Lecture Notes in Computer Science 4582

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Javier Lopez Pierangela Samarati
Josep L. Ferrer (Eds.)

Public Key Infrastructure

4th European PKI Workshop:
Theory and Practice, EuroPKI 2007
Palma de Mallorca, Spain, June 28-30, 2007
Proceedings

Springer

Volume Editors

Javier Lopez
University of Malaga, Computer Science Department, E.T.S.I. Informatica
Campus Teatinos, 29071 Malaga, Spain
E-mail: jlm@lcc.uma.es

Pierangela Samarati
Università degli Studi di Milano, Dipartimento di Tecnologie dell'Informazione
v. Bramante 65, 26013 Crema, Italy
E-mail: samarati@dti.unimi.it

Josep L. Ferrer
University of Balearic Islands, Computer Science Department
07122 Palma, Spain
E-mail: dmijfg0@uib.es

Library of Congress Control Number: 2007929479

CR Subject Classification (1998): E.3, D.4.6, C.2.0, F.2.1, H.3, H.4, K.4.4, K.6.5

LNCS Sublibrary: SL 4 – Security and Cryptology

ISSN 0302-9743
ISBN-10 3-540-73407-4 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-73407-9 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12085347 06/3180 5 4 3 2 1 0

Preface

These proceedings contain the papers accepted at the 2007 European PKI Workshop: Theory and Practice (EuroPKI 2007), held in Palma de Mallorca, Spain, during June 28–30, and hosted by the Computer Science Department of the University of Balearic Islands (UIB) with the support of the Balearic Islands Government and the Private Law Department at UIB. This year's event was the fourth event in the EuroPKI Workshops series. Previous events of the series were held in: Samos, Greece (2004); Kent, UK (2005); and Turin, Italy, (2006).

In response to the call for papers, 77 papers were submitted to this year's workshop, setting a record of the highest number of papers submitted to an EuroPKI event so far and confirming an increased interest in PKI research and in the EuroPKI event. Each paper was reviewed by three members of the Program Committee, and evaluated on the basis of its significance, novelty, technical quality and relevance to the workshop. The paper selection process was very competitive: of the papers submitted, only 21 full papers and 8 short papers were selected for presentation at the workshop and inclusion in this volume.

We would like to thank all those people who, in a different capacity, contributed to the realization of this event. Thank you to all the members of the Program Committee and to the external reviewers for their constructive and insightful comments during the review process. Thank you to the staff of Springer for their co-operation and their excellent work during the publication process. Special thanks to: the members of the Organizing Committee for their hard work in dealing with all matters related to the conference organization; to the sponsors, for their valuable support; and to the invited speakers, Bart Preneel and Gene Tsudik, for accepting our invitation to deliver a talk.

Finally, we would like to thank all the authors who submitted papers to the workshop, including those whose submissions were not selected for publication, and all the workshop attendees. We hope that you find the proceedings interesting.

June 2007

Javier Lopez
Pierangela Samarati
Josep L. Ferrer

EuroPKI 2007
Fourth European PKI Workshop: Theory and Practice

Palma de Mallorca, Spain
28-30 June, 2007

Organized by
Computer Science Department
University of Balearic Islands
Spain

Program Co-chairs

Javier Lopez	University of Malaga, Spain
Pierangela Samarati	University of Milan, Italy

General Chair

Josep L. Ferrer	University of Balearic Islands, Spain

Workshop Chair

Llorenç Huguet	University of Balearic Islands, Spain

Program Committee

Carlisle Adams	University of Ottawa, Canada
Oscar Canovas	University of Murcia, Spain
Sabrina De Capitani di Vimercati	University of Milan, Italy
David Chadwick	University of Kent, UK
Marco Cremonini	University of Milan, Italy
Jorge Davila	Polytechnic University of Madrid, Spain
Ed Dawson	Queensland University of Technology, Australia
Stephen Farrell	Trinity College Dublin, Ireland
Jordi Forne	Polytechnic University of Catalonia , Spain
Dieter Gollmann	Hamburg University of Technology, Germany
Stefanos Gritzalis	University of the Aegean, Greece
Dimitris Gritzalis	AUEB, Greece
Socrates Katsikas	University of the Aegean, Greece
Stephen Kent	BBN Technologies, USA
Kwangjo Kim	ICU, Korea
Chi-Sung Laih	National Cheng Kung University, Taiwan
Antonio Lioy	Politecnico di Torino, Italy

Fabio Martinelli	IIT-CNR, Italy
Apol·lonia Martinez	University of Balearic Islands, Spain
Fabio Massacci	University of Trento, Italy
Stig F. Mjølsnes	NTNU, Norway
Jose A. Montenegro	University of Malaga, Spain
Yi Mu	University of Wollongong, Australia
Rolf Oppliger	eSecurity, Switzerland
Eiji Okamoto	University of Tsukuba, Japan
Jong Hyuk Park	Hanwha S&C Co., Korea
Guenter Pernul	University of Regensburg, Germany
Bart Preneel	Katholieke Universiteit Leuven, Belgium
Chunming Rong	University of Stanvanger, Norway
Kouichi Sakurai	Kyushu University, Japan
Damien Sauveron	University of Limoges, France
Sean Smith	Dartmouth College, USA
Julien Stern	Cryptolog, France
Jianying Zhou	Institute for Infocomm Research, Singapore
Sencun Zhu	Penn State University, USA

Organization Committee

Magdalena Payeras (Chair)	University of Balearic Islands, Spain
Cristina Alcaraz	University of Malaga, Spain
Macia Mut	University of Balearic Islands, Spain
Antonia Paniza	University of Balearic Islands, Spain
Rodrigo Roman	University of Malaga, Spain

External Reviewers

François Arnault, Giulia Boato, Pierre-François Bonnefoi, Serge Chaumette, Wolfgang Dobmeier, Stelios Dritsas, Pierre Dusart, Ludwig Fuchs, Félix J. Garcia-Clemente, Theodoulos Garefalakis, Xinyi Huang, John Iliadis, George Kambourakis, Jan Kolter, Elisavet Konstantinou, Dimitris Lekkas, Ioannis Marias, Antonio Muñoz, Gregory Neven, Sassa Otenko, Vincent Rijmen, Ayda Saidane, Rolf Schillinger, Christian Schläger, Kei Lei Shi, Marianthi Theoharidou, Bill Tsoumas, Frederik Vercauteren, Guilin Wang, Artsiom Yautsiukhin, Nicola Zannone.

Sponsors

Govern de les Illes Balears

Conselleria de d'Economia, Hisenda i Innovació

Direcció General de Recerca, Desenvolupament Tecnològic i Innovació

Table of Contents

Authorization Architectures for Privacy-Respecting Surveillance

Ulrich Flegel and Michael Meier

University of Dortmund, D-44221 Dortmund, Germany
{ulrich.flegel|michael.meier}@udo.edu

Abstract. Even more than in our physical world, in our digital world we need systems that meet the security objective of service providers and users in equal measure. This paper investigates the requirements of secure authorizations with respect to accountability and privacy in the context of surveillance for misuse detection during service utilization. We develop a model of system architectures for secure and privacy-respecting authorizations that allows to derive and compare the properties of available technology. It is shown how the model maps to existing authorization architectures.

Keywords: Architecture, authorization, privacy, pseudonym, surveillance, misuse detection, intrusion detection.

1 From Physical to Digital: A Short Visit to the Zoo

Many safeguards in the digital world mimic safeguards in the physical world. The reason probably is that safeguards are necessary, if the actors do not trust each other. However, at the end of the day, trust is usually anchored in the physical world. Using an example it is shown how we deal with trust in the physical world. In the following, we describe the case of a student who wants to visit the zoo. In the example the zoo serves as a service provider offering free admission to students. Non-students might feel tempted to defraud the zoo by pretending to be a student in order to obtain free admission. Hence, the personnel at the zoo ticket booth is instructed not to trust statements that customers make about their own property as a *student*. For customers it is thus insufficient claiming to be a *student*, also because the ticket booth personnel cannot verify the statement without considering supporting documents. Instead, it is required to show a valid student ID. The student ID is used as a certified property statement that assigns the name of the subject of the statement to the property *student*. At the ticket booth a certified property statement is accepted, if it is a student ID, as a matter of policy the issuing university is trusted to generate useful property statements, the person on the picture visually matches the presenting person, the student ID has not yet expired and looks "genuine".

If the student ID is accepted at the ticket booth, the presenting person is authorized to pass the zoo entrance. The presenting person receives the service-specific property *authorized for zoo entrance*. Therefore customers that are *authorized for zoo entrance* receive an admission ticket at the ticket booth. The

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 1–17, 2007.

ticket is accepted at the zoo entrance, if the stated ticket booth is trusted to issue tickets only to persons that are *authorized for zoo entrance*, the ticket number looks "plausible", the ticket authorizes to pass the zoo entrance, it has not yet expired and looks "genuine".

If the admission ticket is accepted at the zoo entrance, the student may enter the zoo. Right in the front is a sign that specifies behavior that is by policy prohibited in the zoo. Most notably, it is prohibited to tease the monkeys, since they may take revenge using banana peel projectiles. Thus, for the time being, the zoo trusts that the visitors stick to the rules. At critical areas (at the monkey house) the zoo may put a guard in place. The guard observes the behavior of the visitors and reacts, if he detects a violation of the zoo policy.

This paper presents a model for authorization architectures and criteria for deriving and comparing generic high-level properties of existing privacy-enhancing technologies when applied to surveillance for misuse detection. The model and criteria are developed in four steps:

- Generalizing the hybrid PKI model of Biskup and Karabulut [1] by abstracting from PKI-specific technology an architecture model for secure authorizations is developed, which primarily meets the security interests of service providers (see Sect. 2).
- Our previous work on pseudonyms [2] is generalized for the model to solve the privacy problems created by surveillance data, thereby enabling lawful misuse detection. What distinguishes our pseudonym approach from related work is the integrated notion of technical purpose binding for pseudonym disclosure (see Sect. 3).
- Combining the model from Sect. 2 with pseudonyms results in an architecture model for secure and privacy-respecting authorizations (see Sect. 4).
- Given the model, criteria are developed to derive and compare generic high-level properties of privacy-respecting authorization architectures (see Sect. 5). It is shown how the model can be applied to existing privacy-enhancing technologies (see Sect. 6).

The proposed model is compared to existing models in Sect. 7 and the paper concludes in Sect. 8 with a summary of the contributions.

2 An Architecture Model for Authorizations

Based on the assumption that services do not generally trust in property statements that users make on their own behalf, authorization architectures rather rely on property statements that are responsibly certified by agents trusted by the service. In the proposed model individuals, computers and other players in a distributed IT system are denoted as *entities*. A *principal* is a bit string that is unique within its scope of application and it is associated with an entity to serve as its surrogate. An entity can enjoy *properties*, which in turn may be used in conditions in authorization policies, and are taken into account during the trust evaluation.

The terms *certification* and *certificate* in the model denote the process and the result, respectively, when a responsible agent certifies a statement about rather *entity-specific* than service-specific properties in its role as a *certifier*. In Sect. 1 the student ID is a certificate, which expresses the certified statement about the entity-specific property *student*.

In the model the term *authorization* denotes the process and the result, when a responsible agent certifies a statement about *service-specific* properties in its role as an *authorizer*. In Sect. 1 the zoo admission ticket is an authorization, which expresses the certified statement about the service-specific property *authorized for zoo entrance*.

A *responsible agent* (cf. the university or the zoo ticket booth in Sect. 1) verifies that a *subject* entity enjoys certain properties and certifies a statement under one of his own principals, such that the statement assigns a principal of the subject to property attributes that correspond to the verified properties of the subject. The association of the subject principal with the presenting entity is verifiable by means of authentication data. A property statement also contains verifiable data concerning the validity of the statement, where the data can only be generated by the responsible agent and practically cannot be counterfeited. Note that certified property statements come in different forms, such as static documents (e.g. certificates [1]) or as traces of interactive protocols (e.g. anonymous credentials [3]).

Property statements comprise the following *components* (see Fig. 1): A principal of the *responsible agent* for the trust evaluation, parameters for checking the *validity*, *authenticity* parameters for authenticating the presenting entity, a set of *attributes* expressing the subject properties and being evaluated for the access decision, and the *subject* principal, which is used for linking property statements while processing service requests.

The components of certified property statements primarily support security objectives of the service providers. Accordingly the fat light grey frames in Fig. 2 enclose the system components where the service-related security objectives are enforced and which must not be controlled by the user. In Fig. 2 the solid arrows indicate the flow of certified or verified property statements.[1] In the text the arrows are referenced by their identifiers (here: 'A1' to 'C2').

The players in the basic model in Fig. 2 are the user-side management, a certifier, an authorizer and a service. As an example, in *Kerberos* [4] they can be mapped to the client, the *authentication server*, the *ticket granting server*, and the service. The authorization for the utilization of a service can be broken down into the following three phases: 1) The user has his relevant properties certified (see 'A1', 'A2' and 'A3' in Fig. 2). 2) Presenting his relevant certificates the user is authorized for the utilization of the service (see 'B1', 'B2' and 'B3' in Fig. 2). 3) Presenting the authorization the user can utilize the service (see 'C1' and 'C2' in Fig. 2). The management is controlled by the user. It interacts with other players, and based on the policy of the user, and aiming at satisfying the

[1] We assume that the service answer does not include statements about the properties enjoyed by the user. Hence, the service answers are not shown in the model.

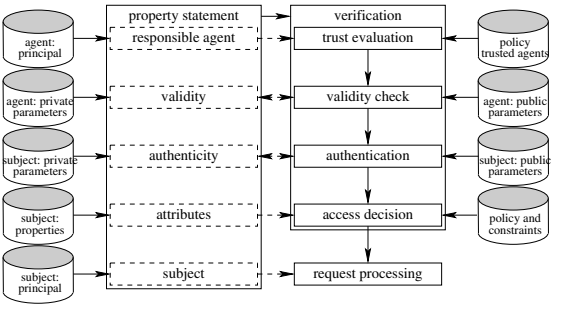


Fig. 1. Verification of property statements

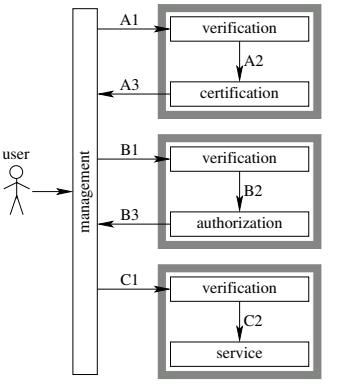


Fig. 2. Basic model

requirements of the service or responsible agent (see the policies in Fig. 1), it chooses property statements and information that are suitable for the respective interaction. There are slightly different variants of the basic model reflecting shortcuts that existing authorization architectures take [5].

3 Pseudonyms with Technical Purpose Binding of Disclosure

As described in Sect. 2 the access decision for service utilization is based on certified property statements including a subject principal. While processing a service request for a subject, in the system occur events that are triggered by the step-wise computation of the answer to the request, where these events usually are linked to the subject principal. That is, system events are accountable, usually with the objective of accounting detected misuse to the perpetrator. The working principle of surveillance technology for misuse detection is based on analyzing relevant system events manifested in *audit data* (cf. guard in Sect. 1). Appropriately responding to detected misuse, i.e. locking out some user or filtering source IP addresses, requires that the manifestation of the misuse in the audit data be accountable.

When collecting and processing audit data the obvious conflict between the individual user's interest in privacy and the overall security objective accountability can be solved by using *pseudonyms* in the audit data instead of subject principals. In the sense of *multilateral security* [6] a fair solution can be achieved by distinguishing the normal case (no accountability) and the exceptional case (accountability can be established) by controlling the external knowledge about (parts of) the *pseudonym mapping*. We discuss the legal foundation of such a solution elsewhere [5] and define a *pseudonym* as a principal that does not allow the identification of the assigned entity, based on the definitions of Pfitzmann and Hansen for *unlinkability* and *anonymity* [7]. Further concepts of

pseudonymization, *pseudonym mapping*, *pseudonym disclosure* and *reidentification* building on the aforementioned definitions are used intuitively here and defined in more detail elsewhere [5].

The *controlled disclosure* of pseudonyms is the controlled ability to make pseudonymized objects accountable again. This ability is controlled by enforcing who can use the pseudonymity mapping. The entity which manages the pseudonym mapping is responsible for performing reidentification for legal purposes of authorized entities only. We say that pseudonym disclosure is subject to *purpose binding*, if it is granted only for some a priori specified purpose, e.g. responding to detected misuse. If the responsible handling is conferred to a person, the reidentification is subject to *organizational purpose binding*. Since this person needs to manually perform the purpose binding, pseudonym disclosure may be delayed and not be sufficiently fast for a timely misuse response. Alternatively the purpose of pseudonym disclosure can already be incorporated during pseudonym generation. The pseudonymized audit data is automatically supplemented with certain information that neutralizes the protection of the pseudonym mapping under certain conditions. The purpose of pseudonym disclosure determines under what conditions the protection becomes ineffective, and (parts of) the pseudonym mapping can be used for reidentification. The pseudonyms can be disclosed only if these conditions are met. If the protection of the pseudonym mapping is customized for the disclosure conditions, such that it cannot be circumvented – e.g. by means of cryptography [2], the pseudonyms are subject to disclosure with *technical purpose binding*. The advantages of technical purpose binding will become apparent in the context of the architecture model in Sect. 4.

4 An Architecture Model for Privacy-Respecting Authorizations

In many cases person-identifying IDs are not necessary to verify certified property statements and to provide a service [8]. If for a given application IDs are not necessary, property statements and their references can be pseudonymized by replacing the subject principal with a pseudonym. As an example, the German act on digital signatures already allows for pseudonymous certificates (§7 Sect. 1-3 SigG [9]). In the example from Sect. 1 the admission ticket to the zoo does not need to contain the name of the ticket owner. Instead, it has a unique ticket number, which can be interpreted as a pseudonym of the ticket owner in the context of the zoo service.

On the one hand, the agent now is additionally responsible to the interest of accountability of the recipients of the property statement, for disclosing pseudonyms in accord with his pre-engaged policy to specific entities for specific purposes only. On the other hand, the agent is also responsible to the interest of the subject entity in pseudonymity, for protecting the pseudonym mapping and adhering to the declared policy w.r.t. pseudonym disclosure and linkability.

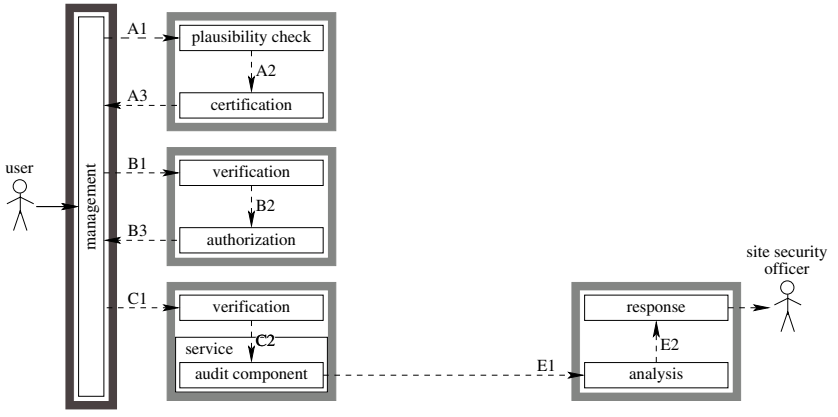


Fig. 3. Unilateral security: management anonymizes

In the following, the basic model from Fig. 2 is extended with the *site security officer* (SSO) of the service provider, who, by means of audit data, observes and analyzes the behavior of the service users, and if necessary, conducts appropriate response (see Fig. 3). The audit data is collected by the *audit component* of the service and is conveyed to the *analysis* component of the SSO (see 'E1' in Fig. 3). According to the *purpose of analysis*, i.e. purpose of processing, the analysis component generates *event reports* and provides them to the *response* component (see 'E2' in Fig. 3). The response component reacts on the event reports, for example by informing the SSO and by suggesting appropriate action. An event report can comprise an *analysis context*, which is a sub-set of the audit data. For this text we consider an intrusion detection system (IDS) as an instance of the described additional components, where the purpose of processing of the analysis component is the detection of misuse scenarios[2] that are caused by the service users.

Fig. 3 to Fig. 6 depict the privacy-respecting versions of the basic model (cf. Fig. 2), where user IDs are pseudonymized before they can be observed by the SSO in the audit data. The graphical elements in the figures call for some explanation. The *solid arrows* indicate the flow of accountable and certified or evidenced property statements. The *dashed arrows* indicate the flow of anonymous or pseudonymous property statements. The *dotted arrows* indicate the flow of the pseudonym mapping. Each *fat grey frame* indicates the control requirement of a certain entity w.r.t. the framed components. An entity B must not control the components implementing the interest I_A of another entity A, which is in conflict with the interest I_B of B. The *dark grey frames* represent the user's interest in pseudonymity. Conversely, the *light grey frames* represent the SSO's interest in accountability. Finally, the *black boxes* together implement a

[2] Models of misuse scenarios are activity patterns that are known to the IDS, i.e., here we consider so-called misuse detection, but not so-called anomaly detection.

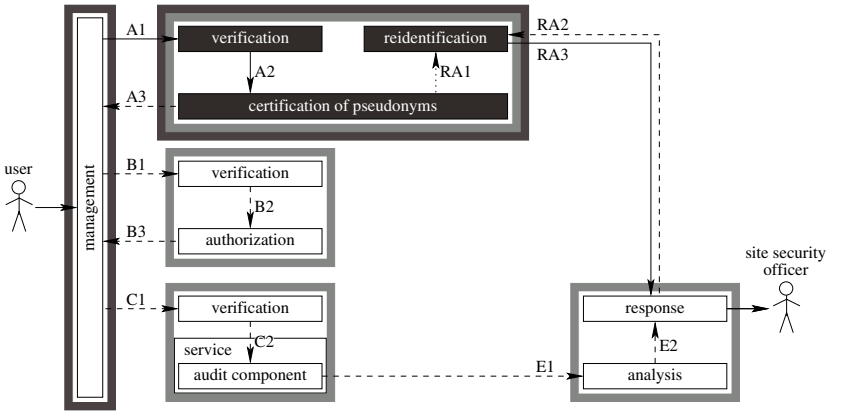


Fig. 4. Multilateral security: certification of pseudonyms

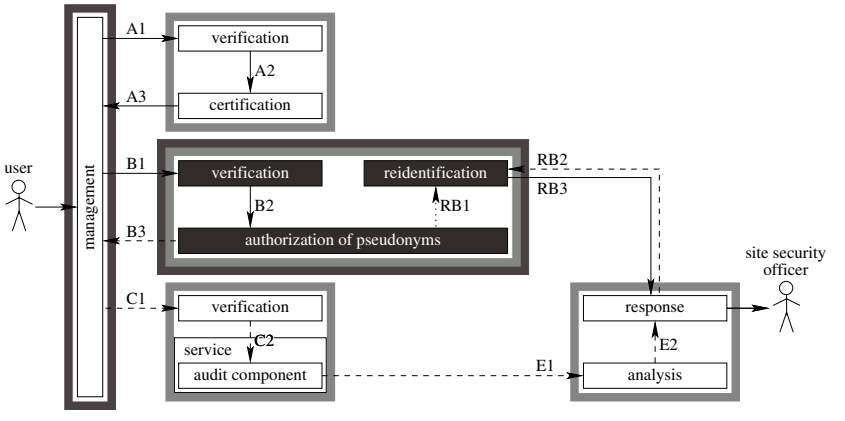


Fig. 5. Multilateral security: authorization of pseudonyms

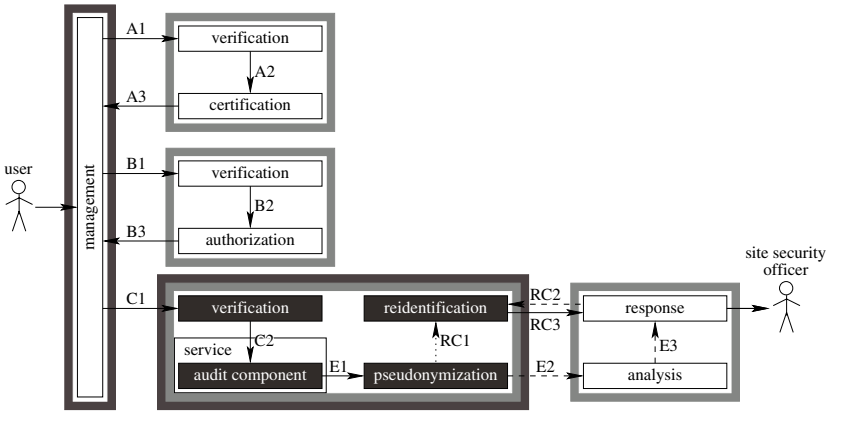
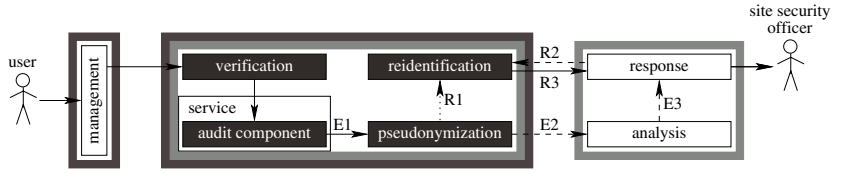
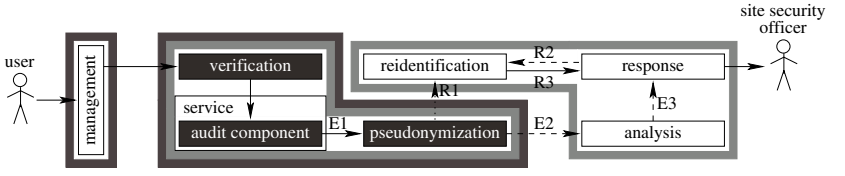


Fig. 6. Multilateral security: pseudonymization of audit data



(a) organizational purpose binding



(b) technical purpose binding

Fig. 7. Purpose binding of controlled pseudonym disclosure

function of multilateral security. Note that they are surrounded by a dark as well as by a light grey frame, i.e., the interests (pseudonymity and accountability) are clashing and need to be balanced.

A unilaterally secure architecture can be built in favor of anonymity. In such an architecture the certifier does not verify that the subject component provided by the user contains an ID which actually identifies the user (see plausibility check in Fig. 3).[3] The user's management component can then choose arbitrary pseudonyms for the property statement and the corresponding pseudonym mapping is also controlled by the user's management component. The SSO would have to rely on the user to disclose his pseudonyms, i.e. dependable accountability is not possible.

Architectures providing multilateral security take conflicting interests into account [6], such that the entities, who pursue the conflicting interests, should not be able to control the objects of interest, i.e. the pseudonyms in the property statements. Instead, for multilateral security the pseudonym mapping could be controlled by one or more agents, which the users and the SSO need to trust (see Fig. 4 to Fig. 6). The presented architectures can also be adapted to the variants mentioned in Sect. 2 [5].

The architectures depicted in Fig. 4 to Fig. 6 only make use of pseudonyms with disclosure subject to organizational purpose binding. Fig. 7 shows for the architecture with audit data pseudonymization at the service layer,[4] how the control requirements can be relaxed by using technical purpose binding for pseudonym disclosure, instead of organizational purpose binding.

For technical purpose binding the pseudonym mapping is provided to the reidentifier in a protected form (see 'R1' in Fig. 7b). Additionally the pseudonymous audit data is supplemented with information needed to neutralize the protection

[3] Note, that this is actually the case for many web-based services on the Internet.

[4] Technical purpose binding is also possible in the certification and authorization layers of the model, but yields varying benefit (see Sect. 5).

of the pseudonym mapping (see 'E2' in Fig. 7b). Due to the nature of the protection of the pseudonym mapping, reidentification is only possible in accordance with the a priori defined purpose of controlled pseudonym disclosure. As a result, the user does not need to trust the entity any more, which controls the reidentification component. Hence, the SSO may control the reidentification component and may disclose pseudonyms in a timely and autonomous fashion, as soon as the respective purpose permits.

5 Comparing Architectures

Fig. 3 to Fig. 6 depict the different phases or layers where pseudonyms can be introduced in the model, such that the analysis component works only on pseudonymized audit data. Introducing pseudonyms in a given layer or phase has specific benefits and disadvantages, which are investigated in the following and summarized in Table 1.

Table 1. Summary of architecture properties, grouped by relation to the issues of trust, security and cost of deployment. Each criterion can be '√'=met, '−'=not met, or '%'=irrelevant in the given context.

property criteria	pseudonymizing entity			
	management	certifier	authorizer	service
multilateral security	−	√	√	√
independence of service	√	√	−	−
dependable attributes	−	√	√	%
technical purpose binding	−	−	√	√
verifiability of pseudonyms b.a.	√	√	√	−
independence of user	−	−	−	√
independence of infrastructure	√	−	−	√

In the model can *multilateral security* only be supported by entities which do not pursue one of the conflicting interests that they are supposed to balance.

Even if an entity does not itself pursue a certain security objective, the organization it is affiliated with and which it depends on still can pursue a certain security objective. Due to the dependence on an organization, the entity's activity could be biased in favor of the organization's interests. In the physical world, one hopes to avoid the problem of biased decision-making by conceding an elected person a secure position within the organization, such that he can make decisions that are in conflict with the organization he depends on, without thereby threatening his own employment (see *independence of service* in Table 1). As an example, such a position has been created by the German labor law for the works council and by the German privacy law for the privacy commissioner.

Depending on which entity responsibly certifies a pseudonymous property statement, can the evaluating party rely on the statement, i.e. that the respective entity or person actually enjoys the certified properties. From the perspective of

the service provider an agent, which pursues the security objectives of the service provider, can be trusted to provide *dependable attributes* in property statements.

While in Sect. 4 *technical purpose binding* of pseudonym disclosure is described for audit data, in principle it can also be realized for the authorizer. Considering technical purpose binding for the certifier, one has to bear in mind that a given certificate is used to acquire authorizations for various services with various purposes for processing and for audit data analysis. The pseudonyms and the respective technical purpose binding would have to support all of these anticipated purposes for disclosure as well as linkability. This would come along with a massive erosion of the pseudonymity of the respective certificates, such that it seems inappropriate to realize technical purpose binding for pseudonymizing certifiers.

As long as pseudonyms are introduced before the service access phase, the service can verify the pseudonymous authorization and the properties of the pseudonyms (see *verifiability of pseudonyms b.a.* in Table 1). Service requests with invalid pseudonyms can be detected by the service's verification component and can be rejected to avoid losses.

If the pseudonymization does not rely on a software component that is controlled or operated by the user, the pseudonymization is said to be *independent of the user*. On the one hand, this leaves the user out of the control loop, and he can independently take additional measures. On the other hand, the service provider is anyway obliged to comply with the privacy law and cannot shift this obligation to the users [5]. Moreover, a software component, which needs to be made available to the user, generates additional cost.

The architectures based on certificates and authorizations require trustworthy agents for certification and authorization, respectively. The effort for establishing such an infrastructure must not be underestimated. Independence of infrastructure therefore is in the interest of a quick and cost-efficient deployment of anonymity or pseudonymity.

6 Mapping Existing Architectures to the Model

In the following, for each of the pseudonymizing entities in Table 1, exemplary privacy-enhancing technologies (PETs) are mapped to the model proposed in Sect. 4. The selection does not claim to give a comprehensive or representative survey over PETs. Rather the intention is to give an impression how the model can be used to classify PETs in the context of authorization.[5]

6.1 Architectures with Pseudonymizing Management

Identity management components installed on the user's personal device (e.g. personal digital assistant, PDA) assists the user with creating and selecting his partial identities or identity profiles, which contain property statements.

[5] The selection intentionally does not cover all possibilities for acting pseudonymously or anonymously, for example anonymous publishing, anonymous elections, anonymous auctions, anonymous (peer-to-peer) file-sharing, Private Information Retrieval (PIR) and its applications are not considered.

Instead of locating this functionality on the user device, it can also be located at one or more third parties, also denoted as *infomediaries*, which the user trusts [10], such as Proxymate a.k.a. Lucent Personalized Web Assistant (LPWA) [11, 12].

Property statements to be sent out are selected, e.g. using P3P, by matching the security requirements and the privacy policy of the given recipient to the privacy requirements tied to the partial identities defined by the user, while considering the actual situation in which the user acts [12, 13]. In analogy to the trust evaluation carried out by the recipients of property statements, the user's management component evaluates the trust w.r.t. the recipient's privacy policy, before selecting and sending a property statement.

6.2 Architectures with Pseudonymizing Certifier

To effectively provide anonymous communication in distributed systems, personally identifying data must be avoided in all layers of the OSI reference model. Hence, anonymous services in the application layer require additional services that provide for anonymous communication. Secure anonymous communication services may also support conditional anonymity [14]. As an example, Mix systems distribute the trust, which the user needs to invest, over several autonomous parties. There are various implementations of Mix systems: Onion Routing/TOR, Hordes, Freedom Network, JAP, Babel and Mixmaster-Remailer. Crowds and Cypherpunk-Remailer are based on similar concepts. Simpler systems, which do not distribute the necessary trust, are or were for example Anonymizer.com, Anonymouse and Anon.penet.fi. Surveys of these technologies have been published by several authors [15, 16, 17, 18, 19].

Anonymous or pseudonymous credentials are introduced as anonymous or pseudonymous property statements in Sect. 4. The literature offers various approaches for implementation [20, 21, 22, 23, 24, 3, 25].

Verifying anonymous or pseudonymous property statements comprises anonymously or pseudonymously authenticating the presenting party (see authenticity component in Sect. 2). There are several proposals for authentication technology subject to controlled identity disclosure [26, 27, 28], or at least with strong mechanisms to discourage the unauthorized sharing of pseudonyms with other users [29]. Anonymous authentication is frequently realized using group signatures.

Fair electronic offline cash usually provides for controlled identity disclosure subject to technical purpose binding in the case that someone spends a given electronic coin more than once (commonly denoted as *double spending*) [17, 30, 31, 32, 33, 34].

For the privacy-enhanced intrusion detection system ANIDA the *Kerberos authentication server* was conceptually extended to use pseudonyms with controlled disclosure subject to organizational purpose binding [35].

6.3 Architectures with Pseudonymizing Authorizer

Anonymous credentials, coins and anonymous authentication may also be employed for authorizers (cf. Sect. 6.2). We give only two examples. Based on the

fair electronic coins of Chaum et al. [36] Internet dial-in users can anonymously log-in to dial-in access points of their Internet providers [37]. A similar approach was proposed as a payment system for wireless LAN Hotspots [38].

Serial transactions can be authorized in a completely unlinkable fashion by extending the validity of one-show credentials at each use for the following transaction only [39].

Büschkes and Kesdogan also proposed a second approach to privacy-enhanced intrusion detection, where the *Kerberos ticket granting server* is complemented with a multilaterally secure Mix [35].

6.4 Architectures with Pseudonymizing Service

In the following is only personal data considered that has already been collected by a service in the form of audit data for misuse detection. When considering service-side anonymization or pseudonymization, it is useful to keep the criteria summarized in Table 1 in mind. To be able to react timely on detected misuse, a timely pseudonym disclosure is desirable, preferably without the need to involve third parties. This can be realized using technical purpose binding of pseudonym disclosure. Also, the solution should be practical and independent from users and expensive infrastructures. As shown in Table 1 these requirements can only be simultaneously met at the service layer. In the following, approaches for anonymization or pseudonymization of audit data at the service layer are summarized.

In her seminal work on *Intrusion Detection and Avoidance* (IDA) Fischer-Hübner proposed the concept of misuse detection using pseudonymized audit data [40, 41]. The concept of pseudonymized audit data for misuse detection is used by Sobirey, showing that it is workable with operational intrusion detection systems. The IDA concepts have been integrated with the fully working IDS *Adaptive Intrusion Detection* (AID) [42]. Lundin developed a simple pseudonymizer for the audit data of an operational firewall, to be able to legally use the pseudonymized audit data for intrusion detection experiments [43]. Rieck developed the pseudonymizer *bsmpseu* to pseudonymize Solaris BSM audit data, which was used for intrusion detection experiments. We introduced an approach for pseudonymizing audit data for misuse detection in a multilaterally secure way, where the controlled pseudonym disclosure and pseudonym linkability are subject to technical purpose binding [2].

Further approaches to pseudonymizing audit data are known for web server log files that are aggregated for statistical purposes, e.g. [44], and for network traffic traces, which need to be shared for research purposes, e.g. [45].

7 Related Work

Other approaches or models have been proposed to describe anonymous or pseudonymous authorizations. In the following they are briefly outlined and mapped to our model.

The Dutch privacy authority *Registratiekamer* together with the information and privacy commissioner of Ontario, Canada, developed a model for information systems with a focus on privacy [8]. Based on this model, the authorization process, including the respective audit data, is described in analogy to the architecture, where the service holds the property statements, such that no further responsible agents are needed and the service needs not to verify the validity of the property statements [5]. Accordingly, the users merely obtain references to the statements about their properties. A so-called *Identity Protector* can be placed at several locations in the model. The Identity Protector acts as a pseudonymizing entity which separates components where user IDs are known from components, where merely the respective pseudonyms are processed. For each proposed placement of the Identity Protector the resulting architecture is described by van Rossum et al. [8], however without distinguishing the respective properties and specifying the control requirements. The Identity Protector corresponds to the management component in our model, when implemented near the user, i.e., in between of the user representation and the service. It corresponds to the certifier or authorizer when implemented as a third party between the user representation and the service. Finally, the Identity Protector corresponds to an audit data pseudonymizer, when implemented between the service representation and the audit data.

Alamäki et al. define various functional components (*Profile Broker, Identity Broker, Authenticator*) that are required for architectures for anonymous or pseudonymous authorizations [46], however without distinguishing the respective properties and specifying the control requirements. Identity Brokers are defined as entities which introduce pseudonyms, and Profile Brokers are user profile access points, where user profiles correspond to the attributes of property statements in our model. Profile Brokers can be complemented with Contract Brokers, which verifyably negotiate the mutual requirements of users and services w.r.t. disclosure of user profiles. In our model these brokers may be part of the user-side management component.[6] Alternatively the Identity Broker may reside at the certifier or authorizer.[7] Alamäki et al. define Authenticators as entities which provide for the authentication of users, which corresponds to the authentication part of the verification boxes in our model (see Fig. 1).

A recent approach describes *Privacy-enhancing Identity Management* (PIM) [12], where the user decides on his discretion, who can get which of his personal data, and where the user can separate his activity in different spheres, such that different addressees of his activity may have a different view of the partial identities (personae) of the user. PIM comprises the applications, the middleware and the communication infrastructure [12]. At the application layer the identity manager of the user (cf. management in our model) and the service provider (cf. service in our model) negotiate the requirements for partial identities (represented by property statements in our model). Beyond anonymous authorizations this approach also addresses e-commerce and e-government. Therefore,

[6] Trusted Mobile Terminal in Alamäki et al. [46].

[7] Physical Separation of Identity and Profile in Alamäki et al. [46].

PIM leverages not only pseudonymizing certifiers and authorizers (see anonymous credentials and authorizations in Sect. 6.2 and Sect. 6.3, respectively) and an infrastructure for anonymous communication (see Sect. 6.2), but also requires additional mediators or trustees for the digital exchange of goods, settling of liabilities, electronic payment (see Sect. 6.2), and finally, the delivery of physical goods in the physical world.

The above mapping shows that wrt. pseudonymous authorization existing models are subsumed by our model, while our model additionally provides advice concerning control requirements and suitability wrt. various high-level properties of authorization architectures.

8 Conclusion

In this paper we present an architecture model for secure and privacy-respecting authorizations. By generalizing the hybrid PKI model of Biskup and Karabulut [1] we firstly develop an architecture model for secure authorizations, which subsequently is extended for pseudonymity. The resulting model is more comprehensive than existing models.

With a focus on surveillance for misuse detection we identify suitable architectures and control requirements for pseudonymous authorization. Moreover, we provide criteria to determine and compare the properties of these architectures. The contribution to the area of privacy-respecting authorizations is threefold:

- The model provides a systematic view on architectures for secure and privacy-respecting authorizations, as well as on their generic high-level properties.
- Starting from a set of required properties it allows to compare and select suitable architectures, either for designing authorization systems from scratch, or to guide product selection.
- For each architecture the control requirements are made explicit, such that they can be taken into account during design, or can be used to verify the appropriateness of control conditions in products.

References

[1] Biskup, J., Karabulut, Y.: A hybrid PKI model with an application for secure meditation. In: Shenoi, S. (ed.) Proceedings of the 16th Annual IFIP WG 11.3 Working Conference on Data and Application Security, Cambridge, England, July 2002, pp. 271–282. Kluwer, Dordrecht (2002)

[2] Flegel, U.: Pseudonymizing Unix log files. In: Davida, G.I., Frankel, Y., Rees, O. (eds.) InfraSec 2002. LNCS, vol. 2437, pp. 162–179. Springer, Heidelberg (2002)

[3] Camenisch, J., Lysyanskaya, A.: An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 93–118. Springer, Heidelberg (2001)

[4] Gollmann, D.: 10.2.1: Kerberos. In: Computer Security, pp. 168–171. John Wiley & Sons, Inc, West Sussex (1999)

[5] Flegel, U.: Pseudonymizing Audit Data for Privacy Respecting Misuse Detection. PhD thesis, University of Dortmund, Dept. of Computer Science (January 2006)
[6] Pfitzmann, A.: Multilateral security: Enabling technologies and their evaluation. In: Wilhelm, R. (ed.) Informatics. LNCS, vol. 2000, pp. 50–62. Springer, Heidelberg (2001)
[7] Pfitzmann, A., Hansen, M.: Anonymity, unlinkability, unobservability, pseudonymity, and identity management - a consolidated proposal for terminology (May 2006) dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.28.pdf
[8] van Rossum, H., Gardeniers, H., Borking, J., et al.: Privacy-enhancing technologies: The path to anonymity, vol. ii, Technical report, Registratiekamer Netherlands and Information and Privacy Commissioner Ontario, Canada, Achtergrondstudies en Verkenningen 5B, Rijswijk, Netherlands (August 1995)
[9] Bundestag, D.D.: Gesetz über Rahmenbedingungen für elektronische Signaturen (SIGG) (in German). Bundesgesetzblatt, Teil I(1) (January 2005) 2, http://bundesrecht.juris.de/bundesrecht/sigg_2001/
[10] Gabber, E., Gibbons, P.B., Kristol, D.M., Matias, Y., Mayer, A.: On secure and pseudonymous client-relationships with multiple servers. ACM Transactions on Information and System Security 2(3), 390–415 (1999)
[11] Cranor, L.F.: Agents of choice: Tools that facilitate notice and choice about web site data practices. In: Proceedings of the 21st International Conference on Privacy and Personal Data Protection, Hong Kong SAR, China, September 1999, pp. 19–25 (1999)
[12] Clauß, S., Köhntopp, M.: Identity management and its support of multilateral security. Computer Networks 37(2), 205–219 (2001)
[13] Köhntopp, M., Berthold, O.: Identity management based on P3P. In: Federrath, H. (ed.) Designing Privacy Enhancing Technologies. LNCS, vol. 2009, pp. 141–160. Springer, Heidelberg (2001)
[14] Köpsell, S., Wendolsky, R., Federrath, H.: Revocable anonymity. In: Müller, G. (ed.) ETRICS 2006. LNCS, vol. 3995, pp. 206–220. Springer, Heidelberg (2006)
[15] Federrath, H.: Privacy enhanced technologies: Methods – markets – misuse. In: Katsikas, S.K., Lopez, J., Pernul, G. (eds.) TrustBus 2005. LNCS, vol. 3592, pp. 1–9. Springer, Heidelberg (2005)
[16] Fischer-Hübner, S.: IT-Security and Privacy: Design and Use of Privacy-Enhancing Security Mechanisms. LNCS, vol. 1958. Springer, Heidelberg (2001)
[17] Seys, S., Díaz, C., De Win, B., Naessens, V., Goemans, C., Claessens, J., Moreau, W., De Decker, B., Dumortier, J., Preneel, B.: Anonymity and privacy in electronic services (APES) Deliverable 2 – Requirement study of different applications. Technical report, K. U. Leuven (May 2001)
[18] Goldberg, I.: Privacy-enhancing technologies for the internet, II: Five years later. In: Dingledine, R., Syverson, P.F. (eds.) PET 2002. LNCS, vol. 2482, pp. 1–12. Springer, Heidelberg (2003)
[19] Goldberg, I., Wagner, D., Brewer, E.: Privacy enhancing technologies for the internet. In: Proceedings of the COMPCON'97, San Jose, California, USA, February 1997, IEEE (1997) http://www.cs.berkeley.edu/ daw/privacy-compcon97-www/privacy-html.html
[20] Chaum, D.: Showing credentials without identification: Transferring signatures between unconditionally unlinkable pseudonyms. In: Seberry, J., Pieprzyk, J.P. (eds.) AUSCRYPT 1990. LNCS, vol. 453, pp. 246–264. Springer, Heidelberg (1990)

[21] Van Herreweghen, E.: Secure anonymous signature-based transactions. In: Goos, G., Hartmanis, J., van Leeuwen, J. (eds.) ESORICS 2000. LNCS, vol. 1895, pp. 55–71. Springer, Heidelberg (2000)
[22] Brands, S.A.: Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy. MIT Press, Cambridge, Massachusetts, USA (2000)
[23] Glenn, A., Goldberg, I., Légaré, F., Stiglic, A.: A description of protocols for private credentials (October 2001) `http://eprint.iacr.org/2001`
[24] Stubblebine, S.G., Syverson, P.F.: Authentic attributes with fine-grained anonymity protection. In: Frankel, Y. (ed.) FC 2000. LNCS, vol. 1962, pp. 276–294. Springer, Heidelberg (2001)
[25] Lysyanskaya, A., Rivest, R.L., Sahai, A., Wolf, S.: Pseudonym systems. In: Heys, H.M., Adams, C.M. (eds.) SAC 1999. LNCS, vol. 1758, pp. 184–199. Springer, Heidelberg (2000)
[26] Schechter, S., Parnell, T., Hartemink, A.: Anonymous authentication of membership in dynamic groups. In: Franklin, M.K. (ed.) FC 1999. LNCS, vol. 1648, pp. 184–195. Springer, Heidelberg (1999)
[27] Gritzalis, D., Moulinos, K., Iliadis, J., Lambrinoudakis, C., Xarhoulakos, S.: Pythia: Towards anonymity in authentication. In: Dupuy, M., Paradinas, P. (eds.) Proceedings of the IFIP TC11 16th International Conference on Information Security (Sec'01), Paris, France, IFIP, June 2001, pp. 1–17. Kluwer Academic Publishers, Dordrecht (2001)
[28] Hirose, S., Yoshida, S.: A user authentication scheme with identity and location privacy. In: Varadharajan, V., Mu, Y. (eds.) ACISP 2001. LNCS, vol. 2119, pp. 235–246. Springer, Heidelberg (2001)
[29] Handley, B.: Resource-efficient anonymous group identification. In: Frankel, Y. (ed.) FC 2000. LNCS, vol. 1962, pp. 295–312. Springer, Heidelberg (2001)
[30] Davida, G., Frankel, Y., Tsiounis, Y., Yung, M.: Anonymity control in e-cash systems. In: Hirschfeld, R. (ed.) FC 1997. LNCS, vol. 1318, pp. 1–16. Springer, Heidelberg (1997)
[31] Camenisch, J., Maurer, U., Stadler, M.: Digital payment systems with passive anonymity-revoking trustees. In: Martella, G., Kurth, H., Montolivo, E., Bertino, E. (eds.) ESORICS 96. LNCS, vol. 1146, pp. 33–43. Springer, Heidelberg (1996)
[32] Claessens, J., Preneel, B., Vandewalle, J.: Anonymity controlled electronic payment systems. In: Proceedings of the 20th Symposium on Information Theory in the Benelux, Haasrode, Belgium, May 1999, pp. 109–116 (1999)
[33] Pointcheval, D.: Self-scrambling anonymizers. In: Frankel, Y. (ed.) FC 2000. LNCS, vol. 1962, pp. 259–275. Springer, Heidelberg (2001)
[34] Nakanishi, T., Haruna, N., Sugiyama, Y.: Unlinkable electronic coupon protocol with anonymity control. In: Zheng, Y., Mambo, M. (eds.) ISW 1999. LNCS, vol. 1729, pp. 37–46. Springer, Heidelberg (1999)
[35] Büschkes, R., Kesdogan, D.: Privacy enhanced intrusion detection. In: Müller, G., Rannenberg, K. (eds.) Multilateral Security in Communications. Information Security, pp. 187–204. Addison Wesley, Reading (1999)
[36] Chaum, D., Fiat, A., Naor, M.: Untraceable electronic cash. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 319–327. Springer, Heidelberg (1990)
[37] Chan, Y.Y.: On privacy issues of internet access services via proxy servers. In: Baumgart, R. (ed.) CQRE (Secure) '99. LNCS, vol. 1740, pp. 183–191. Springer, Heidelberg (1999)
[38] Groß, S., Lein, S., Steinbrecher, S.: A multilateral secure payment system for wireless LAN hotspots. In: Katsikas, S.K., Lopez, J., Pernul, G. (eds.) TrustBus 2005. LNCS, vol. 3592, pp. 80–89. Springer, Heidelberg (2005)

[39] Stubblebine, S.G., Syverson, P.F., Goldschlag, D.M.: Unlinkable serial transactions: Protocols and applications. ACM Transactions on Information and System Security 2(4), 354–389 (1999)
[40] Fischer-Hübner, S., Brunnstein, K.: Opportunities and risks of intrusion detection expert systems. In: Proceedings of the International IFIP-GI-Conference Opportunities and Risks of Artificial Intelligence Systems (ORAIS'89), July 1989, Hamburg, Germany, IFIP (1989)
[41] IDA (Intrusion Detection and Avoidance System): Ein einbruchsentdeckendes und einbruchsvermeidendes System (in German). Reihe Informatik. Shaker (1993)
[42] Sobirey, M., Richter, B., König, H.: The intrusion detection system AID – Architecture and experiences in automated audit trail analysis. In: Horster, P. (ed.) Proceedings of the IFIP TC6/TC11 International Conference on Communications and Multimedia Security, Essen, Germany, IFIP, September 1996, pp. 278–290. Chapman & Hall, London (1996)
[43] Lundin, E., Jonsson, E.: Anomaly-based intrusion detection: privacy concerns and other problems. Computer Networks 34(4), 623–640 (2000)
[44] Eckert, C., Pircher, A.: Internet anonymity: Problems and solutions. In: Dupuy, M., Paradinas, P. (eds.) Proceedings of the IFIP TC11 16th International Conference on Information Security (Sec'01), Paris, France, IFIP, June 2001, pp. 35–50. Kluwer Academic Publishers, Dordrecht (2001)
[45] Pang, R., Paxson, V.: A high-level programming environment for packet trace anonymization and transformation. In: Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, Karlsruhe, Germany, ACM SIGCOMM, August 2003, pp. 339–351. ACM Press, New York (2003)
[46] Alamäki, T., Björksen, M., Dornbach, P., Gripenberg, C., Györbíró, N., Márton, G., Németh, Z., Skyttä, T., Tarkiainen, M.: Privacy enhancing service architectures. In: Dingledine, R., Syverson, P.F. (eds.) PET 2002. LNCS, vol. 2482, pp. 99–109. Springer, Heidelberg (2003)

Privacy-Preserving Revocation Checking with Modified CRLs

Maithili Narasimha and Gene Tsudik

Computer Science Department
University of California, Irvine
{mnarasim,gts}@ics.uci.edu

Abstract. Certificate Revocation Lists (CRLs) are a popular means of revocation checking. A CRL is a signed and time-stamped list containing information about all revoked certificates issued by a certification authority. One of the shortcomings of CRLs is poor scalability, which influences update, bandwidth and storage costs. We claim that other (more efficient) revocation techniques leak potentially sensitive information. Information leaks occur since third parties (agents, servers) of dubious trustworthiness discover the identities of the parties posing revocation check queries as well as identities of the queries' targets. An even more important privacy loss results from the third party's ability to tie the source of the revocation check with the query's target. (Since, most likely, the two are about to communicate.) This paper focuses on privacy and efficiency in revocation checking. Its main contribution is a simple modified CRL structure that allows for efficient revocation checking with customizable levels of privacy.

Keywords: Anonymity and Privacy, Certificate Revocation.

1 Introduction and Motivation

Public key cryptography allows entities to establish secure communication channels without pre-established shared secrets. While entities can be assured that communication is confidential, there is no guarantee of authenticity. Authenticity is obtained by binding a public key to some claimed identity or name which is later verified via digital signatures in conjunction with public key certificates (PKCs). A public key certificate, signed by a recognized certification authority (CA), is used to verify the validity, authenticity and ownership of a public key. As long as the issuing CA is trusted, anyone can verify the CA's certificate signature and bind the included name/identity to the public key. Public key certificates work best in large interconnected open systems, where it is generally infeasible to directly authenticate the owners of all public keys. X.509 [23] is one well-known certificate format widely used in several Internet-related contexts. The peer-based PGP/GPG [2,7] format represents another popular approach.

Since a certificate is a form of a capability, one of the biggest problems associated with large-scale use of certificates is *revocation*. There are many reasons

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 18–33, 2007.

that can lead to a certificate being revoked prematurely. They include [23]: loss or compromise of a private key, change of affiliation or job function, algorithm compromise, or change in security policy. To cope with revocation, it must be possible to check the status of any certificate at any time.

Revocation techniques can be roughly partitioned into implicit and explicit classes. In the former, each certificate owner possesses a timely proof of non-revocation which it supplies on demand to anyone. Lack of such a proof implicitly signifies revocation. An example of implicit revocation is the Certificate Revocation System (CRS) [17]. Most revocation methods are explicit, i.e., they involve generation, maintenance and distribution of various secure data structures that contain revocation information for a given CA or a given range of certificates.

Certificate Revocation Lists (CRLs) represent the most widely used means of explicit revocation checking. Each certificate issuer periodically generates a signed list of revoked certificates and publishes it at (usually untrusted) public directories or servers. Inclusion of a certificate in the list signifies explicit revocation. Verifiers retrieve and cache the latest CRL and use it during certificate validation. Typically, a revoked certificate is included in a CRL from the time it is revoked until its validity period expires. Since certificate lifetime is typically measured in years, even modest revocation rates can result in very long accumulated CRLs. In bandwidth-constrained environments, transferring such CRLs can be expensive. Furthermore, since CRLs are published periodically, another potential concern is that many verifiers may request them around the time of publication. The burst of requests immediately following CRL publication may result in very high network traffic and can cause congestion. Thus, one of the biggest disadvantages of CRLs is the high cost associated with updating and querying the lists and this raises serious scalability concerns.

Other well-known explicit revocation methods include Certificate Revocation Trees (CRTs) [12] and Skip-Lists [8]. Another prominent technique is the On-line Certificate Status Protocol (OCSP) [18] which involves a multitude of validation agents (VAs) which respond to client queries with signed replies indicating current status of a target certificate. However, these explicit revocation methods have an unpleasant side-effect: they divulge too much information. Specifically, a third party (agent, server, responder or distribution point) of dubious trustworthiness knows: (1) the entity requesting the revocation check (source), and (2) the entity whose status is being checked (target). An even more important **loss of privacy** results from the third party tying the source of the revocation checking query to that query's target. This is significant, because revocation status check typically serves as a prelude to actual communication between the two parties. (We assume that communication between verifiers and on-line revocation agents (third parties) is private, i.e., conducted over secure channels protected by tools such as IPsec [10] or SSL/TLS [9,6].)

Given the continual assault on privacy by governments, spammers and just plain hackers, privacy leakage in certificate revocation checking is an important issue worth considering. Consider, for example, a certain country with a

less-than-stellar human rights record where mere intent to communicate (indicated by revocation checking) with an *unsanctioned* or *dissident* web-site may be grounds for arrest or worse. In the same vein, sharp increase in popularity (deduced from being a frequent target of revocation checking) of a web-site may lead authorities to conclude that something *subversive* is going on. Clearly, the problem can also manifest itself in other less sinister settings. For example, many internet service providers already keep detailed statistics and build elaborate profiles based on their clients' communication patterns. Current revocation checking methods – by revealing sources and targets or revocation queries – represent yet another source of easily exploitable (and misused) personal information.

Hiding sources of revocation queries can be easily achieved with modern anonymization techniques, such as onion routing, anonymous web browsing or remailers. While this might protect the source of a revocation query, the target of the query remains known to the third party. Furthermore, although anonymization techniques are well-known in the research community, their overall penetration remains fairly low. Also, in order to take advantage of an existing anonymization infrastructure, one either needs to place some trust in unfamiliar existing entities (e.g., remailers, re-webbers or onion routers) or make the effort to create/configure some of these entities.

Contributions: In this work, we propose privacy-enabling modifications to the well-known CRL structure. . Specifically, we provide a mechanism for verifiers to query untrusted online entities regarding the revocation status of a certificate without revealing to the latter information about the actual certificate of interest. Privacy requirements can be tuned to achieve verifier-specific level of privacy. This is achieved without the need for verifiers to retrieve the entire CRL.

Organization: The remainder of the paper is organized as follows: We discuss relevant prior work in section 2. Section 3 describes the proposed technique. We then discuss some practical implications of the proposed modifications in section 4. We analyze the scheme in section 5 and consider various overhead factors. We next outline some future directions and conclude in section 6. The appendix contains an overview of popular certificate revocation techniques (it is largely similar to the overview in [21]).

2 Related Work

The first effort that considered privacy in revocation checking is the work by Kikuchi [11]. It identified the problem and proposed a fairly heavy-weight (inefficient) cryptographic technique specific to CRLs. The solution relies on cryptographic accumulators [3] which are quite expensive.

The most closely related prior work is the recent paper [21] which proposes privacy-preserving extensions to Certificate Revocation Trees. It proposes that, instead of specifying the actual target certificate, a revocation query should include a range of certificate serial numbers. (The range includes the target

certificate.) As observed in [21], CAs sometimes assign consecutive serial numbers to consecutively issued certificates and groups of related certificates (e.g., issued to the same company) have consecutive serial numbers. Thus, information leaks can occur in cases when the query range subsumes or contains a large block of related consecutive certificate serial numbers. To mitigate this issue, [21] proposes sorting leaf nodes along *permuted* serial numbers. One example of a sufficiently random permutation involves using a block cipher (such as AES) with a fixed key. (Not all CAs assign consecutive numbers to certificates they issue; Verisign and Microsoft are two notable examples of CAs that issue pseudo-random numbered certificates)[1].

The main observation in [21] is that a reasonably large certificate range would normally contain very few revoked certificates since the density of revocation is generally quite sparse. Consequently, if CRTs are used, when one asks for a range of certificates, a small number of leaf and internal tree nodes can be returned in order to allow efficient verification of the entire query reply. In this paper, we use a similar approach to achieve privacy with CRLs. CRLs are a more accepted means of certificate revocation and we investigate techniques needed to achieve the level of privacy similar to that in [21].

A somewhat less related research topic is Private Information Retrieval (PIR) [4,13]. PIR refers to a set of cryptographic techniques and protocols that – in a client-server setting – aim to obscure the actual target(s) of database queries from malicious servers. Although PIR techniques could be applicable in our context, they tend to be relatively inefficient owing to either (or both) many communication rounds/messages or expensive cryptographic operations. As will be seen in subsequent sections, PIR techniques would amount to overkill in the context of privacy-preserving revocation checking.

3 Privacy-Preserving CRL Querying

We start by noting that a CRL trivially meets the privacy requirement, since in the typical usage scenario, a verifier simply downloads and stores the entire CRL. This allows it to locally perform any number of revocation checks against any certificate issued by a particular CA. However, downloading and caching a large CRL in its entirety is neither bandwidth nor storage efficient for the client. Although delta-CRLs tend to be smaller and more bandwidth-efficient, they still impose high storage overhead on the client. In addition, a client is required to periodically download the base-CRL as well as all subsequent delta-CRLs in order to keep the CRL cache recent and accurate. In this section, we propose a new CRL model and outline a mechanism which overcomes the aforementioned drawbacks, while allowing efficient and private revocation checking.

In our model, a trusted RA periodically generates CRLs and publishes them at untrusted online VA-s. A client interested in validating a certificate queries a VA to obtain authentic revocation information corresponding to that certificate.

[1] We thank one of the reviewers for pointing this out.

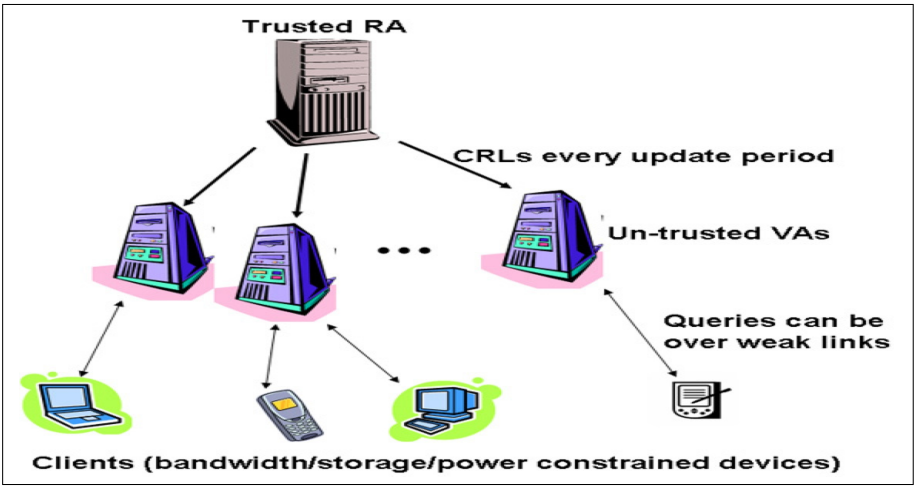


Fig. 1. System Overview

Although this model bears similarities to the OCSP method, unlike OCSP, the VA-s in our model can be third party responders who are not trusted by either clients or RA-s.

Since VA-s are untrusted, it is essential to ensure the integrity and authenticity of the revocation information of a certificate with respect to the RA responsible for that certificate. To achieve this, the RA individually signs each CRL entry. In other words, information regarding each revoked certificate is separately signed by the RA and placed onto the CRL. This differs from the usual CRL structure where all revoked certificates are included in the CRL which is then collectively signed once by the RA. Our modification clearly introduces some additional burden for the RA who is now required to sign each entry. However, we claim that this modification makes certificate revocation checking very efficient for clients; at the same time, it is not prohibitively expensive for a typical RA which is a computationally-powerful machine. (We analyze actual overheads in section 5.) This relatively minor modification obviates the need for the client to download the entire CRL.

3.1 (Non-Privacy-Preserving) Revocation Checking

When a VA receives a revocation status query from a client, it checks to see if there is a CRL entry corresponding to that certificate. If it exists, the VA sends back a **Revoked** reply along with the CRL entry signed by the RA, as the proof. However, if the certificate in question is not revoked, then simply sending a **Not Revoked** reply does not provide integrity and authenticity guarantees

with respect to the RA[2]. To achieve authenticity of **Not Revoked** replies, we suggest chaining of CRL entries, as described below.

Signature Chaining: To provide authenticity of query replies, the RA securely links the signatures of the revoked certificates to form a so-called *signature chain*. To construct a signature chain, the RA generates the individual signature for each revoked certificate in the following way:

Definition 1

$$Sign(i) = h(h(i)||h(IPC(i)))_{SK}$$

where i is the (permuted) serial number of a revoked certificate, $h()$ is a suitable cryptographic hash function, $||$ denotes concatenation, IPC denotes the immediate predecessor certificate and SK is the private signing key of the RA.

The immediate predecessor is a revoked certificate with the highest serial number which is less than the serial number of the current revoked certificate. In other words, the RA sorts all revoked certificates in ascending order by serial number and computes the signature of each revoked certificate by including the hash of the immediate predecessor, thereby explicitly chaining (linking) all signatures.

With signatures chained in the above fashion, when a client queries the status of an un-revoked certificate, the VA composes an **Not Revoked** reply by returning the two boundary certificates $CERT_a$ and $CERT_b$ that subsume the non-existent serial number along with the signature of $CERT_b$.

3.2 Privacy Preserving Revocation Checking

Our CRL modification allows a client to query the VA for revocation status of a certificate and obtain a concise and authentic proof of either revocation or validity. However, this introduces privacy concerns since clients now query the VA by a specific certificate serial number, thus revealing the identity of the target entity. To address this problem we employ the same technique as in [21]. Instead of querying by a specific certificate serial number i, we propose querying by a randomly selected range of serial numbers (j, k) with $j \leq i \leq k$. This effectively hides the certificate of interest. The only information divulged to the VA is that the target certificate lies in the interval $[j, k]$. This translates into the probability of correctly guessing i as: $P_i = \frac{1}{k-j+1}$. Each number in the range is equally likely to be the serial number of interest and the VA has no means, other than guessing, of determining the target certificate. Furthermore, the VA has no way of telling whether the actual query target is a revoked or a valid certificate.

Let n be the total number of certificates and m be the number of revoked certificates. Then, assuming uniform distribution of revoked certificate serial numbers over the entire serial number range, m/n is the fraction of revoked certificates and the very same fraction would be revoked within any $[j, k]$ range.

[2] Recall that VAs are untrusted in our model, and a malicious or lazy VA might not properly execute the query and send incorrect reply to the client.

Clearly, perfect privacy is not attainable with range queries. The highest possible privacy is $1/n$ which corresponds to querying the full certificate serial number range.[3] The lowest privacy level corresponds to querying by a specific serial number, i.e., setting $j = k = i$. The optimal query range is determined by the source of the query, i.e., the client. Several factors must be taken into account: (1) desired level of privacy e.g., if $k-j+1 = 1000$, the probability of correctly guessing i is P_i= 0.001, (2) additional bandwidth and storage overhead stemming from returning a set of CRL entries as a reply.

Once the range size (r) is determined, the client proceeds to set the actual range boundaries: j and k. To do so, it first generates a b-bit random number X where $b = log(r)$ or the bit-length of r. X determines the position, within the range, of the actual target certificate serial number. This step is necessary to randomize/vary the placement of the target. Next, the boundaries are set as: $j = i - X$ and $k = j + r - 1$. The client poses a query to the VA asking for the revocation status of all certificates within $[j, k]$. Since each entry corresponding to a revoked certificate in the $[j, k]$ range is linked only to its IPC (as noted above), it seems that the VA would need to send all signatures to the verifier. This is clearly inefficient for the verifier. However, we can modify the signature generation process as follows:

Sort all revoked certificates in ascending order of permuted serial numbers to obtain $\{CERT_0, CERT_1, CERT_2, \ldots, CERT_m, CERT_{m+1}\}$. The two "dummy" boundary values: $CERT_0$ and $CERT_{m+1}$ represent $-\infty$ and $+\infty$, respecively, i.e., they denote values outside the allowed range. Now, compute the signature of each revoked certificate by signing the **running hash** of all certificates in the chain from the first entry to the current certificate:

Definition 2

$$Sign(CERT_i) = h(CERT_i||h(CERT_{i-1}|| \ldots ||h(-\infty))_{SK}$$

where $h()$ is a suitable cryptographic hash function, $||$ represents concatenation and SK is the private (signing) key of the RA.

