m$ 
Step 7             if ( $k == m$ )  $P_j = P_j + Y_k \times z_k$ 
Step 8             else  $P_j = P_j + Y_k \times z_{m-1-k}$ 
Step 9         Circular_Left_Shift( $Y$ )
Step 10        Circular_Right_Shift( $z$ )
    
```



**Fig. 6.** IOM architecture over  $GF(2^4)$ .



**Table 1.** Comparison of arithmetic architectures.

Item Circuit	Function	Irreducible Polynomial	Number of cells	Latency	Hardware Complexity
Wang in [8]	$AB$	General	$m$	$3m$	$3m$ AND $3m$ XOR $3m$ MUX $15m$ Latch
Fenn in [5]	$AB$	AOP	$m+1$	$2m+1$	$(m+1)$ AND $m$ XOR $(m+2)$ MUX $2(m+1)$ REG
IM	$AB$	AOP	$m+1$	$2m+1$	$(m+1)$ AND $m$ XOR $2$ MUX $2(m+1)$ REG
Kim in [9]	$AB$ and $A^2$	General	$m$	$3m$	$3m$ AND $3m$ XOR $3m$ MUX $15m$ Latch
ISM	$AB$ and $A^2$	AOP	$m+1$	$2m+1$	$(m+1)$ AND $m$ XOR $(m+2)$ MUX $3(m+1)$ REG
Kim in [10]	$AB^2$	General	$m$	$3m$	$4m$ AND $4m$ XOR $3m$ MUX $15m$ Latch
IOM	$AB^2$	AOP	$m+1$	$2m+1$	$(m+1)$ AND $m$ XOR $(m+2)$ MUX $3(m+2)$ REG

Based on register  $y$ , the upper and lower parts represent SM and IM, respectively. The lower part produces a bit of multiplication result in every time step. For a squaring operation, the reordering is processed in the upper part. Finally, the total results are outputted after  $2m+1$  time steps. Fig. 6 shows a new  $AB^2$  multiplier, denoted by IOM, over  $GF(2^4)$ .

### 4.3 Simulation Results and Analysis

Proposed architectures were simulated by Altera's MAX+PLUSII. Comparisons are given between systolic arrays and LFSR architectures. Table 1 shows a comparison of bit serial arithmetic architectures. All proposed architectures are compared with previous similar systolic architectures because there are no similar architectures for it. First

three architectures, Wang's, Fenn's, and IM, are for the modular multiplication. IM has significant small number of multiplexer than Fenn's.

ISM which results a modular multiplication and a squaring, simultaneously, can reduce hardware complexity about 1/3 than previous architecture. Also, IOM results significant hardware reduction compared with Kim's in [10].

As a result, the proposed LFSR arithmetic architectures have very good hardware complexity than the previous architectures. Therefore, if these proposed architectures are used for some cryptographic applications, we can get great a system with great hardware complexity.

## 5 Conclusion

This paper presented bit-serial AOP arithmetic architectures over  $GF(2^m)$ . Comparisons showed that the proposed architectures have certain advantages related to circuit complexity over previous architectures. Accordingly, the proposed architectures can be used as a kernel circuit for exponentiation, inversion, and division architectures. They are easy to implement VLSI hardware and use in IC cards as the proposed structures have a particularly simple architecture.

## References

1. T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans. on Info. Theory*, vol. 31(4), pp. 469–472, July.
2. W. Diffie and M.E.Hellman, "New directions in cryptography," *IEEE Trans. on Info. Theory*, vol. 22, pp. 644–654, Nov. 1976.
3. R.J. McEliece, *Finite Fields for Computer Scientists and Engineers*, New York: Kluwer-Academic, 1987.
4. D.E. Knuth, *The Art of Computer Programming, Vol. 2, Seminumerical Algorithms*, Reading, MA: Addison-Wesley, 1969.
5. S.T.J. Fenn, M.G. Parker, M. Benaissa, and D. Tayler, "Bit-serial multiplication in  $GF(2^m)$  using irreducible all-one opolynomial," *IEE Proc. Comput. Digit. Tech.*, Vol. 144, No.6 pp. 391–393, 1997.
6. T. Itoh and S.Tsujii, "Structure of parallel multipliers for a class of fields  $GF(2^m)$ ," *Info. Comp.*, Vol. 83, pp. 21–40, 1989.
7. C.Y. Lee, E.H.Lu, and J.Y.Lee, "Bit-Parallel Systolic Multipliers for  $GF(2^m)$  Fields Defined by All-One and Equally Spaced Polynomials," *IEEE Trans. on Comp.*, Vol. 50, pp. 385–393, 2001.
8. C.L. Wang and J.L.Lin, "Systolic Array Implementation of Multiplier for Finite Fields  $GF(2^m)$ ," *IEEE Trans. on Circuits and Systems*, Vol. 38, pp. 796–800, July 1991.
9. H.S. Kim and K.Y.Yoo, "Area Efficient Exponentiation using Modular Multiplier/Squarer in  $GF(2^m)$ ," *Lecture Notes in Computer Science* 2180, pp. 262–267, 2001.
10. N.Y. Kim, H.S.Kim, and K.Y.Yoo, "Efficient Systolic Architectures for  $AB^2$  multiplication in  $GF(2^m)$ ," Submitted for publication, Oct. 2001.
11. H.S.Kim, *Bit-Serial AOP Arithmetic Architecture for Modular Exponentiation*, Ph.D Thesis, Kyungpook National University, 2002.

# A Practical Distributed Authorization System for GARA

William A. Adamson and Olga Kornievskaja

Center for Information Technology Integration  
University of Michigan, Ann Arbor, USA  
{andros,aglo}@citi.umich.edu

**Abstract.** Although Quality of Service functionality has become a common feature of network hardware, configuration of QoS parameters is done by hand. There is a critical need for an automated network reservation system to provide reliable *last mile* networking for video, audio, and large data transfers. Security of all communications in the process of automating the network configuration is vital. What makes this security problem difficult is the allocation of end-to-end network resources across security realms and administrative domains.

This paper introduces a practical system that shows a design and implementation of Globus General-purpose Architecture for Reservation and Allocation (GARA) services that offer automated network reservation services to users. The contributions of this paper are twofold. First, we provide a fine-grained cross-domain authorization for GARA that leverages existing institutional security and group services, with universal access for users. We identify and discuss issues involved. Second, we eliminate the need for long term public key credentials and associated overheads that are required by other systems. We describe the implementation of an easy and convenient Web interface for making reservation requests.

## 1 Introduction

Reliable high speed end-to-end network services are increasingly important for scientific collaborators, whether separated by large distances or located just across town or campus. Our experience shows that long haul networks demonstrate good performance (thanks to over provisioning), but the *last mile* – from the edge of the campus network to the desktop – is often a network bottleneck.

Quality of Service functionality is a common feature of network hardware. Recent studies show the viability and utility of these features to control network resources. Currently, QoS configuration of network hardware is done by hand. While several standardization efforts are attempting to produce protocols that enable automated network configuration across administrative domains [12,19], it is not yet clear which protocol(s) will be embraced.

Our work addresses the need for an automated network reservation system to provide reliable *last mile* networking for video, audio, and large data transfers

for the partner institutions. Reliable end-to-end network service between partner institutions is achieved by reserving network resources within the end-point institution networks, coupled with the demonstrated adequate performance of the over provisioned interconnecting long haul networks, wherein no network resource reservation is needed.

In automating network configuration, security of all communications is vital. Network hardware is a prime target for malicious hackers, because controlling the routing and resource allocation of a network enables myriad other attacks. What makes this security problem difficult is the cross-domain nature of end-to-end network resource allocation. Requesting end-to-end network resource allocation between the local domain and a remote domain, a user needs to be authenticated and authorized in both domains before the request can be granted.

Our work is based on the Globus General-purpose Architecture for Reservation and Allocation (GARA) [6,8,7,10]. This is a natural choice because the project partner institutions all run Globus software in either production or pre-production mode. The goal of the GARA architecture is to create a flexible solution that satisfies requirements of different types of resources (networks, CPUs, disks, etc.), while providing a convenient interface for users to create both advance and immediate reservations. GARA uses the Globus Grid Security Infrastructure (GSI) [5] for authentication and authorization. An attractive feature of GSI is that it performs cross-domain authentication by relying on a Public Key Infrastructure (PKI) and requiring users to have long term public key (PK) credentials.

GSI provides coarse-grained access control. A flat file, called the *gridmap* file, stores mappings from PK credentials (Distinguished Names, (DN)) to local user names. A user is allowed access to Globus services if there is an entry corresponding to this user in the *gridmap* file. This all-or-nothing access control policy is extremely limiting. Authorization decisions in QoS are based on many parameters such as the amount of available bandwidth, time of day, system load, and others. We propose to control resource usage with a policy engine and expressive security policies.

In this paper, we describe the design and implementation of a GARA system that automates network reservations. The contributions of this paper are twofold. First, we provide a fine-grained cross-domain authorization for GARA that leverages existing security and group services, with universal access for users. Second, we eliminate the need for long term PK credentials, currently required by the system. We also introduce a secure and convenient Web interface for making reservation requests based on Kerberos credentials.

The remainder of this paper is organized as follows. Section 2 describes the Kerberized Credential Authority (KCA) and Kerberized Credential Translation (KCT) services and shows how they allow universal access to GARA by enabling a reservation to be made via the Web, obviating the need to install Globus software on workstations. Section 3 presents an architecture for distributed authorization that employs a shared namespace, delegated authorization through secure and trusted channels and a signed authorization payload, and the policy engine used



**Fig. 1. KX509 GARA Web Interface.** This figure shows how local network resources are reserved with the GARA Web interface. KX509 junk keys replace long term PK credentials. Communications with the KDC, KCA, and KCT are Kerberos protected.

to make the authorization decision. Section 4 briefly describes implementation of the system. Related work is presented in Section 5. Finally, Section 6 concludes.

## 2 GARA Web Interface

Many sites, such as the University of Michigan, lack a PKI, but they do have an installed Kerberos [14] base. The University of Michigan has developed a service that allows users to access Grid resources based on their Kerberos credentials. The KX509 [9,13] system translates Kerberos credentials into short-lived PK credentials, or *junk keys*, which in turn can be used by browsers for mutual SSL authentication or by GSI for Globus authentication.

Junk keys have several advantages over traditional long-lived PK credentials. They have short lifetimes, so the revocation problem [1] is largely obviated. While, in a traditional PKI, long term credentials put the ease of user mobility in question, KX509 users can obtain new junk keys at each workstation.

KX.509 creates a new public/private keypair and sends the public key to a Kerberized Certificate Authority (KCA) over a Kerberos secured channel. Using the presented public key, the KCA creates and signs a short term X.509 identity certificate.

In order to make network resource reservations convenient for users, we built a GARA Web Interface. A user makes a reservation by filling out a GARA network reservation Web form. All requests are SSL protected and require mutual authentication. As opposed to a traditional password-based user authentication, we use short-lived user certificates, priorly acquired with KX.509. After the Web server authenticates the user, it contacts a Kerberized Credential Translation (KCT) [13] server, presents appropriate credentials, and requests Kerberos credentials on the user's behalf. Next, the Web server runs KX509 on the user's behalf, which creates a new junk key for the user on the Web server. This junk key is then used to create Globus proxy credentials. GARA client code resides on the Web server and uses Globus proxy credentials. Figure 1 gives an overview of the GARA Web Interface.

### 3 Distributed Authorization Design

In a cross domain distributed authorization scheme, authorization decisions are made even if the requestor and resources reside in separate domains. Often authorization decisions are made by a policy engine that applies policy rules to a set of input attributes. These attributes might include user attributes such as group membership or environmental attributes such as time of day. Attribute information can come from a variety of sources: local services, environment, configurations, or attached to the resource request. We separate the authorization process into two phases: gathering of attributes and running of the policy engine.

In designing the distributed authorization system, we must address the location where the authorization decision takes place. We discuss how the use of shared namespace and delegated credentials are the key to creating a practical authorization scheme. We also believe in utilizing existing local authorization services to require as little replication of information as possible.

**Location of authorization decision:** The question that needs to be answered is: where is the best place in GARA to make the authorization decision? Three possible locations exist: Web server, gatekeeper, and resource manager.

Prior to initiating any contact with the desired resource, the Web server can contact an authorization service and provide user's identity and resource request information. Having such an authorization service would perforce need to have a policy for each resource and information about each user. However, this choice presents extra communications when the resource is not available, or when fine-grained authorization is not required.

The gatekeeper is expected to handle numerous requests, so performing the authorization decision at the gatekeeper could have an impact on the gatekeeper's performance. At the gatekeeper, it is still unknown if the resource is available, so as above, the extra communication and work to make an authorization decision could be wasted effort. We conclude that adding authorization at the gatekeeper would be counter productive.

The best place to enforce authorization in the GARA architecture is at the resource manager where each service is capable of stating, enforcing, and modifying its policies without depending on the administration of the Globus architecture at large.

**Shared namespace.** Central to any authorization service design is the formation of an attribute namespace that is understood by policy engines. Frequently, the primary concern in the authorization decision is related to a group membership question: does this user belong to appropriate groups? Consequently, a security policy would enforce the restricted membership for specific actions. Within a domain, the statement of group membership is well defined. Both user identity information and a group namespace are available locally.

A shared group namespace, presented to policy engines in multiple domains and used to control access to resources in multiple domains, is defined by a number of groups with common names across domains. In its existing group service, each domain creates groups with these names and manages user membership as any local group. Other attributes presented to the distributed policy engines

such as the amount of requested bandwidth or start-time of the request are already encapsulated in a shared namespace in that they are coded as name,value pairs in the request.

**Signed authorization payload.** At the remote service, we do not add a callback to the local group service to determine group membership, instead, authorization information is added to the existing resource request.

The local GARA resource manager queries the local group membership service for the subset of shared namespace groups in which the requestor is a member, and passes the group list along with the request parameters to its policy engine to make an authorization decision. If the request succeeds, the local GARA resource manager creates an *authorization payload* consisting of the requestor's distinguished name and the group list. To secure the authorization payload, we require the local GARA resource manager to sign the authorization payload before adding it to the reservation reply returned to the GARA client running on the Web server. The reservation request is forwarded to the remote GARA who validates the signature on the received authorization information before passing the group list as input to its policy engine. Thus we piggy-back the group membership information on the existing reservation request communication.