The resulting CRL is stored at the VA-s as before.

To assert revocation status of all the certificates in the range $[j, k]$, a VA releases the revoked certificates $[CERT_p, \ldots, CERT_q]$ in the actual range $[j, k]$, two sentinel nodes $CERT_{p-1}$ and $CERT_{q+1}$ just outside the range to prove completeness of the reply, running hash for $CERT_{p-2}$, and a **single** signature – $Sign(CERT_{q+1})$. Since all signatures are computed on running hashes, it can be easily seen that $Sign(CERT_{q+1})$ provides a proof of authenticity and completeness for the entire range.

4 Practical Considerations

We now discuss some practical implications of the proposed modifications.

[3] This is equivalent to obtaining an entire CRL.

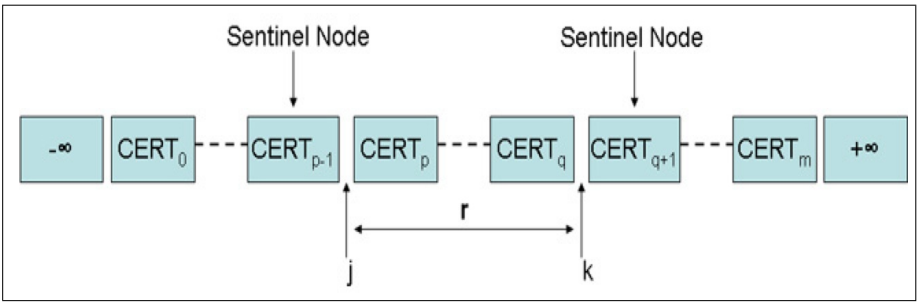


Fig. 2. Revocation Query Reply

4.1 CRL Generation and Size

With the technique described above, the RA signs revoked certificates separately. This increases the overall size of the CRL as well as the computation load on the RA. However, we claim that this overhead is acceptable, for the following reasons:

- Although it might seem that computational overhead of signing each revoked certificate might be significant for the RA, we show that this is not the case in practice. Experiments show that it takes 6.82ms to generate one RSA signature on a "weakling" P3-977MHZ linux PC. Assuming that the update interval for the CRL is one week and the CRL contains 100,000 certificates, this translates to 11.3 minutes of compute time for every CRL update period. Clearly, the same task would take much less (one or two orders of magnitude) time on more powerful platforms readily available today. For example, the Sun Fire T2000 server can compute 12,850 1024-bit RSA signatures and 18,720 1024-bit DSA signatures in one second with 1 UltraSPARC T1 processor and 32 GB memory running Solaris 10[22]. Going back to our example scenario, if the RA were to be deployed on a T2000 server, it could recompute 100,000 signatures in just under 10 seconds.
- The proposed technique requires storing a separate signature per CRL entry. This increases the size of the CRL and, consequently, increases the cost of RA-VA communication. However, it is reasonable to assume that this communication is conducted over fast communication links, in contrast to VA-Client communication which can take place over low-bandwidth channels. Furthermore, RA-VA communication occurs relatively infrequently (once per update period), whereas the frequency of clients-VA communication is significantly higher. We present sample metrics for communication costs in the next section.

4.2 Freshness Considerations

In order to provide freshness and to prevent malicious VA-s from replaying old CRL-s, the RA must re-sign revoked certificates every update period, even if

there are no changes to the CRL. We can reduce the time to re-sign the revoked signatures in the event of no changes to the CRL by using the Condensed RSA signature scheme proposed in [19].

Condensed-RSA Signature Scheme: Given t different messages $\{m_1, ..., m_t\}$ and their corresponding signatures $\{\sigma_1, ..., \sigma_t\}$ generated by the same signer, a Condensed-RSA signature is computed as the product of all t individual signatures:

$$\sigma_{1,t} = \prod_{i=1}^{t} \sigma_i \pmod{n}$$

The resulting aggregated (or condensed) signature $\sigma_{1,t}$ is of the same size as a single standard RSA signature. Verifying an aggregated signature requires the verifier to multiply the hashes of all t messages and checking that:

$$(\sigma_{1,t})^e \equiv \prod_{i=1}^{t} h(m_i) \pmod{n}$$

Condensed-RSA for Re-signing: It is possible to use condensed-RSA for re-signing if the RA signs the revoked certificate data and the time-stamp separately and then aggregates the two by multiplying the resultant signatures as explained above. If there are no changes to the CRL, the RA can re-sign all the revoked certificates by: (1) re-using the signatures for the revoked certificate data; (2) creating a single new signature on the new time-stamp; and (3) aggregating the signature of the new time-stamp with the signatures of all the revoked certificates to re-sign all the revoked certificates.

If we assume, once again, that the CRL contains $100,000$ entries. As mentioned above, it would take 11.3 minutes to re-sign all these certificates with the new timestamp on a P3-977MhZ Linux machine. However, if we use condensed-RSA, re-signing of the same $100,000$ entries with the new time-stamp, using our method can be done in just 4.4 seconds on the same hardware. However, this optimization is less useful if there are changes to the CRL. Because signatures are chained explicitly, any change to the CRL (insertion or deletion of an entry from the chain) causes all the signatures from that point until the end of the chain to be re-computed.

4.3 Forcing Uniform Distribution

In a worst-case scenario, all certificates in the range specified in the query are revoked and corresponding r certificates need to be returned to the client. The simplest solution to this is to force all revoked certificate serial numbers to be uniformly distributed over the entire serial number range. However, this is unrealistic, since, in practice, certificate issuers sometimes assign serial numbers to certificates consecutively over well-defined subranges. Each subrange can be used to indicate a different product or class of products, e.g., Verisign supports the following classes: Standard, Commerce and Premium [5]. Requiring uniform

non-sequential certificate distribution would create a maintenance nightmare for both issuers and certificate-holders. Furthermore, gathering and analysis of statistical data would become problematic.

We use a simple extension to the range query technique that addresses the problem. Instead of sorting according to serial numbers, we sort the revoked certificates along *permuted* serial numbers before creating the signature chain. One obvious choice of suitable permutation that ensures uniformity is a block cipher, e.g., DES, with a known and fixed key. We further observe that a cryptographic hash function is not a good choice for the kind of a PRP we require. Unlike a PRP, a hash function "reduces" its input and collisions are expected, however difficult they might be to compute. Whereas, a PRP resolved with a block cipher such as DES-ECB with a fixed key, guarantees no collisions.

The primary advantage of this extension is that certificate issuers can continue issuing sequentially-numbered certificates over well-defined subranges. As long as an appropriate PRP is used, we can assure uniform distribution of the revoked certificates in the signature chain.

4.4 Query Range Generation

In section 3.2, we outlined a mechanism for positioning the target certificate within the query range by generating a b-bit random number $b = log(r)$ or the bit-length of r, where r is the range of the query. This step is necessary to randomize/vary the placement of the target. We observe that, if a client poses repeated queries against the same target certificate, varying the query range and its boundaries is not advisable, for obvious reasons. In this case, narrowing the overlap of all queries' ranges gradually erodes privacy and might eventually yield a single target certificate. We now discuss one possible solution to eliminate such inference attacks.

Instead of generating a random number (offset) to determine the position of the target certificate within the range, the client generates a b-bit number using a *keyed hash function* h' which takes as input the target certificate serial number i and a secret key sk chosen by the client. $h'(i, sk) = X$ where $|X| = b$. Now, the client sets the range boundaries as described earlier, i.e., $j = i - X$ and $k = j + r - 1$. The client then queries the VA with the range $[j, k]$. The use of the secret key sk to compute range boundaries ensures that the same range is consistently used for repeated queries against a specific certificate,.

5 Analysis

We now briefly analyze overhead factors introduced by our privacy-preserving revocation checking technique.

Client Overhead: With traditional CRLs, clients are required to store entire lists containing all revoked certificates locally. As such lists grow, this can result in significant storage overhead. On the other hand, with our technique, clients

do not incur any storage costs. A client queries the on-line VA for the revocation status of a certificate and does not need to store/cache anything locally.

For traditional CRL, the client is required to periodically contact a CRL distribution point (or directory server) to obtain the most recent CRL. If Δ-CRLs are used, the client needs to periodically download updates in order to keep her copy of the CRL recent and complete. Thus, communication overhead using traditional CRLs and delta-CRLs depends upon the frequency of CRL updates. On the other hand, with our proposed technique, a client incurs one round of communication with the VA for each revocation status check of a certificate.

RA Overhead: In our method, the RA is required to separately sign each entry in the list, whereas, in a traditional CRL, the entire CRL is signed once in its entirety, for each update. This increases both the size of the CRL as well as the computation costs for the RA. However, as discussed in the previous section, we assume that the frequency of querying is much greater than the frequency of CRL updates; thus, we focus our scheme on minimizing costs for the interaction between the VA and its clients.

Communication Costs: We now compare communication costs of the traditional CRL with our modified version. Table 1 summarizes the notation and parameter values assumed for the evaluation. **Update cost** measures RA-to-VA communication and **query cost** measures VA-to-Client communication.

Table 1. Notation

n	# of Certificates $n = 2 * 10^6$
p	Fraction of the certificates revoked $p = 0.1$
t	update period t=weekly
c	# of clients $c = 10^5$
q'	# of queries posed by each client $q' = 100$
q	Total # of queries in an update period $q = c * q' = 10^7$
r	Size of the range query
l_{sig}	Length of a signature $l_{sig} = 1024$ bits
l_{entry}	Length of a CRL entry $l_{entry} = 160$ bits
l_{hash}	Length of a hash digest $l_{hash} = 160$ bits

Traditional CRL: The CRL update cost is $n * p * l_{entry} + l_{sig}$ for each update period since the RA sends the entire CRL to the VAs. We are assuming that there are c clients and each client poses q' certificate validation checks on the cached CRL in a update period. The CRL weekly query cost is $c * (p * n * l_{entry} + l_{sig})$ since for every query the VA sends the whole CRL to the client.

Modified Technique: To update the CRL, the RA sends $n * p * (l_{entry} + l_{sig})$ bits to the VA. To answer a user's query, the VA returns $l_{entry} + l_{sig}$ for a **Revoked** reply and $2l_{entry} + l_{sig}$ for a **Not Revoked** reply if queried by a specific serial number (i.e., for non-privacy preserving querying). Therefore, the VA sends up to $q * (2l_{entry} + l_{sig})$ bits each update period to answer q queries. If we employ range queries to enable privacy-preserving querying, then the communication cost depends on the number of revoked certificates in the range. In general, if we assume uniform distribution of revoked certificates, then the communication cost for user queries for a range of certificates is given by $q*(((r*p)+2)*l_{entry}+l_{sig})$ bits for each update period. Note that $(r * p) + 2$ denotes the total number of revoked certificates in the specified range plus two sentinel nodes.

The following table shows the estimated communication costs (in bits) according to traditional as well as modified CRL schemes. As shown in the table, the modified CRL query costs are orders of magnitude lower than traditional CRL costs. Although the cost of updating a CRL is higher in our scheme, the advantage of our scheme is that it allows the clients to query and obtain portions of the CRL securely thus making our scheme flexible, efficient (for the clients) and privacy preserving.

Table 2. Communication Cost comparison

	Traditional CRL	Modified CRL no privacy	Modified CRL r=100
Update Cost (RA-to-VA)	$3.2 * 10^7$	$2.36 * 10^8$	$2.36 * 10^8$
Query Cost (RA-to-Client)	$3.2 * 10^{12}$	$1.18 * 10^{10}$	$2.94 * 10^{10}$

6 Future Directions and Conclusions

An outstanding issue is assessing loss of privacy in the presence of repeated queries. If we assume that multiple clients, at about the same time, are all interested in a particular target certificate (e.g., because of a breaking news article) and the adversary (third party or VA) is aware of the potential target, correlating multiple range queries does not seem difficult since all the range queries in question would have at least one certificate in common. As part of our future work, we want to study this problem in greater detail to make such inferences more difficult.

In conclusion, we described a very simple (yet novel) approach for privacy-preserving revocation checking using CRLs. We proposed minor modifications to the well-known CRL structure. These modifications enable efficient (for the clients) and privacy-preserving revocation checking. We provided a mechanism for verifiers to query untrusted online entities regarding the revocation status of a certificate without having to retrieve the entire CRL. Each client, depending on the desired level of privacy, can determine a revocation query range that best suits its needs. This results in a trade-off between privacy and bandwidth overhead. In the worst case, the overhead can be significant if the desired privacy level is high as is the number of revoked certificates. However, if only a small fraction of all certificates are revoked, our approach is reasonably efficient.

Acknowledgments

First off, we thank EUROPKI anonymous reviewers for their thorough and helpful comments. We are also grateful to John Solis for his comments on the drafts of this paper.

References

1. Aiello, W., Lodha, S., Ostrovsky, R.: Fast digital identity revocation. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, IACR, vol. 1462, Springer, Heidelberg (1998)
2. The OpenPGP Alliance. Openpgp: Open pretty good privacy `http://www.openpgp.org/`
3. Baric, N., Pfitzmann, B.: Collision-free accumulators and fail-stop signature schemes without trees. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 480–494. Springer, Heidelberg (1997)
4. Cachin, C., Micali, S., Stadler, M.: Computationally private information retrieval with polylog communication. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, IACR, vol. 1592, Springer, Heidelberg (1999)
5. Verisign Corporation. Compare all ssl certificates from verisign, inc. `http://www.verisign.com/products-services/security-services/ssl/buy-ssl-certificates/compare/index.html`
6. Dierks, T., Rescorla, E.: The transport layer security (tls) protocol, version 1.1. Internet Request for Comments: RFC 4346, April 2006, Network Working Group (2006)
7. Inc. Free Software Foundation. Gnu privacy guard `http://www.gnupg.org/`
8. Goodrich, M., Tamassia, R., Schwerin, A.: Implementation of an authenticated dictionary with skip lists and commutative hashing. In: Proceedings of DARPA DISCEX II (2001)
9. OpenSSL User Group. The openssl project web page `http://www.openssl.org`
10. Kent, S., Seo, K.: Security architecture for the internet protocol. Internet Request for Comments: RFC 4301, December 2005, Network Working Group (2005)
11. Kikuchi, H.: Privacy-preserving revocation check in pki. In: 2nd US-Japan Workshop on Critical Information Infrastructure Protection, July 2005, pp. 480–494 (2005)

12. Kocher, P.: On certificate revocation and validation. In: Proceedings of Financial Cryptography 1998, pp. 172–177 (1998)
13. Kushilevitz, E., Ostrovsky, R.: Computationally private information retrieval with polylog communication. In: Proceedings of IEEE Symposium on Foundation of Computer Science, pp. 364–373. IEEE Computer Society Press, Los Alamitos (1997)
14. RSA Laboratories.: Crypto faq: Chapter 4.1.3.16. what are certificate revocation lists (crls)? `http://www.rsa.com/rsalabs/node.asp?id=2283`
15. Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: Handbook of applied cryptography. CRC Press, Boca Raton (1997)
16. Merkle, R.: Secrecy, Authentication, and Public-Key Systems. PhD thesis, Stanford University, PH.D Dissertation, Department of Electrical Engineering (1979)
17. Micali, S.: Certificate revocation system. United States Patent 5666416 (September 1997)
18. Myers, M., Ankney, R., Malpani, A., Galperin, S., Adams, C.: Internet public key infrastructure online certificate status protocol - OCSP. Internet Request for Comments: RFC 2560, 1999. Network Working Group (1999)
19. Mykletun, E., Narasimha, M., Tsudik, G.: Authentication and integrity in outsourced databases. In: Symposium on Network and Distributed Systems Security (NDSS'04) (February 2004)
20. Naor, M., Nissim, K.: Certificate revocation and certificate update. IEEE Journal on Selected Areas in Communications (JSAC) 18(4), 561–570 (2000)
21. Solis, J., Tsudik, G.: Simple and flexible revocation checking with privacy. In: Danezis, G., Golle, P. (eds.) PET 2006. LNCS, vol. 4258, Springer, Heidelberg (2006)
22. Sun Microsystems: Sun Fire T1000, and T2000 Servers Benchmarks `http://www.sun.com/servers/coolthreads/t1000/benchmarks.jsp`
23. International Telecommunication Union. Recommendation x.509: Information technology open systems interconnection - the directory: Authentication framework, 6-1997, 1997. Also published as ISO/IEC International Standard 9594-8 (1997e)

7 Certificate Revocation Techniques

We now briefly overview some popular certificate revocation techniques and associated data structures. In the following, we refer to the entity validating certificates (answering certificate status queries) as a Validation Authority (VA). A distinct entity – Revocation Authority (RA) – is assumed responsible for actually revoking certificates, i.e., generating signed data structures, such as CRLs.

Strictly speaking, a certificate or a public key is never actually revoked. What is revoked is the binding between an identity string and a certificate serial number (which may contain a public key but does not have to, e.g, in case of attribute certificates).

Certificate Revocation Lists (CRLs): CRLs are a common means of checking the revocation status of public key certificates. A CRL is a signed list of certificates that have been revoked before their scheduled expiration date.

Each entry in the CRL indicates the serial number of the revoked certificate. The CRL entry may also contain other relevant information, such as the time, and reason for the revocation. CRLs are usually distributed in one of two ways: In the "pull" model, RA distributes the CRLs via a public directory. The clients/queriers download the CRL from these public databases as needed. In the "push" model, the RA directly sends the CRL to the clients, either at regular intervals or at times indicated in prior CRLs [14]. If CRLs are distributed using a pull model, they should be issued at regular intervals even if there are no changes, to prevent new CRLs being maliciously replaced by old CRLs. Alternatively, if the inter-CRL interval is not fixed; each CRL needs to include the specific time for the issuance of the next CRL.) Since a CRL can get quite long, a RA may instead post a signed Δ-CRL which contains only the new entries consisting of the list of certificates revoked since the last CRL was issued. This requires end-users maintain (and update) secure local images of the CRL. [15] lists some of the other proposed ways to improve the operational efficiency of the CRLs such as Segmented CRLs and CRL distribution points.

Online Certificate Status Protocol (OCSP): this protocol [18] avoids the generation and distribution of long CRLs and can provide more timely revocation information. To validate a certificate in OCSP, a client sends a certificate status request to a VA. The latter sends back a signed response indicating the status (revoked, valid or unknown) of the specified certificate. Note that, in this case, the VA is

- the CA who issued the certificate in question, or
- a Trusted Responder whose public key is trusted by the client, or
- a CA Designated Responder (Authorized Responder) who is authorized to issue OCSP responses for that CA.

In other words, the VA is an on-line authority trusted by both the client and the CA. A VA can also serve multiple CAs. In practice, in order to reduce VA load, pre-signed responses are often used: a VA signs (once) an oft-requested status a given certificate and uses it in to reply to many requests. This helps performance but sacrifices the timeliness of VA's replies.

Certificate Revocation Trees (CRTs): this technique was proposed by Kocher [12] as an improvement for OCSP [12]. Since the VA is a global service, it must be sufficiently replicated in order to handle the load of all validation queries. This means the VA's signature key must be replicated across many servers which is either insecure or expensive. (VA servers typically use tamper-resistance to protect their signing keys). Kocher's idea is a single highly secure RA which periodically posts a signed CRL-like data structure to many insecure VA servers. Users then query these insecure VA servers. The data structure proposed by Kocher is basically a Merkle Hash Tree (MHT) [16] where the leaves represent currently revoked certificate ranges sorted by serial number (lowest serial number is the left-most leaf

and the highest serial number is the right-most leaf). The root of the hash tree is signed by the RA. A client queries to the nearest VA server which produces a short proof that the target certificate is (or is not) on the CRT. If n certificates are revoked, the length of the proof is $O(\log n)$ (In contrast, the proof size in plain OCSP is $O(1)$).

Skip-lists and 2-3 trees: One problem with CRTs is that, each time a certificate is revoked, the whole tree must be recomputed and distributed in its entirety to all VA servers. A data structure allowing for dynamic updates would solve the problem since a secure RA would only need to send small updates to the data structure along with a signature on the new root of the structure. Both 2-3 trees proposed by Naor and Nissim [20] and skip-lists proposed by Goodrich, et al. [8] are natural and efficient for this purpose. Additional data structures were proposed in [1]. When a total of n certificates are already revoked and k new certificates must be revoked during the current time period, the size of the update message to the VA servers is $O(k \log n)$ (as opposed to $O(n)$ with CRT's). The proof of certificate's validity is of size $O(\log n)$, same as with a CRT.

E-Passports as a Means Towards the First World-Wide Public Key Infrastructure

Dimitrios Lekkas[1] and Dimitris Gritzalis[2]

[1] Dept. of Product and Systems Design Engineering, University of the Aegean
Syros GR-84100
dlek@aegean.gr
[2] Information Security and Critical Infrastructure Protection Research Group,
Dept. of Informatics, Athens University of Economics and Business (AUEB)
76 Patission Ave., Athens GR-10434
dgrit@aueb.gr

Abstract. Millions of citizens around the world have already acquired their new electronic passport. The e-passport is equipped with contactless communication capability, as well as with a smart card processor enabling cryptographic functionality. Countries are required to build a Public Key Infrastructure to support digital signatures, as this is considered the basic tool to prove the authenticity and integrity of the Machine Readable Travel Documents. The first large-scale worldwide PKI is currently under construction, by means of bilateral trust relationships between Countries. In this paper, we investigate the good practices, which are essential for the establishment of a global identification scheme based on e-passports, together with the security and privacy issues that may arise. We argue that an e-passport may also be exploited in other applications as a globally interoperable PKI-enabled tamperproof device. The preconditions, the benefits, and the drawbacks of using e-passports in everyday electronic activities are further analyzed and assessed.

Keywords: Security, Trust, Digital Signatures, Machine Readable Travel Documents, PKI, RFID, Smart card, Passport.

1 Introduction

The citizens of many countries around the world obtained their new electronic passport (e-passport), within the last year or so. Most European countries have already implemented the infrastructure for the issuance of the new passports. The requirements for a new type of passport are imposed by the United States and the International Civil Aviation Organization (ICAO), demanding a higher level of security at the inspection points of the countries borders. The e-passport incorporates three state-of-the-art technologies: Radio Frequency Identification (RFID), Biometrics and Public Key Infrastructure (PKI). While RFID is used for practical reasons in the communication with the inspection systems, Biometrics and PKI are considered capable of reducing fraud and enhancing security in worldwide digital identification.

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 34–48, 2007.

From its side, ICAO published a series of technical reports, describing the technical and procedural details on how a Machine Readable Travel Document (MRTD) must be implemented [1]. Face recognition is specified as the only mandatory globally interoperable biometric for identity verification of travelers. MRTDs including e-passports, are equipped with an Integrated Circuit Chip (ICC), where all digital data, including biometric information are stored. Among several other issues, ICAO technical reports describe the details of the communication between the e-passport and the local inspection points, the specifications for biometric data, the structure of the data stored (called the Logical Data Structure – LDS [2]), and the PKI support.

The ICAO PKI Technical Report [3] is intended to provide standards for a simple worldwide Public Key Infrastructure, which should support digital signatures applied to Machine Readable Travel Documents. These digital signatures are intended to permit authentication of basic data produced by the issuing Country and stored in the chip embedded into the e-passport. The stored signed data include the Machine Readable Zone (MRZ) of the passport plus digitized biometric measurements, as well as other personal data of the passport bearer.

Using the digital signature, the receiving Countries can verify that the stored data is authentic (i.e. generated by the issuing Country, and not been altered), just as the physically readable passport booklet is secured from unauthorized alteration or substitution by strong physical security measures. ICAO has recognized that one of the most effective ways of doing this is using Public Key Cryptography to digitally sign the data stored on the chip. Issuing Countries are requested to implement a PKI, following specific interoperable standards, and to properly manage their own keys, which are used to digitally sign the data stored in the e-passports.

Given that the US and the ICAO initially demanded from the Countries to implement the PKI within a very short period (just a few months), the ICAO Technical Report states that it does not aim at describing a full implementation of a PKI within each Country. ICAO states that PKI does not provide the sole measure for determining the authenticity of the passport and, thus, it should not be relied upon as a single determining factor. The passport still maintains its physical security characteristics, and it should be verified by check-points using conventional manual mechanisms, along with the automated check of its electronic contents. Due to this restrained approach, the ICAO report seems that it sacrifices several security characteristics of a strong PKI implementation, such as the existence of client X.509 certificates, as well as the existence of passport revocation mechanism. Moreover, perhaps due to the increased cost of passports with crypto-processor chip, the active security mechanisms, which could protect the e-passport's data against eavesdropping and cloning, are not mandatory, allowing a weak e-passport implementation.

In this paper we focus on the PKI-related issues of e-passports, proposing a series of good practices in order to implement a Country PKI, conforming to ICAO rules. We examine how the required global interoperability can be achieved by building an appropriate worldwide Trust architecture. We then specify some important security and privacy issues, which are emerged by the use of digitized personal data. As the e-passports infrastructure seems to implement the sole globally interoperable PKI of today, we investigate how we can exploit this infrastructure in different areas and applications, by using the e-passport not only as a digital identity, but even as a

signature creation device, or as an Internet authentication certificate, although it was not initially designed for such purposes.

2 Building a Country Public Key Infrastructure

The Public Key Infrastructure for the issuance of e-passports does not issue conventional digital certificates for citizens. However, it has all the characteristics of a full-scale PKI, with only one part missing from this implementation, i.e. the management of end-entity certificates. In other words, it does not maintain a public key directory, and it does not provide a passport revocation mechanism.

As a large-scale PKI, it is necessary to examine whether the ICAO technical report covers the baseline for the implementation, as well as to provide a brief additional set of 'Good Practices' (as described in [4]). The areas where a PKI must focus is (a) the adoption of the proper trust architecture, (b) the legal status of the certification provider, (c) key and certificate management, (d) interoperability, and the technology used.

(a). In respect to the *Trust Architecture*, ICAO proposes a single autonomous hierarchy for each country. The independency of countries in citizen identification is crucial and it is, thus, respected. This hierarchy consists of two levels: The root CA, called Country Signing Certification Authority – CSCA, and one level of one or more subordinate CA, called Document Signing Certification Authorities – DSCA. The Document Signing CA signs the passport's data, including a public key (Active Authentication key) stored in each passport, thus providing a kind of 'identity certificate' to the citizen. ICAO avoids providing any kind of trusted information to Countries (e.g. a directory of Root Country certificates), as it was not desired to establish a worldwide Single-Point-Of-Trust (SPOT) and not even a European Union CA. It is true that the risk taken by a unique organization to serve as a global Trust Anchor is very high and, additionally. it may not be globally acceptable for political reasons. Our view is that this approach is quite reasonable, in terms of flexibility, security, and nations' independency.

(b). As of the *legal status* of the passport issuance service, the organization hosting this activity is always a governmental authority. The authority acts as a Certification Services Provider and it must conform to the legal framework for the provision of certification services. ICAO does not refer to any legal requirements of the issuing authority. However, it briefly describes some security requirements (e.g. the use of secure-signature-creation-devices), that partly conform to the European law for the provision of 'qualified digital signatures'. We argue that if a recognized accreditation scheme exists at the issuing country, then the passport authority must follow the required inspection and accreditation procedure of this country.

(c). The *key and certificate management* obligations of the issuing authority are restricted only to the management of the two levels of CA. The requirements for key protection and renewal for the secure out-of-band distribution of Root certificates and for the issuance of CRL (by the root level only) are well known in PKI [6]. ICAO will provide a Public Key Directory (PKD) for publishing the second level certificates (the DSCA certificates) and the relevant CRL, in order to facilitate the verification

procedure at the local inspection points. Today, the ICAO PKD is not yet operational. As additional good practices in key and certificate management, we may propose:

- The existence of a secondary set of a Root private key and the respective self-signed certificate, which is counter-signed by the primary CSCA (by means of a cross-certificate). This secondary set would enable a quick recovery from an eventual loss of the primary keys. A trust path to the secondary keys is already established and, therefore, there is no need to redistribute the Root certificate by offline cumbersome means.
- As the volume of the signed passport data is expected to be very high, the DCSA keys are heavily exposed to 'known cipher-text' cryptanalysis. The frequent renewal of DSCA keys and certificates (e.g. a monthly basis would be a feasible approach) is a requirement. On the other hand, the CSCA keys must be long lasting (order of decades), as long as we have no dramatic changes in IT.
- Due to the frequent renewal of DSCA certificates, the inspection points have to be aware of a large number of Document Signer certificates (e.g. 5*12=60 different valid DSCA certificates issued within five years) to efficiently perform the verification of the signed passport data. To facilitate the verification procedure (although it is optional for ICAO), the DSCA certificates must be included within the data structure of the passport signature, according to the proposed Cryptographic Message Syntax standard [5].

(d). *Interoperability* is crucial for Machine Readable Travel Documents, since the initial purpose is their use at inspection points outside the home country. ICAO standardizes the technology used, and it enforces a common approach in the algorithms and the data structures [7] used for the security functions of the e-passports. However, ICAO still leaves many optional features, in favor of countries wishing to implement a simpler infrastructure and use a cheaper chip in e-passports (e.g. without processing capabilities). For example, it is possible to avoid the implementation of Basic Access Control and Active Authentication and use a simple memory chip. Additionally, some fields in the data structures of signatures, the LDS, and the certificates are optional, in favor of chips with restricted memory capacity. These options multiply the complexity of an interoperable software package that must successfully read and validate the passports of the countries of the world, but its implementation remains feasible. In a future version of the report, the access control and the active authentication (based on asymmetric key pairs) mechanisms should become mandatory, thus increasing both security and interoperability. The second important factor of the interoperability is the establishment of trust, which is accomplished by means of bilateral cross-certifications (as explained in the next sections).

3 Digital Signatures as the Basic Tool for Passport Authenticity and Integrity

The verification of the authenticity of the data stored in the e-passport is based on digital signatures. The mechanism for signing and validating data is called 'Passive Authentication'. Permitted signature algorithms are RSA, DSA, and Elliptic Curves DSA (ECDSA).

The structure of the data stored in the passport's chip (the LDS – Logical Data Structure), including the digital signature, is briefly illustrated in Figure 1. Data is separated in two parts (Dedicated Files – DF): (a). the user files, which is a writable area, and (b). the LDS, providing read-only access. The LDS contains the cryptographic keys, supporting the Basic Access Control and Active Authentication mechanisms, as well as some general information about itself (EF-COM). Sixteen Data Groups, containing the holder's identification and biometric data, follow. The MRZ (including document number, name of bearer, nationality, date of birth, etc.) is stored in the 1st Data Group, the biometric facial image is stored in the 2nd Data Group, and the Active Authentication public key is stored in the 15th Data Group. The hash values of all the present Data Groups form a new structure, called LDS Security Object. This is, then, signed, according to the 'Cryptographic Message Syntax', thus producing the Document Security Object – SO_D, which is stored in the chip.

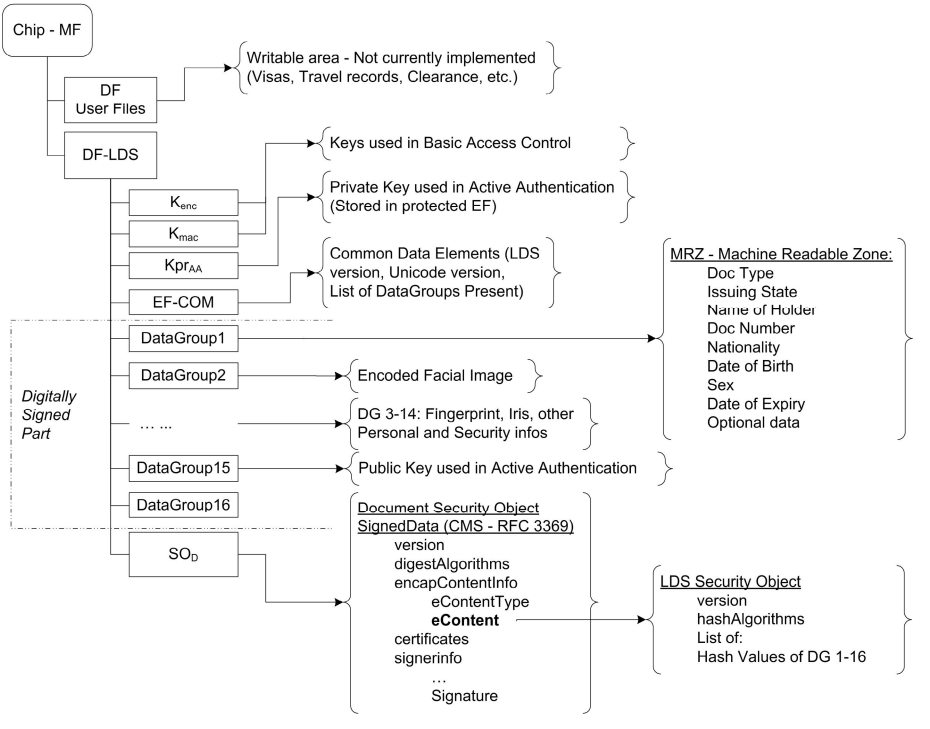


Fig. 1. Overview of data signed and stored in e-passport's chip

A valid digital signature proves that the stored data are produced by the authorized issuer and they are not altered. However, it does not prove that the container of the data (i.e. the e-passport booklet) is authentic. Therefore, the signature alone does not prevent chip cloning or substitution, unless there is a strong binding between signed data, the chip, and the booklet. The signed data must contain information that is also physically and securely printed on the booklet (binding to printed data to avoid substitution), as well as information that is uniquely bound to the chip (to avoid data cloning).

Binding to printed data is mandatory, as the MRZ of the passport is both, printed on the booklet, and included in the signed data. Binding to the chip is based on the existence of 'Active Authentication' keys, which is only optional. The Active Authentication private key is securely created and stored in the chip and it remains secret throughout the lifetime of the e-passport. The respective public key is included in signed data (LDS Data Group 15), thus providing unambiguous connection between the whole signed data set and the chip. Similarly, the serial number of the chip may be stored in the LDS Data Group 13, thus providing secure logical and physical binding (but this also optional).

4 Establishing Global Trust

It is true that ICAO, or any other international organization, cannot and will not play the role of a Single-Point-Of-Trust (SPOT) for the PKI of the whole world, but it will only serve as a regulatory authority. In other words, ICAO will not build a worldwide Root CA, will not server as a Bridge CA, and will not even maintain a Certificate Trust List of countries' root CAs. Each country may build an autonomous Public Key Infrastructure, starting from a top-level Certification Authority (the CSCA). Each country has the possibility to decide about the design parameters of its infrastructure, the implementation of its security policy, and the technology used, conforming to ICAO PKI report and supporting global interoperability.

On the other hand, a global trust infrastructure [8] seems necessary, in order to facilitate the validation of the digitally signed passport of any Country, at the borders of any other Country. Since a global consensus towards an organization which indicates 'who do we trust' may not be feasible in international relationships (even the United Nations could not probably gain that consensus), the most appropriate solution seems to be the establishment of bilateral trust relationships, toward a 'web-of-trust' model. A web-of-trust is built by establishing a subset of NxN trust relationships between Countries.

Technically, a trust relationship is established when a Country decides to trust the root certificate (the certificate of the CSCA) of another Country. First of all, a secure offline channel for the distribution of one country's root CA to another country must be established. This is achieved through out-of-band secure diplomatic means. Given that the whole infrastructure of a country trusts its own root certificate (i.e. the CSCA plays the role of a SPOT for this country), there are two alternative mechanisms to implement the web-of-trust, as shown in Figure 2:

- *Cross-certification:* The Country issues a cross-certificate (signed by the home CSCA) for each root CA of all the countries it trusts. The cross-certificates are then distributed to the inspection systems of the Country, where the cross-certified countries will be trusted. This kind of trust link can be reciprocal or one-way.
- *Certificate Trust List - CTL:* The Country maintains a secure structure (signed by the home CSCA) containing unique referrals to all root CA of the Countries it trusts. The CTL is then distributed to the inspection systems of the Country, where the countries contained in the CTL will be trusted.

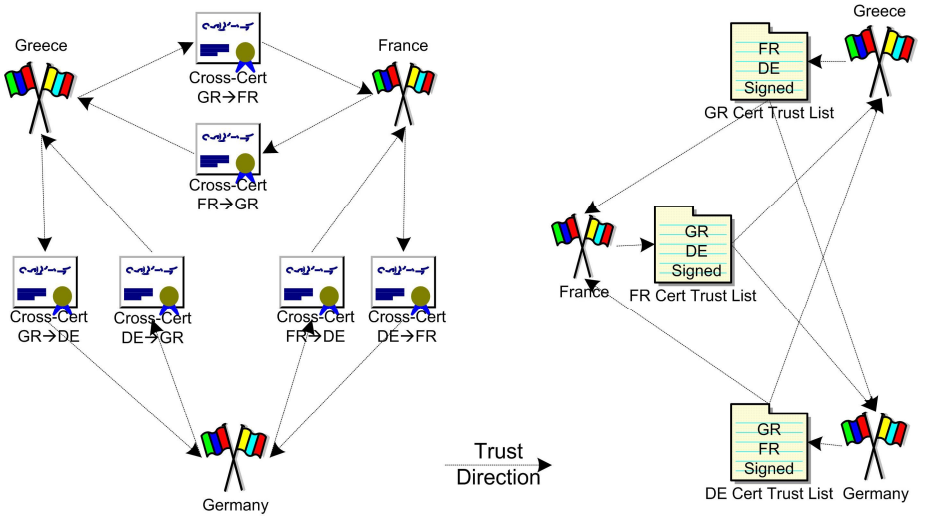


Fig. 2. Alternatives for establishing global Trust:Cross-certification and Certificate Trust Lists

Although the CTL seems to be an easy and neat method to maintain the trust relationships, we consider that the cross-certification is more efficient and countries should prefer it. Three reasons support this argument:

- Only one CTL should be considered as valid at any time. However, since CTL are distributed to inspection points, there is no common mechanism to revoke a previously issued CTL and to impose the use of the newest one. This problem is intensified in case a revocation of a trust relationship occurs. The maintenance of the issued CTLs adds complexity, and it must be based on proprietary protocols. [9]
- CTL is not standardized in a widely accepted format, while cross-certificates are based on the X.509.v3 standard. Cross-certificates are issued and revoked by the existing infrastructure in the same way as issuing and revoking Document Signing certificates.
- Cross-certificates can be autonomously issued, communicated, and revoked for each trust relationship without affecting the other trust relationships. Revocation of cross-certificates can be communicated to the inspection points through the distribution of CSCA CRLs. These CRLs are already distributed to inspection points, informing the status of the Document Signing CA.

5 Security and Privacy Issues

Researchers have already exposed a number of security and privacy issues regarding the possession and the use of e-passports [10, 11, 12]. As expected, there are several potential e-passport threats, due to two factors: (a). the *proximity (RFID)*

communication of the passport with other systems, and (b). the existence of *sensitive biometric data* within its chip.

A basic security concern is the unauthorized skimming or eavesdropping of the information stored in the passport, resulting in an identity theft. This concern is further intensified due the contactless nature of the passport's chip, giving the possibility of skimming its contents without the awareness of its holder. There are reports [13, 14] exhibiting that a passive eavesdropping (while a passport is communicating with a legitimate reader), or an active eavesdropping (the initiator of the communication is the eavesdropper), is possible from a distance of tens of meters.

RFID technology may also provide the means to track the movements of an individual, since the RFID chip transmits a unique anti-collision code during its initial handshaking (clandestine tracking). On the same line, the leakage of biometric information not only consist a violation of privacy, but it may enable forgery and movement tracking as well.

We also focus our attention on three weaknesses related to the cryptographic functionality of the e-passports: (a). the lack of management for 'Active Authentication' keys, (b). the access control on sensitive biometric data, and (c). the low entropy of 'Basic Access Control' keys.

(a). *Active Authentication key management:* Active Authentication renders the e-passport as a strong authentication token. The e-passport securely creates and hosts the private key, while the public key is a part of the signed data and, therefore, it is bound to the identity details. The authentication procedure is based on a challenge-response mechanism, where the passport proves the possession of the private key. Although the active authentication keys upgrade the e-passport to a smart authentication token, the mechanism is not fully implemented, therefore it is weak. Specifically, there is no means to revoke an AA key, in case of passport loss or compromise, although the key pair cannot change during the lifespan of the passport. Secondly, there is no publication mechanism (i.e. public key directory) for AA public keys. The latter may be not affecting the passport functionality, but it constitutes an important drawback for using the e-passport in other applications, such as e-commerce or citizen digital signatures.

(b). *Separation of access control for biometric data:* Access to the data stored in e-passports is allowed at many stages of its use and by many different systems. For example, airport staff and hotel clerks are allowed to read the biometric data stored in the e-passport, since they have physical access to the booklet. In case the e-passport is used in additional applications, such as driving license or national identification, the points able to read sensitive data are multiplied. Extended access control [15] is an additional optional mechanism proposed to address this problem by restricting access to biometrics only to the bearers of a specific cryptographic key. However, the implementation of extended access control requires significant effort and introduces additional key management. We argue that the existence of biometrics (except the facial image) should be avoided, whenever possible, since: (a). it considerably increases the threats against the e-passports; (b). it reduces the value of the e-passport as a public identification token and (c). it restricts its use in other applications.

(c). *Entropy of Basic Access Control keys:* ICAO provides the specifications for implementing an optional mechanism to protect unauthorized reading (and possibly

duplicate) of the contents of an e-passport. It is called 'Basic Access Control'. According to this mechanism, a secure communication channel between the reader and the passport must be established, before reading the identity contents. The secure channel is based on two 3DES keys, which are stored (and protected) in the passport. The same keys are computed at the inspection point. The software at the inspection point computes the symmetric keys, based on information contained in the MRZ of the passport, which must be optically (by an OCR) or manually read from a physically opened passport. In other words, security is based on the fact that nobody can derive the cryptographic keys and read the passport's contents, unless she has physical access to it.

However, it seems that the entropy of the basic access control keys is low [10]. Since the keys are derived from readable, and sometimes known- information stored in the MRZ, it is possible for someone who knows something about the holder and about the policy of the passport issuing authority, to decrease the entropy of the keys at unacceptable levels. Specifically, the derivation of keys is just based on three pieces of information: the passport's serial number, the holder's date of birth, and the expiration date, taken from the MRZ as the one below:

```
P<UTOERIKSSON<<ANNA<MARIA<<<<<<<<<<<<<<<<<<<L898902C<3U
TO6908061F9406236ZE184226B<<<<<14
```

As a real scenario, the above MRZ is typed on a passport of a country called Utopia, which issues passports for its whole population (5 million citizens), the lifetime of the passport is 5 years, and the serial numbers of passports are assigned sequentially. Keys are derived solely from the document number (9 digits - L898902C<), date of birth (6 digits – 690806), and expiry date (6 digits – 940623). In this case, an attacker may estimate the age of the passport's holder after a visual contact, with a variation of ±4 years. Therefore, the entropy of the date of birth is 8 years = 2920 different dates. The entropy of expiration date is by default 5 years = 1825 different dates. Supposing that there are 5 million valid passports (issued within the 5 last years), and their issuance is evenly distributed in time, we have 2740 passports per day. Since the document numbers are issued sequentially, there is a direct connection of the doc number with the date of issuance and, thus, with the expiration date. Having a few information on serial numbers issued on specific days, one can safely estimate the range of serial numbers connected to a given expiration date at 2740 specific serial numbers, with a variation of 800 numbers (30%) – i.e. 3540 in total. Finally, the total entropy is $2920 * 1825 * 3540 = 1.8 * 10^{10} \approx 2^{34}$ different combinations = 34 bits. It is obvious that a 34-bit key does not provide real security and can be broken in the range of minutes by a conventional system.

The problem is further growing, as the access control keys remain the same for the lifetime of the passport. The keys are not only predictable, but they may be also recorded by the inspection systems, as well as by anyone who has a temporary physical access to the inner pages of the passport (such as a hotel receptionist or a car rental clerk).

6 E-Passport as a Client Digital Certificate

The X.509 digital certificates, which are issued for the ICAO PKI implementation, are restricted only to the authorities issuing the passports (i.e. the hierarchy of Country Signing CA and the subordinate Document Signing CA). A strategic decision for the implementation of e-passports is the lack of citizen certificates, in order to facilitate a fast-track implementation and to avoid the complexity of managing client certificates and keys.

ICAO considers the need for CRL for end-entities as a complicating factor, and restricts the usage of CRL only to indicate a CA compromise. Furthermore, even in the unlikely event of a CA compromise, the e-passports remain valid and only a caution mechanism warns the authorities to view these documents more closely.

We consider that the timidity of ICAO to implement client certificates and to use CRLs is not justified. First of all, the e-passport itself is in fact a client digital certificate, in case it contains an Active Authentication key pair. Although the e-passport does not contain an X.509.v3 certificate and it is not designed for everyday Internet transactions, it exhibits all-but-one of the characteristics of a typical PKI-enabled smart card containing a private key and the relevant digital certificate, i.e.:

- The LDS of the e-passport binds a public key (Active Authentication key stored in DG15) to the identity details of a physical person.
- The data structure binding the keys and the identity is digitally signed by a globally trusted authority (the e-passport issuer CA)
- The private key is securely produced and stored within the RFID smart card data file system, as required by European and International laws for digital signature creation devices.
- The e-passport is personalized within a highly secure environment and the identity details are validated according to a strict procedure, conforming to the requirement for secure client registration before obtaining a digital certificate.

The only missing characteristic of digital certificates is a proper publication and revocation mechanism. We argue that the implementation of a baseline certificate management, including the issuance of CRL for e-passports, can be implemented at a considerably low effort, proportionally to the existing infrastructure. At the same time, the existence of a mechanism confirming the good status of an e-passport adds considerable value to the security and the usability of the whole infrastructure. There are several reasons supporting our argument:

- There is already an established semi-manual mechanism in several of the European Union Member States for reporting invalidated or lost passports, based on the "Schengen Convention" [16].
- Issuance of CRL by the e-passport authority is a trivial task, since the existing infrastructure for Document Signing can be used for periodically issuing and signing CRL without additional effort. CRL may include revoked e-passports by referring their serial number, similarly to the X.509 CRL.
- Distribution of CRL can be easily done through the already established Public Key Directory supporting the worldwide distribution of Document Signer certificates.

Inspection points may easily periodically download CRL, which may be subsequently used for off-line passport validation. The size of CRL can be a potential problem, but it can be solved by adopting deltaCRLs (differential CRL, containing only the differences from the previous CRL).

- Existing Public Key Directories may be used for publishing client certificates at a reasonable additional effort.

7 Exploiting E-Passports in Other Applications

While the research community is discussing for years how we can implement a globally acceptable and trusted Public Key Infrastructure, it seems that the e-passports infrastructure - currently under construction in several countries - provides a potentially friendly environment for building the necessary global trust. This 'de-facto' implementation by Countries gives us the opportunity to investigate whether we can exploit the passport's PKI capabilities in other applications, provided that some preconditions are met.

The global e-passports implementation seems to be an attractive PKI establishment, since:

- The passport itself plays the role of a tamperproof device for the storage of private keys and certificates, as already described in the previous section.
- The passport as a digital identity is issued by governmental authorities, under very strict and reliable identification and issuance procedures for the citizens.
- The technology used throughout the world is compatible and, thus, interoperable.
- A worldwide Web-of-Trust is established through a reliable and secure exchange of countries self-signed certificates.
- It provides simultaneously digital identity capabilities and physical identification means (i.e. the printed booklet itself and the facial image)

Of course the e-passport was not initially designed for a wide use in Internet applications or in public points of interest. Some problems which restrict the public or personal use of e-passport include the cost of equipment, the impossible revocation of a lost passport, and the fact that access key to the passport's data cannot change throughout its lifespan.

In order to investigate the possibility to exploit the PKI characteristics of the e-passport in applications other than the Machine Readable Travel Documents, we distinguish three major categories:

- Applications in point-of-sales or other public points of interest (e.g. strong identification, credit card e-payments).
- Applications requiring reading/writing additional data in passport's chip (e.g. traveler's Visas and e-purses).
- Personal use in Internet transactions (web authentication, digital signatures, and encryption)

In the sequel, we describe the preconditions, under which the exploitation is possible for six prominent applications, including the travel documents. We also note

the pro's and con's when using the e-passport as a digital certificate in each of these six applications.

As shown in Table 1, we examine the preconditions for using the e-passport in public points of interest, other than border control, such as the holder identification (e.g. in hotels, banks and other locations), and the e-purse in Points-of-Sales. In both cases, due to the public nature of the system hosting the applications, the access to stored biometric data must be denied. For the same reason, the system must be widely trusted to ensure that the information included in the MRZ will be not misused, as explained earlier. On the other side, when the passport is used in personal systems, the access control is not an issue, but the existence of the (non trivial for an individual) passport reading equipment is necessary.

The most important advantage of all applications is the established worldwide trust, which is based on a web-of-trust consisting of bilateral relationships between countries. This de-facto trust infrastructure is adding high value at the relevant applications and it overcomes the basic drawback of the most commercial or closed-groups PKIs. Thanks to the PKI-enabled smart-card technology used in e-passports, we have a ready-to-use device for e-purse applications, and most importantly a secure-signature-creation-device conforming to security and legal requirements for digital signatures. Furthermore, a digital signature created by the e-passport can be 'qualified' since the certification provider (passport issuing authority) applies strict citizen identification and infrastructure security procedures, conforming to the legal requirements for qualified certification services provision. Another advantage is the high mobility offered for ubiquitous authentication, since it is based on a portable device and on widely acceptable standards.

The basic disadvantage for using the e-passport in a personal computer is the considerable cost of the equipment for reading (RFID) and scanning (OCR) the e-passport. Of course, the cost of this equipment may significantly decrease in case proximity smart cards become a common need. Since the e-passport does not provide a X509 digital certificate, but a proprietary kind of certificate, its use in today's browsers is not possible, unless special add-ons are installed. The lack of a revocation mechanism (and certificate management in general) by the issuing authorities is a problem for automated Internet authentication, as well as for digitally signing, where no visual inspection of the digital ID is possible and where a fraudulent impersonation cannot be identified. The lack of directory adds some complexity to the verification of a digital signature and to the data encryption for a remote recipient, however it does not prohibit both usages.

We are now able to summarize the preconditions in order to exploit the e-passport PKI capabilities in applications other than its initial purpose:

- *Security features:* The chip has processing capabilities and it supports the Basic Access Control and the Active Authentication as described by ICAO.
- *Equipment:* Reading, validating, and using the e-passport's chip data requires a compatible RFID reader, ideally including a scanner and Optical Character Recognition capabilities for digitizing the Machine Readable Zone printed on the passport. Alternatively, the necessary information of the MRZ (specifically, the passport's serial number, the holder's date of birth, and the expiration date) can be entered manually by a common keyboard.

Table 1. Using e-passports in applications other than MRTD

Application	Preconditions	Pros	Cons & Threats
Use in public points of interest			
Identification in public points, other than border control	Separate access control for biometric data Widely trusted hosting systems	Worldwide Trust and Standardization	Wide access to MRZ makes passport vulnerable to skimming and eavesdropping
e-purse for usage at point-of-sales	Separate access control for biometric data Widely trusted hosting systems Additional storage capacity in e-passport's chip	Worldwide Trust and Standardization Ready infrastructure for most PKI-based smart card e-purses	Writing capabilities and special access conditions add complexity
Personal use			
Authentication in Internet applications	Supportive Equipment on personal computer	Worldwide Trust Strong authentication High Mobility	No standard X.509 certificates No support from browsers No revocation – cannot prevent identity theft Cost of equipment
Digital Signature	Supportive Equipment and software on personal computer	Covers most legal requirements for 'qualified signatures' Worldwide Trust Can be based on well established standards and algorithms	No revocation possible – used until expiration No directory of public keys Cost of equipment
Data encryption	Supportive Equipment and software on personal computer	Worldwide Trust Can be based on well established standards and algorithms	No directory of public keys No key escrow possible, possible data loss Cost of equipment

- *Trusted Systems:* The MRZ information must be protected or at least be not publicly available, as it is the only information that prevents the unauthorized use of the e-passport in case it is lost. The requirement for keeping the MRZ secret is intensified by the fact that this information cannot change throughout the lifespan of the passport. Using the MRZ in a personal computer is acceptable, since the MRZ is only locally used and it is not disclosed to other parties. A precondition for

using an e-passport in a public system is that this system is widely trusted (in the same way as we trust an ATM for the card PIN we enter or a secure e-commerce application where we enter our credit card details).

- *No biometrics disclosure:* The biometric data (but not the facial image) are often considered sensitive personal data; therefore they must be further protected, in order to enable additional use of the passport. We consider that the biometric data are not necessary for the additional applications other than the border control. We, therefore, set as a precondition that the sensitive biometric data must not be present, or they must be kept secret for the applications we examine.

8 Conclusions

The International Civil Aviation Organization provides the system developers with the technical standards for the implementation of a simple worldwide Public Key Infrastructure to support digital signatures applied to e-passports. Countries are required to build their hierarchical PKI following specific standards and good practices, in terms of security, trust architectures, key management and interoperability.

Digital signatures are the basic tool to prove passport authenticity and integrity, while additional mechanisms provide access control and authentication. The digital signature unambiguously binds together the stored digital information, the issuing authority, the printed booklet, and the chip itself. Establishing global trust is necessary to efficiently verify the digitally signed e-passports at the inspection points worldwide. Respecting the autonomy of Countries, the establishment of a web-of-trust based on bilateral trust relationships between countries seems to be a promising and appropriate approach.

The e-passport is, in fact, a tamperproof PKI-enabled device, which contains a private key and a kind of digital certificate for the bearer. Since the e-passport is personalized under strict security procedures, as well as there is a worldwide trust and interoperability, it provides an attractive PKI establishment of ‘qualified certification’ that could be exploited in other applications. By examining the use of e-passports in applications such as the Internet or in public Point-Of-Sales, we conclude that their exploitation is possible if some preconditions are met. The wide use of e-passports exhibit important advantages, such as the existing worldwide Trust and standardization, the high level of security, and mobility and the legal conformance for ‘secure signature creation devices’. At the same time, the wide use may be restricted by some drawbacks, such as the need for special equipment, the lack of revocation mechanism, and the non-standard certificates.

Several security and privacy issues are identified, mainly originating from the fact that the e-passport is based on contactless communication and that it contains biometric data. Actions that strengthen the security of the e-passport include the assignment of random document numbers (that increase the entropy of access control keys), the implementation of a revocation mechanism for active authentication keys, and the separation of access control on the biometric data.

E-passports seem to have the potential to be used as global digital identification devices in everyday activities. The infrastructure that supports them may be proved to be the first worldwide PKI that will bring certificates, keys, and digital signatures close to all citizens and applications.

References

1. ICAO: Document 9303, Machine Readable Travel Documents, 6th edn. (December 2005)
2. ICAO: Machine Readable Travel Documents: Development of a Logical Data Structure – LDS, Technical Report, ver. 1.7 (May 2004)
3. ICAO: PKI for Machine Readable Travel Documents offering ICC Read-Only Access, Technical Report, ver. 1.1 (October 2004)
4. Gritzalis, S., Katsikas, S., Gritzalis, D., Stamatiu, Y., Lekkas, D., Marias, Y.: PKI Services in the Public Sector of the EU Member States – Chapter 6: Good Practices. University of the Aegean, Report to the Greek Presidency of the European Union (2003)
5. Housley, R.: Cryptographic Message Syntax, RFC 3369, IETF (2002)
6. Housley, R., Polk, W., Ford, W., Solo, D.: Internet, X.: 509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, RFC 3280, IETF (2002)
7. Jonsson, J., Kaliski, B.: Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1, RFC 3447, IETF (2003)
8. Lekkas, D.: Establishing and managing trust within the Public Key Infrastructure. Computer Communications 26(16), 1815–1825 (2003)
9. Bosworth, K., Tedeschi, N.: Public Key Infrastructures - the Next Generation. BT Technology Journal 19(3), 44–59 (2001)
10. Juels, A., Molmar, D., Wagner, D.: Security and privacy issues in e-passports. In: SecureComm 2005, 1st International Conference on Security and Privacy for Emerging Areas in Communication Networks, Greece (2005)
11. Yoshida, J.: Tests reveal e-passport security flaw. Electronic Engineering Times 1336, 1 (2004)
12. Maurer, U.: Intrinsic limitations of digital signatures and how to cope with them. In: Boyd, C., Mao, W. (eds.) ISC 2003. LNCS, vol. 2851, pp. 180–192. Springer, Heidelberg (2003)
13. Hancke, G.: Practical Attacks on Proximity Identification Systems. In: Proc. of the 2006 IEEE Symposium on Security and Privacy (S&P'06), pp. 328–333 (2006)
14. Dimitriou, T.: A Lightweight RFID Protocol to protect against Traceability and Cloning attacks. In: SecureComm 2005, 1st International Conference on Security and Privacy for Emerging Areas in Communication Networks, Greece (2005)
15. Kugler, D.: Advanced security mechanisms for Machine Readable Travel Documents, Technical Report, Federal Office for Information Security (BSI), Germany (2005)
16. Convey, A., Kupiszewski, M.: Keeping up with Schengen: Migration and policy in the European Union. The International Migration Review 29(112), 939–963 (1995)

An Interdomain PKI Model Based on Trust Lists

Helena Rifà-Pous and Jordi Herrera-Joancomartí

Universitat Oberta de Catalunya, Rbla del Poblenou, 156
08018 Barcelona, Spain
{hrifa,jherreraj}@uoc.edu

Abstract. The penetration of PKI technology in the market is moving slowly due to interoperability concerns. Main causes are not technical but political and social since there is no trust development model that appropriately deals with multidomain PKIs. We propose a new architecture that on one hand considers that trust is not an homogeneous property but tied to a particular relation, and on the other hand, trust management must be performed through specialized entities that can evaluate its risks and threads. The model is based on trust certificate lists that allows users to hold a personalized trust view without having to get involved in technical details. The model dynamically adapts to the context changes thanks to a new certificate extension, we have called TrustProviderLink (TPL).

Keywords: trust lists, reliability in PKI, interoperability, certificate extension.

1 Introduction

PKI technology has been widely accepted as the best solution to provide secure electronic transactions through an insecure channel. However, its global market penetration to common applications of general use it is not yet a fact.

The key reason for the slow adoption of PKI solutions in mass media products [1,2] is due to interoperability concerns. Interoperability can not be simply characterized as a technology-only issue. In fact, it encompasses a wide range of technical, legal and political issues.

The PKI industry has addressed technical interoperability problems through a standardization process of data types and protocols. Nowadays, despite the flexibility of the specifications, PKI technology has achieved maturity and the basic interoperability goals between different vendor solutions are guaranteed.

Therefore, today's major drawback for the PKI adoption is the difficulty to deploy a cross-border solution due to differences in the countries legal and political framework. Governments are reluctant to recognize other nation's CAs if they do not take part in the quality control and, on the other hand, the scope of liabilities of a CA is not clear if the jurisdiction does not regulate it by law. Several proposals have been presented to overcome these problems such as cross-certification and Bridge Certification Authority. Yet, none of them has succeed

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 49–64, 2007.

because of the complexities involved in the management of such infrastructures and the generalist perspective in which they are based.

The contribution of this paper is defining a procedure to facilitate the integration and interoperability of different PKI islands. The aim is being able to deal with elements of external security domains without creating a unique and monolithic infrastructure that is unable to adapt to any change. Users are the last responsible entities of the trust assignment within their context. Facilities are provided to identify the scope and attributions of each authority so that end entities can easily take the more appropriate decision for themselves.

The rest of the paper is structured as follows. In Section 2, the main trust models and their challenges are reviewed. Section 3 presents the proposed architecture and describes the entities and elements involved. In Section 4 we specify the functionality of the proposed architecture. Finally, section 5 concludes the paper and outlines some ideas for future research.

2 Trust Models and Challenges

PKI is intended to establish and maintain trusted relationships. In order to reach such objectives, mechanisms to propagate trust from credited organisms to unknown entities have to be built. See in [3] a survey on interoperability issues of multi-domain PKI. Next, we review the main trust model proposals and we identify their most relevant challenges.

2.1 Trust Models

Single CA
PKI trust development has been studied and analyzed from PKI origins. The most simple topology architecture is the single Certification Authority (CA) model, that is, all certificates are issued by a unique CA. Although simple, this design neither scales well nor adapts to the society patterns, so it leads to the appearance of multiple interconnected CAs which manage communities of users that can not interoperate ones with the others.

Hierarchical PKI
The first attempt to solve the problem of having multiple interconnected trust islands was the hierarchical PKI structure that is managed by a Root Certification Authority (RCA). Trust is established in a tree-like fashion and flows from top to bottom. The RCA public key is the fundamental point of trust, or trust anchor, for evaluating certificate acceptability. In this model the path construction procedure is very simple, as a single path exists from any end entity up to the RCA.

However, deploying a global unique RCA is inappropriate for political reasons. There is not a consensus about whom it would manage the RCA and how it would do it. Thus the conclusion is that this model is only directly applicable within one domain, which is generally supported in one or several communities

generally forming different security domains, but always within one unique administrative domain.

Mesh PKI

There has been multiple efforts to bring trust development to inter-domain levels. In the cross-certification model, also known as mesh model, two CAs cross-certify each other once they agree to trust and rely on each other's issued public key certificates as if they had generated them themselves. Pairs of CAs exchange cross-certificates and enable users from one administrative domain to interact electronically and securely with users from the other.

However, the number of cross-certificates tends to grow exponentially with the number of CAs and policy mappings are very complex. Therefore, there appear scalability problems.

On the other hand, if a PKI domain A wants to join another PKI domain B but restrict or deny trust in one or more other domains that B may have joined, A has to issue a cross-certificate to B where policy constraints shall be explicitly included. This makes the building of certification paths between two generic end entities still more unmanageable.

Mesh model can be also build upon a hierarchical PKI architecture so that the number of required cross-certificates can be reduced and the complexity of the system lightened. Although IETF has included CA cross-certification in its Certificate Management Protocol [4] and there are some implementations of it, cross-certification is still not well supported by common general-purpose applications like browsers or email clients.

Trust Lists

There is not an homogenous way to define or formalize trust lists. While for ones is just a list of certificates (i.e. stores of certificates used in web browsers), for others is a signed list that can contain any trusted information and, in particular, certificates; this is the case of the Certificate Trust List (CTL) format from Microsoft.

In this paper we use the term trust list to designate a signed set of trusted certificates plus information defining properties and constraints on how to apply this trust [5].

There are two types of trust lists:

- User trust lists: managed by a single user.
- Provider trust lists: managed by a trust provider.

The user trust list model is the most common trust development architecture in use today, offered by operating systems and web applications. An example is the list of more than a hundred CAs included in distributions of Microsoft OS. The requirement to appear in this list is paying a fee to Microsoft. On the other hand, end users can modify this list by adding or deleting CAs as needed.

User trust list model does not present technical complexities. However, it has to be noticed that users do not have the means or skills to construct their own trust list from scratch because they do not know the CAs nor are able to

evaluate the risks that entails accepting them. This is, for example, the case of web browser applications, that are distributed with a predefined trust list attached to them so that users do not have to create it.

On the other hand, provider trust lists are created and managed by trust providers (TP). Its aim is to serve as a reference for users; the trust on the certificates of the list is recommended by a TP. From an inter-domain interoperability perspective, the provider trust list essentially replaces the cross-certificate pair in the Mesh model. The user trusts the issuer of the list, adopts the list, and then the trust is extended to the CAs conveyed within it.

Bridge CA

Bridge CA (BCA) trust model was first introduced by the U.S. Federal Government [6] as a way to facilitate the interconnection of CAs through a cross-certification process. Each user only trusts its own CA which in turn trusts the bridge that finally trusts the remote CA, so that each member needs only to maintain a single cross-certification with the BCA and then it is automatically able to build certification paths across all spokes.

It must be noted that the BCA is not intended to be used as a trust anchor by the users of the PKI. It simply acts as a gateway between isolated CAs. Even so, BCA is responsible to map certificate policies and guarantee PKI equivalences adequately. Therefore, users must rely on it regarding these mappings.

Although this model is quite simple from the end user perspective, in fact it presents technical difficulties because the path construction is intrinsically complex and several checks (e.g. policy and name constraints, certificate status, policy mappings) must be performed throughout the certificate chain. Nowadays, the Federal BCA is trying to interconnect with other BCAs, but it is reporting both technical and operational problems (see Top 10 issues from [7]).

European Community took a project for deploying a BCA for the member states. In a way to facilitate the interoperability, their proposal [8] was to create an hybrid BCA that combines the distribution of accredited CA certificates under the form of a signed list, and the cross-certification of the CA with the BCA for those users that do not want to download trust lists.

The European BCA pilot finished in 2004. They concluded [5] cross-certification was more tedious than trust lists and that the model should be restricted to the use of these last ones.

Bridge VA

The Bridge Validation Authority (BVA) trust model [9,10] is a further step to the BCA in which the central entity is not a CA but a Validation Authority (VA). VAs are trusted third parties that offer online services on certificate validation. In general terms, they are responsible of building the certification path, evaluating the quality of the certificates, validating their status, and ensuring they are trustworthy.

In the BVA model, the element that links multiple PKI islands is an VA that gathers and classifies status information of certificates from multiple domains.

BVA becomes the trust anchor for users and admits liabilities for the certificates it works with.

This model solves some of the technical complexities of BCA, for instance, when dealing with path construction, and offers to the users a more comfortable service. Although simple, users are totally relegated to the decisions of the BVA and do not receive any information about the characteristics, quality or application context associated to the CA they are working with.

On the other hand, as we have already seen in other models, it can not exist a unique BVA in the world due to political reasons, but multiple instances of them. Whether users trust one or more BVA, and how they combine the results stated by each authority has to be solved for each user.

2.2 Trust Development Challenges

As we have described above, several solutions for building trust have been proposed, however, trust development still presents challenges. In the following use case, we expose the common problems users face up when dealing with security services.

Lets suppose a user, Bob, who has traveled to a foreign country and is visiting the city. While he is walking in the park, he sees a publicity announcing a music show that night in the theater. He wants to buy a ticket, so he gets his PDA an tries to connect to a wifi network. He finds an ad hoc network that reaches Internet and makes a request for establishing a connection.

Charles is another member of the ad hoc network. He is Bob's neighbor and can provide him with forwarding services so that Bob can connect to the theater. However, Charles would not forward Bob's packets unless he can assure Bob is not a selfish user of ad hoc networks, and that he will be rewarded afterward through a reputation system. Therefore, he wants to verify Bob's identity and that he is registered to a CA that takes liabilities in case of problems.