**Policy Engine.** After all the requestor's attributes such as group membership and request parameters have been established, the fine-grained authorization decision can be made. In general, policy engines accept attribute-value pairs as input, compare the input attributes to a set of policy rules, and return a pass/fail response. The granularity of the authorization decision is embodied in the complexity of the policy rules that must be satisfied by the input attribute-value pairs. To allow different policy engines, the authorization callout has a generic API that passes information about the requestor and the action to the policy engine. We chose KeyNote [2,3,4] for our policy engine because of its flexibility and easy availability.

## 4 Implementation

### 4.1 KeyNote Policy Engine

Applications such as GARA describe policies to KeyNote with a set of attribute-value pairs (called an action condition) creating a *policy namespace*. In Figure 4, the policy namespace consists of the following attributes and their corresponding values: app\_domain, operation, type, location, amount, time, and grid\_bw. To express a security policy, each Globus site is free to choose any descriptive attribute name. However, if any of the attributes are to be included in the authorization data that is passed to or received from a remote domain, then the attributes need to be included in the shared namespace we described previously.

We now describe the basic pseudocode for the KeyNote policy engine call with a group membership action condition.

- *requester*: the requesting principal's identifier.

```

SN_groups = retrieve_group_membership(requestor);
result = authorization_decision(requestor, action_description,
                               policy, credentials);
if(result == "allowed") do the requested action
else report action is not allowed

```

**Fig. 2.** Pseudo-code for the authorization mechanism in *diffserv\_manager*.

```

session_id = kn_init();
kn_add_assertion(session_id, policy[i]);
kn_add_assertion(session_id, credentials[i]);
for all actions in action_description
    kn_add_action(session_id, action_description.attr,
                  action_description.value);
result = kn_do_query(session_id);

```

**Fig. 3.** Pseudo-code for the call to the KeyNote policy engine.

```

keynote-version: 2
local-constants: ADMIN_UM = "x509-base64:MIICrzCC"
authorizer: "POLICY"
licensees: ADMIN_UM
conditions: app_domain == "gara" &&
            operation == "reservation" &&
            type == "bandwidth" &&
            ((location == "local" && @amount <= 100) ||
             (location == "remote" && @amount <= 10) ||
             time == "night") && grid_bw == "yes");

```

**Fig. 4.** Trusted assertion stating KeyNote top level security policy. Note that the value of the key has been truncated.

- *action\_description*: the data structure describing an action contains attribute value pairs which are included in the request. For example, “*system.load* ≤ 70” specifies an environment condition stating that the current system load must not exceed 70%.
- *SN\_groups*: the shared namespace groups in the *action\_description*, also added as attribute value pairs describing the action. For example, “*grid.bw* = yes” states the request is a member of the *grid.bw* group.
- *policy*: the data structure describing local policy, typically read from a local file.
- *credentials*: the data structure with any relevant credentials, typically sent along with the request by the requesting principal. Before making use of these credentials, their validity must be confirmed by verifying a signature included in the credential data structure.



Figure 2 shows the main actions of *diffserv\_manager*.

Figure 3 provides details of the *authorization\_decision* function.

Figure 4 shows an example of a KeyNote top level security policy that allows the action if the following conditions hold: an application domain is called *gara* and the requested operation is *reservation* for the resource of type *bandwidth*. Furthermore, if this is a local request, then bandwidth for more than 100Mb is not allowed. If the request is from a remote user, then amount greater than 10Mb is not allowed. If the current time is after hours, then no restriction on bandwidth is enforced. The requestor must be a member of *grid\_bw* group.

If the KeyNote policy engine states that the action is not allowed, no reservation is made by the local *diffserv\_manager* and an authorization failure is returned to the Web server. As the result, the reservation protocol returns an authorization error back to the client. A success value is returned to the client only if both local and remote authorization have succeeded.

## 4.2 Reservation Flow

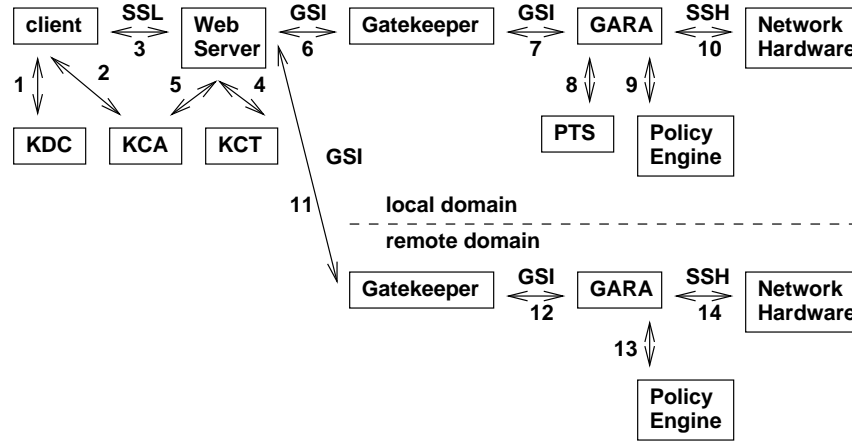
We successfully demonstrated our modifications to GARA by reserving bandwidth for a video application running between the University of Michigan and CERN<sup>1</sup>. Bandwidth is reserved by filling in a Web form served by a modified Apache Web server that runs the GARA client. The GARA client communicates with separate GARA services at each end-point domain, as shown in Figure 5. The GARA services use KeyNote authorization policies configured to require bounded request parameters for bandwidth, time and duration. Group membership is also required. We demonstrated that if any of the policy parameters are not satisfied, e.g. too much requested bandwidth or incorrect AFS PTS group membership, the reservation fails.

A successful reservation results in configuration of the end domain Cisco ingress routers, that marks the packets and polices the flow, with the appropriate Committed Access Rate (CAR) rate.limit. The participating routers are statically configured with WRED, Cisco's implementation of the Random Early Detection (RED) class of congestion avoidance algorithms.

What follows is a step by step description of an end-to-end network reservation using the enhanced GARA, also illustrated in Figure 5.

1. User (locally) executes *kinit* and acquires Kerberos credentials.
2. User (locally) executes *xx509* and acquires junk keys.
3. Using a browser, a user makes an https request for the network resource reservation page. The junk key, obtained in Step 2, is used for mutual SSL authentication. Network reservation parameters such as source and destination IP address, desired bandwidth, start time are entered into a form and sent to the Web server.
4. The Web server *kct\_module* makes a Kerberos authenticated request to the KCT and acquires a service ticket for the KCA service on the user's behalf.

<sup>1</sup> European Organization for Nuclear Research



**Fig. 5. Network Resource Reservation Data Flow.** KDC is a Kerberos Key Distribution Center. KCA is a Kerberized Key Signer. KCT is a Kerberized Credential Translator. KDC and KCT must share hardware because both require access to the Kerberos database. PTS is AFS Protection Server.

5. The Web server *kx509\_module* acquires and caches junk keys on behalf of the user as in Step 2. Then, the Web server *globus\_proxy\_init* module uses the newly created keys to create user's Globus proxy certificate.
6. The Web server *gara\_module* constructs a reservation request to the local gatekeeper using the Globus GSSAPI\_SSLEAY protocol and the proxy certificate. The local GARA gatekeeper looks for an entry in the *gridmap* file that matches the corresponding distinguished name – a field in the Globus proxy certificate (from Step 5). The DN and local id are attached to the RSL (Resource Specification Language) string.
7. Using the Nexus API for interprocess communication, the local gatekeeper forwards the RSL to the resource manager (*diffserv\_manager*).