Charles verifies the signature of Bob's certificate and checks all data is correct. However Charles is unable to verify the certification chain of Bob up to a root CA because he does not know that authority and does not trust it. If Charles' CA has not a binding relationship with Bob's PKI domain, he has no further means to obtain relevant information for extending trust to Bob.

In this example case, let's suppose Charles takes the risk of giving forwarding services to Bob. Now, Bob can reach the Internet and tries to buy theater tickets. He can successfully build the vendor's certification path and validate the certification chain is correct. Still, he does not have any information on the quality of the certificate and if it is appropriate for using it in e-commerce applications. Bob does not want taking the risk of buying the tickets in a foreign country to a vendor that uses a certificate that is not qualified, so pitifully, he will not be able to go to the theater.

The example presented above shows that the challenges of trust models based on PKI can be summarized in:

- Processing certification paths, that is, finding an ordered sequence of certificates from the end entity certificate to a trust anchor.

- Determining the quality of the certificate, which can be usually derived from the certificate policy.
- Deciding if the certificate is trustworthy for the purpose at hand.

Some models like BVA and Trust Lists have a simple certification path management, however, they do not offer users details about quality of service parameters. If a CA issues two types of certificates with different security policies, the BVA party will extend the same type of responses for the two of them, without including quality information that can be used by the user as a qualifying parameter for the trust decision.

BCA model offers some quality of service through cross-certification policies. The BCA is responsible for certification mapping and it has to publish the rules that follows to accept new CAs. However, such architecture presents technical complexities that can harden interoperability.

Several proposals have emerged to formalize a certification policy format [11,12] and define a way to interpret the CA liabilities and the quality of the certificates it issues. In [13] authors define an automatic way to compare certificates from different CAs and determine its relative quality. The use of standard description rules will facilitate certificate evaluation. However, users neither have the knowledge nor the capacity to interpret all the parameters and take a decision about the quality of the certificate. So, they have to delegate this operation in some external entities, which we call Trust Providers (TP).

Another challenge of PKI trust models is how to extend trust and decide if a certificate is trustworthy for the purpose at hand. In spite of some technical complexities, the main problem of existing trust models is that they do not follow the trust building guidelines we use in personal relationships.

Setting up a global uniform network of trust is not viable. The models proposed so far that admit some personalization are the ones based on trust lists. However, current trust lists implementations are very simple and only cover certificate's classification in a few categories: trusted CA certificates, web server certificates, code validation certificates and end user certificates.

3 Architecture Description

In this section we describe the architecture of our trust model aimed to facilitate the multidomain PKI interoperability. The main characteristics of the proposed architecture are:

- Built upon a centralized PKI model that extends trust from well know authorities in which users trust, the TP.
- Let users configure its own trust list based on recommendations, that can be accepted or not.
- Facilitate trust list dissemination and management using core PKI mechanisms.

Different entities interoperate in our architecture using different elements (see figure 1).

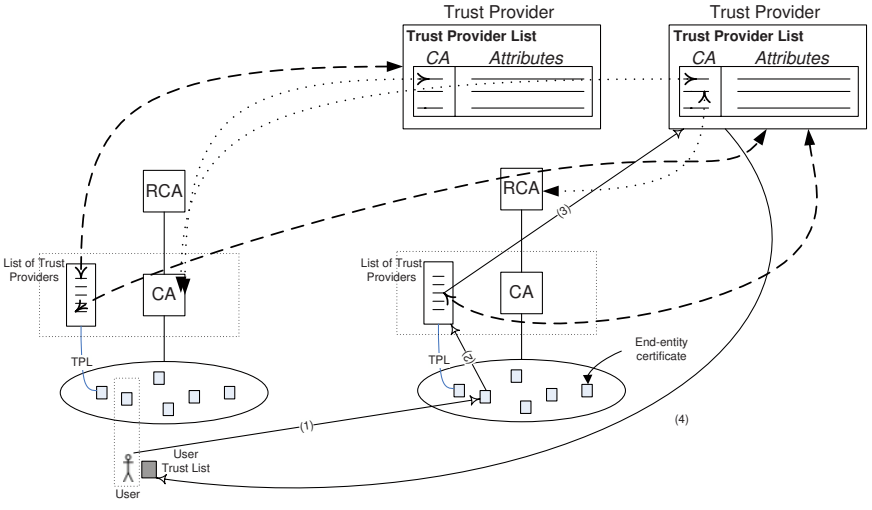


Fig. 1. System architecture

3.1 Architecture Entities

The main architecture entities of our model are users, Certification Authorities (CA) and Trust Providers (TP).

In our architecture we use the term user in a wider sense. A **user** is the one that has a certificate but also the entity that wants to validate a certificate. Notice that the latter does not need to have a certificate, and in some environments entities that validate certificates are called relying parties [14]. However we use the term user in both cases in order to simplify the description.

Certification Authorities (CA). are entities that issue certificates and vouch for the binding between the data items in these certificates. CAs manage the whole life cycle of certificates, revoking them if they are compromised before the validity period, renewing if the validity has to be extended, and publishing updated status information so that users can be accurately informed. Root Certification Authorities (RCA) are a special case of CA with a top hierarchical level.

Trust Providers (TP). are well known entities that have accredited political, legal or social impact (i.e. the Ministry of Law or recognized private enterprise). They manage lists of CA certificates that they consider reliable and that have the required quality for being used in some specific actions. The application context of the certificates in the list is confined to the influence scope of the Trust Provider.

3.2 Architecture Main Elements

Our architecture main elements are certificates, list of trust providers, trust lists, and a trust list enforcement engine.

Certificates
Certificates are the central point of any PKI based model. Our architecture uses X.509 certificates [15] in which a new certificate extension called Trust Provider Link (TPL) has been included.

The TPL is a noncritical certificate extension that contains URL locations where a list of trust providers can be retrieved. The TPL extension shall be identified by a unique object identifier:

```
id-pe-trustProvidersLink OBJECT IDENTIFIER  ::=  { id-pe 20 }
```

The ASN.1 specification of the extension is the following:

```
trustProvidersLink ::= SEQUENCE SIZE (1..MAX) OF
                          TrustProviderLink

trustProviderLink  ::=  SEQUENCE {
        accessMethod          OBJECT IDENTIFIER,
        accessLocation        GeneralName  }
```

`trustProvidersLink` is a list of `trustProviderLink` objects that contain the location where the CA that issued the certificate publishes the list of TP.

List of trust providers
A list of trust providers is the relation of different TP that support the CA policy. Such list is formatted in XML and includes a brief description of the TP included. Table 1 shows the elements of the list in XPath language [16].

Notice that lists of trust providers are signed to avoid fake manipulations in the data. A specific example of a list of providers is shown in figure 2.

Trust lists
The main element of our architecture are trust lists. Two different trust lists can be identified: user trust lists and provider trust lists, which are managed by TPs so that they are also known as TP trust lists.

Trust lists contains three groups of information:

- List owner data: It identifies the entity responsible of the trust list. In case such entity is a TP the list is more reliable and it includes the services offered by the TP, the number of clients that it has, the purpose of the list and its target. It also states if the TP is refining an existing trust list, or it creates it from scratch.
- Certificate list: List of trusted certificates and qualifying information.
- Authenticity data: It includes parameters to validate the integrity and authenticity of the data contained in the list. Such parameters include a hash of the data, a signature and the signing certificate.

Table 1. List of trust providers parameters

XPath	Value
`/List`	List container
`/List/TrustProvider[n]`	Information of a n'th Trust Provider (TP) of the list
`/List/TrustProvider[n]/Name`	TP's distinguished name
`/List/TrustProvider[n]/Context`	The operational context of the TP, i.e, public administrations, e-commerce, ..
`/List/TrustProvider[n]/Scope`	TP's influence ambit. It can be global (i.e. Microsoft) or local (trust list of a European member state)
`/List/TrustProvider[n]/CertProvider`	TP's certification provider, in case it exists
`/List/TrustProvider[n]/Reference`	URL location of the trust lists
`/List/ds:Signature`	XML Signature of the whole list

```
- <List>
  - <TrustProvider>
      <Name>CN=TrustProvider, O=Ministry of Law, C=ES</Name>
      <Context>Law</Context>
      <Scope>Spain</Scope>
      <CertProvider>OU=FNMT Clase 2 CA, O=FNMT, C=ES</CertProvider>
      <Reference>https://www.minlaw.es/trust_provider/trust_list</Reference>
    </TrustProvider>
  - <TrustProvider>
      <Name>CN=TrustProvider, O=UOC, C=ES</Name>
      <Context>E-learning</Context>
      <Scope>Europe South-America</Scope>
      <CertProvider>O=UOC, C=ES</CertProvider>
      <Reference>https://www.uoc.edu/trust_provider/trust_list</Reference>
    </TrustProvider>
  - <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
    - <ds:SignedInfo>
        <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
        <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
      - <ds:Reference URI="">
        - <ds:Transforms>
            <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
          </ds:Transforms>
          <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
          <ds:DigestValue>WScEsaC0y3PzG3cC9nUqLmTrnN8=</ds:DigestValue>
        </ds:Reference>
      </ds:SignedInfo>
      <ds:SignatureValue>gXqJmcIquM0PQjMWIziSXDiLHa6YhEmROorIMh65TEVdvA0c5bH1LJLVinC
        9od5nmarv8fIfF9Y/nDVW7Ad5NHcTYO+EOhyDA==</ds:SignatureValue>
    - <ds:KeyInfo>
      - <ds:X509Data>
          <ds:X509Certificate>MIIEIzCCAwugAwIBAgIBBDANBgkqhkiG9w0BAQUFADCBnjELMAkGAU
            G0+l2k2Y9fzfBqwx7OekTR+OQsMiTHuU/PvQq03uCcZHILF7Qw==</ds:X509Certificate>
        </ds:X509Data>
      </ds:KeyInfo>
    </ds:Signature>
  </List>
```

Fig. 2. List of trust providers example

We introduce a format of trust lists that allows defining quality parameters for the certificates. Table 2 shows the elements of the list in XPath language. The format of the list is based on the ETSI specification [17] that has been extended to include certificates data.

Table 2. Trust lists parameters

XPath	Value
/TrustList	Trust List (TL) container
/TrustList/ThisUpdate	Publication date
/TrustList/Target	Public target
/TrustList/Purpose	Purpose
/TrustList/From	List of Trust Providers (TP) that have contributed to the creation of the present TL
/TrustList/From/TrustProvider[n]	Information of a n'th TP
/TrustList/From/TrustProvider[n]/Id	Distinguished Name
/TrustList/From/TrustProvider[n]/Update	Publication date of the provider's list
/TrustList/Owner	Information of the responsible of the TL
/TrustList/Owner/Id	Distinguished Name
/TrustList/Owner/Name	Name of the responsible legal entity
/TrustList/Owner/Address	Postal address
/TrustList/Owner/URI	Web address
/TrustList/Owner/Services	List of services provided by the entity
/TrustList/Owner/BeginDate	Legal entity creation date
/TrustList/Owner/Context	The operational context of the owner, i.e, public administrations, e-commerce, ..
/TrustList/Owner/Market	Market of the owner
/TrustList/Owner/Status	Financial range of the owner
/TrustList/CertList	List of trusted certificates
/TrustList/CertList/CA[n]	Information of a n'th CA. This element includes trust parameters in semantic languages
/TrustList/CertList/CA[n]/CertProperties	Encoded certificate and certificate fields
/TrustList/CertList/CA[n]Constraints	Information to qualify the certificate
/TrustList/ds:Signature	XML Signature of the whole list

TP evaluate certificates they want to include in their trust list based on their knowledge of the CA holder and the quality of the issued certificates (that is, the CA certificate policy). Results from the evaluation are expressed on the list using an ontology language [18]. Ontology languages are formal languages used to construct ontologies, that is, specifications of some concepts. They allow the encoding of knowledge about specific domains and often include reasoning rules that support the processing of that knowledge.

We use ontologies to include any properties and constraints for the certificates issued by the CAs in the list. Categorization of certificates is achieved with information of two orthogonal fields (see figure 3):

– **Service:** the security services that shall be granted, like identification, data authentication, access control, non-repudiation or establishment of a confidential channel.

– **Sector:** the ambit of the target application, i.e., government, pharmaceutical, banking, transport, leisure, etc.

Finally, TP define a certificate Trust Level for using the stated certificates in each specific environment.

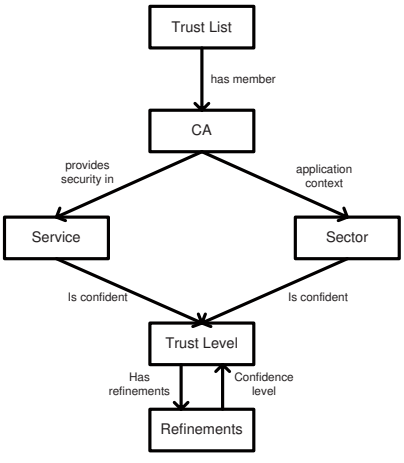


Fig. 3. Trust list ontology schema

Ontologies provide means to define relations between the objects in a domain. This information is exploited in trust lists as a way to automatically propagate trust on certificates used in similar environments. For example, a TP may recommend a certificate to be accepted for non-repudiation services in governmental applications. By extension, if a user asks if he shall accept this certificate for authentication purposes in an e-administration environment, the response will be true.

Extending the context of use of a certificate entails some risks, for this reason, the confidence level of this kind of recommendations will always be lower than the primary goals for it was intended. The client application is responsible to define a threshold above which certificates will be accepted, and rejected otherwise.

Figure 4 shows an example of a TP trust list. The TP is a medical enterprise that offers health services and insurances. It has defined a trust list of certificates to be used in social services. In particular, the example depicts a CA that can be used for non repudiation operations in an e-health context.

Trust list enforcement engine

Since there are many independent PKI domains in the world and new ones are appearing every day, user trust lists have to be updated regularly to meet the changes. New methods have to be provided in order to modify and include the appropriate CAs in the list and trust list enforcement engines are the appropriate tool.

```
- <TrustList>
    <ThisUpdate>2007-03-02T09:16:16Z</ThisUpdate>
    <Target>Spanish citizens</Target>
    <Purpose>Social services access</Purpose>
    <From created="true" />
  - <Owner>
      <Id>CN=Medicus, C=ES</Id>
      <Name>Medicus, S.A.</Name>
      <Address>Calle Alcala, 43. 3-2 Madrid</Address>
      <URI>http://www.medicus.es</URI>
    - <Services>
        <Service>Health</Service>
        <Service>Insurances</Service>
      </Services>
      <BeginDate>1985-05-23T08:00:00Z</BeginDate>
      <Context>Public Health</Context>
    </Owner>
  - <CertList>
    - <CA rdf:ID="AC Raiz DNIE 01" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
      - <CertProperties rdf:resource="#x509cert">
          <Subject>CN=AC Raiz DNIE, OU=DNIE, O=Direccion general de la policia, C=ES</Subject>
          <Issuer>CN=AC Raiz DNIE, OU=DNIE, O=Direccion general de la policia, C=ES</Issuer>
        - <Validity>
            <NotBefore>2006-02-27T11:54:38Z</NotBefore>
            <NotAfter>2021-02-26T23:59:59Z</NotAfter>
          </Validity>
          <EncodedCert>MIIFxTCCA62gAwIBAgIQZCBmyZl7ruFEAtpupCL9w0BAQUFADBdUE9MSU
            MQswCQYDVQQGEwJFUzEoMCYGA1UECgwfRElSRUNQUwgREUgTEEg</EncodedCert>
        </CertProperties>
      - <Constraints>
          <ProvidesSecurity rdf:resource="#NonRepudiation" />
          <ApplicationContext rdf:resource="#eHealth" />
          <TrustLevel>8</TrustLevel>
        </Constraints>
      </CA>
    </CertList>
  + <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  </TrustList>
```

Fig. 4. Trust list example

Trust lists are tied with trust list enforcement engines, that are used to encode users requirements and check if certificates fulfill the stated rules according to the list. The engine is the interface of trust lists that can interpret users requirements in natural language and automatize the process of list modification and update.

4 Architecture Function

4.1 User Trust List Creation

Users trust lists are initially created importing the information from a provider trust list of an entity whom the user knows and trusts. Therefore, users construct they own trust list based on TP recommendations, that are formalized in an TP trust list.

TP can offer a downloading interface that allows exporting only some parts of the list (some selected CAs). Anyway, the information users accept from the provider trust lists, can be then refined and modified.

4.2 User Trust List Management

Trust providers value and classify certificates in the lists they publish. Users can refine or modify this classification in the lists managed by themselves. When a user imports new providers trust lists to his own list, the trust list enforcement engine merges the data with the contents of the other providers.

In our model, the use of user trust lists is not mandatory. There is a way to fasten the trust validation of a certificate and guarantee the service is available anytime. However, users that need to check a certificate and are not using their personal device nor having access to their trust list, can perform the validation directly. The trust list enforcement engine directs the user to the list of TP referenced in the certificate under validation, the user will have to indicate if he trusts any of the TP on the list and, if positive, the trust list of the selected TP will be checked and its recommendations imported.

4.3 Certificate Evaluation

In the general model, users maintain a trust list that contains the PKI domains they trust. When they are involved in an electronic transaction and need to decide the acceptance of a certificate, they delegate to the trust list enforcement engine the certification path construction and the evaluation of whether the certificate is sufficiently trusted and qualified for that specific purpose. The inputs to the engine are the certificate itself, and a description of the context in use. For example, a user can request if a certificate is admissible for non-repudiation purposes in a work contract.

Certificate evaluation can be directly requested from the client application itself (mail client, web browser, document reader, ...). In this case, the client application is the one that passes contextual inputs to the trust list enforcement engine.

Figure 1 shows the certificate evaluation steps:

Step 1. The enforcement engine tries to build a trust certification path from the user certificate to a trust CA in the list. If it is unable to do it, it can be because it does not know that authority, or because it does know it but the user does not explicitly trust in it. In the first case, as the certificate under evaluation holds a TPL extension, the engine can obtain more information about the PKI domain where the certificate belongs through the recommendations made by some TPs. The TPL extension points to a list of TPs that can supply more information about the unrecognized certification chain.

Step 2. The enforcement engine connects to the web page in which the CA that issued the certificate publishes a list of TPs. All TPs in the list support the CA, however, the context and scope of each provider is different. If the list of TPs is not available, the certificate under evaluation is considered untrustful and the process is finished.

Otherwise, after getting the list of TPs and verifying that is authentic and has not been manipulated, the engine searches if the user knows some of the

providers, that is, checks if TPs are registered in the user trust list. If this is the case, the recommendation of the registered TPs is accepted. Otherwise, the engine asks the user if he want to import some new TP in the repository and acts accordingly.

Step 3. TPs are reached. The client verifies the signature of the TP trust lists.

Step 4. TPs trust lists are downloaded and merged with the current information in the user trust list. When recommendations on a PKI domain come from several trust providers, the best trust level values are the ones that prevail. If information of trust providers has to be merged with data stated by the user owning the list, the user opinion takes preference.

Finally, we review the use case example we have introduced in section 2.2 to explain how it can be resolved using our trust list model. Users Bob and Charles are connected in an ad hoc network. Bob requests Charles forwarding access through his node so that he can reach Internet. Charles is not able to construct Bob's certification chain and he does not trust the authority. As Bob's certificate holds a TPL extension, he can get information about the TP that recommend the acceptance of the certificate.

Charles recognizes some of the TP on the list. TP are reached in order, starting by the provider that best meets the requirements of the situation, and ending with the remotest one. For each provider that recommends the use of the certificate, the client application alerts the user and asks for a confirmation.

In our particular case, Charles seeks confidence parameters in the context of ad hoc networks, and particularly in the access control operation. TP that support Bob certification chain are unknown to Charles, and are in the public administration domain. However, because the number of known TP that recommend the use of the certificate is very high, and because the risk of the operation is low, Charles accepts the identity of Bob and grants him the access.

So far, Bob can browse the Internet. He connects to the theater for buying the ticket. However, the vendor presents him a certificate that he does not recognize because does not maintain a user trust list in his handheld device due to its limited capacity. He asks the trust list enforcement engine to verify the validity of the certificate for e-commerce services.

The engine verifies that the vendor certificate is issued by a CA that is member of the certificate trust lists of the major software vendors, and that the context of this CA is e-commerce. It asks Bob if he relies on the TP on the list, and as the response is affirmative, the engine validates positively the vendor certificate and Bob is able to perform the purchase.

5 Conclusions

Interoperability issues of interdomain PKIs are one of the key problems to be solved in order to allow a massive deployment of PKI technology. Although several solutions have been proposed so far, none of them has succeed in the market due to technical, political and social constraints.

In this paper we have presented a new trust development model more flexible than the former ones from the user point of view. The proposed model is based on trust lists in order to resolve interoperability issues of interdomain PKIs. It is built upon a hierarchical PKI model and extends trust using TPs. As in the BVA model, the trust anchor is not a CA, but the TP.

The use of TPs facilitates the adoption of interdomain CAs by the users because they are close entities which users really know and treat. TPs have gained their reputation in a certain community by their demonstrated know-how, activities and projects, and they are trusted in their area of expertise. TPs are used to give recommendations, help users to interpret CAs security policies and give them information about the PKI domains. On the other hand, the promotion and dissemination of TPs is achieved thanks to a new certificate extension that provides information of the entities that support its PKI domain.

Moreover, the presented architecture deals with categorized trust lists expressed in semantic language so that a more specific approach about when is appropriate or not to accept a certificate can be surely stated.

The use of ontologies is twofold: first it permits to describe the complex relations of the real world in a language that is interpretable by computers; Second, it provides the base to deploy natural language interfaces for the TPs so that user-friendly applications can be developed and consumed by anyone, although they do not have technical skills.

Further research will be focused on defining specific ontologies for the trust list enforcement engine to allow a highly configurable and automatic user list management.

Acknowledgement

The work described in this paper has been supported in part by the Spanish MCYT with a grant for the project PROPRIETAS-WIRELESS SEG2004-04352-C04-04.

References

1. Backhouse, J., Hsu, C., Baptista, J., Tseng, J.C.: The key to trust? signalling quality in the PKI market. In: Proceedings of the 11th European Conference on Information Systems, ECIS (2003)
2. Doyle, P., Hanna, S.: Analysis of June 2003 Survey on Obstacles to PKI Deployment and Usage. OASIS Public Key Infrastructure (PKI) Technical Committee (TC) (2003)
3. Shimaoka, M., Hastings, N., Nielsen, R.: Memorandum for multi-domain Public Key Infrastructure Interoperability. IETF Internet Draft (2007)
4. Adams, C., Farrell, S., Kause, T., Mononen, T.: Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP). IETF RFC 4210. Standards Track (2005)
5. Certipost: Trust List Usage Recommendations for a European IDA Bridge/Gateway CA Pilot for Public Administrations. IDA PKI II / EBGCA / WP1.2 (2004)

6. Burr, W.E.: Public Key Infrastructure (PKI) technical Specification: Part A – Technical Concept of Operations. NIST Working Draft (1998)
7. Blanchard, D.: I-CIDM Bridge to Bridge Interoperations. In: 5th Annual PKI R&D Workshop Making PKI Easy to Use (2006)
8. EDS: A bridge CA for Europe's Public Administrations - Feasibility study. European Commission - Enterprise DG. Public Key Infrastructure for Closed User Groups Project (2002)
9. Malpani, A.: Bridge Validation Authority. White Paper, ValiCert (2001)
10. Ølnes, J.: PKI Interoperability by an Independent, Trusted Validation Authority. In: 5th Annual PKI R&D Workshop Making PKI Easy to Use (2006)
11. Casola, V., Mazzeo, A., Mazzocca, N., Vittorini, V.: Policy Formalization to combine separate systems into larger connected network of trust. In: Proc. of Int. Conf. on Network Control and Engineering for QoS, Security and Mobility (Net-Con) (2002)
12. Chokhani, S., Ford, W., Sabett, R., Merrill, C., Wu, S.: Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework. IETF RFC 3647 Informational (2003)
13. Casola, V., Mazzeo, A., Mazzocca, N., Rak, M.: An Innovative Policy-Based Cross Certification Methodology for Public Key Infrastructures. In: Chadwick, D., Zhao, G. (eds.) EuroPKI 2005. LNCS, vol. 3545, pp. 100–117. Springer, Heidelberg (2005)
14. Cooper, M., Dzambasow, Y., Hesse, P., Joseph, S., Nicholas, R.: Internet X.509 Public Key Infrastructure: Certification Path Building. IETF RFC 4158. Informational (2005)
15. Housley, R., Polk, W., Ford, W., Solo, D.: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. IETF RFC 3280. Standards Track (2002)
16. Berglund, A., Boag, S., Chamberlin, D., Fernndez, M.F., Kay, M., Robie, J., Simon, J.: XML Path Language (XPath) 2.0. W3C Recommendation (2007)
17. ETSI: Electronic Signatures and Infrastructures (ESI); Provision of harmonized Trust Service Provider status information. Draft ETSI TS 102 231 V1.2.1 (2005)
18. Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D.L., Patel-Schneider, P.F., Stein, L.A.: OWL Web Ontology Language. W3C Recommendation (2004)

One-More Extension of Paillier Inversion Problem and Concurrent Secure Identification

Yan Song[1,2]

[1] State Key Lab. of Computer Science, Institute of Software,
Chinese Academy of Sciences, P.O.Box 8718, 100080 Beijing, China
[2] Graduate University of Chinese Academy of Sciences, Beijing, China
songyan03@ios.cn

Abstract. In this paper, we revisit Paillier's trapdoor one-way function [15], focusing on the computational problem underlying its one-wayness. We formulate a new computational problem that we call *one-more Paillier inversion problem*. It is a natural extension of Paillier inversion problem to the setting where adversaries have access to an inversion oracle and a challenge oracle. We study the relation between the proposed problem and the one-more RSA inversion problem introduced by Bellare *et al.* in [2]; we prove that the one-more Paillier inversion problem is hard if and only if the one-more RSA inversion problem is hard. Then we propose a new identification scheme; we show the assumed hardness of the one-more Paillier inversion problem leads to a proof that the proposed identification scheme achieves security against concurrent impersonation attack. Compared with the known RSA-related identification schemes, the proposed identification scheme is only slightly inefficient than the best known GQ scheme, but is more efficient than Okamoto's.

1 Introduction

Paillier's cryptosystem [15] is an important member of a family of public-key, probabilistic encryption schemes utilizing a discrete logarithm trapdoor technique modulo a hard-to-factor integer. This family begins when Goldwasser and Micali [10] introduce the notion of probabilistic encryption. The probabilistic encryption scheme of Goldwasser and Micali is based on the quadratic residuosity assumption. Cohen and Fischer [5] improve the limited communications bandwidth of the Goldwasser-Micali scheme; their encryption scheme is based on the prime residuosity assumption. However, the decryption procedure is inefficient since it involves a certain exhaustive search. Naccache and Stern [12] suggest a variant on the Cohen-Fischer scheme; their scheme allows for high communications bandwidth and is proved to be semantically secure under the same assumption, namely the prime residuosity assumption. Independently, Okamoto and Uchiyama [14] propose an improvement on the Cohen-Fischer scheme, this time using a different group structure; the semantic security of the Okamoto-Uchiyama scheme is proved under the p-subgroup assumption. Paillier [15] proposes a new candidate trapdoor one-way function on which he builds a new

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 65–77, 2007.

encryption scheme; the semantic security of Paillier's encryption scheme is proved under the decisional composite residuosity assumption (in contrast to the prime residuosity assumption), and is more efficient than the aforementioned schemes. Following [15], many related works have been done, mainly concerned with modifications, extensions or applications of Paillier's cryptosystem; see, for example, [7,4,9].

In this paper, we revisit Paillier's trapdoor one-way function, focusing on the computational problem underlying its one-wayness, namely the Paillier inversion problem [15]. We formulate a new computational problem that we call *one-more Paillier inversion problem*. It is a natural extension of Paillier inversion problem to the setting where adversaries have access to an inversion oracle and a challenge oracle. We study the relation between the proposed problem and the one-more RSA inversion problem introduced by Bellare *et al.* in [2]. We prove that the one-more Paillier inversion problem is hard if and only if the one-more RSA inversion problem is hard; that is, in regard to intractability the one-more Paillier inversion problem is equivalent to the one-more RSA inversion problem. We then propose a new identification scheme derived from a Σ-protocol for proof of knowledge of pre-image under Paillier's function; we show the assumed hardness of the one-more Paillier inversion problem leads to a proof that the proposed identification scheme is secure against concurrent impersonation attack. Compared with the known RSA-related identification schemes, the proposed identification scheme is only slightly inefficient than the best known GQ scheme [11], but is more efficient than Okamoto's [13].

2 The One-More Paillier Inversion Problem

In Section 2.1, we review the Paillier inversion problem underlying the one-wayness of Paillier's trapdoor one-way function ([15]). In Section 2.2, we formulate the *one-more Paillier inversion problem*, which is analogous to the one-more RSA inversion problem in [2].

Throughout this paper, we let positive integer k denote the *security parameter*. An RSA-type *modulus generator* is a probabilistic, polynomial time algorithm that on input 1^k returns an RSA-type modulus N, which is k-bit long (namely $2^{k-1} \leq N < 2^k$), and is a product of two distinct odd primes. For an RSA-type modulus N, we denote by $\mathbb{Z}_N$ the ring of integers modulo N, and by $\mathbb{Z}_N^*$ the multiplicative group of units (namely invertible elements in $\mathbb{Z}_N$).

An RSA-type *key generator* is a probabilistic, polynomial time algorithm that on input 1^k returns a triple (N, e, d), where the first component $N = pq$ is a k-bit long modulus that is the product of two distinct odd primes p and q, the second component e is an encryption exponent, and the third component d is the matching decryption exponent.

As done in [15], we fix an RSA-type key generator that will be referred to as $\mathcal{K}$ throughout. The generator $\mathcal{K}$ on input 1^k returns a modulus N, an encryption exponent e and the matching decryption exponent d. The modulus N is k-bit long, and is a product of two distinct odd primes. The encryption exponent e

equals N—the modulus itself. The decryption exponent d is the multiplicative inverse of N modulo $\lambda(N)$, where $\lambda(N)$ is Carmichael function on N (namely, in this context, the least common multiple of $p-1$ and $q-1$, $\mathrm{lcm}(p-1, q-1)$).

2.1 The Paillier Inversion Problem

Let $N = pq$ be an RSA modulus and consider the multiplicative group $\mathbb{Z}_{N^2}^*$ of units. Paillier [15] proposes a candidate trapdoor one-way function P from $\mathbb{Z}_N \times \mathbb{Z}_N^*$ to $\mathbb{Z}_{N^2}^*$ defined by

$$P\colon (x, y) \mapsto (1+N)^x y^N \bmod N^2. \tag{1}$$

Paillier actually presents his candidate trapdoor one-way function in terms of an arbitrary base g whose order modulo N^2 is a multiple of N, instead of the specific base $1+N$, which has order N modulo N^2. However, since the Paillier inversion problem is random self-reducible (cf. [15]), it follows that the hardness of the inversion problem is independent of the choice of the base g. Hence, the Paillier inversion problem can be formulated in terms of the key generator $\mathcal{K}$ and the function P.

Definition 1 (Paillier Inversion Problem). Let k be a security parameter, and let $\mathcal{A}$ be an adversary. Consider the following

Experiment $\mathrm{Exp}_{\mathcal{A}}(k)$
$N \leftarrow \mathcal{K}(1^k)$
$z \leftarrow \mathbb{Z}_{N^2}^*$, $(x, y) \leftarrow \mathcal{A}(N, 1^k, z)$
If $z = (1+N)^x y^N \bmod N^2$ then return 1 else return 0.

We define the advantage of $\mathcal{A}$ by

$$\mathrm{Adv}_{\mathcal{A}}(k) = \Pr[\mathrm{Exp}_{\mathcal{A}}(k) = 1].$$

The *Paillier inversion problem* is said to be hard, if the function $\mathrm{Adv}_{\mathcal{A}}(k)$ is negligible in the security parameter k for any adversary whose time complexity is polynomial in k. □

It is implicit in [15] that the Paillier inversion problem is equivalent to the RSA inversion problem as per Definition 3 in Appendix A.

2.2 The One-More Paillier Inversion Problem

Recall that associated to an RSA modulus N is the function P defined by Eq. (1). The Paillier inversion problem involves computing the pre-image of z under P, given z in $\mathbb{Z}_{N^2}^*$. We are also interested in the setting where adversaries may have access to an inversion oracle and a challenge oracle. The adversary may query with a point $z \in \mathbb{Z}_{N^2}^*$ to the inversion oracle and get back the corresponding pre-image under function P. The adversary may also query the challenge oracle who upon request simply returns a random element in $\mathbb{Z}_{N^2}^*$. In this context, we

consider the adversary to be successful if it outputs correct pre-images of all the challenge points returned by the challenge oracle during the interaction, and manages to keep the number of its queries to the inversion oracle strictly fewer than the number of its queries to the challenge oracle. Our definitional approach is to assign to any adversary an *advantage function* that sends the security parameter to the probability that an attack game or experiment returns 1. The problem is said to be hard if and only if any adversary with time complexity polynomial in the security parameter has only negligible advantage.

We now proceed to define the one-more Paillier inversion problem. We denote by I() the *inversion oracle* that on input an element $z \in \mathbb{Z}_{N^2}^*$ returns its pre-image under function P. The adversary is also allowed access to a *challenge oracle* denoted by C(). The challenge oracle simply returns a random challenge point in $\mathbb{Z}_{N^2}^*$ each time being queried; this element is chosen independently and uniformly from $\mathbb{Z}_{N^2}^*$. The adversary is given oracle accesses to both I() and C(). Its goal is to find the correct pre-images of all the challenge points returned by the challenge oracle C() in such a way that the number of its queries to the inversion oracle I() is strictly fewer than the number of its queries to the challenge oracle C().

Definition 2 (One-More Paillier Inversion Problem). Let k be a security parameter. Let $\mathcal{A}$ be an adversary with accesses to the inversion oracle I() and the challenge oracle C(). Consider the following

Experiment $\mathrm{Exp}_{\mathcal{A}}^{\text{om-p}}(k)$
$N \leftarrow \mathcal{K}(1^k)$
$((x_1, y_1), \ldots, (x_m, y_m)) \leftarrow \mathcal{A}^{\mathrm{I}(),\mathrm{C}()}(N, 1^k)$
where m is the number of $\mathcal{A}$'s queries to the challenge oracle C()
Let $z_1, \ldots, z_m$ be the challenge points returned by C()
If the following hold then return 1 else return 0
- For all $i \in \{1, \ldots, m\}$, $z_i = (1+N)^{x_i} y_i^N \bmod N^2$
- The number of queries to I() made by $\mathcal{A}$ is strictly fewer than m.

The advantage function of $\mathcal{A}$ in the above experiment is defined by

$$\mathrm{Adv}_{\mathcal{A}}^{\text{om-p}}(k) = \Pr[\mathrm{Exp}_{\mathcal{A}}^{\text{om-p}}(k) = 1]$$

The *one-more Paillier inversion problem* is said to be hard if $\mathrm{Adv}_{\mathcal{A}}^{\text{om-p}}(k)$ is negligible for any adversary whose time complexity is polynomial in the security parameter k. □

We adopt the convention that the *time complexity* of a one-more Paillier inversion adversary $\mathcal{A}$ is the execution time of the entire experiment, including the time taken for key generation, but only one time unit is charged for each reply to an oracle query (namely the time taken by the oracles to compute replies is one time unit).

3 The Equivalence

In this section, we investigate the relation between the one-more Pailler inversion problem as per Definition 2 and the one-more RSA inversion problem introduced

in [2]; see also Definition 4 in Appendix A. We show that in regard to computational intractability, these two problems are polynomially equivalent.

Theorem 1. The one-more Paillier inversion problem as per Definition 2 is hard if and only if the one-more RSA inversion problem as per Definition 4 is hard. □

Proof. For one implication, we show that given any probabilistic, polynomial time one-more Paillier inversion adversary $\mathcal{A}$, there exists a probabilistic, polynomial time one-more RSA inversion adversary $\mathcal{B}$ such that their advantage functions satisfy the following inequality

$$\mathrm{Adv}_{\mathcal{A}}^{\text{om-p}}(k) \leq \mathrm{Adv}_{\mathcal{B}}^{\text{om-rsa}}(k), \tag{2}$$

which would imply that if the one-more RSA inversion problem is hard, then the one-more Paillier inversion problem is also hard.

To do this, recall that by Definition 4, the one-more RSA inversion adversary $\mathcal{B}$ is given oracle accesses to the RSA inversion oracle RSAI() and the RSA challenge oracle RSAC(). The inversion oracle RSAI() takes an element $\tilde{z} \in \mathbb{Z}_N^*$ and returns its RSA-inverse $\tilde{y} = \tilde{z}^{N^{-1} \bmod \lambda} \bmod N$. The challenge oracle RSAC() returns a point chosen independently and uniformly from $\mathbb{Z}_N^*$ each time being queried. The algorithm of the one-more RSA inversion adversary $\mathcal{B}$ is described as follows:

Adversary $\mathcal{B}^{\mathrm{RSAI}(),\mathrm{RSAC}()}(N, 1^k)$
  Run $\mathcal{A}$ on input $(N, 1^k)$, answer $\mathcal{A}$'s oracle queries as follows
    If $\mathcal{A}$ queries its challenge oracle, then
      Query RSAC() to get $\tilde{z}$, pick a random number x in $\mathbb{Z}_N$,
      and compute $z = (1+N)^x \tilde{z} \bmod N^2$. Return z
    If $\mathcal{A}$ queries z to its inversion oracle, then
      Query RSAI() with $\tilde{z} = z \bmod N$ to obtain $\tilde{y}$. Compute
      $x = (z/\tilde{y}^N - 1)/N \bmod N$. Return $(x, \tilde{y})$
  Until $\mathcal{A}$ halts with some outputs $((x_1, y_1), \ldots, (x_m, y_m))$
  Return $(y_1, \ldots, y_m)$

Notice that the map from $\mathbb{Z}_N \times \mathbb{Z}_N^*$ to $\mathbb{Z}_{N^2}^*$ defined by $(x, y) \mapsto (1+N)^x y \bmod N^2$ is a group isomorphism. It follows that the challenge point z produced by $\mathcal{B}$ is distributed identically to that returned by $\mathcal{A}$'s challenge oracle (namely random element in $\mathbb{Z}_{N^2}^*$).

Now, let $\tilde{z}_1, \ldots, \tilde{z}_m$ be the challenge points returned by the oracle RSAC(), and $z_1, \ldots, z_m$ the corresponding challenges computed by $\mathcal{B}$ (namely, the challenge points $\mathcal{A}$ actually receives). If $\mathcal{A}$ is successful, then the number of inversion queries made by $\mathcal{A}$ is strictly fewer than the number of its challenge queries, and for all $i \in \{1, \ldots, m\}$, the pair (x_i, y_i) is the correct pre-image of z_i under function P, that is, $(1+N)^{x_i} \tilde{z}_i \bmod N^2 = z_i = (1+N)^{x_i} y_i^N \bmod N^2$. Changing modulus from N^2 to N, we have $\tilde{z}_i = y_i^N \bmod N$; namely, y_i is the RSA-inverse of $\tilde{z}_i$ modulo N. Hence, the adversary $\mathcal{B}$ inverts all the m challenges but makes strictly fewer than m queries to its inversion oracle, as adversary $\mathcal{A}$ does. This establishes the desired inequality (2).

For the other implication, we show that given any probabilistic, polynomial time one-more RSA inversion adversary $\mathcal{B}$, there exists a probabilistic, polynomial time one-more Paillier inversion adversary $\mathcal{A}$ such that their advantage functions satisfy the following inequality

$$\mathrm{Adv}_{\mathcal{B}}^{\text{om-rsa}}(k) \leq \mathrm{Adv}_{\mathcal{A}}^{\text{om-p}}(k),$$

which would imply that if the one-more Paillier inversion problem is hard, then the one-more RSA inversion problem is also hard.

Recall that adversary $\mathcal{A}$ is allowed oracle accesses to an inversion oracle I() and a challenge oracle C(). The inversion oracle I() takes input a point $z \in \mathbb{Z}_{N^2}^*$ and returns its pre-image (x, y) under function P. The challenge oracle C() simply returns a point chosen independently and uniformly from $\mathbb{Z}_{N^2}^*$, each time being queried. The algorithm of the one-more Paillier inversion adversary $\mathcal{A}$ is described as follows:

Adversary $\mathcal{A}^{\mathrm{I}(),\mathrm{C}()}(N, 1^k)$
  Run $\mathcal{B}$ on input $(N, 1^k)$, answer $\mathcal{B}$'s oracle queries as follows
    If $\mathcal{B}$ queries its challenge oracle, then
      Query C() to obtain z. Set $\tilde{z} = z \bmod N$. Return $\tilde{z}$
    If $\mathcal{B}$ queries $\tilde{z}$ to its inversion oracle, then
      Query $\tilde{z}$ to I() and get back (x, y). Return y
  Until $\mathcal{B}$ halts with some outputs $(\tilde{y}_1, \ldots, \tilde{y}_m)$
  Let $z_1, \ldots, z_m$ be the challenges returned by C()
  For $i \leftarrow 1$ to m
    $x_i \leftarrow ((z_i/\tilde{y}_i^N - 1)/N) \bmod N$, $y_i \leftarrow \tilde{y}_i$
  Return $((x_1, y_1), \ldots, (x_m, y_m))$.

Let $z_1, \ldots, z_m$ be the challenges returned by C(), and $\tilde{z}_1, \ldots, \tilde{z}_m$ the corresponding challenge points computed by $\mathcal{A}$, namely, the challenges $\mathcal{B}$ gets. If $\mathcal{B}$ is successful, then the number of inversion queries made by $\mathcal{B}$ is strictly fewer than the number of its challenge queries, and for all $i \in \{1, \ldots, m\}$, $\tilde{y}_i = y_i$ is the N-th root of $\tilde{z}_i$ modulo N. This, together with the x_i forms the correct pre-image of z_i under function P. Therefore, $\mathcal{A}$ outputs correct pre-images of all the m challenge points, but makes strictly fewer than m queries to its inversion oracle, as $\mathcal{B}$ does. □

4 The Concurrent Secure Identification Scheme

In this section we present our new identification scheme that is derived from a Σ-protocol (cf. [6]) for proof of knowledge of pre-image under Paillier's function P as per Eq. (1). We show that the assumed hardness of the one-more Paillier inversion problem leads to a proof that the proposed identification scheme is secure against concurrent impersonation attack (see, for example, [3,1] for definition and more discussion). As for efficiency, compared with the known RSA-related

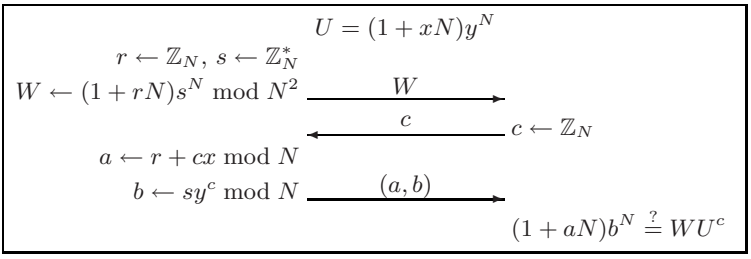


Fig. 1. A concurrent secure identification scheme derived from a Σ-protocol for proof of knowledge of pre-image under function P. Its concurrent security can be proved under the assumed intractability of the one-more Paillier inversion problem. (Operations are understood to be modulo N^2 when not explicitly indicated).

identification schemes, the proposed identification scheme is only slightly inefficient than the well-known GQ scheme [11], but is more efficient than Okamoto's [13].

The scheme (named csID) is depicted in Fig. 1. The system parameter generator proceeds as follows: On input 1^k (where k is a security parameter), it runs $\mathcal{K}(1^k)$ described in Section 2, picks random numbers $x \leftarrow \mathbb{Z}_N$, $y \leftarrow \mathbb{Z}_N^*$, and computes $U = (1 + xN)y^N \bmod N^2$ (recall that $1 + N$ has order N modulo N^2, so it is the case that $(1 + N)^x \equiv 1 + xN \pmod{N^2}$); the public key pk of the prover is (N, U), whereas (N, x, y) is the matching secret key sk.

In a typical execution of the protocol, the prover makes the first move and produces its first message W, which is a random element in $\mathbb{Z}_{N^2}^*$. The verifier chooses a random number $c \in \mathbb{Z}_N$ as a challenge. In response to the challenge c, the prover supplies the verifier with (a, b). The verifier's decision-making leads to acceptance if and only if (a, b) is the pre-image of $WU^c \bmod N^2$ under Paillier's function P.

The following theorem relates the advantage of any concurrent impersonation adversary $\mathcal{I}$ attacking protocol csID to the advantage of a derived one-more Paillier inversion adversary $\mathcal{A}$.

Theorem 2. Given any concurrent impersonation adversary $\mathcal{I} = (\widehat{V}, \widehat{P})$ with time complexity $T()$, there exists a one-more Paillier inversion adversary $\mathcal{A}$ such that their advantage functions are related via the inequality

$$\mathrm{Adv}^{\mathrm{ca}}_{\mathcal{I}}(k) \leq 1/N + \sqrt{\mathrm{Adv}^{\text{om-p}}_{\mathcal{A}}(k)}, \tag{3}$$

where k is any security parameter. Moreover, the time complexity of the one-more Paillier inversion adversary $\mathcal{A}$ is $2T(k) + O(\ell(k)k^3)$, where $\ell(k)$ is the total number of prover instances on which the impersonation adversary $\mathcal{I}$ attacks.

Proof. Without loss of generality we assume that $\widehat{V}$ does not repeat the same oracle query; such repetition can be handled by table look-up.

For security parameter k, let N be the modulus output by the generator $\mathcal{K}$ on input 1^k. Recall that the one-more Paillier inversion adversary $\mathcal{A}$ is given

Adversary $\mathcal{A}^{\mathrm{I}(),\mathrm{C}()}(N, 1^k)$
  Query C() to get U, set $pk \leftarrow (N, 1^k, U)$
  Choose a random tape ρ for $\widehat{V}$, initialize $\widehat{V}$ with (pk, ρ), and set $\ell \leftarrow 0$
  Run $\widehat{V}$, answer $\widehat{V}$'s queries as follows
    If $\widehat{V}$ makes a query of the form (Λ, i)
      Increase ℓ by one, query C() and let W_i be the response.
      Return W_i to $\widehat{V}$
    If $\widehat{V}$ issues a challenge of the form (c, i)
      Set $c_i \leftarrow c$, query I() with $W_i U^{c_i} \bmod N^2$. Let (a_i, b_i)
      be the response. Return (a_i, b_i) to $\widehat{V}$
  Until $\widehat{V}$ outputs some state information σ and stops
  $W \leftarrow \widehat{P}(\Lambda, \sigma)$
  $\widehat{c}_1 \leftarrow \mathbb{Z}_N$, $(\widehat{a}_1, \widehat{b}_1) \leftarrow \widehat{P}(\widehat{c}_1, \sigma)$
  If $(1 + \widehat{a}_1 N)\widehat{b}_1^N \equiv W U^{\widehat{c}_1} \pmod{N^2}$ then $d_1 \leftarrow 1$ else $d_1 \leftarrow 0$
  $\widehat{c}_2 \leftarrow \mathbb{Z}_N$, $(\widehat{a}_2, \widehat{b}_2) \leftarrow \widehat{P}(\widehat{c}_2, \sigma)$
  If $(1 + \widehat{a}_2 N)\widehat{b}_2^N \equiv W U^{\widehat{c}_2} \pmod{N^2}$ then $d_2 \leftarrow 1$ else $d_2 \leftarrow 0$
  If $d_1 = d_2 = 1$, and $\widehat{c}_1 \neq \widehat{c}_2$ then
    Set $\Delta\widehat{a} = (\widehat{a}_1 - \widehat{a}_2) \bmod N$, $\Delta\widehat{b} = (\widehat{b}_1/\widehat{b}_2) \bmod N$, $\Delta\widehat{c} = (\widehat{c}_1 - \widehat{c}_2) \bmod N$
    Compute integers p, t, v such that $p = \gcd(N, \Delta\widehat{c})$, and $p = tN + v(\Delta\widehat{c})$
    If $p = 1$ then
      $x \leftarrow v(\Delta\widehat{a}) \bmod N$, $y \leftarrow (\Delta\widehat{b})^v U^t \bmod N$
    Else
      $q \leftarrow N/p$, $\lambda \leftarrow \mathrm{lcm}(p-1, q-1)$
      $y \leftarrow U^{N^{-1} \bmod \lambda} \bmod N$, $x \leftarrow (U/y^N - 1)/N \bmod N$
    For $i \leftarrow 1$ to ℓ
      $x_i \leftarrow (a_i - c_i x) \bmod N$, $y_i \leftarrow (b_i/y^{c_i}) \bmod N$
    Return $((x, y), (x_1, y_1), \ldots, (x_\ell, y_\ell))$
  Return *failure*

Fig. 2. The algorithm of the one-more inversion adversary $\mathcal{A}$

accesses to an inversion oracle I() and a challenge oracle C(). (When queried with an element z in $\mathbb{Z}_{N^2}^*$, the inversion oracle I() returns the pre-image of z under function P as per Eq. (1); the challenge oracle C() simply returns a random element in $\mathbb{Z}_{N^2}^*$ each time being queried.) The goal of the one-more Paillier inversion adversary $\mathcal{A}$ is to invert all challenges returned by C(), yet making strictly fewer queries to the inversion oracle I() than to the challenge oracle C().

The algorithm of the one-more Paillier inversion adversary $\mathcal{A}$ is described in Fig. 2. The adversary $\mathcal{A}$ first queries its challenge oracle C() to get a random element $U \in \mathbb{Z}_{N^2}^*$, and uses this element along with its own input $(N, 1^k)$ to form a public key $pk = (N, 1^k, U)$ for the impersonation adversary $\mathcal{I}$. Then, to achieve its goal, adversary $\mathcal{A}$ runs $\widehat{V}$, and in every interaction of $\widehat{V}$ and a prover instance, $\mathcal{A}$ emulates the role of the latter. To answer a request of the form (Λ, i) (where Λ denotes the empty string), the one-more adversary $\mathcal{A}$ queries its challenge oracle

C(), and forwards the oracle's answer W_i to $\widehat{V}$. To meet a challenge (c_i, i), $\mathcal{A}$ queries its inversion oracle I() with $W_i U^{c_i} \bmod N^2$, and returns the answer (a_i, b_i) to $\widehat{V}$. Since the answer (a_i, b_i) returned by the inversion oracle I() is the pre-image of $W_i U^{c_i} \bmod N^2$, namely, $(1 + a_i N) b_i^N \bmod N^2 = W_i U^{c_i} \bmod N^2$, the pair (a_i, b_i) is precisely what prover instance i need to provide in order to meet the challenge c_i. Now, from the perspective of $\widehat{V}$, the distribution of the resulting transcript (W_i, c_i, a_i, b_i) is identical to the distribution of transcripts resulting from a real conversation in which prover instance i gets a random tape ρ_i, computes its first message W_i, and meets a challenge c_i by computing the unique pair (a_i, b_i) that is the pre-image of $W_i U^{c_i} \bmod N^2$. It follows that $\mathcal{A}$ perfectly simulates the view of $\widehat{V}$.

Let $\ell(k)$ be the number of prover instances with which $\widehat{V}$ interacts. When $\widehat{V}$ finishes with some piece of state information σ, the one-more inversion adversary $\mathcal{A}$ has made $\ell(k)$ queries to its inversion oracle I() (to get the $\ell(k)$ pairs (a_i, b_i) of pre-images), and has asked for a total number $\ell(k) + 1$ of challenge queries (namely, $U, W_1, \ldots, W_{\ell(k)}$). Recall that $\mathcal{A}$'s goal is to invert all these $\ell(k) + 1$ target points, and at the same time, to make at most $\ell(k)$ queries to its inversion oracle. So at this point, $\mathcal{A}$ cannot make use of its inversion oracle any more, and instead it tries to extract the pre-image (x, y) of U out of $\widehat{P}$, since in that case, the pre-image (x_i, y_i) of all the remaining numbers W_i can be derived by simply computing

$$x_i = (a_i - c_i x) \bmod N,$$
$$y_i = (b_i / y^{c_i}) \bmod N,$$

hence inverting all the $\ell(k) + 1$ challenges successfully.

Now, adversary $\mathcal{A}$ executes the following "extraction" procedure: Run $\widehat{P}$ to get its first message W, issue a random challenge $\widehat{c}_1$, run $\widehat{P}$ to obtain its response, and execute the verifier's checking procedure. Then, select a second random challenge $\widehat{c}_2$, rewind $\widehat{P}$ to get back a second response, and again evaluate the verifier's decision. If both of the decisions come out 1 and the two challenges differ, then proceed to extract the pre-image of U as described below: Let $\Delta\widehat{a} = (\widehat{a}_1 - \widehat{a}_2) \bmod N$, $\Delta\widehat{b} = (\widehat{b}_1 / \widehat{b}_2) \bmod N$ and $\Delta\widehat{c} = (\widehat{c}_1 - \widehat{c}_2) \bmod N$. By the validity of the two challenge/response pairs, we have

$$(1 + (\Delta\widehat{a})N)(\Delta\widehat{b})^N \equiv U^{\Delta\widehat{c}} \pmod{N^2}. \tag{4}$$

Using the extended Euclid's algorithm, compute integers t, v such that $tN + v\Delta\widehat{c} = \gcd(N, \Delta\widehat{c})$. If $\gcd(N, \Delta\widehat{c}) = 1$, then by Eq. (4)

$$U \equiv U^{tN + v\Delta\widehat{c}} \equiv (1 + N)^{v\Delta\widehat{a}} ((\Delta\widehat{b})^v U^t)^N \pmod{N^2}$$

and by the bijectivity of function P we know that

$$x = v(\Delta\widehat{a}) \bmod N,$$
$$y = (\Delta\widehat{b})^v U^t \bmod N$$

is the desired pre-image of U under function P as per Eq. (1). If $\gcd(N, \Delta\widehat{c}) > 1$, then noticing that $0 < |\Delta\widehat{c}| < N$, we can then factor the modulus N into p, q, compute $\lambda = \mathrm{lcm}(p-1, q-1) = (p-1)(q-1)/\gcd(p-1, q-1)$, and compute the modulo inverse $N^{-1} \bmod \lambda$. Now, it is easy to verify that the pair (x, y)

$$y = U^{N^{-1} \bmod \lambda} \bmod N$$
$$x = (U/y^N - 1)/N \bmod N$$

is the desired pre-image of U. In any case, when successfully extracts the pre-image of U, adversary $\mathcal{A}$ can invert all the $\ell(k)+1$ points $U, W_1, \ldots, W_{\ell(k)}$, but queries its inversion oracle only $\ell(k)$ times, provided that both d_1 and d_2 evaluate to 1, and the two challenges $\widehat{c}_1, \widehat{c}_2$ are different. If any of the two decisions evaluates to 0, or the two challenges coincide, then $\mathcal{A}$ fails. It follows that $\mathcal{A}$ succeeds if and only if both $d_1 = d_2 = 1$ and $\widehat{c}_1 \neq \widehat{c}_2$. We next relate the probability of this event to the advantage $\mathrm{Adv}^{\mathrm{ca}}_{\mathcal{I}}(k)$ of the impersonation attacker $\mathcal{I}$.

Observe that the probability distribution of the public key pk formed by $\mathcal{A}$ for $\widehat{V}$ is identical to the distribution of public key in the two-phase defining attack game. Since $\mathcal{A}$ simulates the view of $\widehat{V}$ perfectly, $\widehat{V}$ behaves as it would when playing the two-phase attack game, and provides $\widehat{P}$ with state information distributed identically to what $\widehat{P}$ would receive in their coalition. So, the probability that the first decision d_1 evaluates to 1 is exactly $\mathrm{Adv}^{\mathrm{ca}}_{\mathcal{I}}(k)$.

Write π_1 for $\Pr[d_1 = 1 \mid \sigma, pk]$, the conditional probability that $d_1 = 1$, given that the public key produced by $\mathcal{A}$ is pk, and the state information output by $\widehat{V}$ is σ; here, the probability is over the choice of random challenge $\widehat{c}_1$. Write π_2 for $\Pr[d_1 = d_2 = 1 \text{ and } \widehat{c}_1 \neq \widehat{c}_2 \mid \sigma, pk]$, the conditional probability that $d_1 = d_2 = 1$ and $\widehat{c}_1 \neq \widehat{c}_2$, given that the public key created by $\mathcal{A}$ is pk, and the state information output by $\widehat{V}$ is σ; here, the probability is over the independent and random choices of challenges $\widehat{c}_1$ and $\widehat{c}_2$. It is clear that the expectation E_1 of π_1 is the probability that $\mathcal{I}$ wins, namely, $E_1 = \mathrm{Adv}^{\mathrm{ca}}_{\mathcal{I}}(k)$, and the expectation E_2 of π_2 is the probability that $\mathcal{A}$ succeeds as a one-more inversion adversary, namely $E_2 = \mathrm{Adv}^{\text{om-p}}_{\mathcal{A}}(k)$.

It is obvious that π_1 and π_2 are related via the inequality

$$\pi_2 \geq \pi_1(\pi_1 - 1/N).$$

Notice that for any random variable X, it holds that $\mathrm{E}(X^2) - (\mathrm{E}X)^2 = \mathrm{E}(X - \mathrm{E}X)^2 \geq 0$, where $\mathrm{E}()$ denote the mathematical expectation, we have

$$E_2 \geq \mathrm{E}(\pi_1^2 - \pi_1/N) = \mathrm{E}(\pi_1^2) - E_1/N \geq (E_1)^2 - E_1/N.$$

From that we obtain

$$E_1 - 1/2N \leq \sqrt{E_2 + (1/2N)^2} \leq \sqrt{E_2} + 1/2N,$$

and inequality (3) now follows. This proves the first part of the theorem.

To complete our proof of the theorem, it remains to justify the claim about the time complexity of adversary $\mathcal{A}$. Consider the two-phase attack game defining the

advantage of adversary $\mathcal{A}$. By our conventions for measuring time complexity, the cost of all steps in this game prior to the execution of the next to last If statement is at most $2T(k)$, plus the cost of computing the $\ell(k)$ queries $W_i U^{c_i} \bmod N^2$, plus the cost of twice executing the verifier's checking procedure. The cost of the latter is proportional to $(\ell(k)+2)k^3$. The cost of all the remaining computation is easily seen to be proportional to $(\ell(k)+1)k^3$. Hence, the time complexity of $\mathcal{A}$ is proportional to $(\ell(k)+1)k^3$ plus $2T(k)$, as claimed. □

Using relation (3), we can now provide the following concurrent security property of the proposed identification scheme.

Corollary 1. If the one-more Paillier inversion problem as per Definition 2 is hard, then the identification scheme csID depicted in Fig. 1 is secure against concurrent impersonation attack.

Proof. Let $\mathcal{I}$ be a concurrent impersonation adversary attacking the proposed identification scheme with polynomial time complexity. Then the one-more Paillier inversion adversary $\mathcal{A}$ presented in Fig. 2 also has polynomial time complexity, since both $T()$ and $\ell()$ are polynomially bounded. The assumption that the one-more Paillier inversion problem be hard implies that $\mathrm{Adv}_{\mathcal{A}}^{\text{om-p}}(k)$ is negligible, and obviously $\frac{1}{N}$ is also negligible in k. By relation (3) the advantage function $\mathrm{Adv}_{\mathcal{I}}^{\text{ca}}$ of the concurrent impersonation adversary $\mathcal{I}$ is necessarily negligible. □

5 Conclusion

We formulate a new computational problem, called *one-more Paillier inversion problem*, that extends the Paillier inversion problem underlying the one-wayness of Paillier's trapdoor one-way function in [15]. We investigate the relation between the proposed computational problem and the one-more RSA inversion problem introduced by Bellare *et al.* in [2]; we prove that in regard to intractability the one-more Paillier inversion problem is equivalent to the one-more RSA inversion problem. We also propose a new and efficient identification scheme, and show that under the intractability assumption of the one-more Paillier inversion problem, the proposed identification scheme is secure against concurrent impersonation attack.

References

1. Bellare, M., Namprempre, C., Neven, G.: Security proofs for identity-based identification and signature schemes. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 268–286. Springer, Heidelberg (2004)
2. Bellare, M., Namprempre, C., Pointcheval, D., Semanko, M.: The one-more-RSA inversion problems and the security of Chaum's blind signature scheme. Journal of Cryptology 16(3), 185–215 (2003)

3. Bellare, M., Palacio, A.: GQ and Schnorr identification Schemes: proofs of security against impersonation under active and concurrent attack. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 162–177. Springer, Heidelberg (2002)
4. Catalano, D., Gennaro, R., Howgrave-Graham, N., Nguyen, P.Q.: Paillier's cryptosystem revisited. In: Proceedings of the 8th ACM conference on Computer and Communications Security, pp. 206–214. ACM Press, New York (2001)
5. Cohen, J.D., Fischer, M.: A robust and verifiable cryptographically secure election scheme. In: Proceedings of the 26th Annual IEEE Symposium on Foundations of Computer Science 1985, pp. 372–382 (1985)
6. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 174–187. Springer, Heidelberg (1994)
7. Damgård, I., Jurik, M.: A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In: Kim, K. (ed.) PKC 2001. LNCS, vol. 1992, pp. 119–136. Springer, Heidelberg (2001)
8. Feige, U., Fiat, A., Shamir, A.: Zero knowledge proofs of identity. Journal of Cryptology 1(2), 77–94 (1988)
9. Galbraith, S.D.: Elliptic curve Paillier schemes. Journal of Cryptology 15(2), 129–138 (2002)
10. Goldwasser, S., Micali, S.: Probabilistic encryption. Journal of Computer and System Sciences 28(2), 270–299 (1984)
11. Guillou, L., Quisquater, J.: A practical zero-knowledge protocol fitted to security microprocesors minimizing both transmission and memory. In: Günther, C.G. (ed.) EUROCRYPT 1988. LNCS, vol. 330, pp. 123–128. Springer, Heidelberg (1988)
12. Naccache, D., Stern, J.: A new public key cryptosystem based on higher residues. In: Proceedings of 5th Symposium on Computer and Communications Security, pp. 59–66. ACM Press, New York (1998)
13. Okamoto, T.: Provably secure and practical identification schemes and corresponding signature schemes. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 31–53. Springer, Heidelberg (1993)
14. Okamoto, T., Uchiyama, S.: A new public-key cryptosystem as secure as factoring. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 308–318. Springer, Heidelberg (1998)
15. Paillier, P.: Public-Key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999)

A The RSA Inversion Problem and the One-More RSA Inversion Problem

In this section we briefly review the RSA problem and the one-more RSA inversion problem introduced by Bellare *et al.* in [2].

Definition 3 (RSA Problem). Let k be a security parameter. Let $\mathcal{B}$ be an adversary. Consider the following

Experiment $\mathrm{Exp}^{\mathrm{rsa}}_{\mathcal{B}}(k)$
$(N, e, d) \leftarrow \mathcal{K}(1^k)$
$\tilde{z} \leftarrow \mathbb{Z}^*_N$, $\tilde{y} \leftarrow \mathcal{B}(N, e, 1^k, \tilde{z})$
If $\tilde{z} = \tilde{y}^e \bmod N$ then return 1 else return 0.

The advantage of $\mathcal{B}$ in this experiment is defined by

$$\mathrm{Adv}^{\mathrm{rsa}}_{\mathcal{B}}(k) = \Pr[\mathrm{Exp}^{\mathrm{rsa}}_{\mathcal{B}}(k) = 1].$$

The *RSA inversion problem* is said to be hard, if the function $\mathrm{Adv}^{\mathrm{rsa}}_{\mathcal{B}}(k)$ is negligible for any adversary $\mathcal{B}$ whose time complexity is polynomial in the security parameter k. □

We next briefly review the one-more RSA inversion problem introduced in [2]. We denote by RSAI() the RSA inversion oracle that on input a point $\tilde{z} \in \mathbb{Z}^*_N$ returns the RSA-inverse $\tilde{z}^{N^{-1} \bmod \lambda} \bmod N$. The adversary is also allowed access to an RSA challenge oracle RSAC() that simply returns a random challenge element in $\mathbb{Z}^*_N$ each time being queried. The element is chosen independently and uniformly from $\mathbb{Z}^*_N$ each time being invoked.

Definition 4 (One-More RSA Inversion Problem [2]). Let k be a security parameter. Let $\mathcal{B}$ be an adversary with access to the RSA inversion oracle RSAI() and the RSA challenge oracle RSAC(). Consider the following

Experiment $\mathrm{Exp}^{\text{om-rsa}}_{\mathcal{B}}(k)$
$(N, d) \leftarrow \mathcal{K}(1^k)$, where $d = N^{-1} \bmod \lambda$
$(\tilde{y}_1, \ldots, \tilde{y}_m) \leftarrow \mathcal{B}^{\mathrm{RSAI}(),\mathrm{RSAC}()}(1^k, N)$
where m is the number of queries $\mathcal{B}$ made to RSAC()
Let $\tilde{z}_1, \ldots, \tilde{z}_m$ be the challenges returned by RSAC()
If the following holds then return 1 else return 0
– for all $i \in \{1, \ldots, m\}$, $\tilde{z}_i = \tilde{y}_i^N \bmod N$
– $\mathcal{B}$ made strictly fewer than m queries to RSAI().

The advantage function of $\mathcal{B}$ in the above experiment is defined by

$$\mathrm{Adv}^{\text{om-rsa}}_{\mathcal{B}}(k) = \Pr[\mathrm{Exp}^{\text{om-rsa}}_{\mathcal{B}}(k) = 1].$$

The *one-more RSA inversion problem* is hard if the function $\mathrm{Adv}^{\text{om-rsa}}_{\mathcal{B}}(k)$ is negligible for any adversary whose time complexity is polynomial in the security parameter k. □

An Efficient Signcryption Scheme with Key Privacy

Chung Ki Li[1], Guomin Yang[1], Duncan S. Wong[1,⋆], Xiaotie Deng[1], and Sherman S.M. Chow[2]

[1] Department of Computer Science
City University of Hong Kong
Hong Kong, China
{travisli,csyanggm,duncan,deng}@cs.cityu.edu.hk
[2] Department of Computer Science
Courant Institute of Mathematical Sciences
New York University, NY 10012, USA
schow@cs.nyu.edu

Abstract. In Information Processing Letters 2006, Tan pointed out that the anonymous signcryption scheme proposed by Yang, Wong and Deng (YWD) in ISC 2005 provides neither confidentality nor anonymity. However, no discussion has been made on whether YWD scheme can be made secure. In this paper, we propose a modification of YWD scheme which resolves the security issues of the original scheme without sacrificing its high efficiency and simple design. Indeed, we show that our scheme achieves confidentiality, existential unforgeability and anonymity with more precise reduction bounds. In addition, our scheme further improves the efficiency when compared with YWD, with reduced number of operations for both signcryption and de-signcryption.

Keywords: Signcryption, Key Privacy, Ciphertext Anonymity, Gap Diffie-Hellman.

1 Introduction

Signcryption, introduced by Zheng in 1997 [24], is a cryptographic primitive targeting to provide unforgeability and confidentiality simultaneously as typical signature-then-encryption technique does but with less computational complexity and lower communication cost. Due to these advantages, signcryption is suitable for many applications which require secure and authenticated message delivery using resource limited devices.

There have been many signcryption schemes proposed after Zheng's publication (e.g. [2,11,16,19,23,9,10,15,13,14]). In 2002, Baek *et al.* [3] first formally defined the security notions of signcryption, which are similar to the traditional semantic security against adaptive chosen ciphertext attack (IND-CCA2) [17]

⋆ The author was supported by a grant from CityU (Project No. 7001959).

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 78–93, 2007.

and existential unforgeability against adaptive chosen message attack (EUF-CMA) [12]. The notion of *insider security*[1] was first defined by An *et al.* [1]. The notion allows an adversary to not only access the public keys of both sender and receiver but does also know the sender's private key. For example, a signcryption scheme is said to be 'insider secure' if the adversary cannot compromise the confidentiality of a ciphertext even the adversary knows the sender's private key. Similar notion has later been extended to other security properties for signcryption [9,13,14]. Those properties include unforgeability, anonymity, etc.

In [9], Boyen proposed a new set of security models for signcryption schemes (under the identity-based setting [18]). In particular, a new requirement called *ciphertext anonymity* was proposed. It requires that a ciphertext should appear anonymous to anyone but the actual recipient. It hides the identities of both the sender and the recipient of the ciphertext. This notion can be viewed as an extension of *key privacy* introduced by Bellare et al. [4] for public key encryption. A signcryption scheme with ciphertext anonymity or key privacy, the identities of both sender and recipient are protected from being known from a ciphertext.

In [14], Libert and Quisquater proposed a signcryption scheme with ciphertext anonymity. However, [20] and [22] independently demonstrated that it is neither semantically secure nor anonymous under chosen plaintext attack. In [22], Yang, Wong and Deng also proposed an improvement (hereinafter referred as the YWD scheme) based on [14]. YWD scheme has a special merit on efficiency as it is computationally efficient and supports parallel processing which may help improving the performance further. However, a recent result by Tan [21] showed that the YWD scheme is not semantically secure and does not satisfy ciphertext anonymity, under insider's chosen-ciphertext attack. However, no improved scheme was proposed by Tan.

Our Contribution. It is still not known if the YWD scheme can be improved to a secure one, while maintaining the advantages of the original scheme being highly efficient and simple. In this paper, we propose a modification of the YWD scheme. The modified scheme not only solves the security issues of the original scheme, but also maintains its efficiency. In particular, we show that our scheme achieves confidentiality, existential unforgeability and anonymity with more precise reduction bounds. In addition, our scheme further improves the efficiency with reduced number of operations for both signcryption and de-signcryption.

Organization. We give the definition and security models of a signcryption scheme with ciphertext anonymity (or key privacy) in Sec. 2. It is then followed by the review and discussion of YWD signcryption scheme and Tan's attacks in Sec. 3. This leads us to the description of our method for solving the security issues of the YWD scheme. Our construction and its security analysis are given in Sec. 4. We conclude the paper in Sec. 5.

[1] The original paper of An, *et al.* [1] only presents the insider attack against the integrity of a signcryption. The idea has later been extended to confidentiality and other security properties [9,14].

2 The Definition and Security Models of Signcryption with Key Privacy

A signcryption scheme is a quadruple of probabilistic polynomial time (PPT) algorithms (**Keygen**, **Signcrypt**, **De-signcrypt**, **Verify**).

$(sk, pk) \leftarrow \mathbf{Keygen}(1^k)$ is the key generation algorithm which takes a security parameter $k \in \mathbb{N}$ and generates a private/public key pair (sk, pk).

$\sigma \leftarrow \mathbf{Signcrypt}(1^k, m, sk_U, pk_R)$ takes k, a message m, a private key sk_U and a public key pk_R, outputs a ciphertext σ. m is drawn from a message space M which is defined as $\{0,1\}^n$ where n is some polynomial in k.

$(m, s, pk_U)/\texttt{reject} \leftarrow \mathbf{De\text{-}signcrypt}(1^k, \sigma, sk_R)$ takes k, σ and a private key sk_R, outputs either a triple (m, s, pk_U) where $m \in M$, s is a signature and pk_U is a public key, or `reject` which indicates the failure of de-signcryption.

$\texttt{true}/\texttt{false} \leftarrow \mathbf{Verify}(1^k, m, s, pk_U)$ takes k, $m \in M$, a signature s and a public key pk_U, outputs `true` for a valid signature or `false` for an invalid signature.

For simplicity, we omit the notation of 1^k from the inputs of **Signcrypt**, **De-signcrypt** and **Verify** in the rest of this paper. Note that the specification above requires the corresponding signcryption scheme to support the "unwrapping" option introduced in [15]. The "unwrapping" option allows the receiver of a ciphertext to release the message and derive the embedded sender's signature from the ciphertext for public verification. Early schemes such as [24] do not support the "unwrapping" option and therefore not satisfy this definition.

Definition 1 (Completeness). *For any $m \in M$, $(sk_U, pk_U) \leftarrow \mathbf{Keygen}(1^k)$ and $(sk_R, pk_R) \leftarrow \mathbf{Keygen}(1^k)$ such that $sk_U \neq sk_R$, we have*

$$(m, s, pk_U) \leftarrow \boldsymbol{De\text{-}signcrypt}(\boldsymbol{Signcrypt}(m, sk_U, pk_R), sk_R)$$

and $\texttt{true} \leftarrow \mathbf{Verify}(m, s, pk_U)$.

Informally, we consider a secure signcryption scheme with key privacy to be semantically secure against adaptive chosen ciphertext attack, existentially unforgeable against chosen message attack, and anonymous in the sense that a ciphertext should contain no information in the clear that identifies the author or the recipient of the message and yet be decipherable by the intended recipient without that information. We capture these notions in the following definitions. They are similar to those defined by Libert and Quisquater [14].

Definition 2 (Confidentiality). *A signcryption scheme is semantically secure against chosen ciphertext insider attack (SC-IND-CCA) if no PPT adversary has a non-negligible advantage in the following game:*

1. *The challenger runs **Keygen** to generate a key pair (sk_U, pk_U). sk_U is kept secret while pk_U is given to adversary $\mathcal{A}$.*

2. *In the first stage, $\mathcal{A}$ makes a number of queries to the following oracles:*
   (a) *Signcryption oracle: $\mathcal{A}$ prepares a message $m \in M$ and a public key pk_R, and queries the signcryption oracle (simulated by the challenger) for the result of **Signcrypt**(m, sk_U, pk_R). The result is returned if $pk_R \neq pk_U$ and pk_R is valid in the sense that pk_R is in the range of **Keygen** with respect to the security parameter. Otherwise, a symbol '$\perp$' is returned for rejection.*
   (b) *De-signcryption oracle: $\mathcal{A}$ produces a ciphertext σ and queries for the result of **De-signcrypt**(σ, sk_U). The result is made of a message, a signature and the sender's public key if the de-signcryption is successful and the signature is valid under the recovered sender's public key. Otherwise, a symbol '$\perp$' is returned for rejection.*

   These queries can be asked adaptively: each query may depend on the answers of previous ones.
3. *$\mathcal{A}$ produces two plaintexts $m_0, m_1 \in M$ of equal length and a valid private key sk_S such that sk_S is in the range of **Keygen** with respect to the security parameter. The challenger flips a coin $\check{b} \stackrel{R}{\leftarrow} \{0,1\}$ and computes a signcryption $\sigma^* =$ **Signcrypt**$(m_{\check{b}}, sk_S, pk_U)$ of $m_{\check{b}}$ with the sender's private key sk_S under the receiver's public key pk_U. σ^* is sent to $\mathcal{A}$ as a challenge ciphertext.*
4. *$\mathcal{A}$ makes a number of new queries as in the first stage with the restriction that it cannot query the de-signcryption oracle with σ^*.*
5. *At the end of the game, $\mathcal{A}$ outputs a bit b' and wins if $b' = \check{b}$.*

$\mathcal{A}$'s advantage is defined as $Adv^{ind\text{–}cca}(\mathcal{A}) = \Pr[b' = \check{b}] - \frac{1}{2}$ and the probability that $b' = \check{b}$ is called the probability that $\mathcal{A}$ wins the game.

The definition above captures the advantage of an active adversary over an eavesdropper. That is, the adversary knows and has the full control of the signing key. This also gives us insider-security for confidentiality [1,9,14].

Definition 3 (Unforgeability). *A signcryption scheme is existentially unforgeable against chosen-message insider attack (SC-EUF-CMA) if no PPT forger has a non-negligible advantage in the following game:*

1. *The challenger runs **Keygen** to generate a key pair (sk_U, pk_U). sk_U is kept secret while pk_U is given to forger $\mathcal{F}$.*
2. *The forger $\mathcal{F}$ adaptively makes a number of queries to the signcryption oracle and the de-signcryption oracle as in the confidentiality game.*
3. *$\mathcal{F}$ produces a ciphertext σ and a valid key pair (sk_R, pk_R) in the sense that the key pair is in the range of **Keygen** and wins the game if*
   (a) ***De-signcrypt**(σ, sk_R) returns a tuple (m, s, pk_U) such that* `true` $\leftarrow$ ***Verify**(m, s, pk_U), and*
   (b) *σ is not the output of the signcryption oracle.*

We allow the forger to have the full control of the de-signcryption key pair (sk_R, pk_R). This also captures the notion of insider-security for unforgeability.

Definition 4 (Ciphertext Anonymity). *A signcryption scheme is ciphertext anonymous against chosen-ciphertext insider attack (SC-ANON-CCA) if no PPT distinguisher has a non-negligible advantage in the following game:*

1. *The challenger generates two distinct public key pairs* $(sk_{R,0}, pk_{R,0})$ *and* $(sk_{R,1}, pk_{R,1})$ *using* ***Keygen****, and gives* $pk_{R,0}$ *and* $pk_{R,1}$ *to the distinguisher* $\mathcal{D}$.
2. *In the first stage,* $\mathcal{D}$ *adaptively makes a number of queries in the form of* ***Signcrypt***$(m, sk_{R,c}, pk_R)$ *or* ***De-signcrypt***$(\sigma, sk_{R,c})$*, for* $c = 0$ *or* $c = 1$*.* pk_R *is some arbitrary but valid recipient key such that* $pk_R \neq pk_{R,c}$*.*
3. *After completing the first stage,* $\mathcal{D}$ *outputs two valid and distinct private keys* $sk_{S,0}$ *and* $sk_{S,1}$*, and a plaintext* $m \in M$*.*
4. *The challenger then flips two coins* $b, b' \overset{R}{\leftarrow} \{0,1\}$ *and computes a challenge ciphertext* $\sigma =$ ***Signcrypt***$(m, sk_{S,b}, pk_{R,b'})$ *and sends it to* $\mathcal{D}$*.*
5. $\mathcal{D}$ *adaptively makes a number of new queries as above with the restriction that it is not allowed to ask the de-signcryption oracle of the challenge ciphertext* σ*.*
6. *At the end of the game,* $\mathcal{D}$ *outputs bits* d, d' *and wins the game if* $(d, d') = (b, b')$*.*

$\mathcal{D}$*'s advantage is defined as* $Adv^{anon\text{–}cca}(\mathcal{D}) = \Pr[(d, d') = (b, b')] - \frac{1}{4}$.

The ciphertext anonymity definition above follows that of Libert and Quisquater in [14, Def. 4], which is considered to be an extension of the "key privacy" notion of public key encryption [4]. We only consider this definition for key privacy in this paper rather than also considering an additional one called key invisibility [14, Def. 5]. We believe that the definition above is more intuitive. With only a few differences, one can also consider it as a non-identity based version of Boyen's definition [9] of ciphertext anonymity in the identity-based setting.

3 Preliminaries

Bilinear Pairings. Let k be a system-wide security parameter. Let q be a k-bit prime. Let $\mathbb{G}_1$ be an additive cyclic group of order q and $\mathbb{G}_2$ be a multiplicative cyclic group of the same order. Let P be a generator of $\mathbb{G}_1$. A bilinear map is defined as $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ with the following properties:

1. *Bilinear*: For all $U, V \in \mathbb{G}_1$, and $a, b \in \mathbb{Z}$, we have $e(aU, bV) = e(U, V)^{ab}$.
2. *Non-degenerate*: $e(P, P) \neq 1$.
3. *Computable*: there is an efficient algorithm to compute $e(U, V)$ for any $U, V \in \mathbb{G}_1$.

Modified pairings [7] obtained from the Weil or the Tate pairing provide admissible maps of this kind.

The Gap Diffie-Hellman Problem. The Decisional Diffie-Hellman problem (DDH) [6] in $\mathbb{G}_1$ is to distinguish between the distributions of $\langle P, aP, bP, abP \rangle$

and $\langle P, aP, bP, cP\rangle$ where a, b, c are random in $\mathbb{Z}_q$. The Computational Diffie-Hellman problem (CDH) in $\mathbb{G}_1$ is to compute abP from $\langle P, aP, bP\rangle$ where a, b are random in $\mathbb{Z}_q$. The Gap Diffie-Hellman problem (GDH) is to solve a given random instance $\langle P, aP, bP\rangle$ of the CDH problem with the help of a DDH oracle. The DDH oracle can be implemented through a bilinear map since it suffices to check if the equation $e(P, cP) = e(aP, bP)$ holds for determining if $cP = abP$.

3.1 YWD Signcryption Scheme [22]

Suppose each element in $\mathbb{G}_1$ can be represented distinctly using l bits. Let $H_1 : \{0,1\}^{n+2l} \to \mathbb{G}_1$, $H_2 : \mathbb{G}_1^3 \to \{0,1\}^l$ and $H_3 : \mathbb{G}_1^3 \to \{0,1\}^{n+l}$ be cryptographic hash functions where n denotes the length of a plaintext in binary representation and is in some polynomial of k. The scheme is described as follows:

Keygen: A private key is generated by picking a random $x_u \leftarrow \mathbb{Z}_q$ and the corresponding public key is computed as $Y_u = x_u P$. In the following, the sender and the receiver are denoted by $u = S$ and $u = R$, and their public key pairs are denoted by (x_S, Y_S) and (x_R, Y_R), respectively.

Signcrypt: To signcrypt a message $m \in \{0,1\}^n$ for receiver R, sender S carries out the following steps:

1. Pick a random $r \leftarrow \mathbb{Z}_q$ and compute $U = rP$.
2. Compute $V = x_S H_1(m, U, Y_R)$
3. Compute $W = V \oplus H_2(U, Y_R, rY_R)$ and $Z = (m \| Y_S) \oplus H_3(U, Y_R, rY_R)$.

The ciphertext is $\sigma = (U, W, Z)$.

De-signcrypt: When a ciphertext $\sigma = (U, W, Z)$ is received, receiver R performs the following steps:

1. Compute $V = W \oplus H_2(U, Y_R, x_R U)$
2. Compute $(m \| Y_S) = Z \oplus H_3(U, Y_R, x_R U)$.
3. If $Y_S \notin \mathbb{G}_1$, outputs `reject`. Otherwise, compute $H = H_1(m, U, Y_R)$ and check if $e(Y_S, H) = e(P, V)$.
4. If the equation holds, output $\langle m, (U, Y_R, V), Y_S\rangle$; otherwise, output `reject`.

Verify: For a message-signature pair $(m, (U, Y_R, V))$ and a signer's public key Y_S, the algorithm checks if $e(Y_S, H_1(m, U, Y_R)) = e(P, V)$. If the condition holds, it outputs `true`. Otherwise, it outputs `false`.

3.2 Tan's Attacks Against the YWD Scheme

Adaptive chosen ciphertext attack. Based on the SC-IND-CCA game stated in Definition 2, Tan's attack [21] is carried out as follows.

Given the receiver's public key Y_R, the adversary $\mathcal{A}$ first chooses a sender's private key x_S and two same length messages m_0 and m_1, and sends them to the challenger. Then the challenger randomly picks $b \in \{0,1\}$ and computes the challenge ciphertext of m_b as $\mathcal{C}^* = (U^*, W^*, Z^*)$. After receiving $\mathcal{C}^*$, $\mathcal{A}$ makes a guess of b to be 0 and computes a new ciphertext with a random message

$\bar{m}$ which has the same length as m_0 and a random $\bar{x}_S \leftarrow \mathbb{Z}_q$. $\mathcal{A}$ computes the ciphertext $\bar{\mathcal{C}}$ as follows:

- $\bar{Y_S} = \bar{x}_S P$
- $V^* = x_S H_1(m_0, U^*, Y_R)$
- $\bar{V} = \bar{x}_S H_1(\bar{m}, U^*, Y_R)$
- $\bar{W} = (\bar{V} \oplus V^*) \oplus W^*$
- $\bar{Z} = ((m_0 \oplus \bar{m}) \| (\bar{Y_S} \oplus Y_S)) \oplus Z^*$

At last, $\mathcal{A}$ sends $\bar{C} = (U^*, \bar{W}, \bar{Z})$ to the de-signcryption oracle which computes the following:

- $\hat{m} \| \hat{Y_S} = \bar{Z} \oplus H_3(U^*, Y_R, x_R U^*)$, here $\hat{Y_S} = \bar{Y_S}$
- $\hat{V} = \bar{W} \oplus H_2(U^*, Y_R, x_R U^*)$
- $H = H_1(\hat{m}, U^*, Y_R)$

If $e(\hat{Y_S}, H) = e(P, \hat{V})$, the de-signcryption oracle returns the message $\hat{m}$, otherwise returns '$\perp$' as rejection. If the message $\hat{m}$ is equal to $\bar{m}$, $\mathcal{A}$ knows that m_0 is used for constructing the challenge ciphertext. On the other hand, if the message $\hat{m}$ is not equal to $\bar{m}$, m_1 is used for constructing the challenge ciphertext.

Attack Against Ciphertext Anonymity. Based on the SC-ANON-CCA game stated in Definition 4, Tan's attack [21] is carried out as follows.

Given two receivers' public keys $Y_{R,0}$ and $Y_{R,1}$, the distinguisher $\mathcal{D}$ generates two senders' private keys $x_{S,0}$ and $x_{S,1}$ and a message m^*. Then sends them to the challenger. The challenger picks two random number $b, b' \in \{0, 1\}$ and computes the challenge ciphertext $\mathcal{C}^* = (U^*, W^*, Z^*)$ with sender's private key $x_{S,b}$ and receiver's public key $Y_{R,b'}$. After receiving $\mathcal{C}^*$, $\mathcal{D}$ carries out the similar computation as that of the chosen ciphertext attack and produces four ciphertexts $\bar{\mathcal{C}}_{i,j} = (U^*, \bar{W}_{i,j}, \bar{Z}_i)$ with a random message $\bar{m}$ and a random $\bar{x}_S \in Z_q$, where $i, j \in \{0, 1\}$ and $\bar{m}$'s length is the same as m^*. If $\mathcal{D}$ submits those $\bar{\mathcal{C}}_{i,j}$ to de-signcryption oracle, then the de-signcryption oracle will return either the message $\hat{m}_{i,j}$ or '$\perp$' for rejection. If messages are returned, one of them must be equal to $\bar{m}$. Therefore, $\mathcal{D}$ can make the correct guess of (b, b').

Discussions. The YWD scheme is not secure against Tan's attacks because the component V can easily be reconstructed with the sender's secret. Since the inputs of H_1 do not involve any secret value, the adversary under the notion of "insider security" can easily make use of the malleability property of W and Z to construct a related but different ciphertext. In the next section, we proposed an efficient solution of this problem.

4 Simple and Efficient Signcryption with Key Privacy

Our scheme is based on Boyen-Lynn-Shacham's (BLS) short signature [8] and ElGamal encryption. The recipient can decrypt without knowing who the sender is. Both the sender's public key and the related signature are recovered from the ciphertext such that the recipient can verify its authenticity.

4.1 Our Construction

Suppose each element in $\mathbb{G}_1$ can distinctly be represented using an l-bit long binary string. Let $H_1 : \{0,1\}^n \times \mathbb{G}_1^3 \to \mathbb{G}_1$ and $H_2 : \mathbb{G}_1^3 \to \{0,1\}^{n+2l}$ be hash functions where n denotes the length of a plaintext and is in some polynomial of k. For security analysis, all hash functions are viewed as random oracles [5]. Our proposed scheme is described as follows.

Keygen: The same as that of the YWD scheme.

Signcrypt: To signcrypt a message $m \in \{0,1\}^n$ for receiver R, sender S carries out the following steps:

1. Pick a random $r \leftarrow \mathbb{Z}_q$ and compute $U = rP$.
2. Compute $V = x_S H_1(m, U, Y_R, rY_R)$ and $Z = (m \| Y_S \| V) \oplus H_2(U, Y_R, rY_R)$.

The ciphertext is $\sigma = (U, Z)$.

De-signcrypt: When a ciphertext $\sigma = (U, Z)$ is received, receiver R performs the following steps:

1. Compute $D = x_R U$.
2. Compute $(m \| Y_S \| V) = Z \oplus H_2(U, Y_R, D)$.
3. If $Y_S \notin \mathbb{G}_1$, output `reject`. Otherwise, compute $H = H_1(m, U, Y_R, D)$ and check if (P, Y_S, H, V) is a Diffie-Hellman tuple by $e(Y_S, H) \stackrel{?}{=} e(P, V)$.
4. If the check passes, output $\langle m, (U, Y_R, D, V), Y_S \rangle$; otherwise, output `reject`.

Verify: For a message-signature pair $(m, (U, Y_R, D, V))$ and a signer's public key Y_S, the algorithm checks if $e(Y_S, H_1(m, U, Y_R, D)) = e(P, V)$. If the condition holds, it outputs `true`. Otherwise, it outputs `false`.

Discussions. In our scheme, the signature V is signed on the value D which cannot be computed without the recipient's private key. Even with the knowledge of the sender's private key, the adversary cannot reconstruct V if D remains unknown. On the other hand, the signcryption oracle does not offer much help since no adaptive choice is available to make a signature of D. These give us semantic security under adaptive insider's chosen-ciphertext attack. The adversary needs to know the value of D to come up with a valid ciphertext, which renders the decryption oracle "useless" and explains why we get chosen-ciphertext security from chosen-plaintext secure ElGamal encryption.

4.2 Security Analysis

Theorem 1. *Let k be a security parameter. Under the random oracle model, if there exists a PPT algorithm which can break the SC-IND-CCA security of the proposed signcryption scheme with advantage at least $\rho(k)$, then there exists a PPT algorithm which can solve the Gap Diffie-Hellman Problem with probability at least $2(1 - p2^{-poly(k)})\rho(k)$ where $poly(\cdot)$ is some polynomial and p is the maximum number of de-signcryption queries made in the model of SC-IND-CCA.*

Proof. For contradiction, we assume that there exists an adversary $\mathcal{A}$ who wins the game given in Definition 2 with non-negligible advantage. In the following, we construct an algorithm $\mathcal{B}$ to solve the CDH problem in $\mathbb{G}_1$ with the help of a DDH solver due to the bilinear pairing.

Suppose $\mathcal{B}$ is given a random instance of the CDH problem $(P, aP, bP) \in \mathbb{G}_1^3$, $\mathcal{B}$ runs $\mathcal{A}$ as a subroutine to find the solution abP. $\mathcal{B}$ sets up a simulated environment of SC-IND-CCA model for $\mathcal{A}$ as follows:

$\mathcal{B}$ gives bP to $\mathcal{A}$ as the challenging public key Y_u.

$\mathcal{B}$ maintains two lists L1 and L2 for simulating hash oracles H_1 and H_2, respectively. When a hash query $H_1(m, P_1, P_2, P_3)$ is received, where $m \in \{0,1\}^n$ and $P_1, P_2, P_3 \in \mathbb{G}_1$, $\mathcal{B}$ checks if the query tuple (m, P_1, P_2, P_3) is already in L1. If it exists, the existing result in L1 is returned. If it does not exist but $e(P_1, P_2) = e(P, P_3)$ and $(P_1, P_2, \top)$ is in L1, where '$\top$' is a special symbol, then $\mathcal{B}$ replaces '$\top$' in the entry with P_3 and returns the existing result in the entry. For all other cases, $\mathcal{B}$ randomly chooses $t \leftarrow \mathbb{Z}_q$ and returns tP to $\mathcal{A}$. The query tuple and return value are then saved in L1. For enabling the retrieval of t possibly at some later time of the simulation, the value of t is also stored in L1 along the entry. Hash queries to H_2 are handled similarly.

For a signcryption query on a message m with a receiver's public key Y_R, $\mathcal{B}$ checks if $Y_R \in \mathbb{G}_1$. If it is incorrect or $Y_R = Y_u$, $\mathcal{B}$ returns a symbol '$\perp$' for rejection. Otherwise, $\mathcal{B}$ randomly picks $r \leftarrow \mathbb{Z}_q$, computes $U = rP$ and simulates $H_1(m, U, Y_R, rY_R)$ described as above. Suppose $H_1(m, U, Y_R, rY_R)$ is set to $t'P$. $\mathcal{B}$ then simulates $H_2(U, Y_R, rY_R)$, and computes the ciphertext $\sigma = (U, Z)$ where $Z = (m \| Y_u \| t'(Y_u)) \oplus H_2(U, Y_R, rY_R)$.

De-signcryption query on $\sigma = (U, Z)$ is answered as follows.

1. $\mathcal{B}$ looks for a tuple of the form (U, Y_u, λ) in L2 such that $e(P, \lambda) = e(U, Y_u)$ or $\lambda = \top$.
   - If the tuple (U, Y_u, λ) is not in L2, $\mathcal{B}$ adds a new entry into L2 by storing $(U, Y_u, \top)$ as the query tuple and a value randomly drawn from the range of H_2 as the oracle return. The special symbol '$\top$' is used as a marker for denoting that the real value should be the solution of the CDH problem instance (U, Y_u). This step ensures that the value of $H_2(U, Y_u, \lambda)$ is *fixed* before σ is de-signcrypted.
   - If the tuple (U, Y_u, λ) is already in L2, the existing result will be used as the value of $H_2(U, Y_u, \lambda)$.
2. $\mathcal{B}$ computes $m \| Y_S \| V = Z \oplus H_2(U, Y_u, \lambda)$ and looks for a tuple of the form (m, U, Y_u, λ) in L1 such that $e(P, \lambda) = e(U, Y_u)$ or $\lambda = \top$.
   - If the tuple (m, U, Y_u, λ) is not in L1, $\mathcal{B}$ adds a new entry into L1 by storing (m, U, Y_u, λ) as the query tuple and setting $r'P$ as the oracle return, where $r' \leftarrow \mathbb{Z}_q$. Note that λ can be a value such that $e(P, \lambda) = e(U, Y_u)$. This is because, the value may have been obtained from L2 above.
   - If the tuple (m, U, Y_u, λ) is in L1 and $e(P, \lambda) = e(U, Y_u)$, then besides using the existing result of L1 as the value for $H_1(m, U, Y_u, \lambda)$, the value of λ should also be used to update the corresponding entry in L2.

- If the tuple $(m, U, Y_u, \top)$ is in L1, then the existing result in L1 will be used as the value for $H_1(m, U, Y_u, \top)$. Also if the value of λ can be obtained from L2, then the entry $(m, U, Y_u, \top)$ in L1 should also be updated to (m, U, Y_u, λ).

3. $\mathcal{B}$ checks if $e(P, V) = e(H_1(m, U, Y_u, \lambda), Y_S)$ holds.
   - If the equation holds and $e(P, \lambda) = e(U, Y_u)$, $(m, (U, Y_u, \lambda, V), Y_S)$ are returned as the message-signature pair and the sender's public key.
   - If the equation holds but $\lambda = \top$, then $\mathcal{B}$ halts with failure.
   - Otherwise, the symbol '$\perp$' is returned for rejection.

After completing the first stage of the game, $\mathcal{A}$ chooses two n-bit plaintexts m_0 and m_1 together with a sender's private key x_S^*, and requests $\mathcal{B}$ for a challenge ciphertext built under the receiver's challenging public key Y_u. Suppose the associated signer's public key is $Y_S^* = x_S^* P$.

$\mathcal{B}$ sets the challenge ciphertext to $\sigma^* = (U^*, Z^*)$ where $U^* = aP$ and Z^* is randomly drawn from $\{0,1\}^n \times \mathbb{G}_1^2$. $\mathcal{B}$ also randomly picks $\check{b} \leftarrow 1/0$, and updates L1 by adding in $(m_{\check{b}}, aP, Y_u, \top)$, randomly picking $t^* \leftarrow \mathbb{Z}_q$, and setting the result of $H_1(m_{\check{b}}, aP, Y_u, \top)$ to $t^* P$. Note that this entry will only be added in L1 if it is not in L1 yet. Similarly, L2 will also be updated with $(aP, Y_u, \top)$ and the value of $H_2(aP, Y_u, \top)$ is set to $Z^* \oplus (m_{\check{b}} \| Y_S^* \| t^* Y_S^*)$. Let $V^* = t^* Y_S^*$.

After that, $\mathcal{B}$ answers $\mathcal{A}$'s queries as in the first stage. If $\mathcal{A}$ queries H_1 or H_2 with (aP, Y_u, λ) such that $e(aP, Y_u) = e(\lambda, P)$, then $\mathcal{B}$ outputs λ and halts. If $\mathcal{A}$ halts without making this query, $\mathcal{B}$ outputs a random point in $\mathbb{G}_1$ and halts.

Analysis. The running time of $\mathcal{B}$ is in polynomial of $\mathcal{A}$'s running time. To see that the simulated game is computationally indistinguishable from a real game, we note that H_1, H_2 and signcryption oracle are simulated perfectly. For de-signcryption queries, except the following case, are carried out perfectly too.

The exceptional case is at step 3 of the de-signcryption oracle simulation above when $(U, Y_u, \top)$ is in L2 and $(m, U, Y_u, \top)$ is in L1, while $e(P, V) = e(H_1(m, U, Y_u, \top), Y_S)$ where $m \| Y_S \| V = Z \oplus H_2(U, Y_u, \top)$ (i.e. the equation at step 3 holds). This case implies that $\mathcal{A}$ has never queried H_1 on (m, U, Y_u, λ) nor H_2 on (U, Y_u, λ) such that $e(P, \lambda) = e(U, Y_u)$, while $Z_3 = W_3 \oplus V$ where $Z = (Z_1, Z_2, Z_3) \in \{0,1\}^n \times \mathbb{G}_1^2$ and $(W_1, W_2, W_3) = H_2(U, Y_u, \top) \in \{0,1\}^n \times \mathbb{G}_1^2$. Also note that the de-signcryption oracle would never leak any of $H_1(m, U, Y_u, \top)$ and W_3 to $\mathcal{A}$ at step 3 of the de-signcryption oracle simulation above, since $\mathcal{B}$ either fails or rejects. Therefore, we have

$$\Pr[Z_3 = W_3 \oplus V] \leq p/|\mathbb{G}_1| = p 2^{-poly(k)}$$

where p is the maximum number of de-signcryption queries made by $\mathcal{A}$ and $poly(\cdot)$ is some polynomial function. Hence with probability at least $1 - p 2^{-poly(k)}$, $\mathcal{B}$ does not fail and carries out the simulation perfectly.

Let $\mathbf{E}$ be the event that (aP, Y_u, aY_u) is queried on H_1 or H_2. $\bar{\mathbf{E}}$ denotes the event that (aP, Y_u, aY_u) is not queried on H_1 or H_2. Note that $\mathcal{B}$ solves the CDH problem instance in event $\mathbf{E}$.

We claim that for event $\bar{\mathbf{E}}$, $\mathcal{A}$ does not have any advantage in winning the game over random guessing. Let $V_{\tilde{b}} = x_S^* H_1(m_{\tilde{b}}, aP, Y_u, aY_u)$. Then $\sigma^* = (aP, Z^*)$ is the signcryption of $m_{\tilde{b}}$ if we have

$$(m_{\tilde{b}} \| Y_S^* \| V_{\tilde{b}}) = Z^* \oplus H_2(aP, Y_u, aY_u)$$

If we focus on the output portion of $H_2(aP, Y_u, aY_u)$ that is corresponding to $V_{\tilde{b}}$, we can see that, in event $\bar{\mathbf{E}}$, although $(aP, Y_u, \top)$ is in L2 and $(m_{\tilde{b}}, aP, Y_u, \top)$ is in L1, as argued above, $\mathcal{B}$ does not leak any information of these values to $\mathcal{A}$. Also due to the randomness assumption of H_1 and H_2, $\mathcal{A}$ does not have any advantage in determining the oracle return of $H_1(m_{\tilde{b}}, aP, Y_u, aY_u)$ and the output portion of $H_2(aP, Y_u, aY_u)$ corresponding to $V_{\tilde{b}}$ in the equation above. Hence, $\Pr[\mathcal{A} \text{ wins the game } |\bar{\mathbf{E}}] = \frac{1}{2}$. From the assumption,

$$\Pr[\mathcal{A} \text{ wins the game}] = \frac{1}{2} + \rho(k) \leq \Pr[\mathbf{E}] + \frac{1}{2}(1 - \Pr[\mathbf{E}])$$

where ρ is $\mathcal{A}$'s non-negligible advantage in winning the game defined in Definition 2 and k is the system-wide security parameter. Therefore, $\Pr[\mathbf{E}] \geq 2\rho(k)$. We have $\Pr[\mathbf{E} \wedge \mathcal{B} \text{ does not fail}] \geq 2(1 - p2^{-poly(k)})\rho(k)$. □

Theorem 2. *Let k be the security parameter. Under the random oracle model, if there exists a PPT algorithm which can break the SC-EUF-CMA security of the proposed scheme with advantage at least $\rho(k)$, then there exists a PPT algorithm which can solve the Gap Diffie-Hellman Problem with probability at least $(1 - p2^{-poly(k)})\rho(k)$ where $poly(\cdot)$ is some polynomial and p is the maximum number of de-signcryption queries made in the model of SC-EUF-CMA.*

Proof. We prove it also by contradiction, we assume that there exists a forger $\mathcal{F}$ who can win the game stated in Definition 3. In the following, we construct an algorithm $\mathcal{B}$ that can solve the CDH problem in $\mathbb{G}_1$.

Suppose $\mathcal{B}$ is given a random instance of the CDH problem $(P, aP, bP) \in \mathbb{G}_1^3$, $\mathcal{B}$ runs $\mathcal{F}$ as a subroutine to find abP. $\mathcal{B}$ sets up a simulated environment of SC-EUF-CMA as follows:

$\mathcal{B}$ gives bP to $\mathcal{F}$ as the challenge public key Y_u.

$\mathcal{B}$ maintains two lists L1 and L2 to simulate the hash oracles H_1 and H_2, respectively. In each entry of the lists, it keeps the query and the corresponding return of the oracle. Hash oracles H_2 is simulated as in the proof of Theorem 1.

When a hash query $H_1(m, P_1, P_2, P_3)$ is asked by $\mathcal{F}$, $\mathcal{B}$ first checks if the query tuple (m, P_1, P_2, P_3) is already in L1. If it exists, the existing result in L1 is returned. If it does not exist but $e(P_1, P_2) = e(P, P_3)$ and $(P_1, P_2, \top)$ is in L1, where '$\top$' is a special symbol, then $\mathcal{B}$ replaces '$\top$' in the entry with P_3 and returns the existing result in the entry. For all other cases, $\mathcal{B}$ randomly chooses $t \leftarrow \mathbb{Z}_q$ and returns $t(aP)$ to $\mathcal{F}$. The query tuple and return value are then saved in L1. For enabling the retrieval of t possibly in some later time of the simulation, the value is also saved in L1.

For a signcryption query on a message m with a receiver's public key Y_R both chosen by $\mathcal{F}$, $\mathcal{B}$ first checks if $Y_R \in \mathbb{G}_1$. If it is incorrect or $Y_R = Y_u$, $\mathcal{B}$

returns the symbol '$\perp$' for rejection. Otherwise, $\mathcal{B}$ picks a random $r \leftarrow \mathbb{Z}_q$ and computes $U = rP$. Then $\mathcal{B}$ selects a random $t' \leftarrow \mathbb{Z}_q$ and returns $t'P$ as the value of $H_1(m, U, Y_R, rY_R)$. The query tuple, oracle return and the value of t' are then saved in L1. After obtaining t' such that $t'P = H_1(m, U, Y_R, rY_R)$, $\mathcal{B}$ computes $V = t'(Y_u)$ which is equal to $bH_1(m, U, Y_R, rY_R)$. $\mathcal{B}$ then simulates H_2 as in the proof of Theorem 1 for obtaining $H_2(U, Y_R, rY_R)$, and computes the result ciphertext $\sigma = (U, Z)$ where $Z = (m \| Y_u \| t'(Y_u)) \oplus H_2(U, Y_R, rY_R)$.

When $\mathcal{F}$ performs a De-signcrypt(σ, sk_u) query, where $\sigma = (U, Z)$, the following steps are carried out.

1. $\mathcal{B}$ looks for a tuple of the form (U, Y_u, λ) in L2 such that $e(P, \lambda) = e(U, Y_u)$ or $\lambda = \top$.
   - If the tuple (U, Y_u, λ) is not in L2, $\mathcal{B}$ adds a new entry into L2 by storing $(U, Y_u, \top)$ as the query tuple and a value randomly drawn from the range of H_2 as the oracle return. The special symbol '$\top$' is used as a marker for denoting that the real value should be the solution of the CDH problem instance (U, Y_u). This step ensures that the value of $H_2(U, Y_u, \lambda)$ is *fixed* before σ is de-signcrypted.
   - If the tuple (U, Y_u, λ) is already in L2, the existing result will be used as the value of $H_2(U, Y_u, \lambda)$.
2. $\mathcal{B}$ computes $m \| Y_S \| V = Z \oplus H_2(U, Y_u, \lambda)$ and looks for a tuple of the form (m, U, Y_u, λ) in L1 such that $e(P, \lambda) = e(U, Y_u)$ or $\lambda = \top$.
   - If the tuple (m, U, Y_u, λ) is not in L1, $\mathcal{B}$ adds a new entry into L1 by storing (m, U, Y_u, λ) as the query tuple and setting $t'(aP)$ as the oracle return, where $t' \leftarrow \mathbb{Z}_q$. Note that λ can be a value such that $e(P, \lambda) = e(U, Y_u)$. This is because, the value may have been obtained from L2 above.
   - If the tuple (m, U, Y_u, λ) is in L1 and $e(P, \lambda) = e(U, Y_u)$, then besides using the existing result of L1 as the value for $H_1(m, U, Y_u, \lambda)$, the value of λ should also be used to update the corresponding entry in L2.
   - If the tuple $(m, U, Y_u, \top)$ is in L1, then the existing result in L1 will be used as the value for $H_1(m, U, Y_u, \top)$. Also if the value of λ can be obtained from L2, then the entry $(m, U, Y_u, \top)$ in L1 should also be updated to (m, U, Y_u, λ).
3. $\mathcal{B}$ checks if $e(P, V) = e(H_1(m, U, Y_u, \lambda), Y_S)$ holds.
   - If the equation holds and $e(P, \lambda) = e(U, Y_u)$, $(m, (U, Y_u, \lambda, V), Y_S)$ are returned as the message-signature pair and the sender's public key.
   - If the equation holds but $\lambda = \top$, then $\mathcal{B}$ halts with failure.
   - Otherwise, the symbol '$\perp$' is returned for rejection.

When $\mathcal{F}$ produces a ciphertext $\sigma = (U, Z)$ and a receiver's key pair (x_R, Y_R), $\mathcal{B}$ de-signcrypts the ciphertext as the simulation of de-signcrypt query above. If the forgery is valid, which means $e(Y_u, H_1(m, U, Y_R, \lambda)) = e(P, V) = e(bP, taP)$, $\mathcal{B}$ gets $V = tabP$ and solves the CDH problem from computing $abP = t^{-1}V$. Then $\mathcal{B}$ outputs abP and halts.

Analysis. From the simulation of the de-signcrypt query above, we can see that there must be an entry in L1 for $H_1(m, U, Y_R, \lambda)$. We also claim that the

corresponding oracle return in the entry must be in the form $t(aP)$ for some $t \in \mathbb{Z}_q$, which can be retrieved from L1. Notice that if $H_1(m, U, Y_R, \lambda)$ is equal to $t'P$, which is generated in a signcryption query, the values of Z would also have been determined in that signcryption query, which contradicts the restriction of the game defined in Definition 3.

Obviously, The running time of $\mathcal{B}$ is also in polynomial of $\mathcal{F}$'s running time. As proofed in Theorem 1, the simulated game is also computationally indistinguishable from a real game.

From the proof above, if $\mathcal{F}$ can win the game, then $\mathcal{B}$ can solve the CDH problem. It implies that the probability of $\mathcal{B}$ solving the CDH problem is equals to $\mathcal{F}$'s winning advantage $\rho(k)$, which is non-negligible. In addition, B halts and failure in the same case as mentioned in Theorem 1, with the probability that $p2^{-poly(k)}$. Hence, $\Pr[\mathcal{F} \text{ wins the game} \wedge \mathcal{B} \text{ does not fail}] \geq (1 - p2^{-poly(k)})\rho(k)$. □

Theorem 3. *Let k be a security parameter. Under the random oracle model, if there exists a PPT algorithm which can break the SC-ANON-CCA security of the proposed signcryption scheme with advantage at least $\rho(k)$, then there exists a PPT algorithm which can solve the Gap Diffie-Hellman Problem with probability at least $\frac{4}{3}(1 - p2^{-poly(k)})\rho(k)$ where $poly(\cdot)$ is some polynomial and p is the maximum number of de-signcryption queries made in the model of SC-ANON-CCA.*

Proof. The proof follows that of Theorem 1. Suppose $\mathcal{B}$ is given (aP, cP) as a random instance of the CDH problem, $\mathcal{B}$ runs $\mathcal{D}$ to find the solution acP.

$\mathcal{B}$ picks two random elements $x, y \in \mathbb{Z}_q$ and sets the two challenge public keys as $pk_{R,0} = x(cP)$ and $pk_{R,1} = y(cP)$. $\mathcal{B}$ then simulates all the hash queries, signcryption queries and de-signcryption queries as in the proof of Theorem 1.

After the completion of the first stage, $\mathcal{D}$ chooses two private keys $sk_{S,0}, sk_{S,1}$ and a plaintext $m \in \{0,1\}^n$ and requests a challenge ciphertext built under $sk_{S,b}$ and $pk_{R,b'}$ where $b, b' \stackrel{R}{\leftarrow} \{0,1\}$.

$\mathcal{B}$ sets the challenge ciphertext to $\sigma' = (U', Z')$ where $U' = aP$ and Z' is randomly drawn from $\{0,1\}^n \times \mathbb{G}_1^2$. $\mathcal{B}$ updates L1 by adding in $(m, aP, pk_{R,0}, \top)$ and $(m, aP, pk_{R,1}, \top)$, randomly picking $t', t'' \in \mathbb{Z}_q$ and setting them as the result of $H_1(m, aP, pk_{R,0}, \top)$ and $H_1(m, aP, pk_{R,1}, \top)$ respectively. Note that those entries will only be added in L1 if they are not in L1 yet. Similarly, L2 will also be updated with $(aP, pk_{R,0}, \top)$ and $(aP, pk_{R,1}, \top)$, the value of $H_2(aP, pk_{R,0}, \top)$ and $H_2(aP, pk_{R,1}, \top)$ are set to $Z' \oplus (m \| Y_s' \| t' Y_s')$ and $Z' \oplus (m || Y_s' || t'' Y_s')$ respectively, where $Y_s' = sk_{S,b}$.

$\mathcal{B}$ answers $\mathcal{D}$'s queries as in the first stage. If $\mathcal{D}$ queries H_1 or H_2 with $(aP, pk_{R,0}, \lambda)$ such that $e(aP, pk_{R,0}) = e(P, \lambda)$, $\mathcal{B}$ halts and outputs $x^{-1}\lambda$; If $\mathcal{D}$ queries H_1 or H_2 with $(aP, pk_{R,1}, \lambda)$ such that $e(aP, pk_{R,1}) = e(P, \lambda)$, $\mathcal{B}$ halts and outputs $y^{-1}\lambda$. $\mathcal{B}$ output a random point in $\mathbb{G}_1$ and halts, if $\mathcal{D}$ halts without making those queries.

Analysis. Obviously, the running time of $\mathcal{B}$ is in polynomial of $\mathcal{D}$'s running time. The simulated game is computationally indistinguishable from a real game with

the failure probability $p2^{-poly(k)}$ as proofed in Theorem 1. In the following, we analyze $\mathcal{B}$'s success rate.

Let $\mathbf{E}$ be the event that $(aP, pk_{R,0}, a(pk_{R,0}))$ or $(aP, pk_{R,1}, a(pk_{R,1}))$ has been queried on H_1 or H_2. $\bar{\mathbf{E}}$ denotes event $\mathbf{E}$ does not happen. Note that $\mathcal{B}$ solves the CDH problem instance in event $\mathbf{E}$.

We claim that for event $\bar{\mathbf{E}}$, $\mathcal{D}$ does not have any advantage in winning the game over random guessing: Let $V_{(b,b')} = sk_{S,b}H_1(m, aP, pk_{R,b'}, apk_{R,b'})$. Then $\sigma' = (aP, Z')$ is the signcryption of m under $sk_{S,b}$ and $pk_{R,b'}$ if we have

$$m \| sk_{S,b}P \| V_{b,b'} = Z' \oplus H_2(aP, pk_{R,b'}, apk_{R,b'})$$

In event $\bar{\mathbf{E}}$, since H_1 and H_2 are not queried with $(aP, pk_{R,0}, a(pk_{R,0}))$ or $(aP, pk_{R,1}, a(pk_{R,1}))$, due to the random oracle assumption, $\mathcal{D}$ does not have any advantage in determining the oracle returns of H_1 and H_2 on these query tuples. Hence, $\Pr[\mathcal{D} \text{ wins the game } |\bar{\mathbf{E}}] = \frac{1}{4}$. From the assumption,

$$\Pr[\mathcal{D} \text{ wins the game}] = \frac{1}{4} + \rho(k) \leq \Pr[\mathbf{E}] + \frac{1}{4}(1 - \Pr[\mathbf{E}])$$

where ρ is $\mathcal{D}$'s non-negligible advantage in winning the game defined in Definition 4 and k is the system-wide security parameter. Therefore, $\Pr[\mathbf{E}] \geq \frac{4}{3}\rho(k)$.

We have $\Pr[\mathbf{E} \wedge \mathcal{B} \text{ does not fail}] \geq \frac{4}{3}(1 - p2^{-poly(k)})\rho(k)$. □

4.3 Performance

We consider the costly operations which include point scalar multiplication on $\mathbb{G}_1$ ($\mathbb{G}_1$ Mul), exponentiation on $\mathbb{G}_2$ ($\mathbb{G}_2$ Exp), $MapToPoint$ hash operation [7] (MapToPoint) and pairing operation (Pairing).

Table 1. Efficiency of our proposed scheme

	$\mathbb{G}_1$ Mul	$\mathbb{G}_2$ Exp	MapToPoint	Pairing
Signcrypt	2	0	1	0
De-signcrypt	1	0	1	2
Verify	0	0	0	2

5 Conclusion

We proposed an efficient scheme that is proven secure with respect to confidentiality, unforgeability and ciphertext anonymity under the standard assumption of Gap Diffie-Hellman Problem in the random oracle model. The construction is efficient and requires even less operations than the original YWD scheme.

References

1. An, J.H., Dodis, Y., Rabin, T.: On the security of joint signature and encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 83–107. Springer, Heidelberg (2002)
2. Bao, F., Deng, R.H.: A signcryption scheme with signature directly verifiable by public key. In: Imai, H., Zheng, Y. (eds.) PKC 1998. LNCS, vol. 1431, pp. 55–59. Springer, Heidelberg (1998)
3. Beak, J., Steinfeld, R., Zheng, Y.: Formal proofs for the security of signcryption. In: Naccache, D., Paillier, P. (eds.) PKC 2002. LNCS, vol. 2274, pp. 80–98. Springer, Heidelberg (2002)
4. Bellare, M., Boldyreva, A., Desai, A., Pointcheval, D.: Key-privacy in public-key encryption. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 566–582. Springer, Heidelberg (2001)
5. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: First ACM Conference on Computer and Communications Security, Fairfax, pp. 62–73. ACM Press, New York (1993)
6. Boneh, D.: The decision Diffie-Hellman problem. In: Buhler, J.P. (ed.) Algorithmic Number Theory. LNCS, vol. 1423, pp. 48–63. Springer, Heidelberg (1998)
7. Boneh, D., Franklin, M.: Identity based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
8. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the Weil pairing. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 514–532. Springer, Heidelberg (2001)
9. Boyen, X.: Multipurpose identity-based signcryption: A swiss army knife for identity-based cryptography. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 383–399. Springer, Heidelberg (2003)
10. Chow, S.S.M., Yiu, S.M., Hui, L.C.K., Chow, K.P.: Efficient Forward and Provably Secure ID-Based Signcryption Scheme with Public Verifiability and Public Ciphertext Authenticity. In: Lim, J.-I., Lee, D.-H. (eds.) ICISC 2003. LNCS, vol. 2971, pp. 352–369. Springer, Heidelberg (2004)
11. Gamage, C., Leiwo, J., Zheng, Y.: Encrypted Message Authentication by Firewalls. In: Imai, H., Zheng, Y. (eds.) PKC 1999. LNCS, vol. 1560, pp. 69–81. Springer, Heidelberg (1999)
12. Goldwasser, S., Micali, S., Rivest, R.: A digital signature scheme secure against adaptive chosen-message attack. SIAM J. Computing 17(2), 281–308 (1988)
13. Libert, B., Quisquater, J.-J.: New Identity Based Signcryption Schemes from Pairings. In: IEEE Information Theory Workshop, pp. 155–158. IEEE Computer Society Press, Los Alamitos (2003), Full Version Available at `http://eprint.iacr.org`
14. Libert, B., Quisquater, J.-J.: Efficient signcryption with key privacy from gap Diffie-Hellman groups. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 187–200. Springer, Heidelberg (2004)
15. Malone-Lee, J., Mao, W.: Two birds one stone: Signcryption using RSA. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 211–225. Springer, Heidelberg (2003)
16. Mu, Y., Varadharajan, V.: Distributed signcryption. In: Roy, B., Okamoto, E. (eds.) INDOCRYPT 2000. LNCS, vol. 1977, pp. 155–164. Springer, Heidelberg (2000)
17. Rackoff, C., Simon, D.R.: Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 433–444. Springer, Heidelberg (1992)

18. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
19. Steinfeld, R., Zheng, Y.: A signcryption scheme based on integer factorization. In: Okamoto, E., Pieprzyk, J.P., Seberry, J. (eds.) ISW 2000. LNCS, vol. 1975, pp. 308–322. Springer, Heidelberg (2000)
20. Tan, C.-H.: On the security of signcryption scheme with key privacy. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. E88-A(4), 1093–1095 (2005)
21. Tan, C.-H.: Analysis of improved signcryption scheme with key privacy. Information Processing Letters 99(4), 135–138 (2006)
22. Yang, G., Wong, D.S., Deng, X.: Analysis and improvement of a signcryption scheme with key privacy. In: Zhou, J., Lopez, J., Deng, R.H., Bao, F. (eds.) ISC 2005. LNCS, vol. 3650, pp. 218–232. Springer, Heidelberg (2005)
23. Yum, D.H., Lee, P.J.: New signcryption schemes based on KCDSA. In: Kim, K.-c. (ed.) ICISC 2001. LNCS, vol. 2288, pp. 305–317. Springer, Heidelberg (2002)
24. Zheng, Y.: Digital signcryption or how to achieve cost(signature & encryption). In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 165–179. Springer, Heidelberg (1997)

Direct Chosen-Ciphertext Secure Hierarchical ID-Based Encryption Schemes*

Jong Hwan Park and Dong Hoon Lee

Center for Information Security Technologies(CIST),
Korea University, Seoul, Korea
{decartian,donghlee}@korea.ac.kr

Abstract. We describe two Hierarchical Identity Based Encryption (HIBE) schemes which are selective-ID chosen ciphertext secure. Our constructions are based on the Boneh-Boyen and the Boneh-Boyen-Goh HIBE schemes respectively. We apply the signature-based method to their HIBE schemes. The proposed l-level HIBE schemes are directly derived from l-level HIBE schemes secure against chosen plaintext attacks without padding on identities with one-bit. This is more compact than the known generic transformation suggested by Canetti et al..

Keywords: Hierarchical Identity Based Encryption, Chosen Ciphertext Security.

1 Introduction

Hierarchical Identity Based Encryption (HIBE) [17,16,4,5] is a generalization of Identity Based Encryption (IBE) [18,7,19,15] which allows a sender to encrypt a message for a receiver using the receiver's identity as a public key. In an l-level HIBE scheme, an identity is represented as ID-vectors of length at most l, and a private key for identity at depth $k(< l)$ can be used to derive private keys of its descendant identities. HIBE schemes could be applied to design forward-secure encryption schemes [12,20], and to convert a broadcast encryption scheme in the symmetric key setting into a public key broadcast encryption scheme [14]. Recently, Boyen et al. [11] suggested an anonymous HIBE scheme which mainly gives several application in the public key encryption with keyword search [1].

To prove the security for HIBE schemes without random oracles, Canetti et al. [12] defined a weaker security model called selective-ID security model, and proposed a HIBE scheme. Their scheme is selective-ID secure without random oracles, but that is not efficient. Later, Boneh and Boyen [4] provided an efficient HIBE (denoted by BB_1) scheme, and thereafter Boneh, Boyen, and Goh [5] presented an improved HIBE (denoted by BBG) scheme where the number of

* This research was supported by the MIC(Ministry of Information and Communication), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute of Information Technology Advancement)(IITA-2006-(C1090-0603-0025)).

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 94–109, 2007.

ciphertext elements and pairing operations are independent of the hierarchy depth. These two HIBE schemes suggested by Boneh et al. were provably secure in the selective-ID model without random oracles. More recently, the techniques of constructing the BB_1 and BBG schemes were combined with a public key broadcast encryption scheme [8] in order to achieve the forward security [2].

Chosen ciphertext security of the BB_1 and BBG schemes are obtained from the generic transformation, proposed by Canetti, Halevi, and Katz [13]. The CHK transformation enables construction of an l-level HIBE scheme selective-ID secure against chosen ciphertext attacks based on any $(l+1)$-level HIBE scheme selective-ID secure against chosen plaintext attacks. The CHK transformation, improved upon by [9,10], is generic and extended to the case of adaptive-ID security model (i.e., the full security model) [6].

The CHK transformation requires one-time signature scheme to check the consistency of ciphertext. The important point is that a verification key associated with the one-time signature needs to be embedded into ciphertext in encryption procedure. For this, the authors [13] add one level to an identity hierarchy and set the verification key as an identity. Thus, the CHK transformation considered an $(l+1)$-level HIBE scheme as a subroutine in constructing an l-level HIBE scheme secure against chosen ciphertext attacks. We notice that the CHK transformation needs extra one-bit padding on identities, due to their security proof.

In this paper we construct two HIBE schemes which are provably secure against chosen ciphertext attacks in the selective-ID model. Two schemes are based on the the BB_1 and BBG schemes respectively. We apply the idea of the CHK transformation to their schemes, using one-time signature. At first sight, our constructions appear to apply the CHK transformation to the BB_1 and BBG schemes, but we obtain chosen ciphertext security of l-level HIBE schemes from l-level HIBE schemes secure against chosen plaintext attacks *directly*, without padding on identities with one-bit. Though our approach is not generic, that could be also applied to the concrete schemes [2] with structures of the BB_1 and BBG schemes.

The important algebraic property for security proofs is the one introduced by Boneh et al. [4]. Briefly speaking, for random elements g_1 and g_2 in $\mathbb{G}$ (where $\mathbb{G}$ is generated by a generator g), and random elements r_1, r_2, and r_3 in $\mathbb{Z}_p$ (where r_1 must be non-zero), we have that

$$g_2^{-r_2/r_1}(g_1^{r_1}g^{r_2})^{r_3} = g_2^{u}(g_1^{r_1}g^{r_2})^{r_3 - v/r_1}$$

where $u = \log_g g_1$ and $v = \log_g g_2$. For example, if we let $g_1 = g^a$ and $g_2 = g^b$, the value g_2^u becomes g^{ab}, and if we let $g_1 = g^\alpha$ and $g_2 = g^{\alpha^l}$, the value g_2^u becomes $g^{\alpha^{l+1}}$. The former plays a central role of proving the security of our first construction based on the BB_1 scheme, and the latter does in proving the security of our second construction based on the BBG scheme.

2 Preliminaries

We briefly review the definition of security for HIBE. We also summarize the bilinear maps and the related security assumptions.

2.1 Selective-ID Security Model for HIBE

In a Hierarchical Identity Based Encryption (HIBE) scheme [16,4,5], identities are considered as vectors. That is, an identity of depth l is a tuple $\mathrm{ID} = (\mathrm{I}_1, \ldots, \mathrm{I}_l)$. A HIBE scheme consists of the four algorithms [4,5]: *Setup*, *KeyGen*, *Encrypt*, *Decrypt*. The *Setup* algorithm generates system parameters *params* and a master key *master-key*. The *KeyGen* algorithm takes as input an identity $\mathrm{ID} = (\mathrm{I}_1, \ldots, \mathrm{I}_l)$ at depth l and the private key $d_{\mathrm{ID}|l-1}$ of the parent identity $\mathrm{ID}_{|l-1} = (\mathrm{I}_1, \ldots, \mathrm{I}_{l-1})$ at depth $l-1$. It outputs the private key d_{ID} for identity ID. To encrypt messages, the *Encrypt* algorithm requires a receiver's identity (as a public key) and the system parameters. The *Decrypt* algorithm decrypts ciphertexts with a private key associated with the receiver's identity.

To prove the chosen ciphertext security for HIBE schemes without random oracles, we are interested in the selective-ID security model suggested by Canetti et al. [12,13]. This model is weaker than the full security model (for HIBE schemes, see [5]) in that, in the selective-ID model the adversary commits ahead of time to the identity that it wishes to be challenged on. Since Canetti et al. first proposed the selective-ID model, many cryptographic protocols [4,5,8,11] were proved secure in this weaker security model without random oracles. Selective-ID security model for HIBE schemes is defined via the following game between an adversary $\mathcal{A}$ and a challenger:

Init: $\mathcal{A}$ outputs an identity ID^* where it wishes to be challenged.

Setup: The challenger runs *Setup* algorithm. It gives $\mathcal{A}$ the resulting system parameters *params*. It keeps the *master-key* to itself.

Phase 1: $\mathcal{A}$ issues queries $q_1, ..., q_m$ adaptively where query q_i is one of:

- Private key query on ID_i where $\mathrm{ID}_i \neq \mathrm{ID}^*$ and ID_i is not a prefix of ID^*. The challenger responds by running *KeyGen* algorithm to generate the private key d_i corresponding to the public key ID_i. It sends d_i to $\mathcal{A}$.
- Decryption query CT_i on ID^* or any prefix of ID^*. The challenger responds by running *KeyGen* algorithm to generate the private key d corresponding to ID^*. It then runs *Decrypt* algorithm to decrypt the ciphertext CT_i using the private key d and sends the resulting plaintext to $\mathcal{A}$.

Challenge: Once $\mathcal{A}$ decides that Phase 1 is over, it outputs two equal length plaintexts $M_0, M_1 \in \mathcal{M}$ on which it wishes to be challenged. The challenger picks a random bit $b \in \{0, 1\}$ and computes $\mathsf{CT} = \mathit{Encrypt}(M_b, \mathit{params}, \mathrm{ID}^*)$ as the challenge ciphertext. It sends CT as the challenge to $\mathcal{A}$.

Phase 2: $\mathcal{A}$ issues more queries $q_{m+1}, ..., q_n$ adaptively where q_i is one of:

- Private key query on ID_i where $\mathrm{ID}_i \neq \mathrm{ID}^*$ and ID_i is not a prefix of ID^*. The challenger responds as in Phase 1.

– Decryption query $\mathsf{CT}_i \neq \mathsf{CT}$ on ID^* or any prefix of ID^*. The challenger responds as in Phase 1.

Guess: Finally, $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$. $\mathcal{A}$ wins if $b' = b$.

We refer to such an adversary $\mathcal{A}$ as an IND-sID-CCA adversary. The advantage of $\mathcal{A}$ in breaking the HIBE scheme $\mathcal{E}$ is defined as

$$\mathrm{Adv}_{\mathcal{E},\mathcal{A}} = \left|\Pr[b = b'] - \frac{1}{2}\right|.$$

Definition 1. *We say that a HIBE scheme $\mathcal{E}$ is $(t, q_{ID}, q_C, \epsilon)$-selective-ID, adaptive chosen ciphertext secure if for any t-time IND-sID-CCA adversary $\mathcal{A}$ that makes at most q_{ID} chosen private key queries, at most q_C chosen decryption queries we have that $Adv_{\mathcal{E},\mathcal{A}} < \epsilon$.*

2.2 Complexity Assumptions

We briefly summarize the bilinear maps, and review the Bilinear Diffie-Hellman (BDH) and the Bilinear Diffie-Hellman Exponent (BDHE) assumptions.

Bilinear Groups: We follow the notation in [7,4].

1. $\mathbb{G}$ and $\mathbb{G}_1$ are two (multiplicative) cyclic groups of prime order p.
2. g be a generator of $\mathbb{G}$.
3. e is a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_1$.

A bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_1$ has the following properties:

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

We say that $\mathbb{G}$ is a bilinear group if the group action in $\mathbb{G}$ can be computed efficiently and there exists a group $\mathbb{G}_1$ and an efficiently computable bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_1$ as above. Note that $e(,)$ is symmetric since $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

Bilinear Diffie-Hellman Assumption: The BDH problem in $\mathbb{G}$ is defined as follows: given a tuple $(g, g^a, g^b, g^c) \in \mathbb{G}^4$ as input, compute $e(g, g)^{abc} \in \mathbb{G}_1$. An algorithm $\mathcal{A}$ has advantage ϵ in solving BDH in $\mathbb{G}$ if

$$\Pr\left[\mathcal{A}(g, g^a, g^b, g^c) = e(g, g)^{abc}\right] \geq \epsilon$$

where the probability is over the random choice of a,b,c in $\mathbb{Z}_p$ and the random bits of $\mathcal{A}$. We can also say that an algorithm $\mathcal{B}$ that outputs $b \in \{0,1\}$ has advantage ϵ in solving the *decision* BDH problem in $\mathbb{G}$ if

$$\left|\Pr\left[\mathcal{B}(g, g^a, g^b, g^c,\ e(g, g)^{abc}) = 0\right] - \Pr\left[\mathcal{B}(g, g^a, g^b, g^c,\ T) = 0\right]\right| \geq \epsilon$$

where the probability is over the random choice of a,b,c in $\mathbb{Z}_p$, the random choice of $T \in \mathbb{G}_1$, and the random bits of $\mathcal{B}$.

Definition 2. *We say that the (decision)* (t, ϵ)*-BDH assumption holds in* $\mathbb{G}$ *if no* t*-time algorithm has advantage at least* ϵ *in solving the (decision) BDH problem in* $\mathbb{G}$*.*

Bilinear Diffie-Hellman Exponent Assumption: The l-BDHE problem in $\mathbb{G}$ is defined as follows: given a $(2l+1)$-tuple $(g, h, g^x, \ldots, g^{x^l}, g^{x^{l+2}}, \ldots, g^{x^{2l}}) \in \mathbb{G}^{2l+1}$ as input, compute $e(g,h)^{x^{l+1}} \in \mathbb{G}_1$. An algorithm $\mathcal{A}$ has advantage ϵ in solving q-BDHE in $\mathbb{G}$ if

$$\Pr\Big[\mathcal{A}(g, h, g^x, \ldots, g^{x^l}, g^{x^{l+2}}, \ldots, g^{x^{2l}}) = e(g,h)^{x^{l+1}}\Big] \geq \epsilon$$

where the probability is over the random choice of x in $\mathbb{Z}_p$, the random choice of $h \in \mathbb{G}$, and the random bits of $\mathcal{A}$. Let $\overrightarrow{g}_{x,l} = (g^x, \ldots, g^{x^l}, g^{x^{l+2}}, \ldots, g^{x^{2l}})$. Similarly, we say that an algorithm $\mathcal{B}$ that outputs $b \in \{0,1\}$ has advantage ϵ in solving the *decision* q-BDHE problem in $\mathbb{G}$ if

$$\Big|\Pr\Big[\mathcal{B}(g, h, \overrightarrow{g}_{x,l},\ e(g,h)^{x^{l+1}}) = 0\Big] - \Pr\Big[\mathcal{B}(g, h, \overrightarrow{g}_{x,l},\ T) = 0\Big]\Big| \geq \epsilon$$

where the probability is over the random choice of x in $\mathbb{Z}_p$, the random choice of $h \in \mathbb{G}$, the random choice of $T \in \mathbb{G}_1$, and the random bits of $\mathcal{B}$.

Definition 3. *We say that the (decision)* (t, l, ϵ)*-BDHE assumption holds in* $\mathbb{G}$ *if no* t*-time algorithm has advantage at least* ϵ *in solving the (decision)* l*-BDHE problem in* $\mathbb{G}$*.*

3 Chosen Ciphertext Secure HIBE from the $\mathsf{BB_1}$ Scheme

In this section we present an l-level HIBE scheme that is derived from the l-level $\mathsf{BB_1}$ scheme, using the idea of the CHK transformation. The constructed l-level HIBE scheme is secure against chosen ciphertext attacks in the selective-ID model without random oracles. For the CHK transformation, we need a one-time signature scheme $Sig = (\mathit{SigKeyGen}, \mathit{Sign}, \mathit{Verify})$ which is strongly existentially unforgeable (see the details in [3]). We also need a collision resistant hash function that maps verification keys to $\mathbb{Z}_p$. For simplicity, we assume that the verification keys are elements of $\mathbb{Z}_p$.