8. The local id is passed to the group\_membership function, which performs one of several actions, depending on configuration and on whether the request is from a local or remote user. If the authorization data in this RSL is null, the request is from a user in the local realm. In our settings, group membership function queries a Protection Server (PTS) – a part of AFS. The call can be easily replaced by a query to any local group service such as an LDAP service or a flat file. The call is performed over an authenticated channel using the *diffserv\_manager*'s identity. Prior to accepting any connections, *diffserv\_manager* acquires AFS tokens needed to authenticate with the PTS server.
9. The group membership information acquired in the previous step, the reservation parameters, and a policy file are passed to the KeyNote policy engine that makes an authorization decision. If the request does not satisfy the cur-

rent security policy, an error is returned back to the client and the rest of the steps are not executed.

10. The `setup_flow` Expect script uses SSH to communicate with the appropriate network hardware and configures the network reservation.
11. This step is the same as Step 6 except this time the RSL carries the authorization payload. We use *auth-data* as the attribute name. The value is variable length. In our current implementation, the RSL is limited to 4KB, at least enough to encode information 64 groups (assuming 64 byte names).
12. Same as Step 7.
13. Same as Step 9.
14. Same as Step 10.

This design lets us express many policies, including who can request which network resources and when such requests are valid. In the example we presented, the authorization payload is signed by one certificate, the remote GARA *diffserv manager*. More broadly, a site may require the authorization payload to contain assertions from other services. For example, a site might require that users be U.S. citizens, verified by some specific Certificate Authority. Signed assertions can be added to the authorization payload to accommodate such requirements.

## 5 Related Work

The Globus MyProxy [16] initiative provides a trusted server to store user's delegated credentials, indexed by a tag and a password. Later, a service can contact a MyProxy server, present a tag and a password and receive corresponding credentials (e.g., certificate or Kerberos ticket) on a client's behalf. Each service requires a different tag and password, forcing users to manage many passwords. This approach requires users to type in their passwords into HTML forms. HTML forms are easily reproduced by a malicious hacker who collects passwords. He can obtain a certificate, signed by one of the default Certificate Authorities supported by browsers, and run a Web server, providing a spoofed login HTML form. However, by examining the credential, an activity most users do not bother doing, the user can tell the login form is spoofed.

The Grid Portal Architecture [11] is a Web interface to Grid Computing resources that uses MyProxy Services for client authentication. The GARA Web interface differs from the Grid Portal in several important ways. Access in our scheme is via done an https stream and requires mutual SSL authentication, which in turn requires a user certificate, thus obviating the need for users to type passwords in HTML forms, as it is done in the Grid Portal.

The Community Access Service (CAS) [15] is a proposed Grid authorization service that the user calls prior to making a request for Grid resources. CAS returns a signed capability to indicate a successful authorization request. The capability is then added to the Grid resource request.

The GARA client is designed to contact each end domain GARA service. In the future, GARA client will contact the first GARA service, which in turn will

contact other bandwidth brokers (BB), needed for the end-to-end reservation. Sander *et al.* discusses the bandwidth broker to bandwidth broker protocol in [18]. The Simple Inter-Domain Bandwidth Broker Specification (SIBBS) [19] is a simple request-response bandwidth broker to bandwidth broker protocol being developed by the Internet2 QBone Signaling Design Team. It is anticipated that GARA will be an early development code base for SIBBS.

Our choice of policy engines was influenced by the availability of working code. The modular design allows for use of other policy engines. Akenti [20], and GAA API [17] were also considered. We acknowledge Akenti's strength over KeyNote in terms of credential management. On the other hand, Akenti imposes a lot of overhead, not required by KeyNote, such as creation of certificates for each of the clients.

## 6 Conclusions

We have demonstrated a scalable distributed authorization service that joins local domain group services via a shared namespace, and asserts group membership by adding a signed authorization payload to existing communications.

We showed that authorization succeeds only when the user is a member of the correct groups and the reservation parameters are within bounds as dictated by the policies present at each end-point.

We have focused our distributed authorization service design on the ability to use existing local domain group services, firm in the belief that coalescing and maintaining replicas of user and group information does not scale and is an unnecessary administrative burden.

## References

1. Andre Arnes. *Public Key Certificate Revocation Schemes*. PhD thesis, Norwegian University of Science and Technology, Kingson, Ontario, Canada, February 2000.
2. M. Blaze, J. Feigenbaum, J. Ioannidis, and A. Keromytis. The KeyNote trust management system version 2. RFC 2704, September 1999.
3. M. Blaze, J. Feigenbaum, and A. Keromytis. Keynote: Trust management for public-key infrastructure. In *Proceedings Cambridge 1998 Security Protocols International Workshop*, April 1998.
4. M. Blaze, J. Feigenbaum, and M. Strauss. Compliance checking in the PolicyMaker trust management system. In *Proceedings of Financial Cryptography*, February 1998.
5. R. Butler, D. Engert, I. Foster, C. Kesselman, S. Tuecke, and J. Volmer. A national-scale authentication infrastructure. *IEEE computer*, 33(12):60–66, December 2000.
6. Documentation: A Guide to GARA.  
[http://www-fp.mcs.anl.gov/qos/qos\\_papers.htm](http://www-fp.mcs.anl.gov/qos/qos_papers.htm).
7. Documentation: Administrators Guide to GARA.  
[http://www-fp.mcs.anl.gov/qos/qos\\_papers.htm](http://www-fp.mcs.anl.gov/qos/qos_papers.htm).
8. Documentation: Programmers Guide to GARA.  
[http://www-fp.mcs.anl.gov/qos/qos\\_papers.htm](http://www-fp.mcs.anl.gov/qos/qos_papers.htm).

9. W. Doster, M. Watts, and D. Hyde. The KX.509 protocol. Technical Report 01-2, Center for Information Technology Integration, University of Michigan, February 2001.
10. I. Foster, A. Roy, and V. Sander. A quality of service architecture that combines resource reservation and application adaptation. In *Proceedings of the 8th International Workshop on Quality of Service (IWQOS 2000)*, June 2000.
11. Grid Computing Portal:  
[http://hotpage.npaci.edu/cgi-bin/hotpage\\_top.cgi](http://hotpage.npaci.edu/cgi-bin/hotpage_top.cgi).
12. IETF Internet Traffic Engineering Working Group.  
<http://www.ietf.org/html.charters/tewg-charter.html>.
13. O. Kornievskaja, P. Honeyman, B. Doster, and K. Coffman. Kerberized credential translation: A solution to web access control. In *Proceedings of the 10th USENIX Security Symposium*, August 2001.
14. C. Neuman and T. Ts'o. Kerberos: an authentication service for computer networks. *IEEE Communications*, 32(9):33–38, September 1994.
15. L. Pearlman, V. Welch, I. Foster, C. Kesselman, and S. Tuecke. A community authorization service for group collaboration. In *IEEE Workshop on Policies for Distributed Systems and Networks*, 2002. submitted.
16. MyProxy project. <http://dast.nlanr.net/Projects/MyProxy>.
17. T. Ryutov and C. Neuman. Representation and evaluation of security policies for distributed system services. In *Proceedings of the DISCEX*, January 2000.
18. V. Sander, W. A. Adamson, I. Foster, and A. Roy. End-to-end provision of policy information for network qos. In *Proceedings of the 10th Symposium on High Performance Distributed Computing*, August 2001.