3.1 Construction

Setup(k): To generate HIBE system parameters for maximum depth of l, select random $\alpha \in \mathbb{Z}_p^*$ and set $g_1 = g^\alpha$. Next, pick random elements $h, h_1, \ldots, h_l \in \mathbb{G}$ and a generator $g_2 \in \mathbb{G}$. The public parameters *params* (with the description of ($\mathbb{G}$, $\mathbb{G}_1$, p)) and the secret *master-key* are given by

$$\mathit{params} = (g, g_1, g_2, h, h_1, \ldots, h_l), \qquad \mathit{master\text{-}key} = g_2^\alpha.$$

For $j = 1, \ldots, l$, define $F_j : \mathbb{Z}_p \to \mathbb{G}$ to be the function: $F_j(x) = g_1^x h_j$.

KeyGen($d_{\mathrm{ID}|j-1}$, ID): To create a private key d_{ID} for a user $\mathrm{ID} = (\mathrm{I}_1, \ldots, \mathrm{I}_j) \in \mathbb{Z}_p^j$ of depth $j \leq l$, pick random $r_1, \ldots, r_j \in \mathbb{Z}_p$ and output

$$d_{\mathrm{ID}} = \Big(g_2^{\alpha} \prod_{k=1}^{j} F_k(\mathrm{I}_k)^{r_k},\ g^{r_1}, \ldots, g^{r_j} \Big).$$

The private key for ID can be also generated from a private key for $d_{\mathrm{ID}|j-1}$. Let $d_{\mathrm{ID}|j-1} = (d_0, \ldots, d_{j-1})$ be the private key for $\mathrm{ID}_{j-1} = (\mathrm{I}_1, \ldots, \mathrm{I}_{j-1})$. After selecting random $r_1, \ldots, r_j \in \mathbb{Z}_p$, output d_{ID} as

$$\Big(d_0 \cdot \prod_{k=1}^{j} F_k(\mathrm{I}_k)^{r_k},\ d_1 \cdot g^{r_1}, \ldots,\ d_{j-1} \cdot g^{r_{j-1}},\ g^{r_j} \Big).$$

Encrypt(M, *params*, ID): To encrypt a message $M \in \mathbb{G}_1$ under a public key $\mathrm{ID} = (\mathrm{I}_1, \ldots, \mathrm{I}_j) \in \mathbb{Z}_p^j$,

1. Run the *SigKeyGen* to obtain a signing key SigK and a verification key VerK.
2. Pick a random $s \in \mathbb{Z}_p^*$ and compute
$$C = \Big(g^s,\ e(g_1, g_2)^s \cdot M,\ F_1(I_1)^s, \ldots,\ F_j(I_j)^s,\ (g_1^{\mathsf{VerK}} h)^s \Big).$$
3. Output the ciphertext $\mathsf{CT} = (C, Sign_{\mathsf{SigK}}(C), \mathsf{VerK})$.

Decrypt(CT, *params*, d_{ID}): To decrypt a ciphertext $\mathsf{CT} = (C, \sigma, \mathsf{VerK})$ using the private key $d_{\mathrm{ID}} = (d_0, \ldots, d_j)$,

1. Verify that the signature σ on C is valid under the verification key VerK. If invalid, output $\perp$.
2. Otherwise, let $C = (A, B, C_1, \ldots, C_{j+1})$. Pick a random $r_{j+1} \in \mathbb{Z}_p^*$ and output
$$\frac{\prod_{k=1}^{j} e(C_k,\ d_k) \cdot e(C_{j+1}, g^{r_{j+1}})}{e(A,\ d_0 \cdot (g_1^{\mathsf{VerK}} h)^{r_{j+1}})} \cdot B.$$

The correctness of decryption algorithm is checked as below:

$$\frac{\prod_{k=1}^{j} e(C_k,\ d_k) \cdot e(C_{j+1}, g^{r_{j+1}})}{e(A,\ d_0 \cdot (g_1^{\mathsf{VerK}} h)^{r_{j+1}})} = \frac{\prod_{k=1}^{j} e(F_k(I_k)^s,\ g^{r_k}) \cdot e((g_1^{\mathsf{VerK}} h)^s,\ g^{r_{j+1}})}{e(g^s,\ g_2^{\alpha} \prod_{k=1}^{j} F_k(I_k)^{r_k} \cdot (g_1^{\mathsf{VerK}} h)^{r_{j+1}})}$$
$$= \frac{\prod_{k=1}^{j} e(F_k(I_k)^{r_k},\ g^s) \cdot e((g_1^{\mathsf{VerK}} h)^{r_{j+1}},\ g^s)}{e(g^s,\ g_2^{\alpha}) \cdot e(g^s, \prod_{k=1}^{j} (F_k(I_k))^{r_k} \cdot (g_1^{\mathsf{VerK}} h)^{r_{j+1}})} = \frac{1}{e(g_1, g_2)^s}.$$

At a first glance, the above scheme has a similar structure as the $(l+1)$-level HIBE scheme in that the additional element $h \in \mathbb{G}$ adds to the public parameters and the size of ciphertext increases by one more element. However, the private key for ID is still generated at level $(l-1)$ and is the same as that of chosen plaintext secure l-level HIBE scheme. We note that unlike the BB_1 scheme [4], randomization in the *KeyGen* (in deriving the private keys from its parent identity) and the *Decrypt* algorithms is necessary for the proof of security.

3.2 Security

Theorem 1. *Suppose that the decision (t, ϵ_1)-BDH assumption holds in $\mathbb{G}$ and the signature scheme is $(t, 1, \epsilon_2)$-strongly existentially unforgeable. Then the previous l-HIBE scheme is $(t', q_{\mathrm{ID}}, q_C, \epsilon)$-selective-ID, adaptive chosen ciphertext secure for arbitrary q_{ID}, q_C, and $t' < t - o(t)$, where $\epsilon_1 + q_{\mathrm{ID}}/p + \epsilon_2 \geq \epsilon$.*

Proof. Suppose there exists an adversary $\mathcal{A}$ which has advantage ϵ in attacking the l-level HIBE scheme. We want to build an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the decision BDH problem in $\mathbb{G}$. On input (g, g^a, g^b, g^c, T) for some unknown $a, b, c \in \mathbb{Z}_p^*$, $\mathcal{B}$ outputs 1 if $T = e(g,g)^{abc}$ and 0 otherwise. $\mathcal{B}$ works by interacting with $\mathcal{A}$ in a selective-ID game as follows:

Init: $\mathcal{A}$ outputs an identity $\mathrm{ID}^* = (\mathrm{I}_1^*, \ldots, \mathrm{I}_k^*) \in \mathbb{Z}_p^k$ of depth $k \leq l$ that it intends to attack.

Setup: Let $g_1 = g^a$, $g_2 = g^b$, and $g_3 = g^c$. If the length of ID^* is less than l, $\mathcal{B}$ selects random elements $(\mathrm{I}_{k+1}^*, \ldots, \mathrm{I}_l^*)$ in $\mathbb{Z}_p$. To generate the system parameters, $\mathcal{B}$ first selects random $\alpha_1, \ldots, \alpha_l \in \mathbb{Z}_p$ and defines $h_j = g_1^{-\mathrm{I}_j^*} g^{\alpha_j}$ for $j = 1, \ldots, l$. Next, $\mathcal{B}$ runs $SigKeyGen$ algorithm to gain a signing key SigK^* and a verification key VerK^*, and $\mathcal{B}$ also selects a random $\beta \in \mathbb{Z}_p$ and computes $h = g_1^{-\mathsf{VerK}^*} g^{\beta}$. $\mathcal{B}$ gives $\mathcal{A}$ the system parameters $params = (g, g_1, g_2, h, h_1, \ldots, h_l)$. The master key corresponding to these $params$ is $g_2^a = g^{ab}$, which is unknown to $\mathcal{B}$. For $j = 1, \ldots, l$, the function $F_j : \mathbb{Z}_p \to \mathbb{G}$ is defined as

$$F_j(x) = g_1^x h_j = g_1^{x - \mathrm{I}_j^*} g^{\alpha_j}.$$

Phase 1: $\mathcal{A}$ issues up to q_{ID} private key queries and q_C decryption queries. Consider a query for the private key corresponding to $\mathrm{ID} = (\mathrm{I}_1, \ldots, \mathrm{I}_u) \in \mathbb{Z}_p^u$ where $u \leq l$. We further distinguish two cases according to whether ID^* is not a prefix of ID or not.

First, consider the case ID^* is not a prefix of ID. Then there exists at least one $j \in \{1, \ldots, u\}$ such that $\mathrm{I}_j \neq \mathrm{I}_j^*$. To respond to the query, $\mathcal{B}$ responds to the query by first computing a private key for the identity $(\mathrm{I}_1, \ldots, \mathrm{I}_j)$ from which it derives a private key for the requested identity $\mathrm{ID} = (\mathrm{I}_1, \ldots, \mathrm{I}_j, \ldots, \mathrm{I}_u)$. $\mathcal{B}$ picks random elements $r_1, \ldots, r_j \in \mathbb{Z}_p$ and computes

$$d_0 = g_2^{\frac{-\alpha_j}{\mathrm{I}_j - \mathrm{I}_j^*}} \prod_{v=1}^{j} F_v(\mathrm{I}_v)^{r_v}, \quad d_1 = g^{r_1}, \ldots, \ d_{j-1} = g^{r_{j-1}}, \ d_j = g_2^{\frac{-1}{\mathrm{I}_j - \mathrm{I}_j^*}} g^{r_j}.$$

By the same argument as in [4], we see that $(d_0, d_1, \ldots, d_j)$ is a valid private key for $(\mathrm{I}_1, \ldots, \mathrm{I}_j)$. For the unknown $\widetilde{r}_j = r_j - b/(\mathrm{I}_j - \mathrm{I}_j^*)$, $\mathcal{B}$ has

$$g_2^{\frac{-\alpha_j}{\mathrm{I}_j - \mathrm{I}_j^*}} F_j(\mathrm{I}_j)^{r_j} = g_2^{\frac{-\alpha_j}{\mathrm{I}_j - \mathrm{I}_j^*}} (g_1^{\mathrm{I}_j - \mathrm{I}_j^*} g^{\alpha_j})^{r_j} = g_2^a F_j(\mathrm{I}_j)^{\widetilde{r}_j}, \quad d_j = g^{\widetilde{r}_j}.$$

Then, $\mathcal{B}$ can construct a private key for the requested ID from the above private key $(d_0, d_1, \ldots, d_j)$ and gives $\mathcal{A}$ the obtained private key d_{ID}.

Second, consider the case ID^* is a prefix of ID. Then it satisfies that $k+1 \leq u$. Let $\mathrm{ID} = (\mathrm{I}_1^*, \ldots, \mathrm{I}_k^*, \mathrm{I}_{k+1}, \ldots, \mathrm{I}_u)$. If $\mathrm{I}_j = \mathrm{I}_j^*$ for $j = k+1, \ldots, u$, then $\mathcal{B}$ outputs a random bit $b \in \{0,1\}$ and aborts the simulation. Otherwise, there exists at least one $j \in \{k+1, \ldots, u\}$ such that $\mathrm{I}_j \neq \mathrm{I}_j^*$. $\mathcal{B}$ responds to the query by first computing a private key for $\mathrm{ID} = (\mathrm{I}_1^*, \ldots, \mathrm{I}_k^*, \mathrm{I}_{k+1}, \ldots, \mathrm{I}_j)$ from which it constructs a private key for the requested $\mathrm{ID} = (\mathrm{I}_1^*, \ldots, \mathrm{I}_k^*, \mathrm{I}_{k+1}, \ldots, \mathrm{I}_j, \ldots, \mathrm{I}_u)$. $\mathcal{B}$ picks random elements $r_1, \ldots, r_j \in \mathbb{Z}_p$. Let $\widetilde{r}_j = r_j - b/(\mathrm{I}_j - \mathrm{I}_j^*)$. Then $\mathcal{B}$ generates the private key for $\mathrm{ID} = (\mathrm{I}_1^*, \ldots, \mathrm{I}_k^*, \mathrm{I}_{k+1}, \ldots, \mathrm{I}_j)$ as

$$d_0 = g_2^{\frac{-\alpha_j}{\mathrm{I}_j - \mathrm{I}_j^*}} \prod_{v=1}^{k} F_v(\mathrm{I}_v)^{r_v}, \quad d_1 = g^{r_1}, \ \ldots, \ d_k = g^{r_k},$$

$$d_{k+1} = g^{r_{k+1}}, \ \ldots, \ d_j = g_2^{\frac{-1}{\mathrm{I}_j - \mathrm{I}_j^*}} g^{r_j}.$$

By the similar argument above, this private key has a proper distribution and is computable.

Next, $\mathcal{B}$ responds to decryption queries for $\mathrm{ID}^* = (\mathrm{I}_1^*, \ldots, \mathrm{I}_k^*)$ or any prefix of ID^*. Let $\mathrm{ID}' = (\mathrm{I}_1^*, \ldots, \mathrm{I}_j^*)$ where $j \leq k$ and let $(C, \sigma, \mathsf{VerK})$ be a decryption query for ID' where $C = (A, B, C_1, \ldots, C_{j+1})$. $\mathcal{B}$ does as follows:

1. Run $Verify$ to check the validity of the signature σ on C, using the verification key VerK. If the signature is invalid, $\mathcal{B}$ responds with $\perp$.
2. If $\mathsf{VerK} = \mathsf{VerK}^*$, $\mathcal{B}$ outputs a random bit $b \in \{0,1\}$ and aborts the simulation.
3. Otherwise, $\mathcal{B}$ selects random $\{r_i\}$ for $i = 1, \ldots, j+1$, and computes

$$\widetilde{d}_0 = g_2^{\frac{-\beta}{\mathsf{VerK} - \mathsf{VerK}^*}} (g_1^{\mathsf{VerK} - \mathsf{VerK}^*} g^{\beta})^{r_{j+1}} \cdot \prod_{v=1}^{j} F_v(\mathrm{I}_v^*)^{r_v},$$

$$\widetilde{d}_1 = g^{r_1}, \ \ldots, \ \widetilde{d}_j = g^{r_j}, \ \widetilde{d}_{j+1} = g_2^{\frac{-1}{\mathsf{VerK} - \mathsf{VerK}^*}} g^{r_{j+1}}.$$

As the above, for some (unknown) $\widetilde{r}_{j+1} = r_{j+1} - b/(\mathsf{VerK} - \mathsf{VerK}^*)$, we see that

$$g_2^{\frac{-\beta}{\mathsf{VerK} - \mathsf{VerK}^*}} (g_1^{\mathsf{VerK} - \mathsf{VerK}^*} g^{\beta})^{r_{j+1}} = g_2^a (g_1^{\mathsf{VerK} - \mathsf{VerK}^*} g^{\beta})^{\widetilde{r}_{j+1}} = g_2^a (g_1^{\mathsf{VerK}} h)^{\widetilde{r}_{j+1}},$$

and $\widetilde{d}_{j+1} = g^{\widetilde{r}_{j+1}}$. Then, $\mathcal{B}$ computes the plaintext as

$$\frac{\prod_{v=1}^{j} e(C_v, \ \widetilde{d}_v) \cdot e(C_{j+1}, \widetilde{d}_{j+1})}{e(A, \ \widetilde{d}_0)} \cdot B.$$

This computation is identical to the *Decrypt* algorithm in a real attack, since $\{r_i\}$ for $i = 1, \ldots, j+1$ are uniform in $\mathbb{Z}_p$ and $\widetilde{d}_0 = g_2^a \cdot \prod_{v=1}^{j} F_v(\mathrm{I}_v^*)^{r_v} \cdot (g_1^{\mathsf{VerK}} h)^{\widetilde{r}_{j+1}}$.

Challenge: $\mathcal{A}$ outputs two messages $M_0, M_1 \in \mathbb{G}_1$. To encrypt one of the two messages under the public key ID^*, $\mathcal{B}$ selects a random bit $b \in \{0,1\}$ and computes $C = (g_3,\ M_b \cdot T,\ g_3^{\alpha_1}, \ldots,\ g_3^{\alpha_j},\ g_3^{\beta})$. Next, $\mathcal{B}$ gives the challenge ciphertext $\mathsf{CT} = (C, Sign_{\mathsf{SigK}^*}(C), \mathsf{VerK}^*)$ to $\mathcal{A}$. Since $F_i(\mathrm{I}_i^*) = g^{\alpha_i}$ for $i = 1, \ldots, j$ and $g_1^{\mathsf{VerK}^*} h = g^{\beta}$, we have that

$$C = (g^c,\ M_b \cdot T,\ F_1(\mathrm{I}_1^*)^c, \ldots,\ F_l(\mathrm{I}_l^*)^c,\ (g_1^{\mathsf{VerK}^*} h)^c).$$

If $T = e(g,g)^{abc} = e(g_1, g_2)^c$, then C is a valid encryption of M_b under the public key ID^*. Otherwise, $M_b \cdot T$ is just a random element of $\mathbb{G}_1$ and independent of the bit b in the adversary's view.

Phase 2: $\mathcal{A}$ issues more private key and decryption queries. $\mathcal{B}$ responds as in Phase 1.

Guess : $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$. If $b = b'$ then $\mathcal{B}$ outputs 1, indicating $T = e(g,g)^{abc}$. Otherwise, it outputs 0, indicating $T \neq e(g,g)^{abc}$.

We consider two cases. When T is random in $\mathbb{G}_1$ then $\Pr[\mathcal{B}(g, g^a, g^b, g^c, T) = 0] = 1/2$. Let Iden denote the event that $\mathcal{A}$ issues a private key query for $\mathrm{ID} = (\mathrm{I}_1^*, \ldots, \mathrm{I}_k^*, \mathrm{I}_{k+1}, \ldots, \mathrm{I}_u)$ such that $\mathrm{I}_j = \mathrm{I}_j^*$ for $i = k+1, \ldots, u$. Also, let Forge denote the event that $\mathcal{A}$ submits a valid ciphertext $\mathsf{CT} = (C, \sigma, \mathsf{VerK}^*)$ as a decryption query. In the cases of Iden and Forge, $\mathcal{B}$ cannot reply to the private key and decryption queries, and aborts the simulation. When $T = e(g,g)^{abc}$, $\mathcal{B}$ replied with valid private key and plaintext unless events Iden and Forge occur. Then, $\mathcal{B}$ has

$$\left|\Pr[\mathcal{B}(g, g^a, g^b, g^c, T) = 0] - \frac{1}{2}\right| \geq \left|\Pr[b = b' \wedge \overline{\mathsf{Iden}} \wedge \overline{\mathsf{Forge}}] - \frac{1}{2}\right| - \Pr[\mathsf{Iden}] - \Pr[\mathsf{Forge}].$$

Since $\mathcal{B}$ provided $\mathcal{A}$ with perfect simulation when events Iden and Forge did not occur, $|\Pr[b = b' \wedge \overline{\mathsf{Iden}} \wedge \overline{\mathsf{Forge}}] - 1/2| \geq \epsilon$. From the simple calculation, we know that $\Pr[\mathsf{Iden}]$ is at most q_{ID}/p. Also, note that $\Pr[\mathsf{Forge}]$ is negligible. This means that $\Pr[\mathsf{Forge}] < \epsilon_2$ since otherwise, $\mathcal{B}$ can construct a forger, which is contradiction to the one-time signature. Therefore,

$$\left|\Pr\left[\mathcal{B}(g, g^a, g^b, g^c, e(g,g)^{abc}) = 0\right] - \Pr\left[\mathcal{B}(g, g^a, g^b, g^c, T) = 0\right]\right| \geq \epsilon - \frac{q_{\mathrm{ID}}}{p} - \epsilon_2$$

This completes the proof of Theorem 1. □

4 Chosen Ciphertext Secure HIBE from the BBG Scheme

We present an l-level HIBE scheme secure against chosen ciphertext attacks based on the l-level BBG scheme secure against chosen plaintext attacks. As in the previous section, we need a one-time signature scheme $Sig = (SigKeyGen, Sign, Verify)$, and we assume that verifications keys are elements of $\mathbb{Z}_p$.

4.1 Construction

Setup(k): To generate public parameters for maximum depth of l, select random $\alpha \in \mathbb{Z}_p^*$ and set $g_1 = g^\alpha$. Next, pick random elements $g_2, g_3, v, h_1, \ldots, h_l \in \mathbb{G}$. The public parameters *params* (with the description of $(\mathbb{G}, \mathbb{G}_1, p)$) and the secret *master-key* are given by

$$params = (g, g_1, g_2, g_3, v, h_1, \ldots, h_l), \qquad master\text{-}key = g_4 = g_2^\alpha$$

KeyGen($d_{\mathrm{ID}|j-1}$, ID): To create a private key d_{ID} for a user $\mathrm{ID} = (\mathrm{I}_1, \ldots, \mathrm{I}_j) \in \mathbb{Z}_p^j$ of depth $j \leq l$, pick random $r \in \mathbb{Z}_p$ and output

$$d_{\mathrm{ID}} = \Big(g_2^\alpha \cdot (h_1^{\mathrm{I}_1} \cdots h_j^{\mathrm{I}_j} \cdot g_3)^r,\ g^r,\ v^r,\ h_{j+1}^r, \ldots,\ h_l^r\Big).$$

The private key for ID can be also generated from a private key for $d_{\mathrm{ID}|j-1}$. Let

$$\begin{aligned} d_{\mathrm{ID}|j-1} &= \Big(g_2^\alpha \cdot (h_1^{\mathrm{I}_1} \cdots h_{j-1}^{\mathrm{I}_{j-1}} \cdot g_3)^{r'},\ g^{r'},\ v^{r'},\ h_j^{r'}, \ldots,\ h_l^{r'}\Big) \\ &= (a_0, a_1, a_2, b_j, \ldots, b_l\ ) \end{aligned}$$

be the private key for $\mathrm{ID}_{|j-1} = (\mathrm{I}_1, \ldots, \mathrm{I}_{j-1}) \in \mathbb{Z}_p^{j-1}$. To generate d_{ID}, pick a random $r^* \in \mathbb{Z}_p$ and output

$$d_{\mathrm{ID}} = \Big(a_0 \cdot b_j^{\mathrm{I}_j} \cdot (h_1^{\mathrm{I}_1} \cdots h_j^{\mathrm{I}_j} \cdot g_3)^{r^*},\ a_1 \cdot g^{r^*},\ a_2 \cdot v^{r^*},\ b_{j+1} \cdot h_{j+1}^{r^*}, \ldots,\ b_l \cdot h_l^{r^*}\Big).$$

Since $r = r' + r^*$, we see that this private key is a properly distributed private key for $\mathrm{ID} = (\mathrm{I}_1, \ldots, \mathrm{I}_j)$.

Encrypt(M, *params*, ID): To encrypt a message $M \in \mathbb{G}_1$ under a public key $\mathrm{ID} = (\mathrm{I}_1, \ldots, \mathrm{I}_j) \in \mathbb{Z}_p^j$,

1. Run the *SigKeyGen* to obtain a signing key SigK and a verification key VerK.
2. Pick a random $s \in \mathbb{Z}_p^*$ and compute

$$C = \Big(g^s,\ e(g_1, g_2)^s \cdot M,\ (h_1^{\mathrm{I}_1} \cdots h_j^{\mathrm{I}_j} \cdot v^{\mathsf{VerK}} \cdot g_3)^s\Big).$$

3. Output the ciphertext $\mathsf{CT} = (C, Sign_{\mathsf{SigK}}(C), \mathsf{VerK})$.

Decrypt(CT, *params*, d_{ID}): Consider an identity $\mathrm{ID} = (\mathrm{I}_1, \ldots, \mathrm{I}_j)$. To decrypt a ciphertext $\mathsf{CT} = (C, \sigma, \mathsf{VerK})$ using the private key $d_{\mathrm{ID}} = (a_0, a_1, a_2, b_{j+1}, \ldots, b_l)$,

1. Check that the signature σ on C is valid under the key VerK. If invalid, output $\perp$.
2. Otherwise, let $C = (C_1, C_2, C_3)$. Select a random $w \in \mathbb{Z}_p$ and compute

$$\widetilde{a}_0 = a_0 \cdot a_2^{\mathsf{VerK}} \cdot (h_1^{\mathrm{I}_1} \cdots h_j^{\mathrm{I}_j} \cdot v^{\mathsf{VerK}} \cdot g_3)^w, \qquad \widetilde{a}_1 = a_1 \cdot g^w.$$

3. Output $\big(e(C_1,\ \widetilde{a}_1)/e(C_3,\ \widetilde{a}_0)\big) \cdot C_2$.

Note that the pair $(\widetilde{a}_0,\ \widetilde{a}_1)$ is chosen from the following distribution

$$\left(g_2^{\alpha} \cdot (h_1^{\mathrm{I}_1} \cdots h_j^{\mathrm{I}_j} \cdot v^{\mathsf{VerK}} \cdot g_3)^{\widetilde{r}},\ \ g^{\widetilde{r}} \right)$$

where $\widetilde{r}$ is uniform in $\mathbb{Z}_p$. This distribution is independent of $\mathrm{ID} = (\mathrm{I}_1, \ldots, \mathrm{I}_j)$. Next, the correctness of decryption algorithm is checked as below:

$$\frac{e(C_1,\ \widetilde{a}_1)}{e(C_3,\ \widetilde{a}_0)} = \frac{e((h_1^{\mathrm{I}_1} \cdots h_j^{\mathrm{I}_j} \cdot v^{\mathsf{VerK}} \cdot g_3)^s,\ g^{\widetilde{r}})}{e(g^s,\ g_2^{\alpha} \cdot (h_1^{\mathrm{I}_1} \cdots h_j^{\mathrm{I}_j} \cdot v^{\mathsf{VerK}} \cdot g_3)^{\widetilde{r}})} = \frac{1}{e(g^s, g_2^{\alpha})} = \frac{1}{e(g_1, g_2)^s}.$$

4.2 Security

As opposed to the l-BDHE assumption for the IND-sID-CPA secure BBG scheme in [5], security of the IND-sID-CCA secure HIBE scheme above is based on the $(l+1)$-BDHE assumption.

Theorem 2. *Suppose that the decision $(t, l+1, \epsilon_1)$-BDHE assumption holds in $\mathbb{G}$ and the signature scheme is $(t, 1, \epsilon_2)$-strongly existentially unforgeable. Then the previous l-HIBE scheme is $(t', q_{\mathrm{ID}}, q_C, \epsilon)$-selective-ID, adaptive chosen ciphertext secure for arbitrary q_{ID}, q_C, and $t' < t - \Theta(\tau l q_{\mathrm{ID}})$, where $\epsilon_1 + \epsilon_2 \geq \epsilon$ and τ is the maximum time for an exponentiation in $\mathbb{G}$.*

Proof. Suppose there exists an adversary $\mathcal{A}$ which has advantage ϵ in attacking the l-level HIBE scheme. We want to build an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the decision $(l+1)$-BDHE problem in $\mathbb{G}$. For a generator $g \in \mathbb{G}$ and $\alpha \in \mathbb{Z}_p$, let $y_i = g^{\alpha^i} \in \mathbb{G}$. On input $(g, h, y_1, \ldots, y_{l+1}, y_{l+3}, \ldots, y_{2l+2}, T)$, $\mathcal{B}$ outputs 1 if $T = e(g,h)^{\alpha^{l+2}}$ and 0 otherwise. $\mathcal{B}$ works by interacting with $\mathcal{A}$ in a selective-ID game as follows:

Init: $\mathcal{A}$ outputs an identity $\mathrm{ID}^* = (\mathrm{I}_1^*, \ldots, \mathrm{I}_k^*) \in \mathbb{Z}_p^k$ of depth $k \leq l$ that it intends to attack.

Setup: To generate the system parameters, $\mathcal{B}$ first selects random $\rho, \eta \in \mathbb{Z}_p$ and sets $g_1 = y_1 = g^{\alpha}$, $g_2 = y_{l+1} \cdot g^{\rho}$, and $v = y_{l+1}^{\eta}$. Next, $\mathcal{B}$ runs $SigKeyGen$ algorithm to gain a signing key SigK^* and a verification key VerK^*. Next, $\mathcal{B}$ picks random $\gamma, \gamma_1, \ldots, \gamma_l$ in $\mathbb{Z}_p$, and sets $h_i = g^{\gamma_i} y_i$ for $i = 1, \ldots, l$ and $g_3 = g^{\gamma} \cdot v^{-\mathsf{VerK}^*} \cdot (h_1^{\mathrm{I}_1^*} \ldots, h_k^{\mathrm{I}_k^*})^{-1}$.
Then, it gives $\mathcal{A}$ the system parameters $params = (g, g_1, g_2, g_3, v, h_1, \ldots, h_l)$. The master key corresponding to these $params$ is $g_2^{\alpha} = y_{l+2} \cdot y_1^{\rho}$, which is unknown to $\mathcal{B}$.

Phase 1: $\mathcal{A}$ issues up to q_{ID} private key queries and q_C decryption queries. First, consider a query for the private key corresponding to $\mathrm{ID} = (\mathrm{I}_1, \ldots, \mathrm{I}_u) \in \mathbb{Z}_p^u$ where $u \leq l$. The only restriction is that ID is not a prefix of ID^*. We further distinguish two cases according to whether ID^* is a prefix of ID or not. First, consider the case ID^* is not a prefix of ID. Then there exists $j \in \{1, \ldots, k\}$ such that $\mathrm{I}_j \neq \mathrm{I}_j^*$. To respond to the query, $\mathcal{B}$ first derives a private key

for the identity $(\mathrm{I}_1, \ldots, \mathrm{I}_j)$ from which it constructs a private key for the requested identity $\mathrm{ID} = (\mathrm{I}_1, \ldots, \mathrm{I}_j, \ldots, \mathrm{I}_u)$.

$\mathcal{B}$ picks a random $s \in \mathbb{Z}_p$. Let $\widetilde{s} = s + \alpha^{(l+2-j)}/(\mathrm{I}_j^* - \mathrm{I}_j)$. Next, $\mathcal{B}$ generates the private key for $\mathrm{ID} = (\mathrm{I}_1, \ldots, \mathrm{I}_u)$ as

$$\left(g_2^{\alpha} \cdot (h_1^{\mathrm{I}_1} \cdots h_j^{\mathrm{I}_j} \cdot g_3)^{\widetilde{s}},\ g^{\widetilde{s}},\ v^{\widetilde{s}},\ h_{j+1}^{\widetilde{s}}, \ldots,\ h_l^{\widetilde{s}}\right)$$

which is a properly distributed private key for the identity $\mathrm{ID} = (\mathrm{I}_1, \ldots, \mathrm{I}_j)$. We show that $\mathcal{B}$ can compute all elements of this private key given the values that it knows. To generate the first component of the private key, observe that

$$\begin{aligned}(h_1^{\mathrm{I}_1} \cdots h_j^{\mathrm{I}_j} \cdot g_3)^{\widetilde{s}} &= (h_1^{\mathrm{I}_1} \cdots h_j^{\mathrm{I}_j} \cdot g^{\gamma} \cdot v^{-\mathsf{VerK}^*} \cdot h_1^{-\mathrm{I}_1^*} \cdots h_j^{-\mathrm{I}_j^*} \cdots h_k^{-\mathrm{I}_k^*})^{\widetilde{s}} \\ &= (g^{\gamma} \cdot v^{-\mathsf{VerK}^*} \cdot h_j^{\mathrm{I}_j - \mathrm{I}_j^*} \cdot h_{j+1}^{-\mathrm{I}_{j+1}^*} \cdots h_k^{-\mathrm{I}_k^*})^{\widetilde{s}} \\ &= h_j^{\widetilde{s} \cdot (\mathrm{I}_j - \mathrm{I}_j^*)} \cdot (g^{\gamma} \cdot v^{-\mathsf{VerK}^*} \cdot h_{j+1}^{-\mathrm{I}_{j+1}^*} \cdots h_k^{-\mathrm{I}_k^*})^{\widetilde{s}}.\end{aligned}$$

Note that the value $h_j^{\widetilde{s} \cdot (\mathrm{I}_j - \mathrm{I}_j^*)}$ in the above becomes $y_{l+2}^{-1} \cdot y_j^{s(\mathrm{I}_j - \mathrm{I}_j^*)} \cdot g^{\widetilde{s} \cdot \gamma_j \cdot (\mathrm{I}_j - \mathrm{I}_j^*)}$. Since $g_2^{\alpha} = y_{l+2} \cdot y_1^{\rho}$, the first component can be computed as

$$y_1^{\rho} \cdot y_j^{s(\mathrm{I}_j - \mathrm{I}_j^*)} \cdot g^{\widetilde{s} \cdot \gamma_j \cdot (\mathrm{I}_j - \mathrm{I}_j^*)} \cdot (g^{\gamma} \cdot v^{-\mathsf{VerK}^*} \cdot h_{j+1}^{-\mathrm{I}_{j+1}^*} \cdots h_k^{-\mathrm{I}_k^*})^{\widetilde{s}}$$

where the unknown term y_{l+2} is canceled out. The other terms $g^{\widetilde{s}}$, $v^{\widetilde{s}}$, and $h_i^{\widetilde{s}}$ for $i = j+1, \ldots, k$ are computable since $g^{\widetilde{s}} = g^s \cdot y_{l+2-j}^{1/(\mathrm{I}_j - \mathrm{I}_j^*)}$, $v^{\widetilde{s}} = v^s \cdot y_{2l+3-j}^{\eta/(\mathrm{I}_j - \mathrm{I}_j^*)}$, and $h_i^{\widetilde{s}} = g^{\gamma_i \cdot s} \cdot y_{l+2-j}^{\gamma_i/(\mathrm{I}_j^* - \mathrm{I}_j)} \cdot y_i^s \cdot y_{l+2-j+i}^{1/(\mathrm{I}_j^* - \mathrm{I}_j)}$ for $i = j+1, \ldots, k$. These values do not require knowledge of y_{l+2}. Similarly, the remaining elements $g^{\widetilde{s}}, h_{j+1}^{\widetilde{s}}, \ldots, h_l^{\widetilde{s}}$ can be computed since they do not involve the y_{l+2} term.

Second, consider the case when ID^* is a prefix of ID. Then it holds that $k+1 \leq u$. Let $\mathrm{ID} = (\mathrm{I}_1^*, \ldots, \mathrm{I}_k^*, \mathrm{I}_{k+1}, \ldots, \mathrm{I}_u)$. In this step, we can assume that there exists at least one $j \in \{k+1, \ldots, u\}$ such that $\mathrm{I}_j \neq 0$ in $\mathbb{Z}_p$. Otherwise, for all $j \in \{k+1, \ldots, u\}$, $\mathrm{ID} = (\mathrm{I}_1^*, \ldots, \mathrm{I}_k^*, 0, \ldots, 0)$. Then this private key for ID can be easily used to decrypt the challenge ciphertext. Let j be the smallest index such that $\mathrm{I}_j \neq 0$. $\mathcal{B}$ responds to the query by first computing a private key for $\mathrm{ID} = (\mathrm{I}_1^*, \ldots, \mathrm{I}_k^*, \mathrm{I}_{k+1}, \ldots, \mathrm{I}_j)$ from which it constructs a private key for the requested $\mathrm{ID} = (\mathrm{I}_1^*, \ldots, \mathrm{I}_k^*, \mathrm{I}_{k+1}, \ldots, \mathrm{I}_j, \ldots, \mathrm{I}_u)$. $\mathcal{B}$ selects a random $s \in \mathbb{Z}_p$. Let $\widetilde{s} = s - \alpha^{(l+2-j)}/\mathrm{I}_j$. Then $\mathcal{B}$ generates the private key for $\mathrm{ID} = (\mathrm{I}_1^*, \ldots, \mathrm{I}_k^*, \ldots, \mathrm{I}_j)$ as

$$\left(g_2^{\alpha} \cdot (h_1^{\mathrm{I}_1^*} \cdots h_k^{\mathrm{I}_k^*} \cdots h_j^{\mathrm{I}_j} \cdot g_3)^{\widetilde{s}},\ g^{\widetilde{s}},\ v^{\widetilde{s}},\ h_{j+1}^{\widetilde{s}}, \ldots,\ h_l^{\widetilde{s}}\right).$$

By the similar argument above, this private key has a proper distribution and is computable.

Next, $\mathcal{B}$ responds to decryption queries for $\mathrm{ID}^* = (\mathrm{I}_1^*, \ldots, \mathrm{I}_k^*)$ or any prefix of ID^*. Let $\mathrm{ID}' = (\mathrm{I}_1^*, \ldots, \mathrm{I}_j^*)$ where $j \leq k$ and let $(C, \sigma, \mathsf{VerK})$ be a decryption query for ID' where $C = (C_1, C_2, C_3)$. $\mathcal{B}$ does as follows:

1. Run $Verify$ to check the validity of the signature σ on C, using the verification key VerK. If the signature is invalid, $\mathcal{B}$ responds with $\perp$.
2. If $\mathsf{VerK} = \mathsf{VerK}^*$, $\mathcal{B}$ outputs a random bit $b \in \{0, 1\}$ and aborts the simulation.
3. Otherwise, $\mathcal{B}$ checks that the equality $e(h_1^{\mathrm{I}_1^*} \ldots h_j^{\mathrm{I}_j^*} \cdot v^{\mathsf{VerK}} \cdot g_3, C_1) \stackrel{?}{=} e(C_2, g)$. If it does not hold, $\mathcal{B}$ knows that (C_1, C_2) is not of the right form. Then, $\mathcal{B}$ outputs a random message $M \in \mathbb{G}_1$. Otherwise, for some (unknown) $s \in \mathbb{Z}_p$ such that $C_1 = g^s$, $\mathcal{B}$ has that $C_2 = (h_1^{\mathrm{I}_1^*} \ldots h_j^{\mathrm{I}_j^*} \cdot v^{\mathsf{VerK}} \cdot g_3)^s$. Plugging in the value of g_3, C_2 becomes

$$\begin{aligned} C_2 &= \left(h_1^{\mathrm{I}_1^*} \ldots h_j^{\mathrm{I}_j^*} \cdot v^{\mathsf{VerK}} \cdot g^{\gamma} \cdot v^{-\mathsf{VerK}^*} \cdot h_1^{-\mathrm{I}_1^*} \ldots h_k^{-\mathrm{I}_k^*}\right)^s \\ &= \left(v^{\mathsf{VerK}-\mathsf{VerK}^*} \cdot g^{\gamma} \cdot h_{j+1}^{-\mathrm{I}_{j+1}^*} \ldots h_k^{-\mathrm{I}_k^*}\right)^s \\ &= \left(y_{l+1}^{\eta(\mathsf{VerK}-\mathsf{VerK}^*)} \cdot g^{\gamma}\right)^s \cdot \left(h_{j+1}^{-\mathrm{I}_{j+1}^*} \ldots h_k^{-\mathrm{I}_k^*}\right)^s. \end{aligned}$$

$\mathcal{B}$ computes $\widetilde{a}_0 = y_1^{-\gamma/\eta(\mathsf{VerK}-\mathsf{VerK}^*)} \cdot C_2 \cdot (h_{j+2}^{-\mathrm{I}_{j+1}^*} \ldots h_{k+1}^{-\mathrm{I}_k^*})^{-1/\eta(\mathsf{VerK}-\mathsf{VerK}^*)}$ and $\widetilde{a}_1 = C_1 \cdot y_1^{-1/\eta(\mathsf{VerK}-\mathsf{VerK}^*)}$. Let $\widetilde{r} = s - \alpha/\eta(\mathsf{VerK} - \mathsf{VerK}^*)$. Then,

$$\begin{aligned} \widetilde{a}_0 &= y_1^{-\gamma/\eta(\mathsf{VerK}-\mathsf{VerK}^*)} \cdot \left(y_{l+1}^{\eta(\mathsf{VerK}-\mathsf{VerK}^*)} \cdot g^{\gamma}\right)^s \cdot \left(h_{j+1}^{-\mathrm{I}_{j+1}^*} \ldots h_k^{-\mathrm{I}_k^*}\right)^{\widetilde{r}} \\ &= y_{l+2} \cdot \left(y_{l+1}^{\eta(\mathsf{VerK}-\mathsf{VerK}^*)} \cdot g^{\gamma}\right)^{\widetilde{r}} \cdot \left(h_{j+1}^{-\mathrm{I}_{j+1}^*} \ldots h_k^{-\mathrm{I}_k^*}\right)^{\widetilde{r}} \\ &= y_{l+2} \cdot \left(v^{\mathsf{VerK}} \cdot g^{\gamma} \cdot v^{-\mathsf{VerK}^*} \cdot h_{j+1}^{-\mathrm{I}_{j+1}^*} \ldots h_k^{-\mathrm{I}_k^*}\right)^{\widetilde{r}}, \\ \widetilde{a}_1 &= g^s \cdot y_1^{-1/\eta(\mathsf{VerK}-\mathsf{VerK}^*)} = g^{\widetilde{r}}. \end{aligned}$$

Recall that the master-key is $y_{l+2} \cdot y_1^{\rho}$. For the re-randomization, $\mathcal{B}$ selects a random $r' \in \mathbb{Z}_p$ and computes $\widetilde{a}_0' = \widetilde{a}_0 \cdot y_1^{\rho} \cdot (v^{\mathsf{VerK}} \cdot g^{\gamma} \cdot v^{-\mathsf{VerK}^*} \cdot h_{j+1}^{-\mathrm{I}_{j+1}^*} \ldots h_k^{-\mathrm{I}_k^*})^{r'}$ and $\widetilde{a}_1' = \widetilde{a}_1 \cdot g^{r'}$. For some (unknown) $\widetilde{r}' = \widetilde{r} + r'$,

$$\begin{aligned} \widetilde{a}_0' &= y_{l+2} \cdot y_1^{\rho} \cdot \left(v^{\mathsf{VerK}} \cdot g^{\gamma} \cdot v^{-\mathsf{VerK}^*} \cdot h_{j+1}^{-\mathrm{I}_{j+1}^*} \ldots h_k^{-\mathrm{I}_k^*}\right)^{\widetilde{r}} \\ &= g_2^{\alpha} \cdot \left(h_1^{\mathrm{I}_1^*} \cdots h_j^{\mathrm{I}_j^*} \cdot v^{\mathsf{VerK}} \cdot g_3\right)^{\widetilde{r}'}, \\ \widetilde{a}_1' &= g^{\widetilde{r}} \cdot g^{r'} = g^{\widetilde{r}'}. \end{aligned}$$

$\mathcal{B}$ responds with $\left(e(C_1, \widetilde{a}_1')/e(C_3, \widetilde{a}_0')\right) \cdot C_2$. This response is identical to *Decrypt* algorithm in a real attack, because r' (and $\widetilde{r}'$) is uniform in $\mathbb{Z}_p$.

Challenge: $\mathcal{A}$ outputs two messages $M_0, M_1 \in \mathbb{G}_1$. To encrypt one of the two messages under the public key ID^*, $\mathcal{B}$ selects a random bit $b \in \{0,1\}$ and a random $t \in \mathbb{Z}_p$. $\mathcal{B}$ computes $C = (h^t,\ T^t \cdot e(y_1, h)^{t \cdot \rho} \cdot M_b,\ h^{t \cdot \gamma})$, where T and h are from the input tuple given to $\mathcal{B}$. Next, $\mathcal{B}$ gives the challenge ciphertext $\mathsf{CT} = (C, Sign_{\mathsf{SigK}^*}(C), \mathsf{VerK}^*)$ to $\mathcal{A}$. If $h = g^c$ for some (unknown) $c \in \mathbb{Z}_p$, $h^{t \cdot \gamma} = (h_1^{\mathrm{I}_1^*} \cdots h_k^{\mathrm{I}_k^*} \cdot v^{\mathsf{VerK}} \cdot g_3)^{t \cdot c}$. Define $\mu = t \cdot c \in \mathbb{Z}_p$. On the one hand, if $T = e(g,h)^{\alpha^{l+2}}$, we have that

$$C = \left(g^{\mu},\ e(g_1, g_2)^{\mu} \cdot M_b,\ (h_1^{\mathrm{I}_1^*} \cdots h_k^{\mathrm{I}_k^*} \cdot v^{\mathsf{VerK}^*} \cdot g_3)^{\mu}\right)$$

which is a valid encryption of M_b under the public key $\mathrm{ID}^* = (\mathrm{I}_1^*, \ldots, \mathrm{I}_k^*)$. On the other hand, when T is uniform and independent in $\mathbb{G}_1$, then C (and CT) is independent of b in the adversary's view.

Phase 2: $\mathcal{A}$ issues more private key and decryption queries. $\mathcal{B}$ responds as in Phase 1.

Guess : $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$. If $b = b'$ then $\mathcal{B}$ outputs 1, indicating $T = e(g,h)^{\alpha^{l+2}}$. Otherwise, it outputs 0, indicating $T \neq e(g,h)^{\alpha^{l+2}}$.

When T is random in $\mathbb{G}_1$ then $\Pr[\mathcal{B}(g, h, \overrightarrow{y}_{g,\alpha,l+1}, T) = 0] = 1/2$. Let Forge denote the event that $\mathcal{A}$ submits a valid ciphertext $\mathsf{CT} = (C, \sigma, \mathsf{VerK}^*)$ as a decryption query. In the case of Forge, $\mathcal{B}$ cannot reply to the decryption query and aborts the simulation. When $T = e(g,h)^{\alpha^{l+2}}$, $\mathcal{B}$ replied with a valid plaintext unless event Forge occurs. Then, $\mathcal{B}$ has

$$\left|\Pr[\mathcal{B}(g, h, \overrightarrow{y}_{g,\alpha,l+1}, T) = 0] - \frac{1}{2}\right| \geq \left|\Pr[b = b' \wedge \overline{\mathsf{Forge}}] - \frac{1}{2}\right| - \Pr[\mathsf{Forge}].$$

Since $\mathcal{B}$ provided $\mathcal{A}$ with perfect simulation when event Forge did not occur, $|\Pr[b = b' \wedge \overline{\mathsf{Forge}}] - 1/2| \geq \epsilon$. Also, note that $\Pr[\mathsf{Forge}]$ is negligible. This means that $\Pr[\mathsf{Forge}] < \epsilon_2$ since otherwise, $\mathcal{B}$ can construct a forger, which is contradiction to the one-time signature. Therefore,

$$\left|\Pr\left[\mathcal{B}(g, h, \overrightarrow{y}_{g,\alpha,l+1}, e(g,g)^{abc}) = 0\right] - \Pr\left[\mathcal{B}(g, h, \overrightarrow{y}_{g,\alpha,l+1}, T) = 0\right]\right| \geq \epsilon - \epsilon_2$$

This completes the proof of Theorem 1. □

5 Conclusion

We presented two HIBE schemes that are secure against chosen ciphertext attacks in the selective-ID model, based on the BB_1 and BBG schemes. We obtain chosen ciphertext security of the l-level HIBE schemes by directly applying the idea of the CHK transformation to the l-level BB_1 and BBG schemes. The resulting schemes are more compact than the ones derived from the known generic transformation for chosen ciphertext secure l-level HIBE scheme.

Moreover, our constructions imply that the CHK transformation could be applied to obtain chosen ciphertext security of concrete schemes with the BB_1 and BBG-like structures.

References

1. Abdalla, M., Bellare, M., Catalano, D., Kiltz, E., Kohno, T., Lange, T., Malone-Lee, J., Neven, G., Paillier, P., Shi, H.: Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extentions. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 205–222. Springer, Heidelberg (2005)
2. Attrapadung, N., Furukawa, J., Imai, H.: Forward-secure and searchable broadcast encryption with short ciphertexts and private keys. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 161–177. Springer, Heidelberg (2006)
3. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
4. Boneh, D., Boyen, X.: Efficient selective-ID secure identity based encryption without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004)
5. Boneh, D., Boyen, X., Goh, E.: Hierarchical identity based encryption with constant size ciphertext. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 440–456. Springer, Heidelberg (2005)
6. Boneh, D., Canetti, R., Halevi, S., Katz, J.: Chosen-ciphertext security from identity-based encryption. In: Journal (submission) Available from `http://crypto.stanford.edu/~dabo/pubs.html`
7. Boneh, D., Franklin, M.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
8. Boneh, D., Gentry, C., Waters, B.: Collusion resistant broadcast encryption with short ciphertexts and private keys. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 258–275. Springer, Heidelberg (2005)
9. Boneh, D., Katz, J.: Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. In: Menezes, A.J. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 87–103. Springer, Heidelberg (2005)
10. Boyen, X., Mei, Q., Waters, B.: Direct chosen ciphertext security from identity-based techniques. In: ACM Conference on Computer and Communications Security - CCS'05, pp. 320–329. ACM Press, New York (2005)
11. Boyen, X., Waters, B.: Anonymous Hierarchical Identity-Based Encryption (without random oracles). In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 290–307. Springer, Heidelberg (2006)
12. Canetti, C., Halevi, S., Katz, J.: A forward-secure public-key encryption scheme. In: Biham, E. (ed.) EUROCRPYT 2003. LNCS, vol. 2656, Springer, Heidelberg (2003)
13. Canetti, C., Halevi, S., Katz, J.: Chosen ciphertext security from identity-based encryption. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 207–222. Springer, Heidelberg (2004)
14. Dodis, Y., Fazio, N.: Public key broadcast encryption for stateless receivers. In: Feigenbaum, J. (ed.) DRM 2002. LNCS, vol. 2696, pp. 61–80. Springer, Heidelberg (2003)

15. Gentry, C.: Practical Identity-Based Encryption Without Random Oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 445–464. Springer, Heidelberg (2006)
16. Gentry, C., Silverberg, A.: Hierarchical ID-based cryptography. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 548–566. Springer, Heidelberg (2002)
17. Horwitz, J., Lynn, B.: Toward hierarchical identity-based encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 466–481. Springer, Heidelberg (2002)
18. Shamir, A.: Identity-based cryptosystems and signaure shcemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
19. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
20. Yao, D., Fazio, N., Dodis, Y., Lysyanskaya, A.: ID-based encryption for complex hierarchies with applications to forward security and broadcast encrypption. In: ACM Conference on Computer and Communications Security - CCS'04, pp. 354–363 (2004)

Certificate-Based Signature: Security Model and Efficient Construction*

Jiguo Li[1], Xinyi Huang[2], Yi Mu[2], Willy Susilo[2], and Qianhong Wu[2]

[1] College of Computer and Information Engineering
Hohai University, Nanjing, P.R. China, 210098
lijiguo@hhu.edu.cn
[2] Centre for Computer and Information Security Research
School of Computer Science & Software Engineering
University of Wollongong, Australia
{xh068,ymu,wsusilo,qhw}@uow.edu.au

Abstract. In Eurocrypt 2003, Gentry introduced the notion of certificate-based encryption. The merit of certificate-based encryption lies in the following features: (1) providing more efficient public-key infrastructure (PKI) that requires less infrastructure, (2) solving the certificate revocation problem, and (3) eliminating third-party queries in the traditional PKI. In addition, it also solves the inherent key escrow problem in the identity-based cryptography. In this paper, we first introduce a new attack called the "Key Replacement Attack" in the certificate-based system and refine the security model of certificate-based signature. We show that the certificate-based signature scheme presented by Kang, Park and Hahn in CT-RSA 2004 is insecure against key replacement attacks. We then propose a new certificate-based signature scheme, which is shown to be existentially unforgeable against adaptive chosen message attacks under the computational Diffie-Hellman assumption in the random oracle model. Compared with the certificate-based signature scheme in CT-RSA 2004, our scheme enjoys *shorter* signature length and less operation cost, and hence, our scheme outperforms the existing schemes in the literature.

Keywords: Certificate-based signature, Key replacement attack, PKI.

1 Introduction

In traditional public key signatures (PKS), the public key of a signer is essentially a random string selected from a given set. Therefore, it is infeasible to prove that a party is indeed the signer for a given signature. This problem was solved by assuming the existence of a trusted third party (or often referred to as a Certification Authority - CA) who can issue (sign) public key certificates which

* The work is supported by the National Natural Science Foundation of China (No. 60673070), Natural Science Foundation of Jiangsu Province (No. BK2006217), and the Project of Jiangsu Province Police Ministry (No. 200503002).

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 110–125, 2007.

provide an unforgeable and trusted link between a public key and the identity of a signer. This kind of certificate systems are referred to as the Public Key Infrastructure (PKI). "Third-party query" is considered as a problem in the traditional PKI. Namely, before verifying a signature using the signer's public key, a verifier must obtain the signer's certification status; hence in general he has to make a query on the signer's certificate status to the CA. The verifier must verify the certificate first. If authorization of the CA about the signer's public key is valid, a verifier can then verify the signed message with the given public key from the signer. Therefore, from the verifier's point of view, two verification operations are required. This has been regarded as a drawback due to additional computation time and storage. The apparent need for this infrastructure is often cited as a reason for the widespread use of public-key cryptography. To simplify key management procedures of conventional PKIs, Shamir [1] introduced the concept of Identity-Based Cryptography (IBC) in 1984, which sought to reduce the requirement on the infrastructure by using user's identity as public key. In his seminal paper, Shamir also proposed an identity-based signature scheme. The first practical provably secure Identity-Based Encryption (IBE) scheme [2] was proposed by Boneh and Franklin in 2001. With this approach, certification becomes implicit; that is, the sender of a message does not need to check whether the user is certified or not. Instead, prior to decryption, the receiver must identify himself to a trusted third party called a Private Key Generator (PKG), who will generate and send his private key via an authentication and secure channel. The main practical benefit of IBC lies in greatly reduction of need for public key certification. The PKG can generate the secret keys of all its users, so *private key escrow* becomes an inherent problem in IBC. Moreover, secret keys must be sent over secure channels, which makes secret key distribution a daunting task [9].

To fill the gap between traditional cryptography and identity-based cryptography, Al-Riyami and Paterson proposed a new paradigm called certificateless public key cryptography (CL-PKC) [3]. In CL-PKC, KGC is involved in the process of issuing a partial secret key for each user. The user independently generates a public/private key pair and performs some cryptographic operations in such a way that they can only be carried out when both the partial secret key and the private key are known. Knowing only one of them will not enable anyone to impersonate the user. Therefore, CL-PKC not only solves the key escrow problem, but also eliminates the use of certificates as in traditional digital signature schemes. Due to the lack of public key authentication (certificate), it is important to assume that an adversary in the certificateless system can replace the user's public key with a false key of its choice, which is also known as *key replacement attack*. Cryptographic protocols in certificateless system are easily suffered from this kind of attack. For example, the first certificateless-based signature scheme [3] is not secure against the key replacement attack [5,6]. This problem was later fixed. We will not go into the detail, since it is out of the scope of this paper. Please refer to [5,6] for the detail of the key replacement attack in certificateless system.

In Eurocrypt 2003, Gentry [9] introduced the notion of certificate-based encryption (CBE), which combines public-key encryption (PKE) and IBE while preserving their features. In PKE, each user generates its own public-key/secret-key pair and requests a certificate from the CA. In CBE, the CA generates the certificate as in a conventional PKI system. This certificate has all of the functionalities of a conventional PKI certificate. In addition, it can also be used as a decryption key. This additional functionality gives us an implicit certification so that the sender can doubly encrypt his message so that the recipient can decrypt it using his private key along with an up-to-date certificate from his CA. The feature of implicit certification allows us to eliminate third-party queries for the certificate status. There is no key escrow problem in CBE (since the CA does not know the personal secret keys of users), and there is no secret key distribution problem (since the CA's certificate need not be kept secret). But a CBE scheme is inefficient when a CA has a large number of users and performs frequent certificate updates. Gentry suggests to use *subset covers* to overcome inefficiency. Gentry also demonstrated how certificate-based encryption could be used to construct an efficient PKI [10] requiring less infrastructure than previous proposals, including Micali's Novomodo [8], Naor-Nissim [12,13] and Aiello-Lodha-Ostrovsky [14]. Yum and Lee [15] revisited the definitions and security notions of certificate-less encryption and certificate-based encryption. They provided a formal equivalence theorem among identity-based encryption, certificate-less encryption and certificate-based encryption. Galindo *et al.* [18] pointed out that a dishonest authority could break the security of the three generic constructions of CBE and CL-PKE schemes given in [15,16]. These constructions were inherently flawed due to a naive use of double encryption as highlighted in [17]. Al-Riyami and Paterson [4] gave an analysis of Gentry's CBE concept and repaired a number of problems with the original definition and security model for CBE. They also provided a generic conversion showing that a secure CBE scheme could be constructed from any secure CL-PKE scheme and claimed that the derived CBE scheme was secure and even more efficient than the original scheme of Gentry. Kang and Park [11] pointed out that their conversion was incorrect due to the flaw in their security proof. This implies that the concrete CBE scheme by Al-Riyami and Paterson is therefore invalid. In parallel to CBE, Kang, Park and Hahn [7] proposed the security notion of certificate-based signature that follows the idea of CBE presented by Gentry [9].

Our Contribution

We first introduce a new attack called the "Key Replacement Attack" into the certificate-based system and refine the security model of certificate-based signature. We show that one of the certificate-based signature schemes (CBSm scheme) presented by Kang, Park and Hahn [7] is insecure against the key replacement attack. Furthermore, we construct a new certificate-based signature scheme, which is existentially unforgeable against adaptive chosen message attacks under the computational Diffie-Hellman assumption in the random oracle

model and provide a security proof of the proposed scheme. Compared with the other secure certificate-based signature scheme (CBSa scheme) [7] in CT-RSA 2004, our scheme enjoys shorter signature length and less operation cost.

Organization of the Paper
In the next section, we review some preliminaries required in this paper. We describe security and adversarial model in Section 3. In Section 4, we propose a new certificate-based signature scheme. We provide the security proof in Section 5. In Section 6, we presents a discussion on computation efficiency. Finally, we conclude the paper in Section 7.

2 Models of the Certificate-Based Signature

In this section, we review some background knowledge including the bilinear pairing and computational Diffie-Hellman (CDH) problem, and refine the security model of certificate-based signature schemes.

2.1 Bilinear Pairing

Let $\mathbb{G}_1$ denote an additive group of prime order p and $\mathbb{G}_2$ be a multiplicative group of the same order. Let P be a generator of $\mathbb{G}_1$ and $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be a bilinear mapping with the following properties:

- The map e is bilinear: $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in \mathbb{G}_1, a, b \in \mathbb{Z}_p$.
- The map e is non-degenerate: $e(P, P) \neq 1 \in \mathbb{G}_2$.
- The map e is efficiently computable.

Definition 1. CDH problem in $\mathbb{G}_1$. *Given* (P, aP, bP), *where* $a, b \in \mathbb{Z}_p^*$, *compute* abP.

The success probability of any probabilistic polynomial-time algorithm $\mathcal{A}$ in solving the CDH problem in $\mathbb{G}_1$ is defined to be

$$\mathtt{Succ}_{\mathcal{A},\mathbb{G}_1}^{CDH} = Pr[\mathcal{A}(P, aP, bP) = abP : a, b \in \mathbb{Z}_p^*].$$

The CDH assumption states that for every probabilistic polynomial-time algorithm $\mathcal{A}$, $\mathtt{Succ}_{\mathcal{A},\mathbb{G}_1}^{CDH}$ is negligible.

2.2 Outline of the Certificate-Based Signature

A certificate-based signature scheme consists of the following five algorithms:

1. **Setup:** This algorithm takes as input a security parameter 1^k and returns the certifier's master secret key msk and master public key mpk. It also outputs a public parameter params which is shared in the system.
2. **UserKeyGen:** This algorithm takes as input the master public key mpk and system parameter params. It outputs a user ID's secret/pubic key pair $(SK_{ID}, PK_{ID}) \in \mathcal{SK} \times \mathcal{PK}$. Here, $\mathcal{SK}$ denotes the set of the valid secret key values and $\mathcal{PK}$ denotes the set of the valid public key values.

3. **CertGen:** This algorithm takes as input the maser secret key msk, system parameter params, the identity ID of a user and its public key PK_{ID}. It outputs a certificate $Cert_{ID}$.
4. **Sign:** This algorithm takes as input a message m to be signed, user ID's certificate $Cert_{ID}$ and its secret key SK_{ID}. It outputs a signature σ of m.
5. **Verify:** This algorithm takes as input a message/signature pair (m, σ), user ID's public key PK_{ID}, certifier's public key mpk and system parameter params. It outputs $true$ if (m, σ) is valid. Otherwise, outputs $false$.

Remark. Generally, algorithms (**Setup**, **CertGen**) are run by the certifier. The user himself performs **UserKeyGen** and **Sign**. Additionally, when a user ID requests a certificate of his public key PK_{ID}, he must prove to the certifier of his possession of SK_{ID}. This can be done as the same way in traditional public key system.

2.3 Key Replacement Attack in Certificate-Based System

Before we define the security of certificate-based signature, we first introduce a new attack called "Key Replacement Attack" into the certificate-based system. It refers to the attack that an adversary can replace the target user ID's public key with PK'_{ID} which is chosen by himself. He can also dupe any other third party to encrypt messages using ID's false public key PK'_{ID} or verify signatures with PK'_{ID}.

In a public key system, when one wants to encrypt a message and sends the ciphertext to ID (or check a signature whether it is signed by ID), he must obtain the public key PK_{ID} of ID. Instead of performing the encrypt (or verify) algorithm using PK_{ID} directly, he must check whether PK_{ID} is the genuine public key generated by ID. In traditional public key system, the correctness of PK_{ID} and ID is guaranteed by the certificate. One can check the validity of the certificate which shows the correctness of PK_{ID}. Therefore, a potential adversary can not let others believe a false user/public key pair (ID, PK_{ID}), except that he can forge a certificate of the above pair. However, in the certificateless system [3] where there is no certificate, a potential adversary can replace ID's public key with any value chosen by himself. A cryptographic scheme is required to be secure against this "Key Replacement Attack". That is, even an adversary can replace this user's public key, he can not impersonate a user (say, decrypts a ciphertext intended to this user, generates a valid signature of this user, etc.).

It seems that "Key Replacement Attack" does not exist in the certificate-based system due to the use of certificates. However, as we will explain later, this attack does exist. In a certificate-based system, the certifier does issue the certificate of a user's public key PK_{ID}. However, only the user ID needs to check the validity of its certificate and other users in the system do not need. This is one of the advantages of certificated-based scheme compared with traditional public key where each user must check the validity of all the others' certificates. That's why the certificate-based system is more efficient.

In order to explain the "Key Replacement Attack" in detail, we present a concrete key replacement attack algorithm of CBSm scheme [7].

2.4 A Concrete Key Replacement Attack

In order to facilitate analysis, we review the CBSm scheme in [7] . Given a security parameter 1^k, the algorithm works as follows:

CBS.Setup: Let $\mathbb{G}_1, \mathbb{G}_2$ be groups of a prime order p in which there exists a bilinear pairing map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$. The CA picks an arbitrary generator $P \in \mathbb{G}_1$ and a random secret $s_C \in \mathbb{Z}_q$, and sets $PK_C = s_C P$. Chooses cryptographic hash functions $H_1 : \{0,1\}^* \to \mathbb{G}_1$, and $H_3 : \{0,1\}^* \times \mathbb{G}_1 \to \mathbb{Z}_q$. The system parameters are $\mathsf{params} = (\mathbb{G}_1, \mathbb{G}_2, e, p, P, PK_C, H_1, H_3)$ and the CA's master secret key is $SK_C = s_C \in \mathbb{Z}_q$. The CA uses its parameters and its secret to issue certificates. Alice computers a secret and public key pair as $(SK_A, PK_A) = (s_A, s_A P)$ according to the parameters issued by the CA.

CBS.Cert: Alice obtains a certificate from her CA as follows.

1. Alice sends $Alicesinfo$ to the CA, which includes his public key $s_A P$ and any necessary additional identifying information, such as her name.
2. The CA verifies Alice's information.
3. If satisfied, the CA computes $P_A = H_1(i, PK_C, PK_A, Alicesinfo) \in \mathbb{G}_1$ in period i.
4. The CA then computes $Cert_A = s_C P_A$ and sends this certificate to Alice. Then Alice signs $Alicesinfo$, producing $s_A P_A$, and computes $S_A = s_C P_A + s_A P_A = Cert_A + s_A P_A$. Alice will use this multisignature as her temporary signing key.

CBSm.Sign: To sign $m \in \{0,1\}^*$ using $Alicesinfo$, picks a random $r \in \mathbb{Z}_q$ and outputs a signature $\sigma = (U, V)$ where $U = rP_A, h = H_3(m, U)$ and $V = (r + h)S_A = (r + h)(s_C + s_A)P_A$.

CBSm.Vrfy: To verify a signature $\sigma = (U, V)$ of a message m, checks whether $e(s_C P + s_A P, U + hP_A) = e(P, V)$, where $h = H_3(m, U)$.

We will show that the scheme above is insecure against key replacement attack. The attack method is as follows.

The adversary first chooses a random value $r \in \mathbb{Z}_q^*$, computes $PK'_A = rP - PK_C$ and replaces Alice's public key PK_A with PK'_A. Then the adversary chooses a random value $U \in \mathbb{G}_1$, computes $h = H_3(m, U), P_A = H_1(i, PK_C, PK'_A, Alicesinfo)$. Finally, the adversary computes $V = rU + rhP_A$. Thus, $\sigma = (U, V)$ is a valid certificate-based signature. This is because signature $\sigma = (U, V)$ satisfies the following verification equation.

$$\begin{aligned} & e(PK_C + PK'_A, U + hP_A) \\ = \; & e(PK_C + rP - PK_C, U + hP_A) \\ = \; & e(P, rU + rhP_A) \\ = \; & e(P, V). \end{aligned}$$

Remark: The attack we described above is also mentioned in [7]. In order to ensure a certificate-based signature scheme to be secure against the above attack, the authors in [7] suggested that when A requests the certificate, the certifier

must verify whether the user A knows the corresponding secret key of the registered public key PK_A. However, as we can see, this method does not work. The reason is that an adversary can successfully generate a valid signature without the certificate of the false public key PK'_A.

3 Adversaries and Oracles

Roughly speaking, the security of a certificated-based signature requires that one can generate a valid signature under the public key PK_{ID} if and only if he has $Cert_{ID}$ and SK_{ID}. Only with one of those, one cannot generate a valid signature. The security of certificate-based signatures is defined by the game between the challenger $\mathcal{C}$ and the adversary. Before we start to define its security, we will describe some oracles which are accessible to the adversary.

3.1 Adversary Oracles

We first define the following three oracles that can be accessed by the adversary in the certificate-based system.

UserKeyGen: On a **UserKeyGen** query ID, if ID has already been created, nothing is to be carried out by $\mathcal{C}$. Otherwise, $\mathcal{C}$ runs the algorithm **UserKeyGen** and obtains the a secret/public key pair (SK_{ID}, PK_{ID}). Then it adds (ID, SK_{ID}, PK_{ID}) to the list L. In this case, ID is said to be *created*. In both cases, PK_{ID} is returned.

Corruption: On a **Corruption** query ID where ID denotes the identity which has been created, $\mathcal{C}$ checks the list L and returns the secret key SK_{ID}.

Sign: On a **Sign** query (ID, m) where ID denotes the identity which has been created, $\mathcal{C}$ runs the algorithm **Sign** and returns the signature σ.

3.2 Security Against the Key Replacement Adversary $\mathcal{A}_I$

In this section, we will consider the first kind of Type I adversary $\mathcal{A}_I$. Informally, we want to capture the attack scenarios where an adversary wants to forge a valid signature under the public key PK_{ID^*} whose certificate is not known to him. The pubic key PK_{ID^*} here might be the genuine one generated by the user ID^* or the fake one chosen by the adversary.

1. $\mathcal{A}_I$ can obtain some message/signature pairs (m_i, σ_i) which are generated by the target user ID.
2. $\mathcal{A}_I$ can replace the target user ID^* public key with PK_{ID^*} which is chosen by himself. He can also dupe any other third party to verify user ID^*'s signatures using the false public key PK_{ID^*}.
3. If $\mathcal{A}_I$ has replaced the user ID^* public key, he cannot obtain the certificate of the false public key from the certifier.

The security of a certificated-based signature scheme against a key replacement attack is defined by the following game between $\mathcal{A}_I$ and the challenger $\mathcal{C}$.

Setup: The challenger $\mathcal{C}$ runs the algorithm **Setup** and returns (mpk, params) to $\mathcal{A}_I$.

Query: In polynomial time t, $\mathcal{A}_I$ can adaptively submit queries to the oracles defined in Section 3.1. Additionally, $\mathcal{A}_I$ can also submit the **CertGen** query:

- **CertGen:** On $\mathcal{A}_I$'s **CertGen** query ID where ID denotes the identity which has been created, $\mathcal{C}$ runs the algorithm **CertGen** and returns the certificate $Cert_{ID}$ to $\mathcal{A}_I$. Note that $Cert_{ID}$ is the certificate of the pair (ID, PK_{ID}) where PK_{ID} is the public key returned from the oracle **UserKeyGen**.

Forge: At last, $\mathcal{A}_I$ outputs a forgery $(m^*, \sigma^*, ID^*, PK^*_{ID})$. We say $\mathcal{A}_I$ wins if

- σ^* is a valid signature on the message m^* under the public key PK_{ID^*} and the system's master public key mpk. Here, PK_{ID^*} is chosen by $\mathcal{A}_I$ and might not be the one returned from the oracle **UserKeyGen**.
- ID^* has never been submitted as one of **CertGen** queries.
- (ID^*, m^*) has never been submitted as one of **Sign** queries.

The success probability of adaptively chosen message and chosen identity adversary $\mathcal{A}_I$ wins the above games is defined as $Succ^{cma,cida}_{\mathcal{A}_I}$.

Definition 2. *We say a certificated-based signature scheme is secure against a (t, q) chosen message and chosen identity adversary $\mathcal{A}_I$, if $\mathcal{A}_I$ runs in polynomial time t, makes at most q queries and $Succ^{cma,cida}_{\mathcal{A}_I}$ is negligible.*

Compared with the security model defined in [7], we allow the $\mathcal{A}_I$ to replace the target user's public key with any value chosen by him which captures the essence of the adversaries in the real certificate-based system. However, $\mathcal{A}_I$ cannot obtain the certificate of the target user's public key. In addition, we allow $\mathcal{A}_I$ to corrupt any user in the system which is in order to reflect the malicious user who tries to only use his own secret key (without the knowledge of certificate) to generate valid signatures.

3.3 Security Against the Certifier $\mathcal{A}_{II}$

In this section, we will consider the second kind of Type II adversary $\mathcal{A}_{II}$. Informally, we want to capture the attack scenarios where the certifier wants to generate a valid signature under the public key PK_{ID^*} without the knowledge of the corresponding secret key.

1. $\mathcal{A}_{II}$ has the knowledge of the certifier's secret key msk.
2. $\mathcal{A}_{II}$ can obtain some message/signature pairs (m_i, σ_i) which are generated by the target user ID.
3. $\mathcal{A}_{II}$ cannot replace any user's public key.

The security of a certificated-based signature scheme against the certifier $\mathcal{A}_{II}$ attack is defined by the following game between $\mathcal{A}_{II}$ and the challenger $\mathcal{C}$.

Setup: The challenger $\mathcal{C}$ runs the algorithm **Setup** and returns $(mpk, msk, \mathsf{params})$ to $\mathcal{A}_{II}$.

Query: In polynomial time t, $\mathcal{A}_{II}$ can adaptively submit queries to the oracles defined in Section 3.1.

Forge: At last, $\mathcal{A}_{II}$ outputs a forgery (m^*, σ^*, ID^*). We say $\mathcal{A}_{II}$ wins if

- σ^* is a valid signature on the message m^* under the public key PK_{ID^*} and the system's master public key mpk where PK_{ID^*} is the public key output from the **UserKeyGen** query ID^* .
- (ID^*, m^*) has never been submitted as one of **Sign** queries.

The success probability of adaptively chosen message and chosen identity adversary $\mathcal{A}_{II}$ wins the above games is defined as $Succ^{cma,cida}_{\mathcal{A}_{II}}$.

Definition 3. *We say a certificated-based signature scheme is secure against a (t, q) chosen message and chosen identity adversary $\mathcal{A}_{II}$, if $\mathcal{A}_{II}$ runs in polynomial time t, makes at most q queries and $Succ^{cma,cida}_{\mathcal{A}_{II}}$ is negligible.*

4 Concrete Scheme

Our scheme consists of the following concrete algorithms:

1. **Setup:** Given a security parameter 1^k, the algorithm works as follows:
   (a) Let $\mathbb{G}_1, \mathbb{G}_2$ be groups of a prime order p in which there exists a bilinear pairing map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$.
   (b) Select a random number $s \in \mathbb{Z}_p^*$ as the master secret key msk, choose an arbitrary generator of $P \in \mathbb{G}_1$ and compute the master public key $mpk = sP$.
   (c) Choose three cryptographic hash functions $H_0 : \{0,1\}^* \times \mathbb{G}_1 \to \mathbb{G}_1$, $H_1 : \{0,1\}^* \times \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_1$ and $H_2 : \{0,1\}^* \times \{0,1\}^* \times \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_1$. The system parameters are $\mathsf{params} =< \mathbb{G}_1, \mathbb{G}_2, e, p, P, mpk, H_0, H_1, H_2 >$.
2. **UserKeyGen:** Given params, select a random number $s_{ID} \in \mathbb{Z}_q^*$ as the user secret key and compute the user public key $PK_{ID} = s_{ID}P \in \mathbb{G}_1$.
3. **CertGen:** Given params, master secret key s, user public key PK_{ID} and user identity $ID \in \{0,1\}^*$, compute $Q_{ID} = H_0(ID \| PK_{ID}) \in \mathbb{G}_1$ and output a certificate $Cert_{ID} = sQ_{ID} \in \mathbb{G}_1$.
4. **Sign:** On input params, ID, user secret key s_{ID}, user certificate $Cert_{ID}$ and message $m \in \{0,1\}^*$, the algorithm works as follows:
   (a) Choose a random number $r \in \mathbb{Z}_p^*$ and compute $U = rP$.
   (b) Compute $W_1 = H_1(m \| U \| PK_{ID})$, $W_2 = H_2(m \| ID \| U \| PK_{ID})$.
   (c) Compute $V = Cert_{ID} + s_{ID}W_1 + rW_2$.
   (d) Set $\sigma = (U, V)$ as the signature of m.
5. **Verify:** Given the a message/signature pair $(m, \sigma = (U, V))$, the system parameter params and the public key PK_{ID}, this algorithm works as follows:
   (a) Compute $Q_{ID} = H_0(ID \| PK_{ID}) \in \mathbb{G}_1$, $W_1' = H_1(m \| U \| PK_{ID})$, $W_2' = H_2(m \| ID \| U \| PK_{ID})$.

(b) Check the equation $e(V,P) \stackrel{?}{=} e(mpk, Q_{ID})e(W_1', PK_{ID})e(W_2', U)$. If the equality holds, outputs *true*. Otherwise, reject.

Correctness
If σ is a genuine signature generated from algorithm **Sign**, then $W_1' = W_1$ and $W_2' = W_2$. Therefore,

$$\begin{aligned} & e(mpk, Q_{ID})e(W_1', PK_{ID})e(W_2', U) \\ = & e(mpk, Q_{ID})e(W_1, PK_{ID})e(W_2, U) \\ = & e(sP, H_0(ID, PK_{ID}))e(W_1, s_{ID}P)e(W_2, rP) \\ = & e(sH_0(ID\|PK_{ID}) + s_{ID}W_1 + rW_2, P) = e(V,P). \end{aligned}$$

5 Security Analysis

Theorem 1. *If there is a (t,q) Type I adaptively chosen message and chosen identity adversary $\mathcal{A}_I$ which can submit additional q_R queries to random oracles and win the game defined in Section 3.2 with probability $Succ_{\mathcal{A}_I}^{cma,cida}$, then there exists another algorithm $\mathcal{B}$ which can solve a random instance of Computational Diffie-Hellman problem in polynomial time with success probability $Succ_{\mathcal{B},\mathbb{G}_1}^{CDH} = \frac{1}{q+1}(1-\frac{1}{q+1})^q Succ_{\mathcal{A}_I}^{cma,cida}$.*

Proof. See Appendix A.

Theorem 2. *If there is a (t,q) Type II adaptively chosen message and chosen identity adversary $\mathcal{A}_{II}$ which can submit additional q_R queries to random oracles and win the game defined in Section 3.3 with probability $Succ_{\mathcal{A}_I I}^{cma,cida}$, then there exists another algorithm $\mathcal{B}$ which can solve a random instance of Computational Diffie-Hellman problem in polynomial time with success probability $Succ_{\mathcal{B},\mathbb{G}_1}^{CDH} \geq \frac{1}{q'}(1-\frac{1}{q'})^q Succ_{\mathcal{A}_{II}}^{cma,cida}$, where $1 \neq q' \leq q$ denotes the number of queries submitted to the oracle **UserKeyGen**.*

Proof. See Appendix B.

6 Efficiency Comparison

In this section, we make a comparison between our proposed scheme and CBSa scheme in [7]. According to the analysis given in Section 2.4, the CBSm scheme in [7] is not secure, and therefore we omit it in the comparison. The following notations will be used in the comparison:

$\|\mathbb{G}_1\|$	bit length of an element in $\mathbb{G}_1$	E	exponentiation in $\mathbb{G}_1$
BA	bilinear pairing operation	PA	point addition in $\mathbb{G}_1$

The following table shows the comparison of our scheme and the CBSa scheme proposed in [7].

Scheme	Signature Length	Signature Generation	Verification
CBSa in [7]	$3\|\mathbb{G}_1\|$	$3E$	$2E{+}2PA{+}3BA$
Our scheme	$2\|\mathbb{G}_1\|$	$2E{+}2PA$	$3BA$

As shown in the table, our scheme enjoys shorter signature length under the same system parameters. The signature of our scheme only consists of 3 elements in $\mathbb{G}_1$ and is about 170 bits shorter than the CBSa scheme in [7] when some suitable elliptic curve is used as the underlying building block. Meanwhile, our scheme also requires less operation cost than CBSa scheme under the assumption that the pairing $e(mpk, Q_{ID})$ in verification algorithm can be pre-computed.

7 Conclusion

In this paper, we first introduced a new attack called "Key Replacement Attack" into the certificate-based system and refine the security model of certificate-based signature. Our analysis showed that one of the certificate-based signature schemes proposed by Kang, Park and Hahn is insecure against the key replacement attack. Furthermore, we constructed a new certificate-based signature scheme. Our proposal is proven existentially unforgeable against adaptive chosen message attacks in the random oracle model. The security relies merely well known computational Diffie-Hellman assumption. Compared with the other secure certificate-based signature scheme [7] in CT-RSA 2004, our scheme enjoys shorter signature length and less operation cost. Therefore, our scheme outperforms the existing schemes in the literature.

References

1. Shamir, A.: Identity-based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
2. Boneh, D., Franklin, M.: Identity-based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 229–231. Springer, Heidelberg (2001)
3. Al-Riyami, S.S., Paterson, K.G.: Certificateless Public Key Cryptography. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 452–473. Springer, Heidelberg (2003)
4. Al-Riyami, S.S., Paterson, K.G.: CBE from CL-PKE: A Generic Construction and Efficient Schemes. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 398–415. Springer, Heidelberg (2005)
5. Huang, X., Susilo, W., Mu, Y., Zhang, F.T.: On the Security of Certificateless Signature Schemes from Asiacrypt 2003. In: Desmedt, Y.G., Wang, H., Mu, Y., Li, Y. (eds.) CANS 2005. LNCS, vol. 3810, pp. 13–25. Springer, Heidelberg (2005)

6. Hu, B.C., Wong, D.S., Zhang, Z.F., Deng, X.T.: Key Replacement Attack Against a Generic Construction of Certificateless Signature. In: Batten, L.M., Safavi-Naini, R. (eds.) ACISP 2006. LNCS, vol. 4058, pp. 235–246. Springer, Heidelberg (2006)
7. Kang, B.G., Park, J.H., Hahn, S.G.: A Certificate-based Signature Scheme. In: Okamoto, T. (ed.) CT-RSA 2004. LNCS, vol. 2964, pp. 99–111. Springer, Heidelberg (2004)
8. Micali, S.: Novomodo: Scalable Certificate Validation and Simplified PKI Management. In: Proc. of 1st Annual PKI Research Workshop (2002) available at `http://www.cs.dartmouth.edu/pki02/`
9. Gentry, C.: Certificate-based Encryption and the Certificate Revocation Problem. In: Biham, E. (ed.) EUROCRPYT 2003. LNCS, vol. 2656, pp. 272–293. Springer, Heidelberg (2003)
10. Guttman, P.: PKI: Its not Dead, Just Resting. IEEE Computer 35, 41–49 (2002), Extended version available at `www.cs.auckland.ac.nz/ pgut001/ pubs/notdead.pdf`
11. Kang, B.G., Park, J.H.: Is It Possible to Have CBE from CL-PKE? Cryptology ePrint Archive, Report 2005/431
12. Naor, M., Nissim, K.: Certificate Revocation and Certificate Update. In: Proc. of 7th Annual USENIX Security Symposium (1998) available at `http://www.wisdom.weizmann.ac.il/kobbi/papers.html`
13. Naor, D., Naor, M., Lotspiech, J.: Revocation and Tracing Schemes for Stateless Receivers. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 41–62. Springer, Heidelberg (2001)
14. Aiello, W., Lodha, S., Ostrovsky, R.: Fast Digital Identity Revocation. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 137–152. Springer, Heidelberg (1998)
15. Yum, D.H., Lee, P.J.: Identity-based Cryptography in Public Key Management. In: Katsikas, S.K., Gritzalis, S., Lopez, J. (eds.) EuroPKI 2004. LNCS, vol. 3093, pp. 71–84. Springer, Heidelberg (2004)
16. Yum, D.H., Lee, P.J.: Generic Construction of Certificateless Encryption. In: Laganà, A., Gavrilova, M., Kumar, V., Mun, Y., Tan, C.J.K., Gervasi, O. (eds.) ICCSA 2004. LNCS, vol. 3043, pp. 802–811. Springer, Heidelberg (2004)
17. Dodis, Y., Katz, J.: Chosen-Ciphertext Security of Multiple Encryption. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 188–209. Springer, Heidelberg (2005)
18. Galindo, D., Morillo, P., Ráfols, C.: Breaking Yum and Lee Generic Constructions of Certificate-Less and Certificate-Based Encryption Schemes. In: Atzeni, A.S., Lioy, A. (eds.) EuroPKI 2006. LNCS, vol. 4043, pp. 81–91. Springer, Heidelberg (2006)

A Proof of Theorem 1

Proof. Let P be the generator of $\mathbb{G}_1$. Algorithm $\mathcal{B}$ is given $P_1, P_2 \in \mathbb{G}_1$ where $P_1 = aP, P_2 = bP$, (P, P_1, P_2) is a random instance of the Computational Diffie-Hellman problem. Its goal is to compute abP. Algorithm $\mathcal{B}$ will simulate the oracles and interact with the forger $\mathcal{A}_I$ as described below. In the proof, we regard the hash functions as the random oracles. Algorithm $\mathcal{B}$ starts by setting the master public key $mpk = P_1 = aP$, where P_1 is the input of the CDH problem. $\mathcal{B}$ sends (P, mpk) to the algorithm $\mathcal{A}_I$.

UserKeyGen: On a new **UserKeyGen** query ID_i, $\mathcal{B}$ chooses a random number $s_{ID_i} \in \mathbb{Z}_p$ and sets $(SK_{IDi}, PK_{ID_i}) = (s_{ID_i}, s_{ID_i}P)$. Then, he adds $(ID_i, SK_{IDi}, PK_{ID_i})$ into the list L and returns PK_{ID_i} to $\mathcal{A}_I$.

H_0**:** On a new H_0 query ζ_i, $\mathcal{B}$ first chooses a random number $coin_i \in \{0, 1\}$ such that $\Pr[coin_i = 1] = \delta$ where the value of δ will be determined later.

1. If $coin_i = 1$, $\mathcal{B}$ chooses a random number $c_i \in \mathbb{Z}_p$ and sets $H_0(\zeta_i) = c_iP + P_2$ where P_2 is another output of CDH problem.
2. Else $coin_i = 0$, $\mathcal{B}$ chooses a random number $c_i \in \mathbb{Z}_p$ and sets $H_0(\zeta_i) = c_iP$

In both cases, $\mathcal{B}$ will add (ζ_i, c_i) into H_0-*List* and return $H_0(\zeta_i)$ to $\mathcal{A}_I$

H_1**:** On a new H_1 query θ_i, $\mathcal{B}$ chooses a random number $d_i \in \mathbb{Z}_p$ and sets $H_1(\theta_i) = d_iP$. Then, he adds (θ_i, d_i) into H_1-*List* and returns $H_1(\theta_i)$ as the answer.

H_2**:** On a new H_2 query ξ_i, $\mathcal{B}$ chooses a random number $\lambda_i \in \mathbb{Z}_p$ and sets $H_2(\theta_i) = \lambda_iP$. Then, he adds $(\xi_i, \lambda_i, \lambda_iP)$ into H_2-*List* and returns $H_2(\theta_i)$ as the answer.

CertGen: On a certificate query ID_i, $\mathcal{B}$ first checks the list L to obtain this user's public key PK_{ID_i}. We assume that $(ID_i \| PK_{ID_i}, \cdot)$ has been in H_0-*List*. Otherwise, $\mathcal{B}$ can add $(ID_i \| PK_{ID_i}, c_i)$ into H_1-*List* as the same way he responds to H_0 queries.

1. If $c_i = 0$, which means $Q_{ID_i} = H_0(ID_i \| PK_{ID_i}) = c_iP$, $\mathcal{B}$ returns the certificate $Cert_{ID_i} = c_iP_1$.
2. Otherwise, $\mathcal{B}$ aborts.

Corruption: On a corruption query ID_i, $\mathcal{B}$ will check the list L and return SK_{ID_i} to $\mathcal{A}_I$.

Sign: On a sign query (m_i, ID_j), $\mathcal{B}$ first checks H_0-*List* to obtain $(ID_j \| PK_{ID_j}, c_j)$. If $c_j = 0$, $\mathcal{B}$ can generate the certificate $Cert_{ID_j}$ as he responds the **CertGen** queries and use $(Cert_{ID_j}, SK_{ID_j})$ to sign the message m_i. Otherwise, $c_j = 1$ and $H(ID_j \| PK_{ID_j}) = c_jP + P_2$. Then, $\mathcal{B}$ chooses a random number $r_i \in Z_p$ and sets $U_i = r_iP - P_1$.

1. Firstly, he checks H_1-*List*. If $(m_i \| U_i \| PK_{ID_j}, \cdot)$ does not exist in H_1-*List*, $\mathcal{B}$ will add $(m_i \| U_i \| PK_{ID_j}, d_i)$ into H_1-*List* as the same way he responds to H_1 queries.
2. Secondly, He checks whether $(m_i \| ID_j \| U_i \| PK_{ID_j}, \cdot)$ exists H_2-*List*. If it does, $\mathcal{B}$ must rechoose the number r_i until there is no collision. $\mathcal{B}$ further sets $H_2(m_i \| ID_j \| U_i \| PK_{ID_j}) = \lambda_iP + P_2$ and adds $(m_i \| ID_j \| U_i \| PK_{ID_j}, \lambda_i, \lambda_iP + P_2)$ into H_2-*List*.
3. At last, $\mathcal{B}$ computes $V_i = c_jP_1 + s_{ID_i}H_1(m_i \| U_i \| PK_{ID_j}) + \lambda_iU_i + r_iP_2$ and outputs (U_i, V_i) as the signature.

Correctness

$$\begin{aligned}
&e(V_i, P)\\
&= e(c_j P_1 + s_{ID_i} H_1(m_i \| U_i \| PK_{ID_j}) + \lambda_i U_i + r_i P_2, P)\\
&= e(c_j P_1 + abP + s_{ID_j} H_1(m_i \| U_i \| PK_{ID_j}) + \lambda_i U_i + r_i P_2 - abP, P)\\
&= e(a(c_j P + P_2) + s_{ID_j} H_1(m_i \| U_i \| PK_{ID_j})\\
&\quad + (r_i - a) H_2((m_i \| ID_j \| U_i \| PK_{ID_j}), P)\\
&= e(mpk, Q_{ID_j}) e(H_1(m_i \| U_i \| PK_{ID_j}), PK_{ID_j}) e(H_2((m_i \| ID_j \| U_i \| PK_{ID_j}), U_i)
\end{aligned}$$

At last, $\mathcal{A}_I$ outputs a valid forgery $(m^*, \sigma^* = (U^*, V^*), ID^*, PK_{ID^*})$ with probability $Succ_{\mathcal{A}_I}^{cma,cida}$. Again, PK_{ID^*} is chosen by $\mathcal{A}_I$ and might not be ID^*'s public key output from the oracle **UserKeyGen**. We assume that $(ID^* \| PK_{ID^*}, c^*)$, $(m^* \| U^* \| PK_{ID^*}, d^*)$, $(m^* \| ID^* \| U^* \| PK_{ID^*}, \lambda^*, \lambda^* P)$ have been in H_0-*List*, H_1-*List* and H_2-*List* respectively. If (U^*, V^*) is a valid signature of the message m^*, then $V^* = aH_0(ID^* \| PK_{ID^*}) + d^* PK_{ID^*} + \lambda^* U^*$.

1. If $c^* = 1$, $H_0(ID^* \| PK_{ID^*}) = c^* P + P_2$. Therefore, $\mathcal{B}$ can compute $abP = V^* - (c^* P_1 + d^* PK_{ID^*} + \lambda^* U^*)$.
2. Otherwise, $\mathcal{B}$ fails to solve this instance of CDH problem.

According to the simulation, $\mathcal{B}$ can the value of abP if and only if all the following two events happen:

E_1: $\mathcal{B}$ does not fail during the simulation.
E_2: $\mathcal{A}_I$ output a valid forgery.
E_3: In the forgery output by $\mathcal{A}_I$, $c^* = 1$.

Therefore, the probability that $\mathcal{B}$ can solve this instance of CDH problem is $Succ_{\mathcal{B},\mathbb{G}_1}^{CDH} = \Pr[E_1 \wedge E_2 \wedge E_3] = \Pr[E_1] \Pr[E_2|E_1] \Pr[E_3|E_1 \wedge E_2]$. In addition, all the simulation can be done in polynomial time.

From the simulation, we have $\Pr[E_1] \geq (1-\delta)^q$, $\Pr[E_2|E_1] = Succ_{\mathcal{A}_I}^{cma,cida}$ and $\Pr[E_3|E_1 \wedge E_2] = \delta$. Thus, $Succ_{\mathcal{B},\mathbb{G}_1}^{CDH} \geq \delta(1-\delta)^q Succ_{\mathcal{A}_I}^{cma,cida}$. When $\delta = 1/(q+1)$, this probability is maximized at

$$Succ_{\mathcal{B},\mathbb{G}_1}^{CDH} = \frac{1}{q+1}(1 - \frac{1}{q+1})^q Succ_{\mathcal{A}_I}^{cma,cida}.$$

B Proof of Theorem 2

Proof. Let P be the generator of $\mathbb{G}_1$. Algorithm $\mathcal{B}$ is given $P_1, P_2 \in \mathbb{G}_1$ where $P_1 = aP, P_2 = bP$, (P, P_1, P_2) is a random instance of the Computational Diffie-Hellman problem. Its goal is to compute abP. Algorithm $\mathcal{B}$ will simulate the oracles and interact with the forger $\mathcal{A}_{II}$ as described below. In the proof, we regard the hash functions as the random oracles. Algorithm $\mathcal{B}$ starts by choosing a random number $s \in \mathbb{Z}_p$ and sets $msk = s$, $mpk = sP$. Then $\mathcal{B}$ sends (P, s, sP) to $\mathcal{A}_{II}$.

UserKeyGen: On a new **UserKeyGen** query ID_i, $\mathcal{B}$ acts as following. Suppose there are up to q' **UserKeyGen** queries, $\mathcal{B}$ will choose a random number $\pi \in \{1, 2, \cdots, q'\}$.

1. ID_i is the π^{th} query, $\mathcal{B}$ sets $SK_{ID_i} = \perp$ and $PK_{ID_i} = P_1$ where P_1 is the input of CDH problem. Here, the symbol $\perp$ means $\mathcal{B}$ doesn't know the corresponding value.
2. Otherwise, $\mathcal{B}$ chooses a random number $SK_{ID_i} \in \mathbb{Z}_p$ and sets $PK_{ID_i} = SK_{ID_i}P$.

In both cases, $\mathcal{B}$ adds $(ID_i, SK_{ID_i}, PK_{ID_i})$ into the list L and returns PK_{ID_i} to $\mathcal{A}_{II}$.

H_0: On a new H_0 query ζ_i, $\mathcal{B}$ chooses a random number $c_i \in \mathbb{Z}_p$ and sets $H_0(\zeta_i) = c_iP$. Then, $\mathcal{B}$ adds (ζ_i, c_i) into H_0-*List* and return $H_0(\zeta_i)$ to $\mathcal{A}_{II}$.

H_1: On a new H_1 query θ_i, $\mathcal{B}$ chooses a random number $d_i \in \mathbb{Z}_p$ and sets $H_1(\theta_i) = d_iP + P_2$. Then, he adds $(\theta_i, d_i, d_iP + P_2)$ into H_1-*List* and returns $H_1(\theta_i)$ as the answer.

H_2: On a new H_2 query ξ_i, $\mathcal{B}$ chooses a random number $\lambda_i \in \mathbb{Z}_p$ and sets $H_2(\theta_i) = \lambda_iP$. Then, he adds $(\xi_i, \lambda_i, \lambda_iP)$ into H_2-*List* and returns $H_2(\theta_i)$ as the answer.

Corruption: On a corruption query ID_i, $\mathcal{B}$ will check the list L and return SK_{ID_i} to $\mathcal{A}_{II}$. If $SK_{ID_i} = \perp$, $\mathcal{B}$ fails to solve this problem.

Sign: On a sign query (m_i, ID_j), $\mathcal{B}$ first checks the list L.

1. If $SK_{ID_j} = \perp$, $\mathcal{B}$ will choose two random elements: $U_i = r_iP \in \mathbb{G}_1$ and $d_i \in \mathbb{Z}_p$. Then, he adds $(m_i \| U_i \| PK_{ID_j}, d_iP)$ into H_1-*List*. If a collision occurs, U_i and d_i will be re-chosen. In addition, $\mathcal{B}$ will add $(m_i \| ID_j \| U_i \| PK_{ID_j}, \lambda_i, \lambda_iP)$ to H_2-*List* as the same way he responds to H_2 queries. By assumption, $(ID_j \| PK_{ID_j}, c_j)$ has been in H_0-*List*. Then, $\mathcal{B}$ computes $V_i = Cert_{ID_j} + d_iPK_{ID_j} + \lambda_iU_i$. The signature (U_i, V_i) is returned.
   Correctness

$$
\begin{aligned}
& e(V_i, P) \\
= & e(Cert_{ID_j} + d_iPK_{ID_j} + \lambda_iU_i, P) \\
= & e(Cert_{ID_j} + SK_{ID_j}(d_i)P + r_i(\lambda_i)P, P) \\
= & e(Cert_{ID_j}, P)e(SK_{ID_j}(d_i)P, P)e(r_i(\lambda_i)P, P) \\
= & e(mpk, H_0(ID \| PK_{ID_j}))e(H_1(m_i \| U_i \| PK_{ID_j}), PK_{ID_j}) \\
& e(H_2(m_i \| ID_j \| U_i \| PK_{ID_j}), U_i)
\end{aligned}
$$

2. Otherwise, $\mathcal{B}$ can use SK_{ID_j} and $Cert_{ID_j}$ to generate the signature on this message.

At last, $\mathcal{A}_{II}$ outputs a valid forgery $(m^*, \sigma^* = (U^*, V^*), ID^*)$ with probability $Succ_{A_{II}}^{cma,cida}$. We assume that $(ID^* \| PK_{ID^*}, c^*)$, $(m^* \| U^* \| PK_{ID^*}, d^*, d^*P_2)$, $(m^* \| ID^* \| U^* \| PK_{ID^*}, \lambda^*, \lambda^*P)$ have been in H_0-*List*, H_1-*List* and H_2-*List* respectively. Here PK_{ID^*} is the public key of user ID^* output from the oracle **UserKeyGen**. If (U^*, V^*) is a valid signature of the message m^*, then $V^* = sc^*P + SK_{ID^*}(d^*P + P_2) + \lambda^*U^*$.

1. If $PK^* = P_1$, then SK_{ID^*} should be a. Therefore, $\mathcal{B}$ can compute $abP = V^* - (sc^*P + d^*P_1 + \lambda^* U^*)$.
2. Otherwise, $\mathcal{B}$ fails to solve this instance of CDH problem.

According to the simulation, $\mathcal{B}$ can the value of abP if and only if all the following two events happen:

E_1: $\mathcal{B}$ does not fail during the simulation.
E_2: $\mathcal{A}_{II}$ output a valid forgery.
E_3: In the forgery output by $\mathcal{A}_I$, $PK^* = P_1$.

Therefore, the probability that $\mathcal{B}$ can solve this instance of CDH problem is

$$Succ^{CDH}_{\mathcal{B},\mathbb{G}_1} = \Pr[E_1 \wedge E_2 \wedge E_3] = \Pr[E_1] \Pr[E_2|E_1] \Pr[E_3|E_1 \wedge E_2].$$

In addition, all the simulation can be done in polynomial time.

From the simulation, we have

$$\Pr[E_1] \geq (1 - \frac{1}{q'})^q,$$

$$\Pr[E_2|E_1] = Succ^{cma,cida}_{\mathcal{A}_{II}},$$

$$\Pr[E_3|E_1 \wedge E_2] = \frac{1}{q'}.$$

Thus, $Succ^{CDH}_{\mathcal{B},\mathbb{G}_1} \geq \frac{1}{q'}(1 - \frac{1}{q'})^q Succ^{cma,cida}_{\mathcal{A}_{II}}$ where $1 \neq q' \leq q$ denotes the number of queries submitted to the oracle **UserKeyGen**.

Time Capsule Signature: Efficient and Provably Secure Constructions*

Bessie C. Hu[1], Duncan S. Wong[1], Qiong Huang[2], Guomin Yang[1], and Xiaotie Deng[1]

[1] Department of Computer Science
City University of Hong Kong
Hong Kong, China
{bessiehu,duncan,csyanggm,deng}@cs.cityu.edu.hk
[2] csqhuang@cityu.edu.hk

Abstract. Time Capsule Signature, first formalized by Dodis and Yum in Financial Cryptography 2005, is a digital signature scheme which allows a signature to bear a (future) time t so that the signature will only be valid at time t or later, when a trusted third party called time server releases time-dependent information for checking the validity of a time capsule signature. Also, the actual signer of a time capsule signature has the privilege to make the signature valid *before* time t.

In this paper, we provide a new security model of time capsule signature such that time server is not required to be fully trusted. Moreover, we provide two efficient constructions in random oracle model and standard model. Our improved security model and proven secure constructions have the potential to build some new E-Commerce applications.

Keywords: Time Capsule Signature.

1 Introduction

Modern business is in nature the business for future. A contract signed now is a commitment for some future cooperation; a ticket bought now presents an entry permit at a specific time in the future; an option obtained now, in the derivative markets, ensures the privilege of buying/selling a stock at some time in the future. The success of these practices requires the integrity of credential releasers, and the involvement of an authority who can judge the rules for legal players. To realize these activities in E-Commerce platforms, a new primitive, which has a great promise to be a very useful tool, is called Time Capsule Signature [13].

A time capsule signature involves a signer (known as credential releaser), a verifier (known as credential receiver) and a time server (known as authority). The signer can issue a *future* signature indicated by some time information,

* The second author was supported by a grant from City University of Hong Kong (Project No. 7002001).

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 126–142, 2007.

say t, and enjoys the following properties: 1) The credential receiver can verify immediately that a signature will become valid at time t. 2) The signature will automatically become valid at time t, even without the cooperation of signer. 3) The legal signer has the privilege to make the signature valid before time t.

Property 1 and 2 are easy to comprehend in the current practice. However, in a naive solution of signing a statement that 'the message m will become valid from time t', the verifier is required to be aware of the current time [13]. When time is generalized to arbitrary events, this becomes even more problematic. Moreover, signer has lost control of the validation time t once the statement is produced. For the variety of E-Commerce, we do need to provide signers the power to validate their future signature before the committed time t. For example, in the case of debt repayment, a borrower can sign a check to indicate the repayment day (e.g. due day), he may also have the desire to repay his debt earlier, so to improve his credit history. Of course he can sign another check indicating the actual repayment time, but the original check should be handled carefully to avoid 'double spending'. Time capsule signature supports this desirable feature with a process of making a signature valid at any time by the actual signer known as *prehatch*, as opposed to *hatch* the signature at time t when some additional information is published by the time server. We refer readers to [13] for more discussions on the applications of time capsule signature. Property 3 may also be captured in a signed statement that 'the signature of message m will become valid from time t, or when the signer release some secret information'. Again, such a statement has problems when time is generalized to arbitrary events.

The notion of time capsule signature was first formalized by Dodis and Yum [13] in 2005. Besides the above three properties, they also require that prehatched signature should be indistinguishable from hatched signature. For practical use of time capsule signature as discussed above, the indistinguishability between prehatched signature and hatched signature is actually undesirable. Since the purpose of prehatching is to make a signature valid before time t, the verifier can simply compare the time t with the current time to identify if a signature is prehatched or normally hatched. Furthermore, in some scenarios, we actually need to distinguish a prehatched signature from a hatched signature. In the above debt repayment case, a prehatched signature has to be identified for credit history checking. On the other hand, under the property of indistinguishability, the time server has to be fully trusted, otherwise, there is no way to tell if a signature which becomes valid before time t is generated by the actual signer or a cheating time server.

Therefore, in this paper, we remove the requirement of indistinguishability for time capsule signature while retaining all other properties. This allows us to modify the security model for capturing attacks launched by a cheating time server. Our generic construction is based on a new primitive called identity-based trapdoor relation (IDTR). We propose two efficient implementations for the IDTR primitive, one is proven secure in the random oracle model, the other in the standard model.

1.1 Related Work

The work on timed-release cryptography was first summarized and discussed by May [18] in 1993, and further work was carried out by Rivest et al. [21] in 1996. The main purpose of timed-release cryptography is to ensure that encryption, commitment or signature cannot be opened or valid until a predetermined future time. There are two main approaches for constructing such a scheme. The first approach, categorized as *time-lock puzzles* [21], is to design a computational problem which could be solved by continually computing for at least some required period of time. This approach is widely used in applications, like *verifiable time capsules* [2,3], *timed commitments*[9], and some recently proposed systems [14,15]. The tradeoff of this approach is that immense computational overhead has to be put on the receiver, that makes it impractical for most real-world applications.

The second approach relies on a trusted agent who releases time-dependent information exactly according to a pre-specified schedule. Previous work is mainly on timed-release encryption, which diversifies according to the involvement level of the trusted agent. May [18] suggested that the trusted agent should store messages until the time to release. Rivest et. al. [21] suggested that the agent should pre-compute pairs of public/private keys, publish the public keys first and then release the private keys one by one according to some pre-specified schedule. Most of the recent results [10,5,19,11] are based on Boneh and Franklin's identity-based encryption (IBE) scheme [7]. In this paper, we also use Boneh-Franklin IBE as one of the implementations of our generic construction. In this implementation, we replace the identity in the IBE scheme with the claimed time t, but the technical details are different from previous constructions which are only for timed-release encryption. They will become clear when going through our construction in the subsequent sections of this paper.

Another stream of research based on trusted agents is optimistic fair exchange of digital signatures [1,8,12]. In those constructions, a trusted agent needs to resolve all signatures where the signers are refusing to validate the signatures. Scalability is the main issue of this approach. Recently, there is a new construction of time capsule signature [24] based on ring signature [20]. However, in their system, the time server needs to generate time-dependent information for each individual user, thus scalability is a main problem.

1.2 Paper Organization

In Sec. 2, we introduce some preliminaries. The definition and security model of time capsule signature is specified in Sec. 3. In Sec. 4, we define a new notion called Identity-based Trapdoor Relation (IDTR) and propose two concrete implementations which are proven to be secure in the random oracle model and the the standard model. In Sec. 5, we propose a generic construction of time capsule signature based on IDTR and analyze its security. In Sec. 6, we extend IDTR by adding a new property called Hiding, and use it to construct a distinguishable time capsule signature which could capture an attacker launched by a malicious time server. Finally, we conclude in Sec. 7.

2 Preliminaries

Identity Based Encryption. The notion of Identity Based Encryption (IBE) was introduced by Shamir in 1984 [22]. In such a mechanism, public key could be an arbitrary string, which is chosen from user's name, network address, etc; user private key is properly generated by a trusted third party (Key Generation Center), and the secret can be preserved as long as Key Generation Center does not release its master secret key. For IBE, a message can be encrypted for a receiver even before the corresponding private key is generated. To this extent, IBE is a good candidate of sending a message to the future.

The first practical IBE was proposed by Boneh and Franklin [7] in 2001. They proposed a basic IBE scheme, which is secure against chosen plaintext attack(IND-ID-CPA). By extending the basic scheme, a full scheme could be achieved with security against adaptive chosen ciphertext attack(IND-ID-CCA) in the random oracle model.

In 2005, Brent Waters [23] presented the first efficient Identity-Based Encryption scheme that is fully secure without random oracles. The proof of their scheme makes use of an algebraic method first used by Boneh and Boyen [6] and the security of the scheme is reduced to the decisional Bilinear Diffie-Hellman (BDH) assumption.

Based on IBE, in this paper, we propose a new notion called *Identity Based Trapdoor Relation* (IDTR) which can then be applied to the construction of time capsule signature scheme.

Computational Diffie-Hellman Assumption. Let $\mathbb{G}$ be a group of order p (p is a prime). The challenger chooses $a, b \in \mathbb{Z}_p$ at random and outputs $(g, A = g^a, B = g^b)$, where $g \in \mathbb{G}$. The adversary then attempts to output $g^{ab} \in \mathbb{G}$. An adversary $\mathcal{B}$ has at least ϵ advantage if

$$Pr[\mathcal{B}(g, g^a, g^b) = g^{ab}] \geq \epsilon$$

where the probability is taken over the random choices of a, b and the random bits consumed by $\mathcal{B}$.

Definition: The computational (t, ϵ)-DH assumption holds if no t-time adversary has at least ϵ advantage in the game above.

3 Time Capsule Signature

3.1 Definition

A *time capsule signature scheme* consists of eight PPT algorithms (TSSetup, UserSetup, TSig, TVer, TRelease, Hatch, PreHatch, Ver). The definition below follows that of Dodis and Yum [13].

1. TSSetup (Time Server Key Setup): On input 1^k where $k \in \mathbb{N}$ is a security parameter, it generates a public/secret time release key pair (tpk, tsk).

2. UserSetup (User Key Setup): On input 1^k, it generates a user public/secret key pair (upk, usk).
3. TSig (Time Capsule Signature Generation): On input (m, usk, upk, t), where t is a time value from which the signature will become valid. It outputs a time capsule signature σ'_t.
4. TVer (Time Capsule Signature Verification): On input $(m, \sigma'_t, upk, tpk, t)$, it returns 1 (accept) or 0 (reject). A time capsule signature σ'_t is said to be valid if TVer returns 1 on it.
5. TRelease (Time Release): At the beginning of each time period T, $z_T \leftarrow \text{TRelease}(T, tsk)$ is published by the time server.
6. Hatch (Signature Hatch): On input $(m, \sigma'_t, upk, tpk, z_t)$, anyone can run this algorithm to get a hatched signature σ_t from a valid time capsule signature σ'_t.
7. PreHatch (Signature Prehatch): On input $(m, \sigma'_t, usk, tpk, t)$, the signer can run the algorithm to get a prehatched signature σ_t of a valid time capsule signature σ'_t before time t. However, if σ'_t is not valid, namely, $\text{TVer}(m, \sigma'_t, upk, tpk, t) = 0$, then PreHatch should return $\perp$ which stands for unsuccessful prehatch.
8. Ver (Signature Verification): On input $(m, \sigma_t, upk, tpk, t)$, it returns 1 (accept) or 0 (reject).

Note that Time Server does not contact any user or need to know anything from any user.

3.2 Adversarial Model

There are three types of adversaries, $\mathcal{A}_I$, $\mathcal{A}_{II}$ and $\mathcal{A}_{III}$. $\mathcal{A}_I$ simulates a malicious signer whose aim is to produce a time capsule signature σ'_t, which looks good to a verifier, but cannot be hatched at time t. $\mathcal{A}_{II}$ simulates a malicious verifier who wants to hatch a time capsule signature before time t. $\mathcal{A}_{III}$ simulates a malicious time server who wants to forge a signature. Note that attacks launched by an outsider who wants to forge a signature can also be captured by $\mathcal{A}_{III}$. In the following, let $k \in \mathbb{N}$ be a security parameter.

Game I: Let $\mathcal{S}_I$ be the game simulator.

1. $\mathcal{S}_I$ executes $\text{TSSetup}(1^k)$ to get (tpk, tsk).
2. $\mathcal{S}_I$ runs $\mathcal{A}_I$ on tpk. During the simulation, $\mathcal{A}_I$ can make queries onto TRelease.
3. $\mathcal{A}_I$ is to output $(m^*, t^*, \sigma'^*, upk)$.
4. $\mathcal{S}_I$ executes $\text{TRelease}(t^*, tsk)$ to get z_t^*, and then executes $\text{Hatch}(m^*, \sigma'^*, upk, tpk, z_t^*)$ to get σ^*.

$\mathcal{A}_I$ *wins* if $\text{TVer}(m^*, \sigma'^*, upk, tpk, t) = 1$ and $\text{Ver}(m^*, \sigma^*, upk, tpk, t) = 0$.

A time capsule signature scheme is secure in **Game I** if for every PPT algorithm $\mathcal{A}_I$, it is negligible for $\mathcal{A}_I$ to win the game. Note that we do not put any restriction on the generation of user public key upk. This is natural as in practice, $\mathcal{A}_I$ is normally the one who generates (upk, usk).

Game II: Let $\mathcal{S}_{II}$ be the game simulator.

1. $\mathcal{S}_{II}$ executes $\mathsf{TSSetup}(1^k)$ to get (tpk, tsk) and $\mathsf{UserSetup}(1^k)$ to get (upk, usk).
2. $\mathcal{S}_{II}$ runs $\mathcal{A}_{II}$ on tpk and upk. During the simulation, $\mathcal{A}_{II}$ can make queries onto TSig, $\mathsf{TRelease}$ and $\mathsf{PreHatch}$.
3. $\mathcal{A}_{II}$ is to output (m^*, t^*, σ^*).

$\mathcal{A}_{II}$ *wins* if $\mathsf{Ver}(m^*, \sigma^*, upk, tpk, t^*) = 1$, and $\mathcal{A}_{II}$ has never queried $\mathsf{TRelease}(t^*)$ and $\mathsf{PreHatch}(m^*, t^*, \cdot)$.

A time capsule signature scheme is secure in **Game II** if for every PPT algorithm $\mathcal{A}_{II}$, it is negligible for $\mathcal{A}_{II}$ to win the game.

Game III: Let $\mathcal{S}_{III}$ be the game simulator.

1. $\mathcal{S}_{III}$ executes $\mathsf{TSSetup}(1^k)$ to get (tpk, tsk), and $\mathsf{UserSetup}(1^k)$ to get (upk, usk).
2. $\mathcal{S}_{III}$ runs $\mathcal{A}_{III}$ on upk, tpk and tsk. During the simulation, $\mathcal{A}_{III}$ can make queries onto TSig, and $\mathsf{PreHatch}$.
3. $\mathcal{A}_{III}$ is to output (m^*, t^*, σ^*).

$\mathcal{A}_{III}$ *wins* if $\mathsf{Ver}(m^*, \sigma^*, upk, tpk, t^*) = 1$, and $\mathcal{A}_{III}$ has never queried $\mathsf{TSig}(m^*, \cdot)$ for time t^*.

A time capsule signature scheme is secure in **Game III** if for every PPT algorithm $\mathcal{A}_{III}$, it is negligible for $\mathcal{A}_{III}$ to win the game.

3.3 Discussion

One of the properties of time capsule signature in Dodis-Yum paper is ambiguity which ensures that a prehatched signature is indistinguishable with a hatched signature (with respect to the same message and time value t). Although this property may have independent interest, we notice that in common applications of time capsule signature described in Sec. 1 and in [13], this property is actually undesirable. Since the only purpose of prehatching a signature is to make the signature verifiable before time t. In this case, the verifier can simply check the time t against the current time for finding out if the signature is prehatched or normally hatched.

Our definition, instead, does not requires ambiguity. By this relaxation, we can construct more efficient time capsule signature schemes based on identity-based trapdoor relation (IDTR) in Sec. 4. We will see more discussions on this relaxation and explain that for some applications, it is actually important for the verifier to tell whether a signature is pre-hatched or hatched.

4 Identity-Based Trapdoor Relation (IDTR)

A binary relation R is a subset of $\{0,1\}^* \times \{0,1\}^*$ and the language $\mathcal{L}_R$ is the set of α's for which there exist β such that $(\alpha, \beta) \in R$, i.e., $\mathcal{L}_R = \{\alpha | \exists \beta [(\alpha, \beta) \in R]\}$. We assume that (1) there is an efficient algorithm to decide whether $\alpha \in \mathcal{L}_R$ or

not, (2) if $(\alpha, \beta) \in R$, then the length of β is polynomially bounded in $|\alpha|$, and (3) there exists a short description D_R which specifies the relation R.

An *identity-based trapdoor relation* (IDTR) is a set of relations $\mathcal{R} = \{R_{id} | id \in I_{\mathcal{R}}\}$, where each relation R_{id} is called a trapdoor relation and there is a master trapdoor $mtd_{\mathcal{R}}$ for extracting the trapdoor td_{id} of each R_{id}. Formally, IDTR is specified by the following five probabilistic polynomial-time (PPT) algorithms (Gen, Sample, Extract, Invert, Check).

1. Gen : This algorithm is used to generate $\mathcal{R} = \{R_{id} | id \in I_{\mathcal{R}}\}$ where $I_{\mathcal{R}}$ is a finite set of indices. $\mathsf{Gen}(1^k)$ returns $D_{\mathcal{R}}$ (the description of $\mathcal{R}$) and $mtd_{\mathcal{R}}$ (the master trapdoor).
2. Sample : This sampling algorithm takes $(D_{\mathcal{R}}, id)$ as input and $\mathsf{Sample}_{D_{\mathcal{R}}}(id)$ returns a random commitment c and witness d such that $(c, d) \in R_{id}$.
3. Extract : This algorithm is used to extract the trapdoor of each relation by using $mtd_{\mathcal{R}}$. $\mathsf{Extract}_{mtd_{\mathcal{R}}}(id)$ returns the trapdoor $\mathsf{td}_{R_{id}}$ of relation R_{id}.
4. Invert : This algorithm is used to find a witness d for a given $c \in \mathcal{L}_{R_{id}}$ by using the trapdoor $\mathsf{td}_{R_{id}}$. If $c \in \mathcal{L}_{R_{id}}$, then $\mathsf{Invert}_{\mathsf{td}_{R_{id}}}(c)$ returns a witness $\hat{d}$ such that $(c, \hat{d}) \in R_{id}$.
5. Check : This algorithm is used to check the validity of a witness d on the commitment c. If $(c, d) \in R_{id}$, then $\mathsf{Check}_{D_{\mathcal{R}}, id}(c, d)$ returns 1 (accept). Otherwise, it returns 0 (reject).

Properties: One-wayness requires that no one is able to find the witness of a commitment if the trapdoor information is not given. Soundness requires that no one can produce a commitment whose witness cannot be found using Invert.

– One-wayness: Let O_{Extract} be an oracle simulating the trapdoor extraction procedure Extract and $Query(A, O_{\mathsf{Extract}})$ the set of queries an algorithm A asked to O_{Extract}. It states that the following probability is negligible for all PPT algorithm $A = (A_1, A_2)$:

$$\begin{aligned} &Pr[\mathsf{Check}_{D_{\mathcal{R}}, id^*}(c^*, \tilde{d}) = 1 \wedge id^* \notin Query(A, O_{\mathsf{Extract}}) | \\ &(D_{\mathcal{R}}, mtd_{\mathcal{R}}) \leftarrow \mathsf{Gen}(1^k); (id^*, h) \leftarrow A_1^{O_{\mathsf{Extract}}}(D_{\mathcal{R}}); \\ &(c^*, d) \leftarrow \mathsf{Sample}_{D_{\mathcal{R}}}(id^*); \tilde{d} \leftarrow A_2^{O_{\mathsf{Extract}}}(id^*, c^*, h)] \end{aligned}$$

– Soundness: We require that the following probability should be negligible for all algorithm B:

$$\begin{aligned} &Pr[R_{id^*} \in \mathcal{R} \wedge c^* \in \mathcal{L}_{R_{id^*}} \wedge \mathsf{Check}_{D_{\mathcal{R}}, id^*}(c^*, \tilde{d}) = 0 | \\ &(D_{\mathcal{R}}, mtd_{\mathcal{R}}) \leftarrow \mathsf{Gen}(1^k); (c^*, id^*) \leftarrow B(D_{\mathcal{R}}, mtd_{\mathcal{R}}); \\ &\mathsf{td}_{R_{id^*}} \leftarrow \mathsf{Extract}_{mtd_{\mathcal{R}}}(id^*); \tilde{d} \leftarrow \mathsf{Invert}_{\mathsf{td}_{R_{id^*}}}(c^*)] \end{aligned}$$

Discussion: The definition of IDTR above is much like the definition of Dodis and Yum's Identity-based Hard-to-Invert Relation (ID-THIR) [13]. ID-THIR has an *ambiguity* property which requires that witness $\hat{d}$ inverted from c given $\mathsf{td}_{R_{id}}$ is computationally indistinguishable from d obtained from $\mathsf{Sample}_{D_{\mathcal{R}}}(id)$ for the

same commitment c. To facilitate our construction of time capsule signature under new definition in Sec. 3, we do not require ambiguity property in the definition of IDTR above. We will see that with this relaxation we can construct much more efficient schemes then that in [13].

4.1 Implementations of IDTR

In this section, we propose two concrete constructions of IDTR, one based on Boneh and Franklin's IBE whose security has been proven in the random oracle model [7], and the other one based on Waters' IBE whose security has been proven in the standard model [23].

Implementation 1: In Random Oracle Model. An IBE scheme consists of four PPT algorithms (Setup, KeyGen, Encrypt, Decrypt). The Boneh-Franklin scheme [7] is described as follows:

1. Setup: Given a security parameter $k \in \mathbb{N}$, generate a prime q, two groups $\mathbb{G}_1$, $\mathbb{G}_2$ of order q, and an admissible bilinear map $\hat{e}$: $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$, where $|q|$ is some polynomial in k. Choose a random generator $P \in \mathbb{G}_1$, pick a random $s \in \mathbb{Z}_q^*$ and set $P_{pub} = sP$. Choose a cryptographic hash function H_1: $\{0,1\}^* \rightarrow \mathbb{G}_1$, another hash function H_2: $\mathbb{G}_2 \rightarrow \{0,1\}^k$, and the security analysis will view H_1, H_2 as random oracles [4]. The message space is $\mathcal{M} = \{0,1\}^k$. The ciphertext space is $\mathcal{C} = \mathbb{G}_1 \times \{0,1\}^k$. The system public key is $mpk = \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, k, P, P_{pub}, H_1, H_2 \rangle$. The master secret key msk is $s \in \mathbb{Z}_q^*$.
2. KeyGen: For a given string $id \in \{0,1\}^*$ the algorithm computes $Q_{id} = H_1(id) \in \mathbb{G}_1$, and sets the private key sk_{id} to be sQ_{id} where s is the master secret key.
3. Encrypt: To encrypt $m \in \mathcal{M}$ under the public key id, the algorithm computes $Q_{id} = H_1(id) \in \mathbb{G}_1$, chooses a random $r \in \mathbb{Z}_q^*$, and sets the ciphertext to be $c = \langle rP, m \oplus H_2(g_{id}^r) \rangle$ where $g_{id} = \hat{e}(Q_{id}, P_{pub}) \in \mathbb{G}_2$.
4. Decrypt: Given the private key $sk_{id} \in \mathbb{G}_1$, a ciphertext $c = \langle c_1, c_2 \rangle \in \mathcal{C}$ can be decrypted by computing $c_2 \oplus H_2(\hat{e}(sk_{id}, c_1)) = m$.

An IDTR based on the IBE above is constructed as follows:

1. Gen: Run $\mathsf{Setup}(1^k)$, and set $\mathcal{L}_R = \mathcal{C}$, $D_{\mathcal{R}} = mpk$, and $mtd_{\mathcal{R}} = msk$.
2. Sample: Given $D_{\mathcal{R}}$ and id, randomly pick $m \in \mathcal{M}$ and compute $c = \mathsf{Encrypt}_{id}(m)$. Let $r \in \mathbb{Z}_q^*$ be the randomness used in Encrypt. Set witness $d = \langle rQ_{id}, P_{pub}, m \rangle$.
3. Extract: Given a string $id \in \{0,1\}^*$, compute $sk_{id} = \mathsf{KenGen}_{mpk,msk}(id)$, and set $\mathsf{td}_{R_{id}} = sk_{id}$.
4. Invert: Given trapdoor $\mathsf{td}_{R_{id}} \in \mathbb{G}_1$ and a commitment $c = \langle c_1, c_2 \rangle \in \mathcal{C}$, compute $m = \mathsf{Decrypt}_{sk_{id}}(c)$, and set the witness $\hat{d} = \langle \mathsf{td}_{R_{id}}, c_1, m \rangle$.
5. Check: Given $D_{\mathcal{R}}$, id, $c = \langle c_1, c_2 \rangle \in \mathcal{C}$, $d = \langle d_1, d_2, d_3 \rangle$ (where $d_1, d_2 \in \mathbb{G}_1$, and $d_3 \in \mathcal{M}$) , if $d_2 = P_{pub}$, $\hat{e}(d_1, P) = \hat{e}(c_1, Q_{id})$, and $c_2 = d_3 \oplus H_2(\hat{e}(d_1, d_2)$, return 1. Else if $d_1 = \mathsf{td}_{R_{id}}$, $d_2 = c_1$, $\hat{e}(d_1, P) = \hat{e}(P_{pub}, Q_{id})$ and $c_2 = d_3 \oplus H_2(\hat{e}(d_1, d_2))$, return 1. Otherwise, return 0.

One-wayness. In the game of one-wayness, an adversary A has access to the Extract oracle of all id other than id^*. This oracle is simulated by performing KeyGen of the underlying IBE scheme. A wins if it can find secret key sk_{id^*} and plaintext m^*. However, the semantic security (IND-ID-CPA) [7] of the underlying IBE attains that any PPT adversary will have negligible advantage in distinguishing m^* with another m in $\mathcal{M}$. If A succeeds, it is easy to see that we can also distinguish m^*, which contradicts the security of the underlying IBE scheme.

Soundness. An adversary B wins if it can generate a value c^* which is not able to decrypt under sk_{id^*}. In the underlying IBE scheme, this will not be the case even when B knows msk. Given id^*, sk_{id^*} can always be properly generated with the knowledge of msk. As long as c^* is in the ciphertext domain, a valid plaintext m^* can always be retrieved.

Remark: This construction of IDTR in random oracle model based on the Boneh-Franklin IBE scheme is much more efficient than the OR-proof for ID-THIR [13].

Implementation 2: In Standard Model. We now review Waters' IBE [23] and propose a construction for IDTR based on this scheme.

1. Setup: Given a security parameter $k \in \mathbb{N}$, generate a prime p, two groups $\mathbb{G}_1, \mathbb{G}_2$ of order p, and an admissible bilinear map $\hat{e}$: $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$, where $|p|$ is some polynomial in k. Choose a random generator $g \in \mathbb{G}_1$, pick a random $\alpha \in \mathbb{Z}_p^*$ and set $g_1 = g^\alpha$. Choose random values g_2, $u' \in \mathbb{G}_1$, and a random k-length vector $U = (u_i)$, whose elements are chosen uniformly at random from $\mathbb{G}_1$. The message space is $\mathcal{M} \subseteq \mathbb{G}_2$. The ciphertext space is $\mathcal{C} = \mathbb{G}_2 \times \mathbb{G}_1 \times \mathbb{G}_1$. The system public key is $mpk = \langle p, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, k, g, g_1, g_2, u', U \rangle$. The master secret key msk is g_2^α.
2. KeyGen: Let v be an k-bit string representing an identity id, v_i denote the ith bit of v, and $\mathcal{V} \subseteq 1, ..., k$ be the set of all i for which $v_i = 1$. ($\mathcal{V}$ is the set of indices for which the bitstring v is set to 1.) Randomly select $r \in \mathbb{Z}_p^*$ and construct the private key sk_{id} as:

$$sk_{id} = \langle g_2^\alpha (u' \prod_{i \in \mathcal{V}} u_i)^r, g^r \rangle$$

3. Encrypt: To encrypt $m \in \mathcal{M}$ for an identity v, randomly select $t \in \mathbb{Z}_p^*$ and construct the ciphertext c as:

$$c = \langle \hat{e}(g_1, g_2)^t m, g^t, (u' \prod_{i \in \mathcal{V}} u_i)^t \rangle$$

4. Decrypt: Given the private key $sk_{id} = \langle sk_1, sk_2 \rangle$, a ciphertext $c = \langle c_1, c_2, c_3 \rangle \in \mathcal{C}$ can be decrypted as:

$$c_1 \frac{\hat{e}(sk_2, c_3)}{\hat{e}(sk_1, c_2)} = (\hat{e}(g_1, g_2)^t m) \frac{\hat{e}(g^r, (u' \prod_{i \in \mathcal{V}} u_i)^t)}{\hat{e}(g_2^{\alpha} (u' \prod_{i \in \mathcal{V}} u_i)^r, g^t)}$$
$$= (\hat{e}(g_1, g_2)^t m) \frac{\hat{e}(g, (u' \prod_{i \in \mathcal{V}} u_i)^{rt})}{(\hat{e}(g_1, g_2)^t) \hat{e}((u' \prod_{i \in \mathcal{V}} u_i)^{rt}, g)} = m$$

Based on Waters' IBE: $(mpk, msk) \leftarrow \mathsf{Setup}(1^k)$; $sk_{id} = \mathsf{KenGen}_{mpk,msk}(id)$; $c = \mathsf{Encrypt}_{id}(m)$; $m = \mathsf{Decrypt}_{sk_{id}}(c)$, an IDTR can be constructed as follows:

1. Gen: Given $k \in \mathbb{N}$, execute $(mpk, msk) \leftarrow \mathsf{Setup}(1^k)$ and set $\mathcal{L}_R = \mathcal{C}$, $D_{\mathcal{R}} = mpk$, and $mtd_{\mathcal{R}} = msk$.
2. Sample: Given $D_{\mathcal{R}}$ and id, randomly pick $m \in \mathbb{G}_2$ and compute $c = \mathsf{Encrypt}_{id}(m)$. Let $t \in \mathbb{Z}_p^*$ be the randomness in producing c. Set witness d to

$$d = \langle g_2^t, g_1 (u' \prod_{i \in \mathcal{V}} u_i), g_2, c_3, m \rangle$$

3. Extract: Let v be an k-bit identity id, compute $sk_{id} = \mathsf{KenGen}_{mpk,msk}(id)$ and set $\mathsf{td}_{R_{id}} = sk_{id}$.
4. Invert: Given trapdoor $\mathsf{td}_{R_{id}}$ and a commitment $c = \langle c_1, c_2, c_3 \rangle \in \mathcal{C}$, compute $m = \mathsf{Decrypt}_{sk_{id}}(c)$, and set the witness to

$$\hat{d} = \langle sk_1, c_2, sk_2, c_3, m \rangle$$

5. Check: Given $D_{\mathcal{R}}$, id, $c = \langle c_1, c_2, c_3 \rangle \in \mathcal{C}$, $d = \langle d_1, d_2, d_3, d_4, d_5 \rangle$ (where $d_1, d_2, d_3, d_4 \in \mathbb{G}_1$, and $d_5 \in \mathcal{M}$), if $d_2 = g_1 (u' \prod_{i \in \mathcal{V}} u_i)$, $d_3 = g_2$,$d_4 = c_3$, $\hat{e}(d_1, u' \prod_{i \in \mathcal{V}} u_i) = \hat{e}(g_2, c_3)$, and $c_1 = M \frac{\hat{e}(d_1, d_2)}{\hat{e}(d_3, d_4)}$, return 1(∗∗). Else if $d_1 = sk_1$, $d_2 = c_2$, $d_3 = sk_2$, $d_4 = c_3$, $\hat{e}(sk_1, g) = \hat{e}(g_2, g_1) \cdot \hat{e}(u' \prod_{i \in \mathcal{V}} u_i, sk_2)$, and $c_1 = m \frac{\hat{e}(d_1, d_2)}{\hat{e}(d_3, d_4)}$, return 1. Otherwise, return 0.

(∗∗):The check will pass because:

$$m \frac{\hat{e}(d_1, d_2)}{\hat{e}(d_3, d_4)} = m \frac{\hat{e}(g_2^t, g_1 (u' \prod_{i \in \mathcal{V}} u_i))}{\hat{e}(g_2, (u' \prod_{i \in \mathcal{V}} u_i)^t)}$$
$$= m \frac{\hat{e}(g_2^t, g_1) \cdot \hat{e}(g_2^t, (u' \prod_{i \in \mathcal{V}} u_i))}{\hat{e}(g_2, (u' \prod_{i \in \mathcal{V}} u_i)^t)} = m \cdot \hat{e}(g_2^t, g_1) = c_1$$

Similar to the first implementation, the proof of One-wayness can be reduced to IND-ID-CPA security of Waters' IBE scheme. Soundness also holds since a valid $c \in \mathcal{C}$ can always be decrypted to a message m for a given sk_{id}.

Discussion: Note that given $c \in \mathcal{C}$, we only require that an adversary is not able to compute the entire m for a randomly chosen $m \in \mathbb{G}_2$. In other words, we do not need IND-ID-CPA [7] security. Although both of the constructions could achieve IND-ID-CPA, this is not a necessity in our security notion.

5 Generic Construction of Time Capsule Signature

We now describe our generic construction of time capsule signature scheme. Our construction is based on the identity-based trapdoor relation (IDTR) defined in Sec. 4.

Let (Set, Sig, Verify) be the key generation, signature generation and verification algorithms of an ordinary signature scheme, and (Gen, Sample, Extract, Invert, Check) be the tuples of IDTR.

1. TSSetup: Let $k \in \mathbb{N}$ be a security parameter. The Time Sever gets $(D_{\mathcal{R}}, mtd_{\mathcal{R}}) \leftarrow \mathsf{Gen}(1^k)$ and sets public/secret time release key pair $(tpk, tsk) = (D_{\mathcal{R}}, mtd_{\mathcal{R}})$.
2. UserSetup: Each user runs $(pk, sk) \leftarrow \mathsf{Set}(1^k)$ and sets $(upk, usk) = (pk, sk)$.
3. TSig: To generate a time capsule signature on a message m for a future time t, the signer gets a commitment/witness pair $(c, d) \leftarrow \mathsf{Sample}_{D_{\mathcal{R}}}(t)$, then computes $s \leftarrow \mathsf{Sig}_{usk}(m\|c\|t)$. The time capsule signature σ'_t is (s, c). The signer stores the witness d.
4. TVer: A verifier checks if $\sigma'_t{=}(s, c)$ is a valid time capsule signature by checking whether $c \in \mathcal{L}_{R_t}$ and s is a valid standard signature under public key upk, that is, check if $\mathsf{Verify}_{upk}(m\|c\|t, s) = 1$. If both are correct, output 1; otherwise, output 0.
5. TRelease: At the beginning of each time period T, the Time Server gets $td_{R_T} \leftarrow \mathsf{Extract}_{tsk}(T)$ and publishes td_{R_T} as z_T.
6. Hatch: To hatch a time capsule signature $\sigma'_t = (s, c)$, a party computes $\hat{d} \leftarrow \mathsf{Invert}_{td_{R_t}}(c)$. The hatched signature is $\sigma_t = (s, c, \hat{d})$.
7. PreHatch: To prehatch a valid time capsule signature $\sigma'_t{=}(s, c)$, the signer retrieves stored value d, and sets the prehatched signature to $\sigma_t = (s, c, d)$. However, if $\mathsf{TVer}(m, \sigma'_t, upk, tpk, t) = 0$, then the algorithm outputs $\perp$.
8. Ver: For a given prehatched (or hatched) signature $\sigma_t = (s, c, d)$, the verifier checks the validity of (c, d) by running $\mathsf{Check}_{tpk,t}(c, d)$. Then, it verifies s on $m\|c\|t$ by running $\mathsf{Verify}_{upk}(m\|c\|t, s)$. If both verifications are correct, output 1; otherwise, output 0.

5.1 Security Analysis

Theorem 1. *The proposed time capsule signature scheme is secure if the underlying public key signature scheme is existentially unforgeable against adaptive chosen message attacks (euf-cma) [16] and the IDTR has the properties of one-wayness and soundness.*

Proof. We prove the security of our proposed time capsule signature scheme against Game I, Game II and Game III.