19. SIBBS. The simple inter-domain bandwidth broker specification.  
<http://qbone.internet2.edu/bb/>.
20. M. Thompson, W. Johnson, S. Mudumbai, G. Hoo, K. Jackson, and A. Essiari. Certificate based access control for widely distributed resources. In *Proceedings of the 8th USENIX Security Symposium*, August 1999.

# Design of a VPN Software Solution Integrating TCP and UDP Services

Javier Lopez<sup>1</sup>, Jose A. Montenegro<sup>1</sup>, Rodrigo Roman<sup>1</sup>, and Jorge Davila<sup>2</sup>

<sup>1</sup> Computer Science Department

Universidad de Malaga, Spain

{jlm,monte,roman}@lcc.uma.es

<sup>2</sup> Computer Science Department

Universidad Politecnica de Madrid, Spain

jdavila@fi.upm.es

**Abstract.** The main aims of Virtual Private Network (VPN) are to isolate a distributed network from outsiders, as well as to protect the confidentiality and integrity of sensitive information traversing a non-trusted network such as the Internet. However, some problems arise when security is considered as the unique problem because VPN users suffer from restrictions in their access to the network. They are not free to use traditional Internet services such as electronic mail exchange with non-VPN users, and to access Web and FTP servers external to the organization. This paper presents a new solution that allows the open use of traditional network services running over TCP and UDP layers, while maintaining strong security features. The new scheme works at the TCP/IP transport layer and does not require the addition of new hardware because it is a totally software solution. As a consequence, the application is totally portable. Moreover, and because of its implementation at the transport layer, there is no need to modify any traditional communication applications previously installed in the network system.

## 1 Introduction

The needs of digital communications for most of organizations have grown very much during last years because of different reasons. For instance, globalization of economy increases the demand of telecommunication among branch offices, and inter-companies agreements impose that resources are shared. Thus, decreasing the cost of telecommunication infrastructures is imperative.

Traditionally, companies have used leased lines with that purpose. The most representative example is Frame-Relay service [2],[5], which is based on the transfer of information frames between intermediate switching offices. The service, that uses permanent virtual circuits (PVCs) through telephone network routers, presents some drawbacks:

- It becomes expensive because connections remain open permanently.
- The architecture creates large latency periods because of the poor connectivity between intermediate routers.

- Full connectivity requires the increment of PVCs and, hence, of intermediate network routers; but the cost of avoiding routing problems in this way is high.
- The number of companies that offer Frame-Relay services is small compared to the number of Internet Service Providers (ISPs), so competitiveness is more limited.

On the other hand, open networks offer a more profitable solution than leased lines. Thus, for example, *Virtual Private Networks* (VPNs) use relatively low-cost, widely available access to public networks to connect remote sites together safely. Network architectures defined by VPNs are inherently more scalable and flexible than classical WANs, and they allow organizations to add and remove branch offices into their systems in an easy way.

However, and as shown later, the study of the different TCP/IP stack layers reveals that the different solutions that enable establishing a VPN essentially focus on security aspects. Their main aims are to isolate a distributed network from outsiders and to protect the privacy and integrity of sensitive information traversing the non-trusted open networks, as the Internet. These approaches fail to be complete.

The main drawback in conceiving the security problem as the unique target is that VPN users suffer from restrictions in accessing the Internet. That is, they cannot freely use traditional services such as electronic mail exchange with non-VPN users, and cannot freely access Web and FTP servers external to the organization. Actually, within the same application, it is a difficult task to enable generic Internet access to VPN users and, at the same time, to provide a strong enough security model for the organization.

This work presents an approach to this problem, that is an extension to a previous constrained approach [3]. We have developed a new security integrated solution for VPNs that, while maintaining strong security features, allows the open use of traditional Internet services that run not only over TCP (the case of our previous solution), but also over UDP. The solution does not require the addition of new hardware because it is an exclusively software solution. As a consequence, the application is totally portable. Moreover, the implementation is located at the transport layer; thus, it allows using those services running over TCP (like FTP, Telnet, WWW, etc.) and also those running over UDP (that is, real-time applications like audio and video-conferences), which are increasingly being used in the Internet. Additionally, and as we will show, there is no need to modify any software previously installed.

The paper is organized in the following way: Section 2 introduces the different cryptographic solutions used in the TCP/IP stack layers to establish private network communications over the Internet, focusing on their advantages and disadvantages. Section 3 explains the two-module architecture of the new system. Section 4 describes the operation of the first module, called *Secsockets*, an extension of the traditional socket interface to which security features have been added. Section 5 outlines the operation of *VPN-Insel*, the second module. This is the part of the system that really establishes the VPN, and resides on the

interface of the previous module. Section 6 shows a real example of how the new solution works, and section 7 presents concluding remarks.

## 2 Security Solutions in the TCP/IP Architecture

There are different security solutions in the TCP/IP stack layers that can be used to establish confidential and authenticated communications over Internet. In this section we briefly review the most relevant solutions in those layers, pointing out their advantages and disadvantages.

It is necessary to point out that all the solutions that have been developed to work above the network layer are oriented to TCP services. To the best of our knowledge there is no widely used standard mechanism that provides security to UDP services.

### 2.1 Data Link Layer

The method used in the lowest layer of the stack is point-to-point encryption. The architecture of link level encryption defines a completely isolated connection between two systems. On the one hand, physical access is restricted because every connected host belongs to the organization; on the other hand, logical access is restricted too because information is encrypted throughout the transmission. This solution does not necessarily require the use of the TCP/IP stack protocols because these ones work at higher levels.

Point-to-point encryption is a simple concept that makes it, from a security standpoint, a good VPN solution. However, it has certain properties that make it hard to use in every application. Firstly, system users have no choice regarding encryption. They can either link to a host using encryption or they cannot communicate at all. Therefore, it does not provide generic Internet access because it protects from hostile outsiders, but also blocks access to beneficial outsiders. Secondly, point-to-point encryption scales very poorly. It becomes complex and expensive to establish links to new hosts that are installed in the network system.

### 2.2 Network Layer

The solution used at the network layer is the encryption of the data fields in the IP packets. The set of *IPSEC protocols* (IP security protocols) [1], is a part of IPv6, the IP's future version and it is designed to provide privacy and/or data forgery detection.

For this purpose, IPSEC defines two optional packet headers: *Authentication Header* (AH), and *Encapsulating Security Payload* (ESP). Both headers contain a numeric value called the *Security Parameter Index* (SPI), which is used by a host to identify the cryptographic keys and the security procedures to be used.

In order to communicate, each pair of hosts using IPSEC must negotiate a security association. The security association establishes what types of protection to apply, how to encrypt or authenticate, and which keys are needed. The packet



IP address and the packet SPI in the packet header determine the security association applied to an IPSEC header.

The combination of IPSEC protocols with routers or firewalls constitutes a VPN solution, and it enables the use of traditional Internet services. Nevertheless, and because of its location, it is difficult to establish a security relationship between two applications, which makes difficult to provide the same granularity and control over authentication and encryption. Subsequently, an important disadvantage of this solution is that the system automatically applies protection according to the security associations between the host; users cannot decide when to apply security mechanisms, which is inefficient for most cases. An additional disadvantage is that architecture blocks communications to other non-trusted computer systems and networks, so it does not enable generic Internet access.

### 2.3 Transport Layer

There are some standardized solutions at the transport layer for TCP, but as stated, not for UDP. The implementation of Netscape's *Secure Socket Layer* (SSL) [12] protocol stands out from other solutions. It is widely used in conjunction with World Wide Web service and includes capabilities for authentication, encryption and key exchange. Some similar alternatives to this solution are *Transport Layer Security* (TLS) [4], proposed by the IETF TLS working group, and the Microsoft compatible protocol Private Communication Technology (PCT) [9].

The SSL protocol is integrated in both Web client and server software, and protects data transfer between them. Client and server negotiate the type of protection by setting up a group of cryptographic parameters that includes a strong encryption algorithm and a shared secret key. A public key cryptosystem is used to facilitate authentication and distribution of the secret key.

Applying security at the transport level in this way provides good control over the mechanisms to be used, because a web browser can choose whether a particular connection will be secure or not.

However, a serious disadvantage of this scheme is that it requires the integration of SSL protocol software into the application itself; that is, it is necessary to modify every network application that needs to use these security mechanisms. Therefore, this is not a general solution for a VPN because it only solves specific network services.

### 2.4 Application Layer

There are specific solutions at the application layer, but again, only for TCP-based services, not for UDP-based ones. These solutions are, for instance, electronic mail (PGP, PEM, S-MIME) [18] [8] [13], the file transfer protocol (Secure-FTP) [6], the hypertext transfer protocol (S-HTTP) [16], etc. Regarding UDP, each application based on this protocol must provide its own security solution.

The lack of a homogeneous solution becomes a serious problem because each application provides its particular encryption and key management mechanisms.

Moreover, even if only solution existed, providing a VPN solution at this level would make necessary to control that every single user updates his/her non-secure applications with the selected security mechanisms. There is no doubt that this management would be hard, not only for the establishment of the VPN but also for any little change in the configuration of the VPN.

### 3 Architecture of the New Scheme

The scheme we introduce is an entirely software solution that works at the transport level. The solution adopts the advantages of IPSEC (it facilitates the use of traditional Internet services) and, at the same time, solves some of its drawbacks. For instance, in IPSEC, security associations are only negotiated among hosts but not among services. Also communications with hosts outside the VPN is not allowed. These problems are solved in our new scheme where, additionally, VPN users decide when to apply security mechanisms. Thus, in our solution any VPN user can connect to other VPN and non-VPN users. Moreover, it does not require the modification of the software of traditionally insecure applications such as FTP, HTTP, Telnet, electronic mail, etc. As a result, the scheme is a simple and cheap solution for those organizations that want to install a VPN.

A generic organization scenario that has been used for the development of the new solution is depicted in figure 1. There is a main entity that centralizes control of the organization and represents the Headquarters. There are other entities that represent the branch offices of the organization, the secondary entities. Every secondary entity is located in a different LAN.

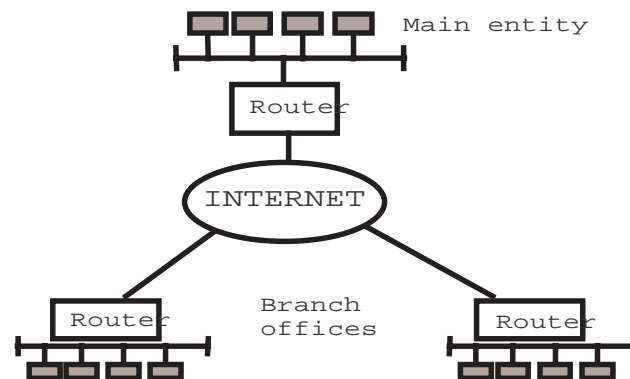


Fig. 1. Scenario of a VPN

The new design is divided into two sublevels, Secsockets y RPV-Insel. The module Secsockets is the lower sublevel. It is an interface that works on top of the traditional socket interface that provides access to TCP. The service offered by Secsockets is the transmission of data from an origin process to a target one through a secure and authenticated channel.

The module VPN-Insel is the upper sublevel. It uses the services provided by Secsockets and, at the same time, it supports the applications with the VPN services that these ones require. Next sections shows the operation of both modules in detail.

## 4 SecSockets

Secsockets is an extension of the traditional socket interface, and enables authenticated and confidential communications. That is, Secsockets provides the same functions as the socket interface and, in addition, provides authentication, confidentiality, and integrity.

The goal of this interface is to provide a secure delivery service for both TCP and UDP, while facilitating its use. This interface is based on a client/server architecture, in such a way that the process that starts the communication plays the role of the client, and the process that attends to the connection at the other end plays the role of the server. The operation of Secsockets is divided into two phases: firstly a connection phase, and secondly, the communication phase itself.

### 4.1 Connection Phase

Firstly, during the connection phase, mutual authentication is carried out. Secondly, there is a negotiation of the parameters (hash function, encryption algorithm, and optional compression algorithm) that will be used during the communication phase.

In this phase the two interacting parts have no secret key to share as yet. So, the use of public key certificates provides mutual authentication and confidentiality during parameter negotiation. This enables communicating the secret key, so avoiding the risk of a third party interception, and facilitating the identification of VPN users because it provides a method to implement digital signatures.

We must point out that during this phase the TCP protocol is used regardless the type of service that needs the creation of a secure channel. The reason of using TCP is that we need a reliable protocol for parameters interchange, and is widely known that UDP is not reliable at all.

The following are the steps that are done:

1. The client and the server interchange the digital certificates.
2. The client sends, encrypted with the server's public key, a connection request. This message contains the security parameters (hash function, symmetric key algorithm, a random value and, optionally, data compression). The hash functions that can be used are MD5 [14] and SHA [11]. The symmetric key

encryption algorithms that can be used are DES [10], IDEA [7], Blowfish [17] and RC4 [15]. In case data compression is selected, the GZIP algorithm is used.

3. Once the request is processed, the server sends a message to the client indicating acceptance or rejection, encrypted with the client's public key. In case negotiation is accepted, it includes a random number that will be used for the calculation of the key. If the channel to be opened is and UDP one, then the message will include the port number where the server will listen messages from the client.
4. Both client and server calculate the secret key that will be used during the next phase: the communication phase. Random values exchanged in step 2 and 3 and the hash function negotiated in these steps are used to calculate a bit string. The string is obtained from the following expression:

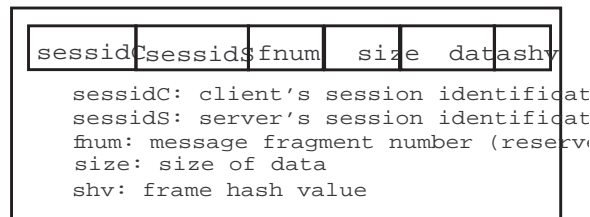
$$H(H(\text{Client\_random}) \text{ XOR } H(\text{Server\_Random}))$$

Additionally, and in the case the channel to be opened is UDP, the server closes the TCP connection with the client, and opens a UDP connection to wait messages from this one.

## 4.2 Communication Phase

The operations performed during the communication phase (or transmission phase) are, for each message: a) fragmentation of the message into frames; b) data field compression, if this was decided during the initial negotiation; c) calculation of hash value for each frame; and, d) frame encryption and MAC calculation.

Figure 2 shows the composition of each frame. Maximum size of the message depends on the length of packets accepted by the network; thus it is a value that can be configured. Obviously, after the reception of each packet, the receiver has to undo all previous operations.