Security Against Game I: $\mathcal{A}_I$ wins the game if he can generate a valid time capsule signature $\sigma'_t{=}(s, c)$ such that $c \in \mathcal{L}_{R_t}$, and $\mathsf{Ver}_{upk}(m||c||t, s) {=}1$. Moreover, no party can obtain a witness $\tilde{d} = \mathsf{Invert}_{td_{R_t}}(c)$ such that $\mathsf{Check}_{tpk,t}(c, \tilde{d}) = 1$, where $td_{R_t} \leftarrow \mathsf{Extract}_{tsk}(t)$ is released by the Time Server. This contradicts

the Soundness property of IDTR. Thus, the proposed time capsule signature scheme is secure against **Game I** if underlying IDTR satisfies the Soundness property.

Security Against Game II: We construct an adversary $\mathcal{B}$ which breaks the One-wayness of IDTR with non-negligible advantage if $\mathcal{A}_{II}$ forges a valid signature σ. Let (m^*, t^*, σ^*) be a successful forgery generated by $\mathcal{A}_{II}$. Since the underlying standard signature scheme $(\mathsf{Set}, \mathsf{Sig}, \mathsf{Verify})$ is euf-cma, $\mathcal{A}_{II}$ has overwhelming probability to have obtained the corresponding time capsule signature σ'^* from oracle TSig rather than forging σ' on its own.

The game between the IDTR One-wayness challenger and adversary $\mathcal{B}$ starts when the challenger generates $D_{\mathcal{R}}$ and $mtd_{\mathcal{R}}$ by running $\mathsf{Gen}(1^k)$. After receiving $D_{\mathcal{R}}$ from the challenger, $\mathcal{B}$ interacts with $\mathcal{A}_{II}$ in Game II as follows:

$\mathcal{B}$ gets a random public/private key pair $(pk, sk) \leftarrow \mathsf{Set}(1^k)$, sets $(upk, usk) = (pk, sk)$, $tpk = D_{\mathcal{R}}$, and gives (tpk, upk) to $\mathcal{A}_{II}$.

$\mathcal{B}$ manages a list $L = \{(m_i, t_i, s_i, c_i, d_i)\}$ for answering $\mathcal{A}_{II}$'s queries on PreHatch. Let q_{TSig} be the total number of TSig queries made by $\mathcal{A}_{II}$ and r be the random number chosen by $\mathcal{B}$ in the interval of $[1, q_{\mathsf{TSig}}]$. $\mathcal{B}$ responds to the i-th TSig query (m_i, t_i) as follows:

- If $i = r$, $\mathcal{B}$ sends t_r to the IDTR One-wayness challenger and receives a random commitment $c \in R_{t_r}$ from the challenger. $\mathcal{B}$ sets $c_r = c$ and computes $s_r = \mathsf{Sig}_{usk}(m_r \| c_r \| t_r)$. $\mathcal{B}$ returns $\sigma'_{t_r} = (s_r, c_r)$ to $\mathcal{A}_{II}$ and stores $(m_r, t_r, s_r, c_r, \perp)$ in the list L.
- If $i \neq r$, $\mathcal{B}$ gets a random commitment/witness pair (c_i, d_i) generated from $\mathsf{Sample}_{D_{\mathcal{R}}}$ and computes $s_i = \mathsf{Sig}_{usk}(m_i \| c_i \| t_i)$. $\mathcal{B}$ returns $\sigma'_{t_i} = (s_i, c_i)$ to $\mathcal{A}_{II}$ and stores $(m_i, t_i, s_i, c_i, d_i)$ in L.

To simulate oracle TRelease, say on query t_i from $\mathcal{A}_{II}$, $\mathcal{B}$ relays t_i to the trapdoor extraction oracle Extract simulated by the IDTR One-wayness challenger and gets $td_{\mathcal{R}_{t_i}}$. If $t_i = t_r$, $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ returns $z_{t_i} = td_{\mathcal{R}_{t_i}}$ to $\mathcal{A}_{II}$.

To simulate oracle PreHatch, say on query (m_i, t_i, s_i, c_i), $\mathcal{B}$ checks whether the query is in the list L or not. If (m_i, t_i, s_i, c_i) is in the list L, and equal to (m_r, t_r, s_r, c_r), $\mathcal{B}$ aborts. If (m_i, t_i, s_i, c_i) is in the list L, and not equal to (m_r, t_r, s_r, c_r), $\mathcal{B}$ extracts d_i from L and gives a prehatched signature $\sigma_{t_i} = (s_i, c_i, d_i)$ to $\mathcal{A}_{II}$. If (m_i, t_i, s_i, c_i) is not in L, since $\mathcal{A}_{II}$ does not know usk and this case implies that s_i is not generated by $\mathcal{B}$ on $m_i \| c_i \| t_i$, due to the euf-cma assumption of the underlying standard signature, it is negligible to have s_i be valid. Hence this case will happen with negligible chance. Therefore, for this case, $\mathcal{B}$ returns $\perp$.

When $\mathcal{A}_{II}$ outputs the forgery (m^*, t^*, σ^*) where $\sigma^* = (s^*, c^*, d^*)$, $\mathcal{B}$ verifies whether the forgery passes the verification algorithm Ver, and $(m^*, t^*, s^*, c^*) = (m_r, t_r, s_r, c_r)$. If so, $\mathcal{B}$ outputs the witness d^*. Otherwise, it chooses a d_B randomly and outputs d_B. The probability that $\mathcal{B}$ does not abort during the simulation and has a right guess of r is at least $1/q_{\mathsf{TSig}}$ since r is randomly chosen. Therefore, if $\mathcal{A}_{II}$ forges with a probability ϵ, $\mathcal{B}$ succeeds in breaking the One-wayness of IDTR with probability $\epsilon \geq \epsilon / q_{\mathsf{TSig}}$.

Security Against Game III: To show the security against Game III, we convert any adversary $\mathcal{A}_{III}$ which wins in Game III to a forger $\mathcal{F}$ against the underlying standard signature scheme. $\mathcal{F}$ gets pk as an input, and has access to signing oracle Sig of the signature scheme as described in the euf-cma model [16]. $\mathcal{F}$ simulates Game III for $\mathcal{A}_{III}$ as follows:

$\mathcal{F}$ gets $(D_{\mathcal{R}}, mtd_{\mathcal{R}}) \leftarrow \mathsf{Gen}(1^k)$ and gives $(upk, tpk, tsk) = (pk, D_{\mathcal{R}}, mtd_{\mathcal{R}})$ to $\mathcal{A}_{III}$. $\mathcal{F}$ simulates TSig on query (m_i, t_i) by getting $(c_i, d_i) \leftarrow \mathsf{Sample}_{D_{\mathcal{R}}}(t_i)$ and obtaining $s_i \leftarrow \mathsf{Sig}(m_i \| c_i \| t_i)$ from signing oracle Sig. $\mathcal{F}$ stores (m_i, c_i, d_i, t_i) in a list $L = \{(m_i, c_i, d_i, t_i)\}$ for answering $\mathcal{A}_{III}$'s queries to PreHatch. To simulate PreHatch on query (m_i, t_i, s_i, c_i), $\mathcal{F}$ verifies if s_i is a valid signature on $m_i \| c_i \| t_i$.

- If s_i is valid, $\mathcal{F}$ checks if (m_i, c_i, t_i) is in the list L. If so, $\mathcal{F}$ gives the corresponding d_i to $\mathcal{A}_{III}$. Otherwise, s_i is a new signature value and $\mathcal{F}$ succeeds in producing a new forgery s_i on $m_i \| c_i \| t_i$.
- If s_i is not valid, $\mathcal{F}$ returns $\perp$ due to the same reason as shown above in the Security Against Game II.

Finally, when $\mathcal{A}_{III}$ outputs a forgery (m^*, t^*, σ_t^*) where $\sigma_t^* = (s^*, c^*, d^*)$, $\mathcal{F}$ outputs a signature s^* on message $m^* \| c^* \| t^*$. Therefore, if $\mathcal{A}_{III}$ succeeds with a probability ϵ, $\mathcal{F}$ succeeds in producing a new forgery with at least probability ϵ. □

6 Distinguishable Time Capsule Signature

As discussed in Sec 3.3, the ambiguity between a prehatched signature and a hatched signature may not be desirable in practice. Moreover, in some scenarios, there are demands to distinguish a prehatched signature from a hatched signature. In the case of debt repayment, as an example, if a borrower repays his debt before the actual due date, he can improve his credit history or get extra reward. Then the signature for validating the payment check should be determined on whether it is prehatched or hatched.

Our generic construction of time capsule signature can be extended to capture the need of distinguishability. In the following, we first extend the IDTR (identity-based trapdoor relation). We then modify our construction based on the extended IDTR.

6.1 Extended IDTR

The extended IDTR (identity-based trapdoor relation) has seven PPT algorithms associated (Gen, Sample, Reveal, Extract, Invert, CheckS, CheckI). The settings of Gen, Sample, Extract, and Invert remain the same as in IDTR. Reveal is used to print out a 'sampled' witness. Check in IDTR is replaced by two separated functions CheckS and CheckI, which are used to check the validity of sampled witness and inverted witness, respectively.

- Reveal: Given $c \in \mathcal{L}_{R_{id}}$, if there is a pair (c, d) in a sampling list defined by $List = \{(c, d, id)\}$ where $(c, d) \leftarrow \mathsf{Sample}_{D_{\mathcal{R}}}(id)$, $\mathsf{Reveal}_{id}(c)$ returns witness d. Otherwise, it returns $\perp$.

– CheckS: For any $(c, d) \leftarrow \mathsf{Sample}_{D_{\mathcal{R}}}(id)$, we have $\mathsf{CheckS}_{D_{\mathcal{R}},id}(c, d)$ return 1 (accept); otherwise, it returns 0 (reject).
– CheckI: Given $(c, \hat{d}) \in R_{id}$, where $\hat{d} \leftarrow \mathsf{Invert}_{td_{\mathcal{R}_{id}}}(c)$, $\mathsf{CheckI}_{D_{\mathcal{R}},id}(c, \hat{d})$ returns 1 (accept). Otherwise, it returns 0 (reject).

With this modification, the extended IDTR can be used to achieve another property called Hiding, which is beyond One-wayness and Soundness. Hiding captures a malicious system master (e.g. a malicious Time Server) who aims to forge a sampled witness for a given commitment.

– Hiding: Let O_{Sample} and O_{Reveal} be oracles simulating the procedures of Sample and Reveal, respectively, where O_{Sample} only returns a commitment for each query. Let $Query(A, O_{\mathsf{X}})$ be the set of queries an algorithm A asked to O_{X}, where X can be Sample or Reveal. Note that A can only obtain commitment c from O_{Sample}. It states that the following probability is negligible for all PPT algorithm A:

$$Pr[\mathsf{CheckS}_{D_{\mathcal{R}},id^*}(c^*, d^*) = 1 \wedge c^* \in Query(A, O_{\mathsf{Sample}}) \wedge c^* \notin Query(A, O_{\mathsf{Reveal}}) | (D_{\mathcal{R}}, mtd_{\mathcal{R}}) \leftarrow \mathsf{Gen}(1^k); (c^*, d^*, id^*) \leftarrow A^{O_{\mathsf{Sample}} O_{\mathsf{Reveal}}}(D_{\mathcal{R}}, mtd_{\mathcal{R}})]$$

For One-wayness and Soundness, we refer readers to Sec. 4 for their definitions while replacing Check in One-wayness with CheckS and CheckI, and replacing Check in Soundness with CheckI.

6.2 A Generic Construction of Extended IDTR

Let $\mathcal{E}$ be an IBE scheme. Let $\mathcal{E}.\mathsf{Enc}(mpk, id, m; r)$ be $\mathcal{E}$'s encryption algorithm which encrypts message m under identity id and master public key mpk using randomness r. We say that $\mathcal{E}$ is injective if it satisfies the following condition:

Injective: For every master public key mpk and every identity id, for every ciphertext e of a message m under mpk and id, there exists *at most one* randomness r such that $e = \mathcal{E}.\mathsf{Enc}(mpk, id, m; r)$.

In the literature, many IBE schemes are injective, like BasicIdent and FullIdent proposed by Boneh and Franklin [7], and Waters' IBE [23].

Suppose $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ is an injective encryption scheme with IND-ID-CPA security [7], $\mathcal{MSP}$ is the message space, and $\mathcal{RSP}$ is the space of randomness used in $\mathcal{E}.\mathsf{Enc}$. Let $f : \{0,1\}^{\ell(k)} \rightarrow \mathcal{RSP}$ be a one-way function (or a hash function). We now give a generic construction of extended IDTR as follows.

– Gen: On input 1^k, run $\mathcal{E}.\mathsf{Setup}(1^k)$ to generate a master key pair (mpk, msk) and set $\mathcal{D}_{\mathcal{R}} = mpk$ and $mtd_{\mathcal{R}} = msk$.
– Sample: On input $D_{\mathcal{R}}$ and id, randomly select $m \in \mathcal{MSP}$ and $s \in \{0,1\}^{\ell(k)}$, compute $r = f(s)$, and run $\mathcal{E}.\mathsf{Enc}(\mathcal{D}_{\mathcal{R}}, id, m; r)$ to generate a ciphertext e of m under the identity id. Store $(id, c, d) = (id, (e), (m, s))$ into a sampling list List and return (c, d).

- Extract: Given $mtd_{\mathcal{R}}$ and id, run $\mathcal{E}.\mathsf{Extract}(mtd_{\mathcal{R}}, id)$ to generate the corresponding private key sk_{id} with respect to the identity id, and return $td_{\mathcal{R}_{id}} = sk_{id}$.
- Invert: Given $td_{\mathcal{R}_{id}}$ and c, run $\mathcal{E}.\mathsf{Dec}(\mathcal{D}_{\mathcal{R}}, td_{\mathcal{R}_{id}}, c)$ to get the plaintext m, and return $\hat{d} = (td_{\mathcal{R}_{id}}, m)$.
- Reveal: Given $c \in \mathcal{L}_{\mathcal{R}_{id}}$, check if there is an entry for c in the sampling list $\mathsf{List} = \{(id, c, d)\}$. If so, return the corresponding d; otherwise return $\perp$.
- CheckS: For any pair (c, d) output by algorithm Sample on input $\mathcal{D}_{\mathcal{R}}$ and id, we have that $(c, d) = ((e), (m, s))$. Check if $\mathcal{E}.\mathsf{Enc}(D_{\mathcal{R}}, id, m; f(s)) = e$. If so, return 1 (accept); otherwise return 0 (reject).
- CheckI: For any $(c, \hat{d}) \in \mathcal{R}_{id}$, where $\hat{d} \leftarrow \mathsf{Invert}_{td_{\mathcal{R}_{id}}}(c)$, we have that $(c, \hat{d}) = ((e), (sk_{id}, m))$. Check if $m = \mathcal{E}.\mathsf{Dec}(D_{\mathcal{R}}, sk_{id}, e)$. If so, return 1 (accept); otherwise, return 0 (reject).

Theorem 2. *The above scheme is a secure extended IDTR scheme, provided that the underlying IBE scheme $\mathcal{E}$ is IND-ID-CPA secure, and function f is one-way.*

Due to page limitation, we will provide the proof of Theorem 2 in the full paper [17].

6.3 Extended Time Capsule Signature

The Ver function in time capsule signature can also be separated into two functions accordingly: VerP is to verify the prehatched signature, VerH is to verify the hatched signature. The generic construction of time capsule signature based on IDTR can then be modified as follows:

- VerP: For a given prehatched signature $\sigma_t = (s, c, d)$ on m, a verifier checks if $\mathsf{CheckS}_{tpk,t}(c, d)$ outputs 1 and $\mathsf{Verify}_{upk}(m\|c\|t, s)$ outputs 1. If both of the verifications are correct, output 1; otherwise, output 0.
- VerH: For a given hatched signature $\sigma_t = (s, c, \hat{d})$ on m, the verifier compares the current time with t. If the current time is smaller than t, it returns $\perp$ indicating that hatching cannot be done at the moment. Otherwise, the verifier determines if $\mathsf{CheckI}_{tpk,t}(c, \hat{d})$ outputs 1 and $\mathsf{Verify}_{upk}(m\|c\|t, s)$ outputs 1. If both of the verifications are correct, output 1; otherwise, output 0.

In the construction of [13], the Time Server should be fully trusted and it is assumed that the Time Server would not collude with any malicious user and release some time trapdoor z_t before t. Otherwise, there is no way to distinguish whether a signature is pre-hatched by the actual signer or hatched by a malicious Time Server. In our distinguishable time capsule signature, we make this act of a malicious Time Server distinguishable. Below is the formal security model. Let $k \in \mathbb{N}$ be a security parameter.

Game IV: Let $\mathcal{S}_{IV}$ be the game simulator.

1. $\mathcal{S}_{IV}$ executes $\mathsf{TSSetup}(1^k)$ to get (tpk, tsk) and $\mathsf{UserSetup}(1^k)$ to get (upk, usk).

2. $\mathcal{S}_{IV}$ runs $\mathcal{A}_{IV}$ on upk, tpk and tsk. During the simulation, $\mathcal{A}_{IV}$ can make queries onto TSig, and $\mathsf{PreHatch}$.
3. $\mathcal{A}_{IV}$ is to output (m^*, t^*, σ^*).

$\mathcal{A}_{IV}$ *wins* if $\mathsf{VerP}(m^*, \sigma^*, upk, tpk, t^*) = 1$, and $\mathcal{A}_{IV}$ has never queried $\mathsf{PreHatch}(m^*, t^*, \cdot)$.

A time capsule signature scheme is secure in **Game IV** if for all PPT algorithm $\mathcal{A}_{IV}$, it is negligible for $\mathcal{A}_{IV}$ to win the game.

Now we prove the security of our proposed time capsule signature scheme against Game IV.

Theorem 3. *The extended time capsule signature scheme is secure in Game IV if the underlying extended IDTR scheme has the Hiding property, and the standard signature scheme is existentially unforgeable against adaptive chosen message attacks (euf-cma) [16].*

Due to page limitation, we will provide the proof of Theorem 3 in the full paper [17].

7 Conclusion

Time Capsule Signature is a promising technique for various E-Commerce applications. In this paper, we improve the security model of time capsule signature, construct a generic and provably secure time capsule signature scheme based on a new primitive called identity-based trapdoor relation (IDTR), and show that IDTR can be implemented efficiently by proposing two instantiations. We believe that the IDTR itself is of independent interest and may be implemented by other techniques. We leave these as our further investigations.

References

1. Asokan, N., Shoup, V., Waidner, M.: Optimistic fair exchange of digital signatures. IEEE Journal on Selected Areas in Communications 18(4), 593–610 (2000)
2. Bellare, M., Goldwasser, S.: Encapsulated key escrow. Technical Report 688, MIT/LCS/TR (1996)
3. Bellare, M., Goldwasser, S.: Verifiable partial key escrow. In: ACM Conference on Computer and Communications Security, pp. 78–91. ACM Press, New York (1997)
4. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: ACM Conference on Computer and Communications Security, pp. 62–73. ACM Press, New York (1993)
5. Blake, I.F., Chan, A.C.-F.: Scalable, server-passive, user-anonymous timed release public key encryption from bilinear pairing. In: ICDCS (2005)
6. Boneh, D., Boyen, X.: Efficient selective-id secure identity based encryption without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, Springer, Heidelberg (2004)
7. Boneh, D., Franklin, M.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)

8. Boneh, D., Gentry, C., Shacham, H., Lynn, B.: Aggregate and verifiably encrypted signatures from bilinear maps. In: Biham, E. (ed.) Advances in Cryptology – EUROCRPYT 2003. LNCS, vol. 2656, pp. 416–432. Springer, Heidelberg (2003)
9. Boneh, D., Naor, M.: Timed commitments. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, p. 236. Springer, Heidelberg (2000)
10. Chen, L., Harrison, K., Smart, N., Soldera, D.: Applications of multiple trust authorities in pairing based cryptosystems. In: Davida, G.I., Frankel, Y., Rees, O. (eds.) InfraSec 2002. LNCS, vol. 2437, pp. 260–275. Springer, Heidelberg (2002)
11. Cheon, J.H., Hopper, N., Kim, Y., Osipkov, I.: Timed-release and key-insulated public key encryption. Cryptology ePrint Archive, Report 2004/231 (2004)
12. Dodis, Y., Reyzin, L.: Breaking and repairing optimistic fair exchange from PODC 2003. In: ACM Workshop on Digital Rights Management (DRM), October 2003, ACM Press, New York (2003)
13. Dodis, Y., Yum, D.: Time capsule signature. In: Patrick, A.S., Yung, M. (eds.) FC 2005. LNCS, vol. 3570, pp. 57–71. Springer, Heidelberg (2005)
14. Garay, J.A., Jakobsson, M.: Timed release of standard digital signatures. In: Blaze, M. (ed.) FC 2002. LNCS, vol. 2357, pp. 168–182. Springer, Heidelberg (2003)
15. Garay, J.A., Pomerance, C.: Timed fair exchange of standard signatures. In: Wright, R.N. (ed.) FC 2003. LNCS, vol. 2742, pp. 190–207. Springer, Heidelberg (2003)
16. Goldwasser, S., Micali, S., Rivest, R.: A digital signature scheme secure against adaptive chosen-message attack. SIAM J. Computing 17(2), 281–308 (1988)
17. Hu, B.C., Wong, D.S., Huang, Q., Yang, G., Deng, X.: Time capsule signature: Efficient and provably secure constructions. Cryptology ePrint Archive (2007) http://eprint.iacr.org
18. May, T.C.: Timed-release crypto (1993) http://www.cyphernet.org/cyphernomicon/chapter14/14.5.html
19. Mont, M.C., Harrison, K., Sadler, M.: The HP time vault service: Exploiting IBE for timed release of confidential information. In: WWW (2003)
20. Rivest, R.L., Shamir, A., Tauman, Y.: How to leak a secret. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer, Heidelberg (2001)
21. Rivest, R.L., Shamir, A., Wagner, D.A.: Time-lock puzzles and timed-release crypto. Technical Report 684, MIT/LCS/TR (1996)
22. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
23. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
24. Zhang, M., Chen, G., Li, J., Wang, L., Qian, H.: A new construction of time capsule signature. Cryptology ePrint Archive, Report, 2006/113 (2006) http://eprint.iacr.org

A New Variant for an Attack Against RSA Signature Verification Using Parameter Field

Yutaka Oiwa, Kazukuni Kobara, and Hajime Watanabe

Research Center for Information Security (RCIS),
National Institute of Advanced Industrial Science and Technology (AIST),
1-18-13-1102 Sotokanda, Chiyoda-ku, Tokyo, Japan

Abstract. We present a method to create a forged signature which will be verified to a syntactically well-formed ASN.1 datum, when certificate authorities use small RSA public exponents such as 3. Our attack is related to the technique which Daniel Bleichenbacher reported recently, but our forged signature is well-formed ASN.1 datum, unlike Bleichenbacher's original attack: thus our new attack is still applicable to certain implementations even if these are immune to the Bleichenbacher's attack. We have also analyzed the parameters which enable our attack and Bleichenbacher's, and found that both attacks are possible with the combination of existing public keys of widely-trusted certificate authorities and existing real-world implementations. We have already reported the vulnerability to developers of both GNUTLS and Mozilla NSS to fix their implementations.

List of Keywords: vulnerability, attacks, certificate verification.

1 Introduction

In this paper we present an attack against some implementations of RSA signature verification for small public exponents. More precisely, given (n, e) an RSA public key of a certificate authority (CA), and given h a digest value of a certificate, we present a method to generate a forged signature which, after verified by the public key, will be accepted as a "correct" RSA signature data by several RSA implementations.

The attack is a variant of the attack which Daniel Bleichenbacher presented at the rump session of CRYPTO2006 [1]. His attack generates a syntactically ill-formed signature with some garbage data at the tail of decoded data. Our version of the attack, instead, generates at least syntactically well-formed data, enclosing similar "garbage" data inside the DER data packet. Therefore, some software (e.g. GNUTLS) is vulnerable to our attack, although it is not vulnerable for Bleichenbacher's attack.

The possibility of exploits using such garbage data seems to be understood by several independent parties: at least OpenSSL has added a check routine for those garbage data before we reported a specific attack [8]. Our contributions are (1) we found similar misimplementation in other two SSL implementations

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 143–153, 2007.

(GNUTLS [4] and Mozilla NSS), and (2) we constructed a practical attack and give an analysis on it.

In October 2006, NIST has published a technical notice [6] about the vulnerability which Bleichenbacher has discovered. However, the notice urges implementors to ensure that they checks the non-existence of garbages at the tail of signature messages, which Bleichenbacher has used for constructing the attack, but it overlooks about other possibilities for attacks, like one presented in this paper. Thus, for example, the notice cannot address the issue on GNUTLS (which is not vulnerable for Bleichenbacher's original attack). We believe that it is important to share the technical backgrounds of such vulnerabilities with researchers and implementors by putting it into the form which can be easily referred, so that they can not only fix their implementations but also avoid resurrecting the bugs in future software.

The rest of the paper is organized as follows: we describe Bleichenbacher's original attack in Section 2. In Section 3 we present our new attack and the precise way of generating forged signatures. The next section analyzes the conditions and possible extensions for our attack. A real case of misimplementations and measures we have taken are described in Section 5. Section 6 discusses about possibilities of similar vulnerabilities caused by other misimplementations. In Section 7 we conclude this paper and gives some suggestions for implementors of RSA signature verification.

2 The Bleichenbacher's Attack

In this section we describe the original attack which Bleichenbacher presented at the rump session of CRYPTO2006. It uses the fact that some TLS implementations do not check whether there are any excess data after the ASN.1 packet in the decoded message datum. A correct RSASSA-PKCS1-v1_5 signature message (see [10] for details) with MD5 digest, after RSA verification primitive (*RSAVP1*) is applied, looks like the following (shown in hexadecimal representation):

```
0x0001FF.....................FF
003020300c06082A864886F70D020505000410[ h ],
```

where [h] is the 128-bit MD5 digest value of the certificate signed. A DER-encoded DigestInfo block for signature data containing h, which is shown in the second line, is padded with the PKCS #1 block type 1 padding in the first line. The length of the whole data is adjusted to the length of CA's public key before applying the signing primitive using the CA's secret key.

Bleichenbacher's attack generates a forged signature which will be decoded by the verification primitive to the message

```
0x0001FF......FF
003020300C06082A864886F70D020505000410[ h ]
garbage
```

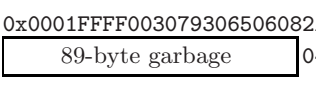
Since the RSA verification primitive $RSAVP1(s) = s^e \bmod n$, if the above message become a perfect cube of an integer, and if the public exponent e of CA's key is 3, to reverse the verification operation (to make a signature which will generate the above message) is just to take a cubic root, disregarding n. As there are a lot of garbage bits in the above message, it is easy to adjust the garbage so that the whole datum becomes a perfect cube. Most simply, it can be done by setting the "garbage" to 0, taking its cubic root, then taking its ceiling. Of course the datum above is an invalid message as a verified signature, but many existing implementations such as OpenSSL and Mozilla NSS have accepted such a signature.

Bleichenbacher's presentation at CRYPTO2006 was a bit more interesting than the above description: he says that even taking a cubic root is not required when the public key length is a multiple of 3, because if the padding length is neatly adjusted, it can be factored to a perfect cube by using a pencil and paper. Note that it is not the key element of the attack, however.

3 The Variant: The New Attack on "Parameters" Field

In this section we describe the attack we have reported to the developers of GNUTLS [7] and Mozilla NSS [5]. Similarly to the Bleichenbacher's attack, our attack targets a CA's public key with a small exponent. This attack is applicable for both MD5 and SHA-1 hash algorithms when the public key length is 1024 bits or more, and also for other hash algorithms such as SHA-256 and SHA-512 if the public key length is long enough. For simplicity, the rest of this section assumes that the key length of CA's public key is 1024 bits, the exponent e is 3, and the MD5 digest algorithm is used.

The attack aims to create a signature which generates the following message after decoded by *RSAVP1* verification primitive (i.e., *cubed*),



The "meaning" of this datum is as follows:

```
0x0001FFFF00                        ;; PKCS#1 padding
  3079                              ;; digestInfo: SEQUENCE
    3065                            ;; digestAlgorithm: SEQUENCE
      06082A864886F70D0205          ;; algorithm: OID (MD5)
      0459 (89 bytes of garbage)    ;; [parameters]
    0410 (16 bytes of h)            ;; digest: OCTET STRING
```

Unlike Bleichenbacher's attack, the whole datum is now a well-formed PKCS #1-padded DER datum[1]. The total length of the datum is adjusted to 1024 bits.

[1] PKCS #1 states that FFs in the padding must at least be 8 bytes. In other words, it requires *signers* to choose long-enough keys to make a signature, depending on the length of the datum to be signed. Some implementations (e.g. OpenSSL) check this condition also as a *verifier*, but some implementations (e.g. GNUTLS) do not.

There is a usually unused field called "parameters" inside "digestAlgorithm" field. The purpose of that field seems to be putting additional data such as initial vectors required by some digest algorithms (e.g., HMAC-based one, we guess). The datatype of this field is depending on the hash algorithm specified in the "algorithm" field, and it is defined as "ANY DEFINED BY algorithm OPTIONAL". For simple digest algorithms such as MD5 and SHA-1, this field must be NULL (`05 00`). Correct implementations must check the type of this field after reading the object identifier (OID) in "algorithm". However, many implementations omit this check and simply treat this field as "ANY"[2]. The above exploit datum put a meaningless 89-byte OCTET-STRING in that field instead of NULL, which is later adjusted to make the whole datum a perfect cube, like Bleichenbacher's original attack.

3.1 The Algorithm

The construction of forged signature is now slightly complicated than that for the original: we now have to fix the bits both before and after the garbage in the verified message. For fixing the prefix part, we use the same method as that of Bleichenbacher's. And, for the postfix part, we combine another calculation.

When the digest value of the certificate to be forged (h) is an odd number, the following algorithm generates the forged signature s:

1. Let $M_{\min}$ be `0x0001FFFF00307930650608 2A864886F70D02050459` $\times 2^{856}$. This is the possible minimum value for the decoded message (after *RSAVP1* is applied) for this attack.
2. Let $S_{\min}$ be $\lceil \sqrt[3]{M_{\min}} \rceil$. This is the possible minimum value for the signed signature (before *RSAVP1* is applied).
3. Let h' be `0x0410` $\times 2^{128} + h$, where h is the digest value of the certificate to be forged.

   The value of 144-bit integer h' is the fixed postfix for the signature we are generating. (128 bits for the hash value and 16 bits for two fixed bytes "`0x0410`".)
4. Construct a 144-bit integer y which satisfies $y^3 \equiv h' \pmod{2^{144}}$. The integer y exists when h' is an odd number, and can be obtained by the following algorithm.
   (a) Let $y \leftarrow 1$.
   (b) Iterate i from 1 to 143, and for each i do:
   - If $y^3 \equiv h' \pmod{2^{i+1}}$, do nothing.
   - Otherwise, assign $y \leftarrow y + 2^i$. The relation $(y+2^i)^3 \equiv h' \pmod{2^{i+1}}$ always holds in this case.

   This step takes at most only 143 steps of cubing operation.[3]

[2] Some reasons of this misimplementations might be that (1) if a generic DER parser is used, it is not possible to check the type of the parameters field inside the parser (because the parser does not know the correct type), and (2) there is no digestAlgorithm which uses this field in their protocol implementations.

[3] In fact, for $e = 3$, lowest 3 bits of y can be simply computed as $h' \bmod 8$.

5. If $y < (S_{\min} \bmod 2^{144})$, let x be $\lceil S_{\min}/2^{144} \rceil \times 2^{144}$. Otherwise, let x be $\lfloor S_{\min}/2^{144} \rfloor \times 2^{144}$.
6. Let s be $x + y$. This s is the smallest integer not less than $S_{\min}$ and whose least significant 144 bits are equal to y. This s is the forged signature for the certificate with the digest value h.

If h is even, the step 4. in the above algorithm gives no solution. For this case, we can still perform the similar operation after adding the public modulus n of the CA's key to $M_{\min}$, and adding the lower 144 bits of n (this is an odd number) to h (so that the value $(s^3 \bmod n) = (s^3 - n)$ becomes the forged signature, instead of s^3).

3.2 An Example

The signature datum

```
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
0000000000014289F9E149612CAEC814
A4EEA8793CFDEBBBC8DCD61DCD56CB40
DEC7DBEE995670D0F2594318F68AB50D
```

is a forged signature for the certificate of digest

```
ABA548B7F12DF4577C26A9274CA37F95
```

signed by a CA with $e = 3$, because the cube of the datum becomes the message below.

```
0001FFFF003079306506082A864886F7
0D020504590000000000E25E4BE37765
C524507FBB23560E94B179E9D91E0BC0
2B900C78139FD1032E1E5AFF0B5B0ABC
0BA7FC8B7D019A67DF3116B536DE7018
0BF7D3EBC543FF46CAD6228C07D2D33E
27776166FE9708DCF6ABE89FEE080410
ABA548B7F12DF4577C26A9274CA37F95
```

The italic part above is the garbage parameter field. The calculation can be done instantly even by using any scripting languages with a big-integer functionality (e.g. Ruby).

3.3 Extentions to Different Parameters

This method is also applicable for other digest algorithms or other public key lengths, by taking an appropriate M_{min} so that it is a "correct" prefix for the

resulting signature as a DER packet. To use another digest algorithm, constants such as 144 and 128 must also be changed according to the digest length.

The method can also be extended to the attacks against CA's keys which uses public exponent e which is other than 3. However, it makes real attacks difficult, as discussed in the next section.

4 Analysis

4.1 Conditions for Successful Forgeries

For the resulting signature s to be accepted as a "valid" signature, the value decoded by the verification primitive (s^3) must have a correct ASN.1 format shown in Section 3. This gives some restrictions on public key lengths and digest algorithms.

Let x and y be the value computed in the previous section, then

$$\begin{aligned} s^3 &= (x+y)^3 \\ &= x^3 + 3x^2y + 3xy^2 + y^3 \\ &= M_{\min} + (x^3 - M_{\min}) + 3x^2y + 3xy^2 + y^3, \end{aligned}$$

holds. The value of last four terms must be fit in to the space shown as "garbage" and the digest value, and must not overwrite the fixed prefix part of the decoded ASN.1 message. Thus, the sum of four terms $(x^3 - M_{\min})$, $3x^2y$, $3xy^2$, and y^3 must not overwrite the bits from $M_{\min}$. In addition, the lower bits of s^3 must hold valid information for the hash value (h').

For a 1024-bit public key with the MD5 algorithm, the condition always holds (at least when GNUTLS or Mozilla NSS is used as a target: see Section 5 for the OpenSSL's case). This can be checked in the following way.

- Since x is about 337 bits and y is at most 144 bits, $3x^2y$ generates at most 820-bit value. Because the space left in $M_{\min}$ is 856 bits (see the definition of $M_{\min}$), this term does not overwrite the bits from $M_{\min}$.
- Other terms $x^3 - M_{\min}$, $3xy^2$, and y^3 are negligible compared to the above.
- As the lower 144 bits of x is zero, the lower 144 bits of s^3 matches the value h' ($= y^3 \bmod 2^{144}$).

We have also checked that the attack is also possible for SHA-1 hash algorithm with 1024-bit (or longer) public keys, in this case y is 176 bits, x's length is the same as that for MD5's case, and the space left for garbages and the digest is 880 bits (because the object identifier for SHA-1 is 24-bit shorter than that for MD5). If any hash algorithms which have longer digest values (for example, SHA-256) are used, the length of public key must become larger.

4.2 Conditions with Other Public Exponents

If the public exponents get much larger than 3, the exploit become infeasible. we analyze the conditions for the same attack with the case of $e > 3$. Hereafter,

let L_P be the length of the fixed prefix in $M_{\min}$ which must be unmodified by other terms[4], L_H be the length of the fixed postfix (the length of y), and $|n|$ be the length of the public key.

For a valid forged signature to exist for any h, it is sufficient that any integer between $\lceil \sqrt[e]{M_{\min}} \rceil$ and $\lfloor \sqrt[e]{M_{\min}} \rfloor + 2^{L_H}$ becomes a well-formed signature. It means that

$$M_{\min} + 2^{|n|-L_P} > \left(\sqrt[e]{M_{\min}} + 2^{L_H} \right)^e$$

must hold, where $M_{\min} \sim 2^{|n|-15}$ (see the example value of the decoded forged signature in Section 3). We can see obviously that we need $\sqrt[e]{M_{\min}} \gg 2^{L_H}$. Assuming this, the right-hand side can be expanded as

$$M_{\min} + 2^{|n|-L_P} > M_{\min} + e \cdot 2^{L_H} \left(\sqrt[e]{M_{\min}} \right)^{e-1} + \cdots,$$

thus

$$\begin{aligned} 2^{|n|-L_P-L_H} &> e \left(\sqrt[e]{M_{\min}} \right)^{e-1} + \cdots \\ |n| - L_P - L_H &> \log_2 e + \frac{e-1}{e} \log_2 M_{\min} + \cdots \\ e|n| - e(L_P + L_H) &> e \log_2 e + (e-1)(|n| - 15) + \cdots \\ \therefore \quad |n| &> e \log_2 e - 15(e-1) + e(L_P + L_H) + \cdots \end{aligned}$$

is required.

For MD5's case (where $L_P = 168$ and $L_H = 144$) this condition means that public keys longer than about 1512 bits are required when $e = 5$. The required key length will become about 2120 bits when $e = 7$, 12416 bits when $e = 41$, and about 20.5 million bits when $e = 65537$. Considering current trends of the public key lengths, the attack is not realistic when e is about 33 (which makes the attack impossible even for 8192-bit public keys) or more.

Figure 1 shows the required public key lengthes for various e and hash algorithms used.

4.3 Comparison with the Bleichenbacher's Original Attack

In Bleichenbacher's original attack, as shown in Section 2, all garbages are put at the tail of the decoded signature. Izu et al. [3] have analyzed this case (named "extension 1" in their paper) based on information amounts analysis, but in this section we do the same thing in a bit more detailed way.

The Bleichenbacher's attack is correspond to the case where the L_P-bits prefix contains both DER.1 header and hash value (that is, L_P equals to $(L_P + L_H)$ in our attack), and there is no fixed postfix (i.e. $L_H = 0$).

[4] As DER format (ASN.1) uses a variable-length representation of integers, the actual value of L_P gets few bytes larger when the number of bytes in the garbages exceeds 127. We ignore this because the change contributes little to the result.

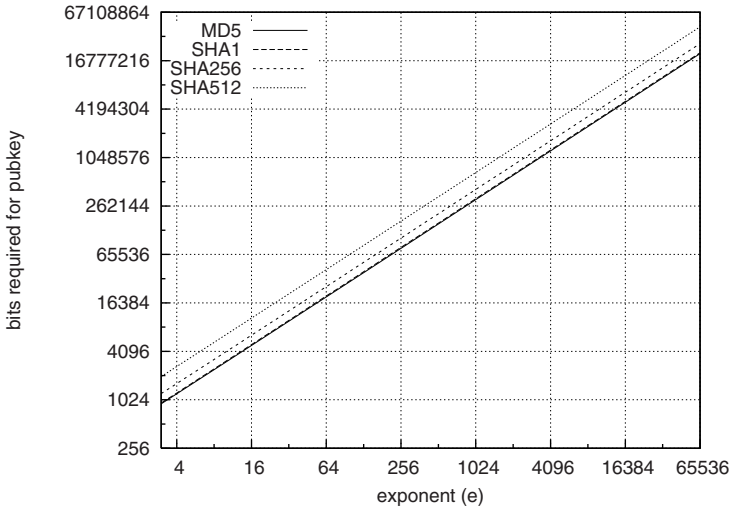


Fig. 1. Required public key lengthes for various e and hash algorithm used for signatures

As you can see, the condition for a successful attack only depends on the sum of fixed bits ($L_P + L_H$), and not on how those bits are distributed in the target value. Thus, those two attacks are as same as easy, regarding the required length of public keys.

Note that Izu et al. state that the attack is impossible for 1024-bit public keys, which contradicts our result. This is because they assume the OpenSSL's behavior (checking the length of padding as a signature verifier).

5 Vulnerabilities in Several Existing Libraries

Several existing libraries were vulnerable to this attack. We have reported this vulnerability to the developers of GNUTLS and Mozilla NSS (Netscape Security Service).

GNUTLS was vulnerable to this variant of attack and fixed in version 1.4.3 [4]. GNUTLS was not vulnerable for the original Bleichenbacher's attack.

Mozilla NSS was vulnerable for both our variant and the original attacks. The developers say that this variant was reported by several parties [5] independently and was fixed in Firefox 1.5.0.7.

OpenSSL was also vulnerable for both attacks. The developers found (at least possibility for) this variant by themselves and fixed their implementation in version 0.9.8c [9]. Interestingly, the variant attack for OpenSSL requires public key slightly longer than 1024 bits, because the software requires at least 8 bytes of `FF`s for PKCS#1 padding, which reduces the garbage space by 48 bits.

The impact of the variant is almost the same as Bleichenbacher's original attack: if any of trusted root certificate authorities (CAs) has used an RSA key

with its public exponent $e = 3$, anyone can create a forged certificate which seems to be signed by that CA. Among the root certificates distributed with Debian GNU/Linux, four root CAs use RSA keys with $e = 3$ (one is 1000-bit long, and the other three are 1024-bit long). In addition to this, if any trusted CAs (that can be using exponent other than 3) had signed a public key with $e = 3$ as an intermediate CA, there are also possibilities for forgeries. It is unknown there are such keys in the world.

6 Possibilities for Other Similar Exploits

We have also investigated several other possibilities for exploits. As the DER data format, which ASN.1 uses, is not always canonical, the representation for the signature has some flexibilities. Furthermore, there are several programming pitfalls which many implementors fall into. Currently, the following observations have been achieved:

- In GNUTLS, a parsing routine for OIDs suffers from integer overflow conditions. It does not properly check the overflow of integer elements of identifiers. For example, the datum "90 80 80 80 00" (represents 2^{32}) is parsed as 0. Furthermore, if several bytes, with most-significant bits set to 1, are prepended to the above datum, it will also be parsed as the same identifier. Fortunately, it seems difficult to exploit this mis-implementation by a similar method as our exploit, since attackers must control every eighth bits of the "garbages" to 1.
  OpenSSL does not suffer from this property, because it handles an OID as a DER-encoded string (in the canonical (shortest) form), not as a list of encoded integers.
- GNUTLS has another mis-implementation in the handling of integer overflow in the length field of DER elements. There is a code which checks for the overflow, but it does not detect every cases of overflow. However, by the same reason as the above, it seems not to be exploitable.

7 Conclusion

7.1 Summary

We have constructed a new variant of Bleichenbacher's recent attack against RSA signature verification process. The attack uses the "parameters" field inside the DER encoding of ASN.1 signature datum. All of three famous open-source TLS libraries were vulnerable to this attack, and these are now fixed.

7.2 Recommendations for Implementors

After announcement of the original attack by Bleichenbacher, NIST has released an announcement letter [6] regarding the attack. However, the announcement

only refers to garbage in the tail of signature message, but not to the parameters field inside the ASN.1 data structure. We suggest that they should also mention to the variant described as well as other possible fields which can have redundant representations.

We also propose several recommendations for possible implementors of RSA signature verification routines.

- First of all, implementors must carefully put every possible checks for the validness of the input data. Often it is said that protocol implementations should "be liberal in what you accept, and conservative in what you send" [2], but this should not be applicable for implementations of security protocols.
- When verifying PKCS #1 signature using RSASSA-PKCS1-v1_5, it is better to follow the method show in the PKCS #1 v2.1 document: i.e. instead of using generic ASN.1 parsers, the decoded signature should be directly compared against fixed bit patterns of valid signatures.
- They may reject signatures which is too small: The difficulties of reversing RSA signing operation depends on the use of the public modulus n in the verification primitive. Rejecting too small value for signatures (see Section 3.2 for an example of such signatures) will reject both Bleichenbacher's and our variants of the attack. The possibility to reject the valid signature in this method can be made negligible: for example, when signatures smaller than $2^{|n|-96}$ are rejected, there is only 2^{-80} possibility for rejecting a true signature.
- If it is possible, they should avoid use of CAs with small public exponents.
- The current version of OpenSSL implements much stricter checking for the length of the PKCS #1 padding than one specified in [10]. Paddings are specified to be at least 8 bytes, but these always become much longer in real applications. The stricter check effectively checks the length of the signature message before a padding is added. This might also be effective for preventing similar attacks.

References

1. Bleichenbacher, D.: Forging some RSA signatures with pencil and paper. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, Springer, Heidelberg (2006)
2. IETF Internet Engineering Task Force. Requirements for Internet Hosts—Application and Support. RFC 1123 (October 1989) http://www.ietf.org/rfc/rfc1123.txt
3. Izu, T., Shimoyama, T., Takenaka, M.: Analysis on Bleichenbacher's Forgery Attack. SCIS 2007 (January 2007)
4. Josefsson, S.: [gnutls-dev] Variant of Bleichenbacher's crypto 06 rump session attack. A security advisory posted to gnutls-dev mailing list (September 8, 2006) http://lists.gnupg.org/pipermail/gnutls-dev/2006-September/001205.html
5. Mozilla development team. Bugzilla bug #351079
6. National Institute of Standards and Technology (NIST, USA). An Attack on RSA Digital Signature (October 2006) http://csrc.nist.gov/news-highlights/RSA-statement_10-17-06_.pdf

7. Oiwa, Y. (reposted by Josefsson, S.): A repost of the original report for the GNUTLS's bug (26th September 2006) http://lists.gnupg.org/pipermail/gnutls-dev/2006-September/001240.html
8. The OpenSSL Team. A patch for RSA Signature Forgery bug (5th September 2006) http://www.openssl.org/news/patch-CVE-2006-4339.txt
9. The OpenSSL Team. An advisory for RSA Signature Forgery bug (CVE-2006-4339) (5th September 2006) http://www.openssl.org/news/secadv_20060905.txt
10. RSA Security Inc.: PKCS #1 v2.1: RSA Cryptography Standard (14th June 2002)

AutoPKI: A PKI Resources Discovery System*

Massimiliano Pala and Sean W. Smith

Dartmouth College, Computer Science Department,
6211 Sudikoff, Hanover, NH 03755, US
{pala,sws}@cs.dartmouth.edu
http://www.cs.dartmouth.edu

Abstract. The central goal of *Public Key Infrastructure (PKI)* is to enable trust judgments between distributed users. Although *certificates* play a central role in making such judgments, a PKI's users need more than just knowledge of certificates. Minimally, a relying party must able to locate critical parameters such the certificate repositories and certificate validation servers relevant to the trust path under consideration. Users in other scenarios may require other resources and services.

Surprisingly, locating these resources and services remains a largely unsolved problem in real-world X.509 PKI deployment. In this paper, we present the design and prototype of a new and flexible solution for automatic discovery of the services and data repositories are available from a *Certificate Service Provider* (CSP). This contribution will take real-world PKI one step closer to achieving its goal.

Keywords: PKI, Service Discovery, Certification Authority, Digital Certificates.

1 Introduction

The central goal of *Public Key Infrastructure (PKI)* is to enable trust judgments between distributed users. At is core, PKI depends on certificates: signed bindings of public keys to keyholder properties. Effective use of PKI requires use of these certificates; however, effective use of certificates requires many additional services, such as OCSP servers, CRL repositories, timestamping services, etc. As a consequence, client-side PKI tools need to be able to discover and use these services; server-side PKI tools need to be able to provide these services and enable client tools to discover them.

Unfortunately configuring these tools to carry out these tasks is painful for both server administrators and end users, thanks to badly written User Interfaces

* The authors would like to thank Stephen Kent, Frank Pooth, Ashad Noor, Sravan and all the PKIX WG for several discussions and comments. This work was supported in part by the NSF (under grant CNS-0448499), the U.S. Department of Homeland Security (under Grant Award Number 2006-CS-001-000001), and Sun. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of any of the sponsors.

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 154–169, 2007.

(UI) and overly detailed configurations. Certification Authorities barely publish access details on their official websites; even data as basic as the URLs for provided services and repositories are usually omitted. As a result, if a CA provides a new service (e.g. OCSP [1]) or a new data repository (e.g. LDAP [2]), users and administrators have difficulty learning of these changes. Furthermore, certificates already issued could not carry any sign of the new services. It is unlikely that users (and applications) will be easily aware of the new services if not directly contacted. This problem impacts even more on users from enterprises other than the issuing organization, as they have very limited knowledge about CA's practices and service locations.

In this paper, we present a new approach to provide a flexible way to automatically discover which services and data repositories are available from a CA. This flexibility would also facilitate interoperability across different infrastructures. Section 2 presents the core aspect of our solution: the design and the implementation of a new (and simple) *PKI Resource Query Protocol* (PRQP) easing PKI management both for administrators and final users. Section 3 presents our prototypes. Section 4 evaluates the performance of our prototypes and the effectiveness of our solutions. Section 5 reviews other approaches to solving the problem. Section 6 concludes with some directions for future work.

2 The PKI Resource Query Protocol

To solve this problem, we define the *PKI Resource Query Protocol* (PRQP) for finding any available PKI resource from a particular CA. In PQRP, the client and a *Resource Query Authority* (RQA) exchange a single round of messages:

1. the client requests a resource token by sending a request to the server;
2. the server replies back by sending a response to the requesting entity.

The client embeds zero or more resource identifiers (OIDs)—when specifying exactly the data the client is interested into—in the request token, in order to specify which subset of CA resources she wants. If the client does not specify any services by providing an empty list of OIDs in the request, all of the available data for a particular CA should be returned by the server in the response. The resources might be items that are (occasionally) embedded in certificates today—such as URLs for CRLs or OCSP or SCVP—as well as items such as addresses of the CA homepage address, the subscription service, or the revocation request.

Fig. 1 shows an example of this protocol: an SSL web server needs to retrieve the revocation status of a user's certificate. (Here, the Web server is the PQRP requesting client.) At first (step 1), the web server receives the user's certificate from the browser. The web server looks at the issuer identifier in the certificate and builds up a PRQP request asking the RQA for the location of the OCSP server of the issuing CA (step 2). The RQA provides (step 3) the web server with the URL of the requested service, as configured on the RQA. In this particular example only the OCSP URL is requested, and therefore only the locator for such service is put in the response. The web server, then, continues with the

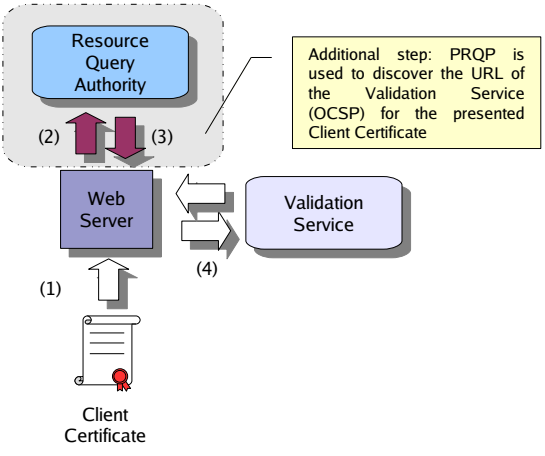


Fig. 1. A web server uses PRQP to help verify a user certificate

normal validation procedures (step 4) by using the provided URL to directly access the OCSP server.

2.1 Resource Query Authority (RQA)

In our protocol, an RQA can play two roles. First, a CA can directly delegate an RQA as the party who can answer queries about its certificates, by issuing a certificate to the RQA with a unique value set in the *extendedKeyUsage* (i.e. `prqpSigning`). The RQA will provide authoritative responses for requests regarding the CA that issued the RQA certificate. Alternatively, an RQA can act as *Trusted Authority* (TA) ("trusted" in the sense that a client simply chooses to trust the RQA's judgment). In this case, the RQA may provide responses about multiple CAs without the need to have been directly certified by them. In this case, provided responses are referred to as *non-authoritative*, meaning that no explicit trust relationship exists between the RQA and the CA. To operate as a TA, a specific extension (*prqpTrustedAuthority*) should be present in the RQA's certificate and its value should be set to `TRUE`. In this configuration the RQA may be configured to respond for different CAs which may or may not belong to the same PKI as the RQA's one.

2.2 The Message Format

A PRQP request contains several elements. The *protocol version* is used to identify whether the client or the RQA is capable to handle the request format. (Currently, `v1` is the only allowable value.) The `NONCE` (optional) is a random number long enough to assure that the client will produce it only once. The `ResourceRequestToken` identifies the resource (e.g. the CA and the service itself). The `MaxResponse` identifier tells the RQA the maximum number of `ResourceResponseToken` that may be present in the response.

The `ResourceRequestToken` contains a CA's target certificate identifier and optionally one or more `ResourceIdentifier` fields. If one or more are provided in the request, the RQA should report back the location for each of the requested services. If no `ResourceIdentifier` is present in the request, the response should carry all the available service locations for the specified CA (with respect to the `MaxResponse` constrain). Extensions can be used for future protocol enhancement.

The ***PRQP response*** also contains several elements. Again, the *protocol version* identifies the response's version. The `NONCE`, if present, binds the response to a specific request. The usage of the `NONCE` is meaningful only in signed responses and its value must be copied directly from the corresponding request. The status data structure (`PKIStatusInfo`) carries the response status and, in case of error, a description of the cause. The `ResourceResponseToken` is used to provide the pointers to the requested resources (one for each requested service). Optional Extensions may be added if requested.

Discussion. When designing the protocol, we paid special attention to several aspects: simplicity, security, message complexity, and RQA address distribution.

An important target of the protocol design was simplicity. By keeping the protocol very simple, its adoption would not add a big additional burden to PKI management, nor to applications and developers.

Security was another major concern. The PRQP provides URLs to PKI resources, therefore it only provides locators to data and services, not the real data. It still remains client's job to access the provided URLs to gather the needed data, and validate the data (e.g., via signatures or SSL). Because of this consideration, both the `NONCE` and the signature are optional in order to provide flexibility in how requests and responses are generated. Also, it is then possible to provide pre-computed responses in case the `NONCE` is not provided by the client. If an authenticated secure channel is used at the transport level between the client and the RQA (e.g. HTTPS or SFTP) signatures in requests and responses can be safely omitted.

We also analyzed the level of complexity of messages. Some type of services, e.g. delta CRLs, can be directly detected upon data downloading. However, if a client is looking for a specific version of a protocol or data type, a fine-grained query system can reduce server load by only permitting data download when the requesting client actually supports that version.

We considered two different candidates for the PRQP message format: *eXtensible Markup Language (XML)* and *Distinguished Encoding Rules (DER)*. The adoption of the *Abstract Syntax Notation (ASN.1)* to describe the data structures would let the software developer to provide either DER or XML-based implementations of the protocol. However we think that a DER-based implementation of PRQP is the best choice because of compatibility considerations with existing applications and APIs. Moreover DER encoded messages are smaller in size then XML encoded ones and almost all PKI aware applications already support it.

Last but not least considered issue was the distribution of the RQA's address. We envisage two different approaches. A first option would be to use the AIA

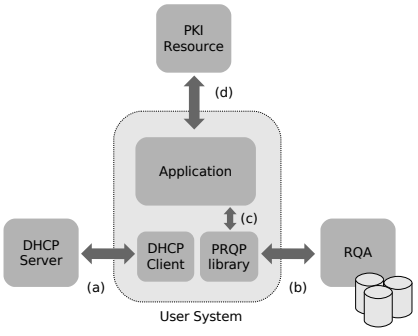


Fig. 2. AutoPKI General Design

and SIA extensions to provide pointers to RQAs. Although this approach seems to be in contrast with considerations provided in 5.1, we believe that by using only one extension to locate the RQA would provide an easy way to distribute the RQA's URL. The size of issued certificates would be smaller, thus providing a more space efficient solution. A second option is applicable mostly in LANs and consists in providing the RQA's address by means of DHCP. This method would be mostly used when a trusted RQA is locally available. These two techniques can then be combined together.

3 Prototype

To bring our solution into practice, we built *AutoPKI*, a prototype implementation of the libraries and software support to carry out PQRP in real PKI applications. The basic idea behind AutoPKI is to provide clients with addresses of PKI resources and to ease administrators and users from PKI configuration issues.

Our system differs from previously presented work in that it is aimed to provide an easy to use location service without providing 3^{rd} party validation or proxying services (e.g. does not provide services as SCVP [3]).

Our AutoPKI prototypes makes use of the three principal components (Fig. 2): an Extended DHCP client and server, a Resource Query Authority server, and a PRQP library.

3.1 The Extended DHCP Client and Server

To bring PRQP into the real world, we need to distribute the addresses of available RQAs. A naive option is to include the AIA and SIA extensions to carry the pointer to the RQA directly in digital certificates. This approach works if the CA of the target certificate provides an RQA. The extension contents would point to the available RQA and the client could directly discover services provided by the CA by querying the RQA.

```
# generated by /sbin/dhclient-script
queryauthority 130.192.1.23
queryauthority 130.192.1.59
```

Fig. 3. Example configuration file originated by the extended DHCP client (dhclient)

However, today we could not rely on the presence of RQA pointers in certificates, yet. Therefore we needed a way to provide clients with a pointer to a *local RQA* to query for resources provided by CAs that do not have RQAs. In fact if no RQA address is present in the certificate, a client application could use a default configured one.

The DHCP protocol provides sufficient flexibility for this purpose. In particular it allows the client to request the server to send specific information if needed. By modifying the configuration (to add specific options both to the client and the server) it is possible to store the provided addresses in a system-wide configuration file where applications could retrieve the local RQA address. Fig. 3 reports an example configuration file for PRQP[1]. In case no DHCP server is available, configuration can be provided by using a simple user interface, also common practice for DNS configuration on many systems.

3.2 PRQP Library

Our *PRQP library* can be invoked by applications in order to discover the address of a repository or a service. The implemented library provides applications with easy-to-use functions that handle both the generation and parsing of requests/responses as well as communication with the designated RQA. The library makes use of OpenSSL [4] for cryptographic operations such as signature generation and verification.

The library uses the configuration file generated by the DHCP client in order to retrieve the address of the RQA. Besides the low-level functionality needed to manage the PRQP data structures, we also implemented several high-level ones that help developers to integrate PRQP in their applications.

Along with the library, we built a *command line tool* that accepts an X.509 certificate and configuration options (e.g. names of requested resources) as input, and outputs the response both in PEM/DER and in a human-readable format. The output could then be parsed by any calling application in order to use the response's data.

When the command line tool is executed, it performs the following steps: (1) verifies the user input and load the certificate(s) whose services and/or data are requested; (2) builds up the PRQP request; (3) parses the global pki configuration file; (4) connects to the configured RQA server via TCP sockets; (5) sends the request to the server by using the HTTP protocol, in particular we use the POST method to upload the request to the server; (6) retrieves and parse the

[1] Our implementation stores the file as `/etc/pki.conf`

RQA response; (7) eventually saves the request and the response in separate files; (8) prints out the response details in text format;

By using the client library, steps from two throughout six can be performed automatically. For this purpose we provided the library with the `getpkiresources` function which handles all the PRQP details and returns a stack of `URL` structures back to the calling application. As the address of the RQA is directly taken by a global pki configuration file located in `/etc/pki.conf` which is generated by the extended DHCP client, no specific knowledge about the PRQP protocol is required out of the application.

3.3 RQA Server

Because of many similarities between PRQP and OCSP in the basic design we decided to implement our PRQP responder by using the OpenCA [5] OCSPD [6] package. This software uses OpenSSL and implements an OCSP responder over HTTP. To implement PRQP, we modified the software by leveraging the functionality provided by the PRQP library: ASN.1 functions capable to load, parse and save PRQP data structures by using the I/O abstraction layer of OpenSSL (i.e. the `BIO` interface); request and response processing functions; network communication functions to manage the simple HTTP POST method used between the client and the RQA.

Because of the simple design of OpenCA's OCSP responder, we could reuse much part of the original code in order to build PRQP responses instead of OCSP ones. Currently we support PRQP over HTTP only. We also defined the "`application/prqp−request`" and "`application/prqp−response`" HTTP content types for PRQP requests and responses, respectively.

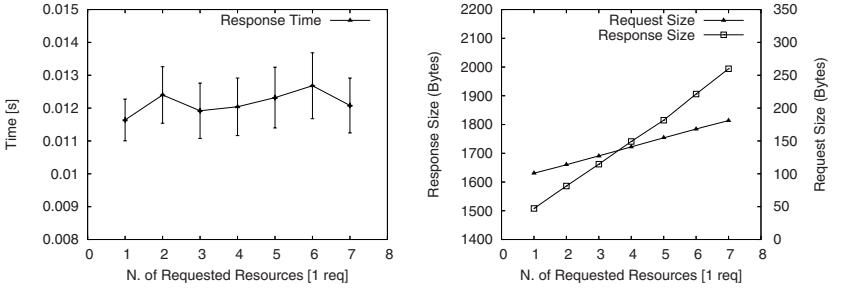
Our server is capable to act also as a TA by supporting multiple CAs by setting the appropriate configuration options. Each configured CA and its provided services have been grouped together in separated sections of the configuration file, thus being very easy to add new CAs to the server.

4 Evaluation

4.1 Performance

To test the system, we set up a testbed consisting of two computers connected over a switched Fast Ethernet LAN. On the first machine (Intel Core Duo @ 2.13 GHz, 4GB Ram) we installed the PRQP library and the PRQP server, while on the second one (Intel Pentium M @ 600MHz, 512MB Ram) we installed the PRQP library and the command line tool. Both systems were running Linux 2.6.18.3 Kernels on a Fedora Core 6 distribution. On the RQA server, we configured the pointers to services provided by our CA. Each response was digitally signed by using the RSA algorithm and 1024 bit keys, no crypto hardware was used.

On the client, we ran several tests that made use of the command line application to query the RQA server; in particular, we queried the server with an



(a) *PRQP Response times.* **(b)** *PRQP Message sizes.*

Fig. 4. PRQP Performance Stats

increasing number of requested pointers and repeated the experiment fifty times each. Although the PRQP enables for caching of responses during their validity period, no caching of responses has been used during our tests. Response times are reported in Fig. 4/a.

The results show that the overhead introduced by our system is small—and is almost negligible for the majority of today's applications. Moreover, no increase in response time has been noticed with the number of requested locators.

We also analyzed the size of PRQP requests and responses. Collected data are shown in Fig. 4/b. Generated requests are considerably smaller in size in respect to responses. The main reason for this is that we decided not to sign requests and not to include any certificate in the request (because we envisage this would be the most common scenario), while the server was signing and including its own certificate into generated responses.

We also noticed that the size of the responses grows more rapidly than the size of the requests. This is easily explained by the fact that in the request a single OID is used to identify the service, whilst in the response a more complex data structure is used that comprises the actual locators and the validity period for that information.

4.2 Solving the Problem

To demonstrate that PRQP solves the resource discovery problem, we analyzed the profile of a population of widely deployed CA certificates.

Our analysis has been focused on two different set of certificates, the ones embedded into popular browsers (i.e. Firefox, IE and Konqueror) and Mail User Agents (i.e., Thunderbird, Outlook and KMail) and the ones used in the main webpages of universities in USA and Europe. Table 1 shows the results coming from the study of the certificate profiles from the first set. Most of the certificates do not provide any pointers, thus making it really difficult for applications to correctly reach PKI related resources. For instance, in the Firefox/Thunderbird certificate store 66% of certificates has no pointers to any service or data repository

Table 1. Profile analysis for certificates embedded in major applications

	Firefox and Thunderbird	IE7 and Outlook	Konqueror and KMail
Total Certs	103	105	112
Self Signed Certs	98	105	103
Non Self Signed	5	0	9
Certs Without Pointers	68	87	89

(not even to CRLs), while for IE7/Outlook this percentage goes up to 82%. This problem is even worse when taking into account the lifetime of the certificates. Fig. 5 shows that the majority of the analyzed certificates present a validity period that spans over twenty or more years. Indeed, most certificates have lifetimes far longer than a typical URL—making it risky to solve the resource discovery problem by simply listing the URL in a certificate. The combination of the two analysis suggests that updating the contents of embedded CA certificates could be really difficult. PQRP would solve this problem.

To understand if these results were biased by the requirements imposed by the application policies, we turned our attention to the second set of certificates. By contacting all the universities websites [7,8] by using the HTTPS protocol—where supported—we were able to dump the list of certificates from the servers. After having retrieved all the certificates, we analyzed the results. From a pool of 2013 US universities, 1016 support HTTPS. The retrieved certificates were primarily issued by organizations external to the university (91.4%). In this scenario many certificates were pointing to the same providers, only 35 different CAs provide certificates for 929 different universities. Most of the certificates were providing pointers to CRLs and OCSP servers which where, most of the time, the same across different organizations. We think that the usage of certificates from commercial vendors, even when an internal CA exists, is due to the lack of real solutions to achieve interoperability between PKIs.

Results for European universities were quite different. In fact out of 2541 universities, only 745 support HTTPS. However, differently from the US case, the number of internally[2] issued certificates exceeds the number of certificates from external vendors. We were able to count 414 different providers of which 332 were "internal". In this environment, where there are many different vendors, we discovered that more that 54% of certificates did not provided any pointer to PKI resources. From this results it is therefore evident that also for EE certificates, such as the ones from university websites, solving the resource discovery problem by simply listing URLs in the certificate does not provide a working solution.

OASIS conducted a survey [9] about PKI deployment. This survey found that support for PKI is often missing from applications and operating systems and, when present, it is always inconsistent in the sense that it differs widely in what

[2] Issued by the university's internal CA.

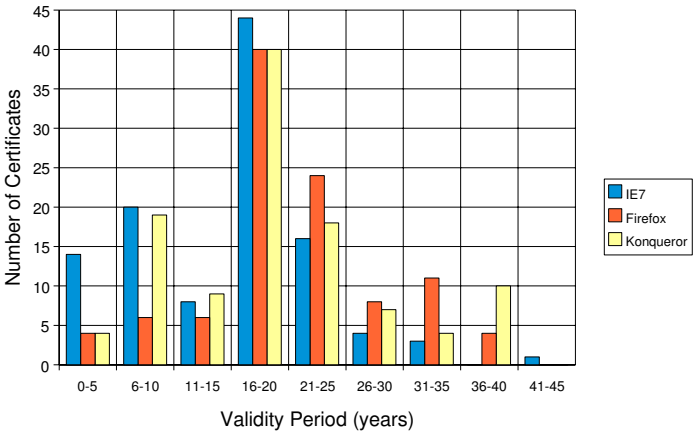


Fig. 5. Validity periods of CA certificates present in today browsers

is supported. This survey also found that current PKI standards are inadequate as they are often too complicated and implementations from different vendors rarely interoperate. It is interesting to notice that seven out of ten reported problems in the list are related to PKI usability and interoperability.

PRQP could help the deployment of PKIs and PKI-aware applications by providing a flexible way to automatically discover which services and data repositories are available from a CA. By providing such a mechanism, support for PKI basic operations (e.g. certificates validation) could be easily implemented also at the operating system level.