**Fig. 2.** Format of communication frame

### 4.3 Functions of SecSockets

The service offered by Secsockets is to send data from a source process to a target one through a secure and authenticated channel. Secsockets works as a connection oriented communication protocol based on the client/server paradigm.

Secsockets provides a group of functions. From the programming interface point of view these functions work in a client/server mode. As we will see later, the reason is that levels above Secsockets need an interface that helps in the creation of parallel client/server systems. Also, working on this way, it is possible to use either TCP or UDP for the communication by changing only one parameter.

The functions are the following ones:

- *sec\_init( )*: This function creates a connection point at the server's end. The point is initialized and the function assigns it to a local communication port. Then the server remains in an idle state waiting for a client connection-request through that port.
- *sec\_accept( )*: This function is invoked by the server to positively acknowledge an incoming connection and immediately provides it with a new socket. After accepting the connection the negotiation phase, under TCP, starts at server's end. A socket address, either TCP or UDP, according to the mentioned parameter, is returned. Security parameters are returned too.
- *sec\_connect( )*: The client invokes this function in order to create a final connection point. After a negotiation phase under TCP starts at the client's end, authenticating the other end and agreeing on the security parameters. A TCP or UDP socket address and security parameters are returned.

From this point on, and following the parameters settled on during the negotiation phase, it will be possible to perform securely the communication phase. The functions of this phase are described next:

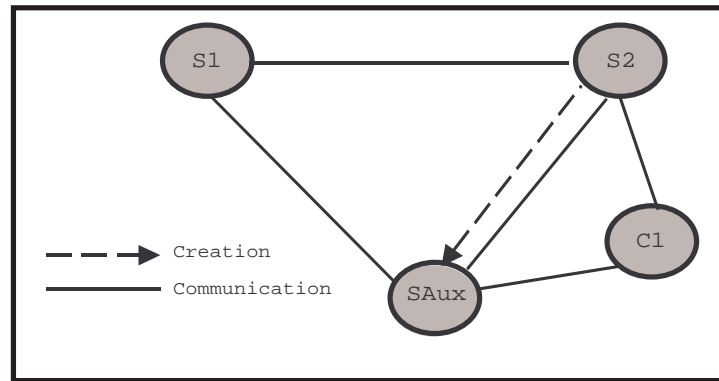
- *sec\_recv( )*: This function enables data reception through a secure socket connection, decrypts them, checks their authenticity, and decompresses them if necessary
- *sec\_send( )*: This function is symmetrical to the previous one. It compresses data if this was negotiated, calculates the hash value, encrypts data, and finally, sends them through the socket. These operations are performed according to the negotiations of the client and server.
- *sec\_close( )*: This function erases the socket.

## 5 RPV-Insel

The module VPN-Insel uses the services provided by the interface Secsockets. At the same time it supports the communication applications that run at the upper level and provides them with the characteristic services of a VPN. So, by making use of Secsockets security mechanisms, VPN-Insel offers users the

possibility of registration into the VPN, participating in on-line connections, like HTTP, Telnet, or FTP, and off-line connections, such as electronic mail. Moreover, this module manages the VPN. That is, VPN-Insel initializes the private network, setting the configuration parameters. Once the private network is working, VPN-Insel controls which users belong to it and can use its services, and which new users and secondary entities can register. It also allows system supervisors to install servers for HTTP, Telnet, FTP, SMTP and videoconference which are of exclusive use for users belonging to the VPN.

Basically, the VPN-Insel processes and their interrelations are those shown in figure 3. Now we detail the operation of these processes:



**Fig. 3.** RPV-Insel scheme

1. *Clients* (C1): represents the client software for those users who communicate from a LAN.
2. *Main server* (S1). It runs in the main entity's computer system (main LAN in the organization). This type of process centralizes the VPN management, controls other types of server processes, maintains a database with the information related to these servers (IP addresses), and provides information about the VPN. There is only one of these processes in each VPN.
3. *Secondary Server* (S2): This server controls the local users (the users inside its LAN) and decides which of them can log in to the VPN. It also attends to their requests in the same way as the main server does. One secondary server exists in each entity, i.e., in each LAN of the organization.
4. *Auxiliary Servers* (SAux): Auxiliary servers are created by the secondary server to attend to users during a particular connection. They are intermediate processes within secure communications and make use of the functions that the SecSockets interface provides, in such a way that security mechanisms are transparent to client processes.

### 5.1 Detailed Operation of VPN-Insel

RPV-Insel starts working when the VPN administrator starts-up the main server (S1). After this, the following sequence of events takes place:

1. S1 initializes the secondary servers database.
2. S1 starts a secondary server (S2) for its own LAN (main entity's LAN).
3. S1 remains open waiting for requests. The requests come from other LAN's secondary servers or remote users.
4. Each secondary server, started by the corresponding LAN administrator, is registered in the main server, in order to integrate the LAN into the VPN.
5. Each secondary server stays open for: a) requests from users in its own LAN; or b) requests from remote auxiliary servers that want to use its LAN's resources.

Then clients can start using the VPN to communicate securely with other members of the private network. The communication among VPN members follows these steps:

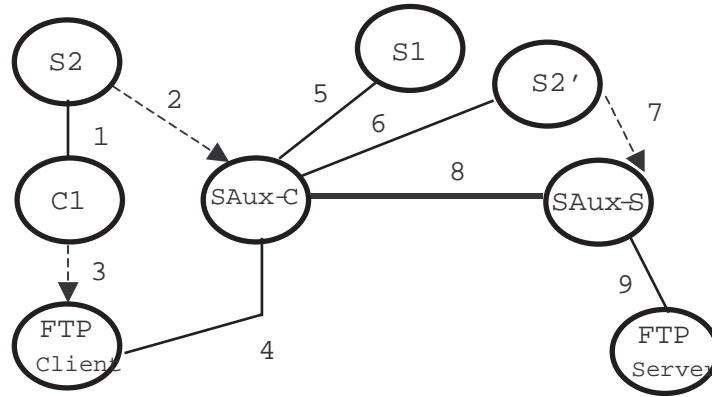
1. The user decides, by using the application interface of VPN-Insel, which network application to use (e.g. FTP) and the name of the computer (or IP address) where the real server (e.g. FTP server) is running.
2. The application interface of VPN-Insel sends a communication request to its secondary server. This one creates an auxiliary server that exclusively manages that secure connection. Afterwards, the secondary server remains open for new requests
3. The auxiliary server connects with the secondary server in the other end, and a secure channel is established.
4. At this moment, VPN-Insel runs the client program, but it does not provide it with the computer address where the real server is running. VPN-Insel provides the client with the address where the auxiliary server is running inside its own client LAN.
5. From this point on, the communication between the client and the server processes takes place in a secure way through the channel that connects both auxiliary servers.

## 6 Examples

In this section we show two scenarios that clarify how our solution works for TCP and UDP services. In the first scenario we show how an FTP session is secured inside the VPN. The second scenario describes an audio-conference between two users.

The example scenario for FTP service is shown in figure 4. A client C1 wants to connect to one FTP server in the VPN. The sequence of events is the following one:

1. C1, started by the user, request a connection to S2, its secondary server.



**Fig. 4.** Scheme of processes for a FTP service

2. If the user is a legitimate VPN user, S2 creates an auxiliary server, SAux-C, to manage the connection.
3. At the same time, process C1 runs the FTP client software.
4. FTP client software connects to SAux-C.
5. SAux-C request S1 for the number of the port and the host where process S2' is running.
6. SAux-C request S2' to create SAux-S, the remote auxiliary server.
7. S2' creates SAux-S.
8. SAux-C connects to SAux-S, and they set up the secure channel.
9. SAux-S connects to the FTP server.

Regarding the audio-conference service, the example scenario is shown in figure 5. Two clients, C1 and C2 want to establish a secure communication. The steps performed are the following ones (we suppose that C1 is the one that starts the communication):

Steps 1 and 2 are the same one as in the previous scenario.

3. SAux-S request S1 for the number of the port and the host where process S2' is running.
4. SAux-S notifies S2', y S2' notifies user C2. SAux-S adopts the role of server, and waits for the request to start the communication.
5. C2, started by the user, request a connection to S2, its secondary server.
6. If the user belongs to the VPN, S2' creates an auxiliary server SAux-C, that will communicate with to SAux-S.
7. SAux-C connects to SAux-S, and they set up the secure channel.
8. The audioconference program runs in both ends.

In both cases, the secure communication is established between the auxiliary servers (thick line). They work as gateways, receiving the frames, decrypting the information and sending them to the specific receiver.



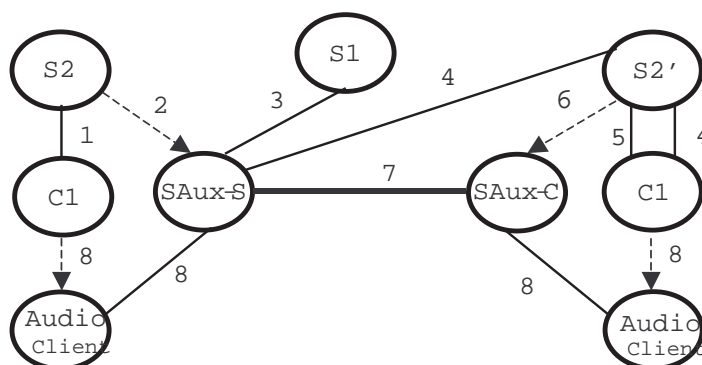


Fig. 5. Scheme for a videoconference service

## 7 Conclusions

The study of the different TCP/IP stack layers reveals that the different solutions that enable establishing a VPN pay most of their attention to security aspects. These solutions focus on the isolation of a distributed network from outsiders and on the protection of sensitive information traversing Internet. But the users of these systems cannot freely use traditional services such as electronic mail exchange with non-VPN users, and cannot freely access Web and FTP servers external to the organization.

In this paper we have presented a new solution for the implementation of VPNs at the transport layer that, while maintaining strong security features, allows the open use of traditional Internet services that run over TCP and UDP layers. As a consequence, it enables access to any remote node in the VPN and outside the VPN. The solution is exclusively software, so its deployment does not require the addition of new hardware or the modification of any existing one.

The solution allows an homogeneous control of all network services, which is necessary to gain higher user confidence regarding the entire system. Moreover, it does not require the modification of the software of traditionally insecure applications such as FTP, HTTP, Telnet, electronic mail, etc. As a result, the scheme is a simple and cheap solution for those organizations that want to install a VPN.

## References

1. Atkinson, R. Security Architecture for the Internet Protocol. RFC 1825, August 1995.
2. Black, U. Frame-Relay: Specifications and Implementations, McGraw-Hill, 1994.
3. Davila, J., Lopez, J., Peralta, R. Implementation of Virtual Private Networks at the Transport Layer, Second International Information Security Workshop (ISW'99), LNCS 1729, November 1999, pp. 85–102

4. Dierks, T., Allern, C. The TLS Protocol Version 1.0. RFC2246, January 1999.
5. Harbison, R. Frame-Relay: Technology for our Time, LAN Technology, December 1992.
6. Horowitz, M., Lunt, S. FTP Security Extensions. RFC 2228, October 1997.
7. Lai, X.; Massey, J. Hash Functions Based on Block Ciphers. Advances in Cryptology. EUROCRYPT '92, Springer-Verlag, 1992, pp. 55–70.
8. Linn, J. Privacy Enhancement for Internet Electronic Mail: Part I – Message Encipherment and Authentication Procedures. RFC 989, February 1987.
9. Microsoft Corporation. The Private Communication Technology, 1997.
10. National Bureau of Standards. Data Encryption Standard. U.S. Department of Commerce, FIPS pub. 46, January 1977.
11. National Institute of Standards and Technology, NIST FIPS PUB 180. Secure Hash Standard. U.S. Department of Commerce, May 1993.
12. Netscape Communications. SSL 3.0 Specification.
13. Ramsdell, B. “S/MIME Version 3.1 Message Specification”, Internet Draft, February 2002.
14. Rivest, R. The MD5 Message Digest Algorithm. RFC 1321, April 1992.
15. Rivest, R. “The RC4 Encryption Algorithm”, RSA Data Security, Mar 1992.
16. Schiffman, A., Rescorla, E. The Secure Hypertext Transfer Protocol. Internet Draft, June 1998.
17. Schneier, B. Description of a New Variable-Lenght Key, 64-Bit Block Cipher (Blowfish). Fast Security Workshop Proceedings, Springer-Verlag, 1994, pp. 191–204.
18. Zimmermann, P.R. The Official PGP User’s Guide. MIT Press, 1995.

## Author Index

- Adamson, William A. 314  
Armington, John 1
- Blundo, Carlo 30  
Brown, Richard 59
- Cánovas, Óscar 214  
Casassa Mont, Marco 59  
Chapman, Mark 276  
Chen, L. 260  
Clercq, Jan De 40
- D'Arco, Paolo 30  
Das, Devaraj 115  
Davida, George 276  
Davila, Jorge 325
- Enzmann, Matthias 199
- Fenkam, Pascal 180  
Flegel, Ulrich 162
- Galdi, Clemente 30  
Gall, Harald 180  
Gómez, Antonio F. 214  
Gürgens, Sigrid 227
- Harrison, K. 260  
Ho, Purdy 1
- Jazayeri, Mehdi 180
- Kim, HongGeun 145  
Kim, Hyun-Sung 303  
Kim, HyungJong 145  
Koh, KyungHee 145  
Kornievskaia, Olga 314  
Koznek, Paul 1  
Krishnan, Venky 115
- Kruegel, Christopher 180  
Kunz, Thomas 199  
Kwon, Taekyoung 288
- Lopez, Javier 246, 325
- Martínez, Gregorio 214  
Martínez, Humberto 214  
Martinez, Richard 1  
McEvoy, Neil 88  
Montenegro, Jose A. 325
- Ochsenschläger, Peter 227  
Ortega, Juan J. 246
- Pagliusi, Paulo S. 129
- Reddy, Prakash 115  
Rila, Luciano 19  
Roman, Rodrigo 325  
Rudolph, Carsten 227
- Santis, Alfredo De 30  
Schechter, Stuart 73  
Schneider, Markus 199  
Shin, DongHoon 145  
Siougle, E.S. 104  
Smart, N.P. 260  
Soldera, D. 260
- Troya, Jose M. 246
- Whitcombe, Andrew 88
- Yoo, Kee-Young 303
- Zhang, Kan 115  
Zorkadis, V.C. 104