5 Related Work

Our work focuses on the PKI resources look up problem. This problem does not only involves the certificate retieval, but also the discovery of new services whenever they are made available by service providers. In prior work, we see three primary methods for clients to obtain pointers to PKI data: adopting specific certificate extensions; looking at easily accessible repositories (e.g. DNS, local database, etc.); and adapting existing protocols (e.g. Web Services).

5.1 Certificate Extensions

To provide pointers to published data, a CA could use the *Authority Information Access* (AIA) and *Subject Information Access* (SIA) extensions as detailed in RFC 3280 [10]. The former can provide information about the issuer of the certificate while the latter carries information (inside CA certificates) about offered services. The *Subject Information Access* extension can carry a URI to point to certificate repositories and timestamping services. Hence this extension allows to access services by several different protocols (e.g. `HTTP`, `LDAP` or `SMTP`).

Table 2. Analysis of AIA statistics

AIA Datatype	Firefox	IE7	Konqueror
OCSP	12	0	1
caIssuers	0	0	0
timeStamping	0	0	0
DVCS	0	0	0

Although encouraged, usage of the AIA and SIA extension is still not widely deployed. There are two main reasons for this. The first is the lack of support for such extensions in available clients. The second reason is that extensions are static, i.e. not modifiable. Indeed to modify or add new extensions, in order to have users and applications to be aware of new services or their dismissal, the certificate must be re-issued.

This would not be feasible for *End Entities* (EE) certificates, except during periodic reissuing, but it would be feasible for the CA certificate itself. The CA could retain the same public key and name and just add new values to the AIA extension in the new certificate. If users fetch the CA cert regularly, rather than caching it, this would enable them to become aware of the new services. Although this is possible, almost every available clients do not look for CAs certificates if they are already stored in clients' local database.

In any case, since URLs tend to change quite often while certificates persist for longer time frames, experience suggests that these extensions invariably point to URLs that no longer exist. Moreover considering the fact that the entity that issues the certificates and the one who runs the services may not be the same, it is infeasible that the issuing CA will reissue all of its certificate in case a server URL's changes. Therefore it is not wise to depend on the usage of AIA or SIA extensions for available services and repositories look up.

In Table 2 we report the contents of the AIA extensions in most diffused applications. As expected only OCSP pointers are present in a very small number of certificates (i.e., 11% for Firefox/Thunderbird, 0% for IE7/Outlook and Konqueror/KMail), whilst no pointer to other services are provided.

5.2 DNS Service Records

The SRV record or *Service record* technique is thought to provide pointers to servers directly in the DNS [11]. As defined in RFC 2782 [12], the introduction of this type of record allows administrators to perform operations rather similar to the ones needed to solve the problem we are addressing in this paper, i.e. an easily configurable PKI discovery service.

The basic idea is to have the client query the DNS for a specific SRV record. For example if an SRV-aware LDAP client wants to discover an LDAP server for a certain domain, it performs a DNS look up for *_ldap._tcp.example.com*

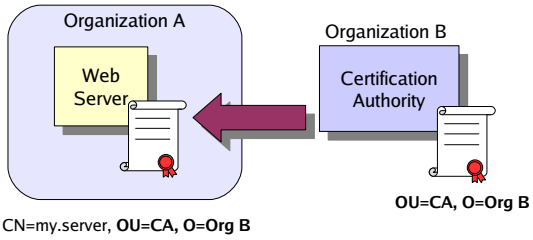


Fig. 6. The Certificate Authority from Organization "B" issues a certificate to the web server from Organization "A"

(the "_tcp" means the client requesting a TCP enabled LDAP server). The returned record contains information on the priority, the weight, the port and the target for the service in that domain.

The problem in the adoption of this mechanism is that in PKIs (unlike DNS) there is usually no fixed requirement for the name space used. Most of the time, there is no correspondence between DNS structure and data contained in the certificates. The only exception is when the *Domain Component* (DC) attributes are used in the certificate's Subject.

The DC attributes are used to specify domain components of a DNS name, for example the domain name "example.com" could be represented by using the `dc=com, dc=example` format. If the CA's subject field would make use of such a format, the *Issuer* field would allow client applications to perform DNS lookups for the provided domain where the information about repositories and services could be stored.

However, currently, the practice is very different. In fact it is extremely difficult for a client to map digital certificates to DNS records because the DC format is not widely adopted by existing CAs. As shown by our analysis, only one certificate[3] from IE7/Outlook store uses the domain components to provide a mapping between the certificate and an Internet Domain.

Recently a new proposal has been presented by the IETF PKIX Working Group [13] to standardize the usage of DNS records to locate PKI repositories. It emerged from discussion that, although a client has been implemented that is capable to locate an LDAP service for a specific e-mail address, the authors were not able to find anyone who announces their directory service in the DNS according to the specification.

Another example of the infeasibility of this solution is presented in Fig. 6. The figure depicts a very common scenario where an organization "A" buys a certificate for its web server from a CA ran by organization "B". Neither the contents of the distinguished name nor the contents of other fields in the certificate (e.g. `subjectAltName`) provide a pointer to the right domain where the query for RR records should be made.

[3] `/DC=com/DC=microsoft/CN=Microsoft Root Certificate Authority`

Moreover, the issuing organization may not even have control over the DNS records in case they need to be updated. In our example, if RR records are put in the DNS under the domain identified in the *Common Name* (CN) attribute of the web server's certificate, i.e. "`my.server`", the management of such records is not under control of the issuing organization ("B").

5.3 Web Services

Web Services [14] is a new technology using three different components to allow applications to exchange data: *SOAP (Simple Object Access Protocol)* [15], *WSDL (Web Services Description Language)* [16,17] and *UDDI (Universal Description Discovery and Integration)* [18].

By using UDDI, applications discover available Web Services (described by using the WSDL language) and interact with them by using SOAP to exchange data. Although Web Services provide a good tradeoff between flexibility and complexity (e.g. CORBA [19] offers much more possibilities but CORBA-oriented applications are difficult to implement), the format of exchanged messages is still complex. In fact, communication is handled by using XML [20] which is quite complex when compared to other binary formats like DER [21,22]. These aspects are to be considered with special care when it comes to mobile devices. XML-formatted messages require a large amount of computational power to be correctly processed and large bandwidth (messages are usually bigger in size). From our experience a message encoded by using the DER format is less than the 30% in size when compared to the corresponding XML format.

Another important aspect to be considered here is the *ease of integration* into existing applications. Every application dealing with digital certificates already have its own implementation for DER, while it is not true that XML is widely supported as well.

5.4 Local Network Oriented Solutions

Another approach to provide reliable information is to use existing protocols for service location such as *Jini* [23,24], *Universal Plug and Play* protocol (UPnP) [25,26] or *Service Location Protocol (SLP)* [27,28,29].

Jini is used to locate and interact with Java-based services. The main disadvantage of Jini is that it is tied to a specific programming language and it requires a lot of Java-specific mechanisms (e.g. object serialization, RMI [30] and code downloading) in order to function properly. In addition it provides many communication services which are quite complex and not really needed in our environment.

Like Jini, UPnP provides a mechanism to locate and to interact with services over a network. UPnP is also very complex as it involves the usage of different techniques like XML (SOAP) over HTTP. The protocol is peer-to-peer and it is aimed for home environments. There exists a service-discovery subset of UPnP, the *Simple Service Discovery Protocol (SSDP)* [31], which operates on HTTP over UDP. As UPnP, the SSDP is thought to be operated in small environments

and it is possible that administrators block UPnP from leaving the LAN or disable it for security reasons, in the same way they currently block/disable NetBIOS from leaving local networks.

The IETF defined the SLP to provide a service location mechanism that is language and technology independent. Some issues, however, make it not the right choice to solve our problem. First of all, the protocol is very complex to implement, although a freely available reference implementation [32] exists. Moreover there is little deployment of SLP and there is little knowledge of its existence.

Indeed, we believe that the definition of a specific and simple protocol for PKI resources location is needed to ease its integration into existing and future applications, especially for mobile devices which have limited computational power and communication bandwidth.

6 Conclusions

The lack of interoperability among closed PKI islands is a very urgent problem and demands a solution.

One example of an environment where our system could provide measurable improvements is the Grid community, which already make heavy use of X.509. One of the most sensitive technical issue to be solved is related to the availability of revocation data and validation services in big Grids. The *Grid Security Infrastructure* (GSI) uses proxy certificates to allow an entity to temporary delegate its rights to remote processes or resources on the Internet. Such a certificate is derived from, and signed by, a normal EE certificate. Therefore an easy way to find validation services and CRLs for EE certificates is needed in order to verify their validity. Administrators decide a set of CAs, and therefore users, to be trusted for accessing the shared resources.

PRQP could help automatic configuration of validation services by providing updated URLs to OCSP, CRLs repositories, or other services (e.g. SCVP). This would increase data availability and possibility to securely use existing PKIs for Grid Computing. Moreover, a party like the International Grid Trust Federation (IGTF) [33], established in October 2005, could run a centralized RQA to provide URLs about federated CAs to all users and resource managers.

Wireless is another very interesting scenario for the deployment of PRQP. Usage of digital certificates in open environments (e.g. university and enterprise WLANs) is strongly limited by interoperability issues. Access Points (or radius servers) could leverage the use of PRQP to discover services and, then, retrieve PKI data needed for validation of client certificates. For example support for visiting students or professors to access the University's network could be easily managed without requiring complex authentication infrastructures (e.g. EduRoam [34]) and without delegating credentials validation to third parties.

The PRQP protocol provides a PKI-specific protocol for resource discovery, and offers a starting point for the development of a PKI Resource Discovery Architecture where different RQAs cooperate to access data which is not locally

available. Our research will next proceed by evaluating the usage of an authenticated *Peer-To-Peer (P2P)* network for distribution of URLs of available services between RQAs. These authorities would share data about configured services with other peers in the P2P network. In this scenario, each client would use one of the configured RQAs as an entry point where all its requests will be sent to. Thus the P2P network would map network addresses to services mostly like the DNS maps logical names to IP addresses. Current research is focused both on the study and the implementation of such a network.

References

1. Myers, M., Ankney, R., Malpani, A., Galperin, S., Adams, C.: Online Certificate Status Protocol - OCSP. Internet Engineering Task Force: RFC 2560 (June 1999)
2. Wahl, M., Howes, T., Kille, S.: Lightweight Directory Access Protocol (v3). Internet Engineering Task Force: RFC 2251 (December 1997)
3. Freeman, T., Housley, R., Malpani, A., Cooper, D., Polk, W.: Server-based Certificate Validation Protocol (SCVP). IETF Draft (January 2007). [Online] Available `http://www.ietf.org/internet-drafts/draft-ietf-pkix-scvp-31.txt`
4. OpenSSL Homepage. [Online] Available: `http://www.openssl.org/`
5. OpenCA Project Homepage. [Online] Available: `http://www.openca.org/`
6. OpenCA OCSPD. [Online] Available: `http://www.openca.org/ocspd/`
7. World List of Universities. [Online] Available: `http://www.unesco.org/iau/`
8. Universities Worldwide. [Online] Available: `http://univ.cc/`
9. Hanna, S.: Follow-up Survey on Obstacles to PKI Deployment and Usage (October 2003). [Online] Available: `http://www.oasis-open.org/committees/pki/pkiobstaclesaugust2003surveyr eport.pdf`
10. Housley, R., Polk, W., Ford, W., Solo, D.: Certificate and Certificate Revocation List (CRL) Profile. Internet Engineering Task Force: RFC 3280 (2002)
11. Mockapetris, P.: Domain Names - Implementation and Specification. Internet Engineering Task Force: RFC 1035, Request for Comments (November 1987)
12. Gulbrandsen, A., Vixie, P., Esibov, L.: A DNS RR for specifying the location of services (DNS SRV). Internet Engineering Task Force: RFC 2782 (February 2000)
13. Boeyen, S., Hallam-Baker, P.: Internet X.509 Public Key Infrastructure Repository Locator Service. IETF Experimental (September 2005). [Online] Available: `http://tools.ietf.org/wg/pkix/draft-ietf-pkix-pkixrep/draft-ietf-pkix-pkixrep-04.txt`
14. Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., Weerawarana, S.: Unraveling the Web Services Web: An Introduction to SOAP, WSDL, and UDDI. IEEE Internet Computing 6(2), 86–93 (2002). [Online] Available: `http://dx.doi.org/10.1109/4236.991449`
15. Martin, G., Marc, H., Noah, M., Jean-Jacques, M., Henrik Frystyk, N.: SOAP Version 1.2. W3C Recommendation (June 2003). [Online] Available: `http://www.w3.org/TR/`
16. Christensen, E., Curbera, F., Meredith, G., Weerawarana, S.: PWeb Services Description Language (WSDL) 1.1. W3C Note (March 2001). [Online] Available: `http://www.w3.org/TR/2001/NOTE-wsdl-20010315`
17. Chinnici, R., Gudgin, M., Moreau, J.-J., Weerawarana, S.: Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language. W3C Working (May 2005). [Online] Available: `http://www.w3.org/TR/wsdl20`

18. Clement, L., Hately, A., von Riegen, C., Rogers, T.: UDDI Version 3.0.2. [Online] Available (October 2004) http://uddi.org/pubs/uddi_v3.htm
19. Common Object Request Broker Architecture: Core Specification (March, 2004) [Online] Available: http://www.omg.org/technology/documents/corba_spec_catalog.htm
20. Yergeau, F., Cowan, J., Bray, T., Paoli, J., Sperberg-McQueen, C.M., Maler, E.: Extensible Markup Language (XML) 1.1. W3C Recommendation (2004, February). [Online] Available: http://www.omg.org/technology/documents/corba_spec_catalog.htm
21. Information Technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)ITU-T Recommendation X.690 (1994) | ISO/Uniform Resource Locators (URL)IEC 8825-1:1995 (1994)
22. Information Technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)ITU-T Recommendation X.690 (1994) | ISO/Uniform Resource Locators (URL)IEC 8825-1:1995 (1994)
23. Edwards, W.: Core Jini, 2nd edn. Prentice-Hall, Englewood Cliffs (2000)
24. Arnold, K.: The Jini Specification, 2nd edn. Addison-Wesley, Reading (2000)
25. Universal Plug and Play Specifications. [Online] Available: http://www.upnp.org/resources/specifications.asp
26. Jenronimo, M., Weast, J.: UPnP Design by Example: A Software Developer's Guide to Universal Plug and Play (2003)
27. Guttman, E., Perkins, C., Veizades, J., Day, M.: Service Location Protocol, version 2. Internet Engineering Task Force: RFC 2608 (June 1999)
28. Guttman, E., Perkins, C., Kempf, J.: Service Templates and Schemes. Internet Engineering Task Force: RFC 2609 (June 1999)
29. Guttman, E.: Service Location Protocol: Automatic Discovery of IP Network Services. IEEE Internet Computing 3(4), 71–80 (1999)
30. Java RMI Specification (2003). [Online] Available: http://java.sun.com/j2se/1.4.2/docs/guide/rmi/spec/rmiTOC.html
31. Goland, Y., Cai, T., Leach, P., Gu, Y., Albright, S.: Simple Service Discovery Protocol. IETF Draft (October 1999). [Online] Available: http://www.ietf.org/internet-drafts/draft-cai-ssdp-v1-03.txt
32. OpenSLP Project. [Online] Available: http://www.openspl.org
33. International Grid Trust Federation. [Online] Available: http://www.gridpma.org
34. Education Roaming (Eduroam) Homepage. [Online] Available: http://www.eduroam.org/

Bootstrapping a Global SSO from Network Access Control Mechanisms

Manuel Sánchez[1], Gabriel López[1], Óscar Cánovas[2], and Antonio F. Gómez-Skarmeta[1]

[1] Department of Information and Communications Engineering
[2] Department of Computer Engineering
University of Murcia, Spain
{msc,gabilm,skarmeta}@dif.um.es,
ocanovas@ditec.um.es

Abstract. This paper presents the details of a Single Sign On proposal which takes advantage of previously deployed authentication mechanisms. The main goal is to establish a link between authentication methods at different levels in order to provide a seamless global SSO. Specifically, the users will be authenticated once, during the network access control phase. Next, having authenticated to get on to the network using 802.1X, that authentication will automatically fetch the necessary signed tokens so that there would be no need to repeat the login at the application layer. Therefore, the application level authentication would be bootstrapped from the network access. As we will see, this involves the generation of SAML signed tokens that will be obtained by the users using a PEAP channel able to deliver the appropriate authentication credentials. Then, users will contact a federation-level validation service and there will no need to re-authenticate the user, only a query of the related user attributes will be necessary in some cases.

Keywords: SSO, authorization, SAML, federation.

1 Introduction

In the last years, we have experienced the emergence of federated approaches to resource sharing. In this approaches, trust links are established among different autonomous organizations in order to grant users in any of them access to shared resources with a single identity, stated by the organization the user belongs to. Important examples of these approaches are the establishment of academic federations worldwide, like eduroam, InCommon, HAKA or SWITCH. In those scenarios where users are moving among the different organizations pertaining to the federation, authorized users may also have additional resources at their disposal at the visited institutions.

Despite many aspects of federations have been addressed by several projects, other issues generally related with integral identity management are still open. For example, in those scenarios where authentication mechanisms have been included for network access control purposes, it would be interesting to create a seamless link between the network-layer authentication mechanism and any additional authentication step that will be needed when users try to gain access to application-level resources. This would involve the extension of the network access mechanism in order to deliver additional

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 170–180, 2007.

information (some kind of security credentials) that might be used at service-level in order to avoid further user re-authentication.

The work presented in this paper is one of the objectives defined by the DAMe project [1]. Once the eduroam infrastructure has been extended so that user mobility will be controlled by security assertions and policies expressed in standard and extensible languages, such as SAML [5] and XACML [4], the next phase is to provide a global Single Sign On (SSO) mechanism based on that extension. Eduroam constitutes an exceptional starting point in order to provide a mechanism for transmitting the user credentials that will be used by the application-level authorization systems in order to offer a full and integrated network access experience to the users. As we will see, we have defined two different steps in order to achieve the SSO. The first one is related to the delivery of a security token during the network access that will be later used to avoid unnecessary re-authentication. Then, once that token has been transmitted to the user, it will be necessary to define how an application-level service will be able to validate that information using a federation-level service. Specifically, we will follow some guidelines already stated by the technical community in order to provide global SSO.

The rest of this paper is structured as follow. Section 2 provides an overview of the DAMe project and introduces the underlying roaming infrastructure. Section 3 describes eduGAIN, an authentication and authorization infrastructure that will be used in order to provide a common validation service. Section 4 points out the set of requirements derived from a SSO scenario and section 5 describes the proposed architecture. Then, section 6 contains a survey of other proposals that informed our work. Finally, we conclude the paper with our remarks and some future directions.

2 DAMe Project: Adding Authorization to Eduroam

The eduroam network [15] is an inter-institutional roaming service based on the 802.1X architecture [8] and a hierarchical RADIUS-based infrastructure. This initiative allows users of participating institutions to access the Internet at other participants using their home institution's credentials, all this with a minimal administrative overhead. Nowadays, eduroam is a production service that is used in more than 350 institutions over 19 countries (European and Australian-Pacific) with a great success.

Figure 1 depicts the typical scenario in eduroam. It shows a user from Institution A who moves to Institution B, both pertaining to the eduroam federation. In the new institution, the user wants to get access to the wireless network. In this situation, access control is carried out following the 802.1X standard. That is, the user associates with the wireless access point (AP), which contacts its local RADIUS server in order to authenticate the user. But when this server identifies that the user belongs to a different domain, for example based on the user identifier, the authentication request is forwarded through the RADIUS hierarchy to the server located in the user's home institution. Then, the user is authenticated and the response is routed back to Institution B, where the AP enables the requested connection.

However, eduroam only takes into account the identity in order to carry out the access control process. In this way, it is not possible to offer different services or restrict the access to some resources based on the user profile, defined for example by means of

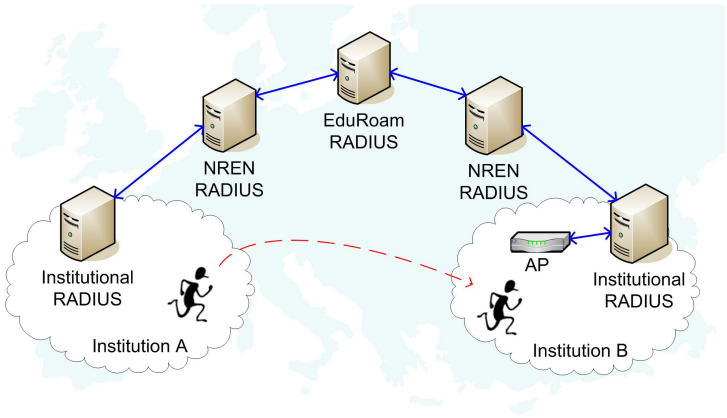


Fig. 1. eduroam infrastructure

some attributes in the home institution. Therefore, the main objective of DAMe [1] is the definition of a unified authentication and authorization system for federated services hosted in the eduroam network. Those federated services can range from network access control to other high level distributed services such as Grid Computing or web services.

Due to wide deployment of eduroam, it should not be convenient to modify the eduroam infrastructure in such a way that the resulting system could be incompatible with the initial one. Therefore, DAMe defines several steps that can be applied to eduroam gradually, in order to obtain the new functionality. Besides, backward compatibility must be maintained to allow some institutions to keep on using the initial eduroam system, providing only authentication.

The first goal of DAMe is to extend the eduroam infrastructure using NAS-SAML [10] to provide a fine grained authorization system. In this way, user mobility can be controlled by security assertions and policies expressed in standard and extensible languages, such as SAML [5] and XACML [4]. The result of this work is described in [14].

The next step consists of taking advantage of confederation mechanisms such as those defined by eduGAIN [9], which is described in more detail in the next section. In this way, federations based on different technologies such as NAS-SAML and Shibboleth [13] can cooperate building a confederation. The idea is that eduGAIN can act as intermediary infrastructure between these federations, carrying and translating the different messages and attributes.

The following step is to provide Single Sign-On (SSO) from a global point of view. That is, global authentication will be bootstrapped during the network access control phase, providing the necessary security information so that there would be no need to repeat the authentication phase at the application layer. Therefore, once institutions have deployed the complete infrastructure developed in DAMe, users will access to high level resources and applications after the initial network authentication. This paper introduces the details of that SSO system.

3 eduGAIN

The main goal of eduGAIN [9] is to build an interoperable authentication and authorization infrastructure to interconnect different existing federations. In this way, eduGAIN is responsible for finding the federation where a roaming user belongs to, translate the messages between the federation internal protocols and eduGAIN and vice versa, and guarantee the trust among the participating institutions.

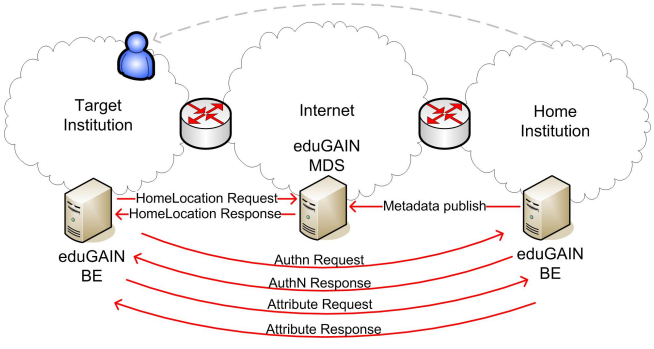


Fig. 2. eduGAIN infrastructure

The main goal is achieved defining a set of common services, where it is included the MetaData Service (MDS), and a confederation-aware element called Bridging Element (BE) responsible for connecting the different federations to eduGAIN. In eduGAIN, metadata related to federations are published by means of a MDS. These metadata include information for locating the authentication and authorization points of the federation. In this way, the home federation of a roaming user is located by the BE obtaining the information published in the MDS. Then, the appropriate authentication and authorization requests are translated and routed by the remote BE toward the user's home institution. As we will see in this paper, this scheme is also valid in order to communicate different institutions pertaining to the same federation, as [6] describes for Universal Single Sign On purposes.

The specific way the authentication and authorization processes are carried out in eduGAIN are defined by different profiles. Currently, a profile compatible with Shibboleth, called *Web SSO*, and another one that does not require human intervention, called *Automated Client*, are defined. Moreover, additional profiles are being developed, as for instance a DAMe profile based on NAS-SAML and DIAMETER.

4 Single Sign-On Scenario

The proposal presented in this paper follows the guidelines defined by the uSSO framework [6]. It is centered in a multidomain scenario where different authorization systems

are defined to access to the resources. These are the main requirements considered for the SSO:

- The user has the appropriate credentials in its home institution (HI).
- HI belongs to a federation where its issued credentials are valid.
- When the user tries to access to a resource (R) in another institution (RI), the decision about the access is taken by an Authorization Service at the RI.
- The authentication and authorization information about users is exchanged between institutions through a mechanism provided by the confederation.
- There is a confederation-aware component called "Local Federation Adaptor" (LFA) which decides if an authentication request can be handled locally or must be sent to another institution. The functionality related to This LFA is commonly performed by the eduGAIN BE.

Considering eduroam, once the user is associated to a wireless access point or is connected to a wired switch in the remote institution, he is requested for authentication information. When provided, those credentials are forwarded to the user home institution to be authenticated. But now, to enable the SSO, an authentication service is responsible for authenticating the user instead of the RADIUS server. Besides, this service should return to the user some kind of statement that can be used later to achieve the SSO. At this point, the user is allowed to access to the network.

Lately, he might want to access to a protected resource in this institution or in another one belonging to the same federation. Then the user's credentials are sent to the resource in order to be validated to gain access. Depending on the type of resource, additional steps for obtaining user attributes would be needed.

This generic pattern presented here is detailed using specific technologies in the next section, where we present the architecture for SSO and the interaction among the different elements.

5 Single Sign-On Proposal for DAMe

Since the interaction among these elements must follow the pattern defined in the previous section, it is necessary to define two different processes. On one hand, we need to authenticate the user in order to access to the network and to receive the SSO credentials or token. On the other hand, protected resources have to use the token to validate the user's identity.

5.1 SSO Architecture

The architecture for this proposal, which is shown in Figure 3, starts with the elements needed to authenticate the user using eduroam, that is, the access point with 802.1X support and the hierarchy of RADIUS servers. Moreover, according to the previous section, it is necessary to include a new service to authenticate the user and generate the SSO token. In this proposal, this element is a SAML authority called AuthN Authority, that receives a SAML *AuthN Request* and responds with a SAML *AuthN Statement*. Besides, this statement is the SSO token that will be used later for accessing the protected resources.

These elements constitute the core of the user authentication phase, but new elements must be defined to allow institutions based on different authorization infrastructures to create a global SSO scenario. This cooperation can be achieved using a confederation management infrastructure such as eduGAIN. In this way, the Bridging Elements (BE) from eduGAIN will be responsible for translating the requests and responses from the specific institution authentication or authorization mechanism to the eduGAIN protocols, and vice versa. Besides, the BE is responsible for finding the corresponding BE from the user home institution by means of queries to the MetaData Service (MDS). Finally, in relation to the authorization decision based on user's attributes, two NAS-SAML entities are introduced. The first one, called Attribute Authority, is responsible for providing user attributes and the second one, called Policy Decision Point (PDP), is responsible for taking the authorization decisions.

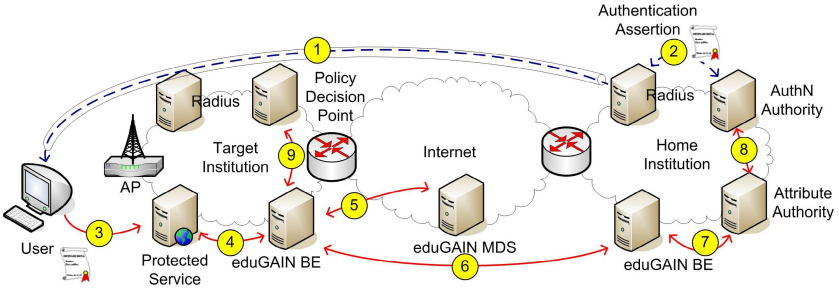


Fig. 3. DAMe SSO architecture

5.2 Network Authentication

Once described the architectural elements, the next step is to define the specific protocols implementing the profile depicted in previous sections. First, the system authenticates the user by means of some authentication method, for example PEAP [12] as we will see, and then generates some security data which is returned to him.

In DAMe, this first phase is based on the eduroam authentication through the RADIUS hierarchy (dotted lines in Figure 3), but it is enhanced with the delivery of security data to the user. Specifically, as Figure 4 details, when the user tries to access to the network in the remote institution, the remote RADIUS server forwards the authentication request to the RADIUS server located in the home institution. Then, this server asks the AuthN Authority using an *AuthN Request* message, that contains the user credentials, and receives an *AuthN Statement*. This statement, which is the data the user receives to enable SSO, is a SAML sentence which contains information about the user's identity, his credentials and the kind of authentication being performed. To protect the user privacy in the SSO, this statement does not contain his subject, but a handle generated by the AuthN Authority. Besides, to validate the handle later, the authority stores the matching handle-subject. Finally, the remote RADIUS server returns the statement to the user through a TLS tunnel created by the authentication method. The creation of that tunnel is detailed in the next section.

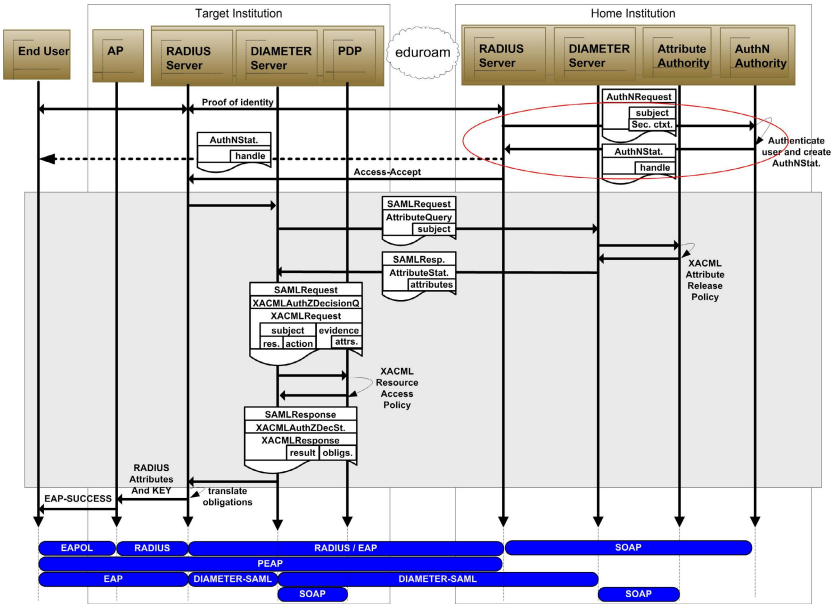


Fig. 4. Initial authentication

As Figure 4 shows in the grey area, this phase includes some messages that are not directly related to the SSO mechanism. They constitute the network access control authorization in a roaming scenario, which is a previous result of the DAMe project. More specifically, the remote RADIUS server makes use of a DIAMETER infrastructure between the two institutions to requests the user's attributes. Finally, an authorization decision is taken consulting the PDP.

5.3 Token Delivery

The TLS tunnel mentioned before is part of the PEAPv2 [12] authentication method. The reason to use PEAPv2 to authenticate the user is the need for a protected channel to deliver the SSO statement. Figure 5 shows the sequence of messages needed to authenticate the user using this method. It has the special feature that after an initial handshake creates a protected tunnel between the peer and the authenticator. Then, this tunnel is used to exchange the authentication information in a secure way. Therefore the SSO mechanism can take advantage of this tunnel and use it to deliver the statement to the user in a secure way. Specifically, elements are transmitted through the tunnel by means of Type Length Value (TLV) objects. Besides, there is a special TLV, named vendor specific, which can be used to carry non standard elements. In this way, the authenticator can add the security data inside a vendor specific TLV to the last message sent to the peer before closing the tunnel.

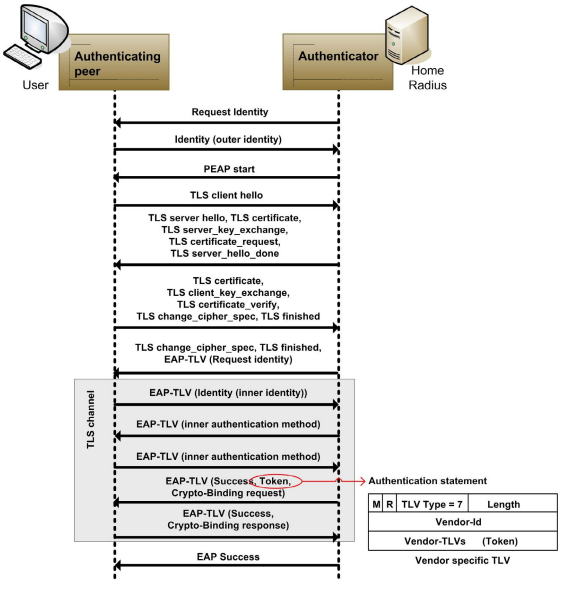


Fig. 5. PEAP authentication method and vendor specific TLV

5.4 Resource Access Using SSO

The second phase starts when the user tries to gain access to a protected resource after receiving the SSO token. Due to this data is used as a proof of user's authentication, it is not necessary to ask the user again for his credentials. In this way, once the user has received the SSO data, he can access to whatever service that supports this SSO mechanism.

The specific process is shown in Figure 6, where the user tries to access to some protected service in the remote institution and the service rejects the user connection because it is unauthorized. Then, the user sends the data received in the previous phase to the protected service. Alternatively, the user could add that data directly to the access request. At this point, the service recovers and validates that information. Optionally, the resource might impose a fine grain user authorization based on more information besides the identity. In order to do this, the service sends an *Attribute Query* to the BE, which includes the handle from the SSO statement to identify the user. Now the BE queries the eduGAIN MDS service to know how to find the BE belonging to the user's home institution, and then sends the query to it. When the home institution BE receives the query, it is forwarded to the Attribute Authority. This entity firstly validates the handle consulting the AuthN Authority by means of an *AuthN Query*. To validate the handle, the AuthN Authority check if it has a matching between this handle and a user subject, returning in that case the subject to the Attribute Authority. Then, this entity recovers the user's attributes and returns them to the remote institution, but including in the statement the handle instead of the subject to maintain the user privacy. When the

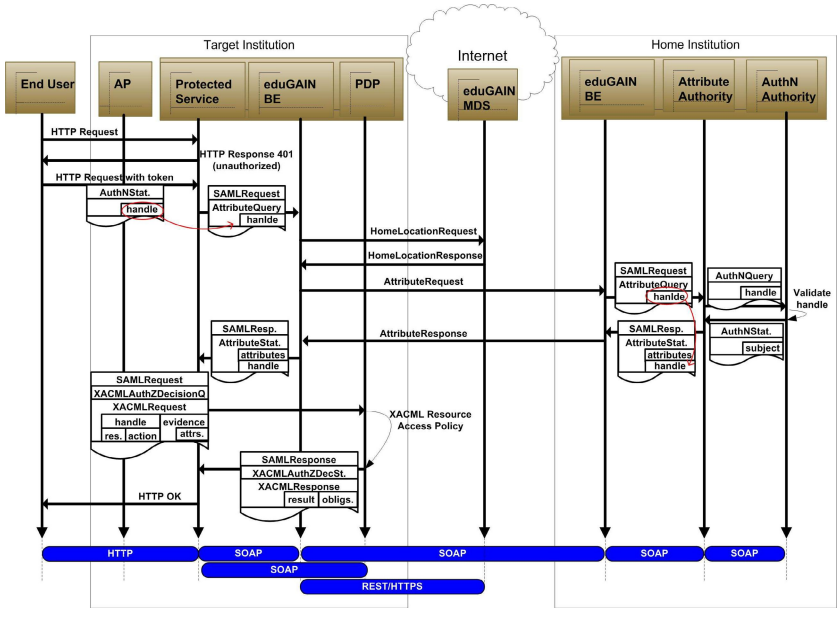


Fig. 6. Authentication using the token

protected service receives the attributes through the BE, it finalizes the authorization process consulting the Policy Decision Point (PDP) to take the decision.

Finally, once we have presented the architecture and the protocols for including SSO in the eduroam network, some other SSO mechanisms that informed our work are included in the next section.

6 Related Work

One of the first SSO mechanisms was Kerberos [11]. This system is based on the use of a centralized Key Distribution Center (KDC), which is composed by an Authentication Server (AS) and a Ticket Granting Server (TGS). The KDC maintains a shared secret with each server and user in the system to authenticate them. The system works as follow: Firstly the user authenticates to the AS and receives a ticket encrypted with the TGS key and a TGS session key. Secondly, using the session key to authenticate to the TGS, the user sends the encrypted ticket and the id of the server that is willing to access to the TGS. Then, the TGS returns to the user a new token encrypted with the server key and a session key for the server. Finally, using this information, the user can authenticate and access to the server. Later, if the user wants to access to a new server, he can use the first ticket to request to the TGS a new ticket for the new server. But it has several drawbacks such as that the TGS becomes a bottleneck, and it must trust all the servers and vice versa. Besides, Kerberos requires synchronized clocks. Therefore,

although there are proposals trying to make the protocol scalable for inter-networking, it is only usefully for single networks.

Nowadays there are several SSO mechanisms, mainly for web environments, such as Shibboleth and Microsoft Passport. Shibboleth [13] is a web authorization infrastructure based on the use of SAML and web redirections to determine if a user can access to a resource through its web browser. This process is based on the user's information that is maintained in his home institution. This mechanism enables the definition of identity federations, in such a way that the user always authenticates to his home institution, and then the needed information is sent where it is necessary to authorize him. Shibboleth defines the Service Provider (SP) and the Identity Provider (IdP). The former is the institution providing resources and the latter is the institution managing user's identities. In this way, when the user accesses to some protected resource from a SP, he asks the IdP where the user belongs to for information about him. If the user has previously authenticated, the IdP returns the needed information, but elsewhere the user is redirected to the IdP to be authenticated. In this way, web SSO is provided. The main difference from Shibboleth with the proposal presented in this paper is that this mechanism do not include the initial network access authentication. Therefore, the user has to authenticate twice whereas we are taking advantage of a mandatory network access control system in order to distribute valid credentials for SSO. However, some efforts are being made by Internet2 [2] to develop a universal SSO mechanism similar to our proposal [3], where the network access authentication via RADIUS returns information to contact with user's IdP.

Microsoft Passport [7] offers a SSO service for websites that do not have any relationship among them. They only have to trust the entity providing the SSO service. In this way, when a user accesses to a protected website, he is redirected to the passport server to be authenticated. Then, if the authentication is successful, the user is redirected to the initial website. Besides the user's Internet browser receives two encrypted cookies, one from the website and another one from the passport server. This mechanism, like the previous one, do not take into account any previous authentication.

7 Conclusions and Future Work

As we have presented in this paper, there are several scenarios in federations where authentication is required in order to protect shared resources, ranging from network access to distributed web resources or grid computing platforms. Despite one of the main benefits from federations is the harmonization of procedures, information schemas, and protocols, we identified the design of a unified SSO mechanism as an interesting research activity.

One of the main features of this SSO proposal is the seamless link of authentication processes performed at different levels. Since eduroam constitutes an exceptional federated service for supporting user mobility, we demonstrate that the delivery of signed tokens during the network access phase provides the desired functionality in order to bootstrap the SSO system. Moreover, the definition of PEAP-based protocols does not impose the use of new authentication methods (from the organization point of view) but provides the required distribution channel. Additionally, the choice of eduGAIN as

the authentication infrastructure for validation purposes guarantees the interoperability among different type of resources located at different organizations.

As a statement of direction, we are implementing the PEAP supplicant and defining also the required middleware for managing the different security tokens obtained by the users.

Acknowledgements

This work has been partially funded by Daidalos FP6-IP-506997 and DAMe.

References

1. DAMe Project. http://dame.inf.um.es
2. Internet 2 Home Page. http://www.internet2.edu
3. Carmody, S.: Radius profile of SAML. Revision 2 (October 2006) http://stc.cis.brown.edu/ stc/Projects/Projects-using-Shib/eduRoam/Radius-SAML-Profile-v1.html
4. Anderson, A., et al.: EXtensible Access Control Markup Language (XACML) Version 1.0, OASIS Standard (February 2003)
5. Eve, M., Prateek, M., Rob, P.: Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML)v1.1, OASIS Standard (September 2003)
6. Kerver, B., Stanica, M., Rauschenbach, J., Wierenga, K.: Deliverable DJ5.3.1: Documentation on GÉANT2 Universal Single Sign-On (uSSO) Requirements, GN2 JRA5. Geant 2 (February 2007)
7. Kormann, D.P., Rubin, A.D.: Risks of the passport single signon protocol. Computer Networks 33, 51–58 (2000)
8. LAN MAN Standards Committee of the IEEE Computer Society. IEEE Draft P802.1X/D11: Standard for Port based Network Access Control (March 2001)
9. López, D.R., Macias, J., Molina, M., Rauschenbach, J., Solberg, A., Stanica, M.: Deliverable DJ5.2.3.1: Best Practice Guide - AAI Cookbook, 1st edn., GN2 JRA5. Geant 2 (September 2006)
10. López, G., Cánovas, O., Gómez, A.F., Jimenez, J.D., Marín, R.: A network access control approach based on the aaa architecture and authorzation attributes. Journal of Network and Computer Applications JNCA (to be published, 2006)
11. Newman, B.C., Ts'o, T.: Kerberos: An authentication service for computer networks. IEEE Communications 32, 33–38 (1994)
12. Palekar, A., Simon, D., Salowey, J., Zhou, H., Zorn, G., Josefsson, S.: Protected EAP Protocol (PEAP) Version 2, Internet Draft (October 2004)
13. Scavo, T., Cantor, S.: Shibboleth Architecture. Technical Overview, Working Draft 02 (June 2005)
14. Sánchez, M., López, G., Cánovas, O., Gómez-Skarmeta, A.F.: A proposal for extending the eduroam infrastructure with authorization mechanisms. In: 5th International Workshop on Security in Information Systems (submitted 2007)
15. Wierenga, K., Florio, L.: Eduroam: past, present and future. In: TERENA Networking Conference (2005)

Anonymous k-Show Credentials

Mohamed Layouni and Hans Vangheluwe

School of Computer Science, McGill University,
3480 University Street, Montreal, H3A 2A7, Quebec, Canada

Abstract. Privacy-preserving digital credentials are cryptographic tools that allow a user to prove a predicate about his/her identity or qualifications, without the verifying party learning additional information beyond the status of that predicate. The Identity Mixer (Idemix) [CL01] is a framework providing such credentials. In Idemix, we can distinguish two types of credentials: (1) one-time show credentials which can be shown only once before unveiling the identity of their holder, and (2) multi-show credentials which can be shown infinitely many times without the showings being linked to each other, or to the identity of their holder. In this paper, we bridge the gap between the two previous types of credentials, and extend Idemix to k-show credentials (for $k > 1$.) The k-show credentials we propose can be shown anonymously, but linkably, up to k times.

Keywords: Privacy-preserving digital credentials, anonymity, multiple-show credentials.

1 Introduction

With the increasing digitization of society, and the continuous migration of day-to-day services from the paper world to the digital world, digital credentials have become a very important tool. Similar to their paper counterparts, digital credentials are special documents, issued by a certification authority, that may contain a variety of information about their holder (e.g., identity attributes, qualifications, privileges, etc.) In addition, digital credentials have attractive features, that make them superior to their paper counterparts, such as searchability, large-scale data-mining, and knowledge discovery, just to name a few. With the latter features, comes also the disadvantage that credential holders are now a lot easier to monitor, and to have their privacy violated. Furthermore, digital credentials – by their very nature – are easy to clone and copy, and using them without proper safeguards could lead to serious security problems. To address this set of conflicting requirements, namely privacy and security requirements, privacy-preserving credentials have been invented [Cha85, CP92, Bra94, Bra00, CL01, CL04]. In a privacy-preserving digital credential system, one can generally distinguish three types of players: a certification authority, a user, and a verifier. In some cases, the certification authority and the verifier are controlled by the same entity. The certification authority issues a credential to a user who fulfills certain conditions. In exchange for goods and services, the user may be required to prove,

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 181–192, 2007.

to a service provider (the verifier), possession of a valid credential from the certification authority. The user may also be required to prove a predicate on the attributes encoded in his credential. The service provider may later decide to deposit a transcript of the interaction it had with the user, to the certification authority. The main requirements the credential system should satisfy are: (1) Non-forgeability: the user should not be able to succeed in proving the validity of forged credentials, or in proving predicates that are not satisfied by the attributes encoded on his CA-issued credential, and (2) Privacy: the verifier should not be able to learn any information about the user's credentials beyond what can be naturally inferred from the status of the proven predicate. The latter requirement can be refined even further, by adding constraints on the number of times a credential can be used. Based on this last criterion, we can distinguish three types of credentials:

1. Multiple-show credentials: they can be shown infinitely-many times without the showings being linked to each other, or to the issuing protocol instance where they were generated.
2. One-show credentials: they can be shown anonymously only once, before the identity of their holder is unveiled.
3. Limited- or k-show credentials, for ($k > 1$): they can be shown anonymously up to k times, after which the identity of the holder is revealed.

Privacy-preserving credential systems are becoming increasingly popular, and there is a growing interest in concrete implementations [Ide07, UPr07, Hig07]. The Identity Mixer [Ide07] is based on Camenisch and Lysyanskaya's credentials [CL01], and is one of today's most complete credential systems. Idemix provides a framework supporting only the first two types of credentials, namely multi-show credentials, and one-time show credentials.

Our Contribution: In this paper, we bridge the gap between the two first types of credentials, and extend the Idemix framework to k-show credentials (for $k > 1$.) A naive way to construct k-show credentials is by issuing k separate copies of one-show credentials, but this option obviously lacks efficiency. The solution we propose in this paper extends the one-time show credentials of [CL01] to k-show credentials without a significant increase in complexity. Compared to the protocols of [CL01], we only add 2 extra exponentiations and 1 proof of discrete logarithm knowledge to the user in the pseudonym creation protocol. For the issuing protocol, the user performs 3 more exponentiantions and a proof of knowledge for each additional showing allowed. Finally, the complexity of the showing protocol can be made very close to that of one-time show credentials [CL01] by using precomputations and fast exponentitation methods [Gor98].

Anonymous k-show credentials may be used in a variety of applications. They can be used for instance to build public transit passes, where a user is allowed to make up to k rides anonymously, after which the pass serial number will be uncovered, revoked, and added to a black list. In order to count the number of times a credential is shown, the issuing organization is able to link different showings of the same credential to each other, but not to the identity of the

credential holder, or for that matter, to the instance of the issuing protocol that generated the credential. This linking feature can also be found in the one-show credentials of [CL01] and [Bra94] where issuers rely on it to detect double-spending. The work in [LTW05] follows the same principle to recover lost electronic cash.

The remainder of this paper is organized as follows. In section 2 we give an overview of the basic credential system of [CL01], as well as the general setting. In section 3 we present our extension to k-show credentials. In section 4, we position our contribution with respect to related work. We conclude in section 5.

2 Review of the Idemix Credential System

2.1 General Setting

There are two main types of players in the Idemix system: users and organizations. Organizations offer both material (e.g., goods) and non-material (e.g., assertions) services to users. To provide those services, organizations require users to fulfill certain conditions. These conditions may include paying a fee, fulfilling a predicate about one's identity, or proving possession of an authorization from some recognized authority. More generally, this process consists in showing one or a set of credentials.

Users in turn want to benefit from those services without revealing unnecessary information about their identity. To remain anonymous, each user U possesses a different pseudonym $N_{(U,O_i)}$ with each organization O_i. For example, if an organization O_1 requires a user U to show that he has a valid credential from organization O_2, then U should be able to do so without O_1 and O_2 being capable of linking his pseudonyms $N_{(U,O_1)}$ and $N_{(U,O_2)}$. This property is called *unlinkability*. It should also be possible for user U to show a credential he obtained from organization O without O being able to retrace that credential back to the protocol instance where the credential was issued. This property is called *untraceability*.

In the Idemix system [CL01], a user U first registers with an organization O and obtains a pseudonym $N_{(U,O)}$, and a validating tag $P_{(U,O)}$ on it. The validating tag allows the user to prove that he actually owns the pseudonym. Next, user U obtains a credential from organization O. The credential is a pair denoted $(c_{(U,O)}, e_{(U,O)})$, such that $c_{(U,O)}^{e_{(U,O)}} = P_{(U,O)} d_O$, where d_O is a system parameter. Later, user U can prove to a verifying organization V that he holds a valid credential from organization O without either revealing the credential $(c_{(U,O)}, e_{(U,O)})$ or his pseudonym $N_{(U,O)}$ with organization O.

The Idemix framework is based on the strong RSA assumption and the decisions Diffie-Hellman assumption modulo a safe prime product.

In the following we begin with an overview of the system parameters. Then we give a brief description of the pseudonym generation protocol, the credential issuing protocol, and the credential showing protocol.

2.2 System Parameters and Notations

All RSA moduli used in the system are of length ℓ_n. Let the intervals $\Gamma =]-2^{\ell_\Gamma}, 2^{\ell_\Gamma}[$, $\Delta =]-2^{\ell_\Delta}, 2^{\ell_\Delta}[$, and $\Lambda =]-2^{\ell_\Lambda}, 2^{\ell_\Lambda+\ell_\Sigma}[$ be such that $\ell_\Delta = \epsilon(\ell_\Lambda + \ell_n) + 1$, where $\epsilon > 1$ is a security parameter, and $\ell_\Lambda > \ell_\Sigma + \ell_\Delta + 4$. Each organization O chooses a safe prime product modulus n_O and keeps the factorization secret. It also chooses random elements $a_O, b_O, d_O, g_O, h_O, v_O, z_O \in QR_{n_O}$ and publishes them along with n_O. The parameter ℓ_Λ is chosen such that computing discrete logarithms in QR_{n_O} with ℓ_Λ-bits exponents is hard.

In the remainder of this paper, $PK\{(\alpha, \beta, \cdots) : \mathcal{P}(A, B, \cdots; \alpha, \beta, \cdots)\}$, denotes, for public parameters $A, B, \cdots$, a proof of knowledge of secrets $\alpha, \beta, \cdots$, for which the public predicate $\mathcal{P}(\cdots)$ is satisfied.

2.3 Establishing a Pseudonym with an Organization

Let U be a user who wants to establish a pseudonym with organization O. Let $x_u \in \Gamma$ be U's master secret key (or ID). The protocol shown in figure 1 allows U to obtain a pseudonym $N_{(U,O)}$ and a validating tag $P_{(U,O)}$ such that $P_{(U,O)} = a_O^{x_u} b_O^{s_{(U,O)}}$, where $s_{(U,O)}$ is jointly chosen by U and O. Organization O learns nothing about the values of x_u and $s_{(U,O)}$.

2.4 Obtaining a Credential from an Organization

Organization O can issue a credential to user U who proves ownership of a previously established pseudonym and validating tag $(N_{(U,O)}, P_{(U,O)})$. The credential is a pair $(c_{(U,O)}, e_{(U,O)}) \in \mathbb{Z}_{n_O}^* \times \Lambda$ such that $P_{(U,O)} d_O = c_{(U,O)}^{e_{(U,O)}}$. Figure 2 describes how the issuing protocol works.

2.5 Showing a Credential to an Organization

User U can prove to any verifying organization V that he possesses a valid credential from organization O. In this proof V does not learn anything about the user's pseudonym with O, the credential he holds from O, or the secrets underlying that credential. Figure 3 explains how the show protocol works for one-time show credentials. Showing multiple-show credentials is simpler and can be easily derived.

If user U shows his one-time show credential more than once, organization O will have two challenge-response pairs (c_1, r_1) and (c_2, r_2), from which it can retrieve x_u and $s_{(U,O)}$, which in turn will determine $P_{(U,O)}$ and identify U.

3 Extension to *k*-Show Credentials

In this section we present an extension of the credential system described above. In addition to the multiple-show and one-show modes, where credentials can be spent either infinitely many times or only one time, we now present k-show credentials, with $k > 1$. k-show credentials can be spent k times without being

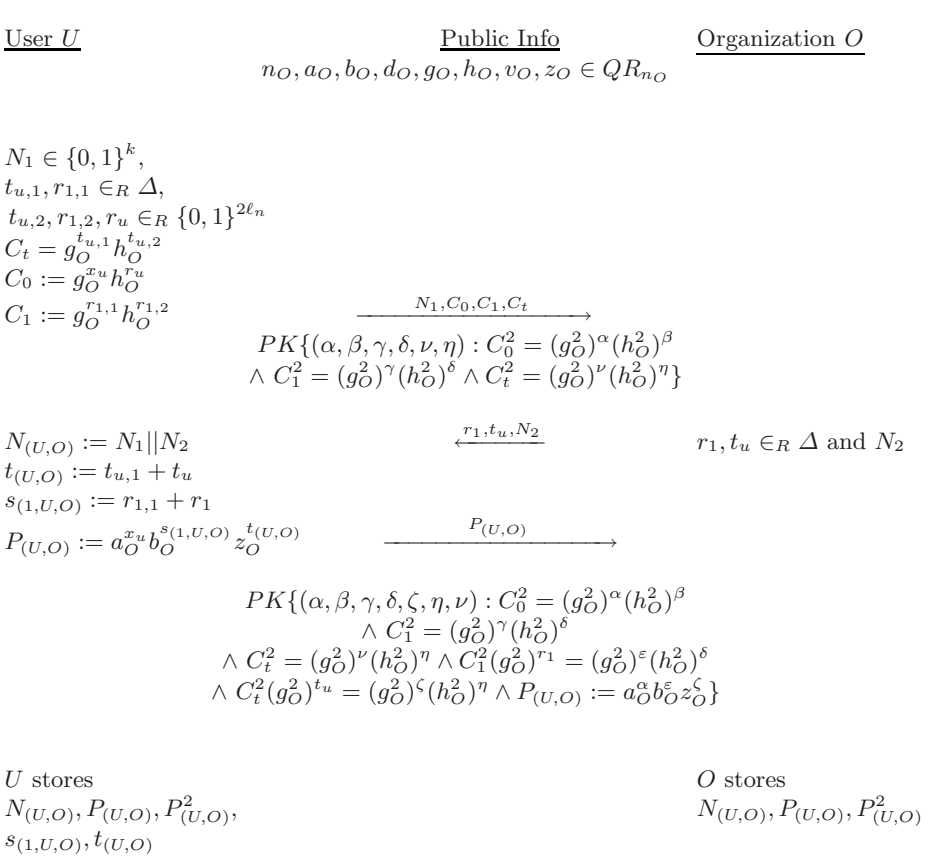


Fig. 1. Idemix pseudonym creation protocol

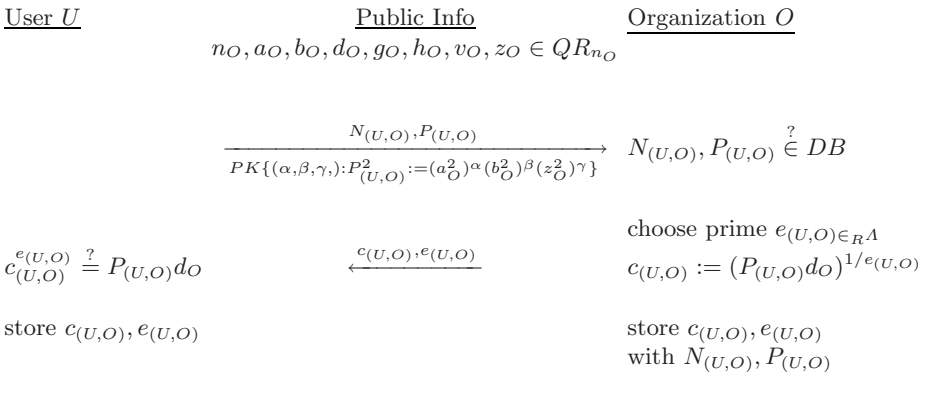


Fig. 2. Idemix credential issuing protocol

linked to the instance of the issuing protocol that generated them or to the identity of their owner. The identity of the owner is revealed only if the credential is spent more than k times.

User U	Public Info	Organization O
	$n_O, a_O, b_O, d_O, g_O, h_O, v_O, z_O \in QR_{n_O}$	
choose $r_1, r_2 \in_R \{0,1\}^{2\ell_n}$		
$A := c_{(U,O)} h_O^{r_1}$		
$B := h_O^{r_1} g_O^{r_2}$		
$H_{(u,O)} := h_O^{t_{(U,O)}}$	$\xrightarrow{A,B,H_{(u,O)}}$	
	$\xleftarrow{c}$	choose $c \in_R \{0,1\}^{\ell_c}$
$r := cx_u + s_{(U,O)}$	$\xrightarrow{r}$	
mark credential as void	$PK\{(\alpha,\beta,\gamma,\delta,\zeta,\xi,\varepsilon,\varphi) : H_{(u,O)} = h_O^{\varphi}$	
	$\wedge\ d_O^2 = (A^2)^{\alpha}(\frac{1}{a_O^2})^{\beta}(\frac{1}{b_O^2})^{\gamma}(\frac{1}{z_O^2})^{\varphi}(\frac{1}{h_O^2})^{\delta}$	
	$\wedge\ B^2 = (h_O^2)^{\varepsilon}(g_O^2)^{\zeta} \wedge 1 = (B^2)^{\alpha}(\frac{1}{h_O^2})^{\delta}(\frac{1}{g_O^2})^{\xi}$	If accepted
	$\wedge\ g_O^r = g_O^{\gamma}(g_O^c)^{\beta}$	store transcript
	$\wedge\ \beta \in \Gamma \wedge \gamma \in \Delta \wedge \varphi \in \Delta \wedge \alpha \in \Lambda\}$	for later checkings

Fig. 3. Idemix credential showing protocol (one-time show mode)

For the purpose of credential revocation, an additional party will be added to the setting. This party is an independent outsider to the system, and is denoted CA. When showing a credential, a user verifiably encrypts a piece of identifying information under the CA's public key. The encryption also specifies the condition under which the ciphertext is decrypted. This is achieved using the Cramer-Shoup encryption scheme [CS98].

Additional system parameters. These are parameters to be used in the verifiable encryption scheme. The CA chooses a group $G = \langle g \rangle = \langle h \rangle$ of prime order $q > 2^{\ell_\Gamma}$. Then he chooses $x_1, x_2, x_3, x_4, x_5 \in_R \mathbb{Z}_q$ and keeps them secret. The CA then computes the tuple $(y_1, y_2, y_3) := (g^{x_1}h^{x_2}, g^{x_3}h^{x_4}, g^{x_5})$ and publishes it as his public key.

3.1 Establishing a Pseudonym with an Organization for Obtaining Revocable Credentials

This protocol can be used not only for k-show credentials but for multiple-show credentials as well. It simply gives an organization the necessary information it could use to identify a user and revoke his credentials in case a certain revocation condition is fulfilled. The protocol is shown on figure 4.

Several variables are used in this protocol. Following are the roles played by the most important ones.

- $t_{(U,O)}$ which is only known to the user will be used to link the showings of the same k-show credential to each other.
- x_u is used to identify the user globally by the CA. x_u is only known to the user.

User U	Public Info	Organization O
	$n_O, a_O, b_O, d_O, g_O, h_O, v_O, z_O \in QR_{n_O}$	
		G, g, h, y_1, y_2, y_3
$N_1 \in \{0,1\}^k$,		
$x_u, x_{(u,O)} \in \Gamma$		
$t_{u,1}, r_{1,1} \in_R \Delta$,		
$t_{u,2}, r_{1,2}, r_u \in_R \{0,1\}^{2\ell_n}$		
$C_t = g_O^{t_{u,1}} h_O^{t_{u,2}}$		
$C_0 := g_O^{x_u} h_O^{r_u}$		
$C_1 := g_O^{r_{1,1}} h_O^{r_{1,2}}$	$\xrightarrow{N_1, C_0, C_1, C_t}$	
	$PK\{(\alpha,\beta,\gamma,\delta,\nu,\eta) : C_0^2 = (g_O^2)^\alpha (h_O^2)^\beta$	
	$\wedge\, C_1^2 = (g_O^2)^\gamma (h_O^2)^\delta \wedge C_t^2 = (g_O^2)^\nu (h_O^2)^\eta\}$	
	$\xleftarrow{r_1, t_u, N_2}$	$r_1, t_u \in_R \Delta$ and N_2
$N_{(U,O)} := N_1 \| N_2$		
$t_{(U,O)} := t_{u,1} + t_u$		
$s_{(1,U,O)} := r_{1,1} + r_1$		
$P_{(U,O)} := a_O^{x_u} b_O^{s_{(1,U,O)}} .z_O^{t_{(U,O)}} v_O^{x_{(u,O)}}$		
If $O =$ CA then		
$Y_u := g^{x_u}$	$\xrightarrow{P_{(U,O)}, Y_u}$	
	$PK\{(\alpha,\beta,\gamma,\delta,\zeta,\eta,\nu,\xi) : C_0^2 = (g_O^2)^\alpha (h_O^2)^\beta$	
	$\wedge\, C_1^2 = (g_O^2)^\gamma (h_O^2)^\delta \wedge C_1^2 (g_O^2)^{r_1} = (g_O^2)^\varepsilon (h_O^2)^\delta$	
	$\wedge\, C_t^2 = (g_O^2)^\nu (h_O^2)^\eta \wedge\ C_t^2 (g_O^2)^{t_u} = (g_O^2)^\zeta (h_O^2)^\eta$	
	$\wedge\, P_{(U,O)} := a_O^\alpha b_O^\varepsilon z_O^\zeta v_O^\xi \wedge Y_u = g^\alpha\}$	
else		
$Y_{(u,O)} := g^{x_{(u,O)}}$	$\xrightarrow{P_{(U,O)}, Y_{(u,O)}}$	
	$PK\{(\alpha,\beta,\gamma,\delta,\zeta,\eta,\nu,\xi) : C_0^2 = (g_O^2)^\alpha (h_O^2)^\beta$	
	$\wedge\, C_1^2 = (g_O^2)^\gamma (h_O^2)^\delta \wedge C_1^2 (g_O^2)^{r_1} = (g_O^2)^\varepsilon (h_O^2)^\delta$	
	$\wedge\, C_t^2 = (g_O^2)^\nu (h_O^2)^\eta \wedge\ C_t^2 (g_O^2)^{t_u} = (g_O^2)^\zeta (h_O^2)^\eta$	
	$\wedge\, P_{(U,O)} := a_O^\alpha b_O^\varepsilon z_O^\zeta v_O^\xi \wedge Y_{(u,O)} = g^\xi\}$	
U stores $N_{(U,O)}, P_{(U,O)}, P_{(U,O)}^2$, $s_{(1,U,O)}, t_{(U,O)}$, Y_u or $Y_{(u,O)}$		O stores $N_{(U,O)}, P_{(U,O)}, P_{(U,O)}^2$, Y_u or $Y_{(u,O)}$

Fig. 4. Pseudonym creation protocol for revocable k-show credentials

- $x_{(U,O)}$ will be used to identify the user locally at the issuing organization. $x_{(U,O)}$ is only known to the user.
- Y_u can be used only when a user violates the agreement governing the usage of his credential (e.g., overshowing). Y_u allows the CA to reveal a user's identity.
- $Y_{(u,O)}$ is similar to Y_u, but de-anonymizes the user only locally by revealing his pseudonym to the issuing organization O.

3.2 Obtaining a *k*-Show Revocable Credential from an Organization

User U executes the protocol of Figure 5 to obtain a k-show credential with organization O. The tuple $(P_{(U,O)}, Q_{(k,U,O)}, c_{(k,U,O)}, e_{(k,U,O)})$ is U's k-show credential with organization O. The pseudonym $P_{(U,O)}$ does not depend on k and can therefore be used both in the multiple-show as well as the k-show setting.

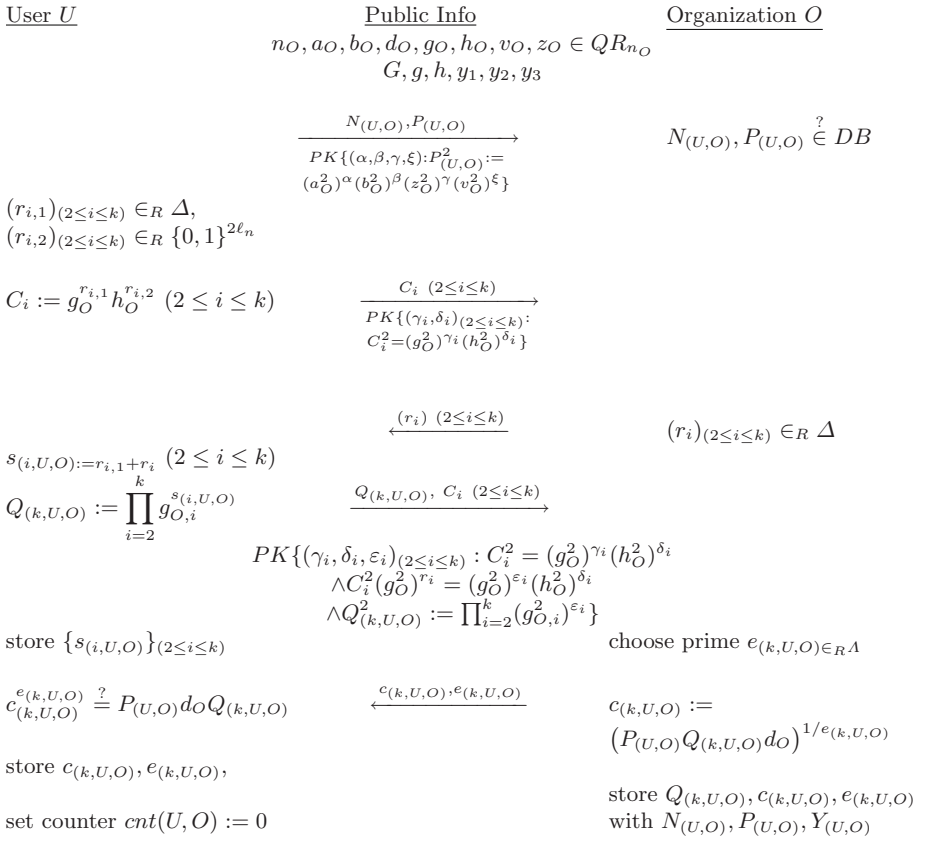


Fig. 5. Issuing protocol for revocable k-show credentials

3.3 Showing a Revocable *k*-Show Credential to an Organization

User U is allowed to show his credential up to k times without the showings being linked to his identity or to the instance of the issuing protocol by which it has obtained the credential. Figure 6 describes how the showing protocol works.

3.4 Local User Identification and Credential Revocation

A verifying organization V in the showing protocol above, submits the showing transcript offline to the issuing organization O, which in turn will be able to

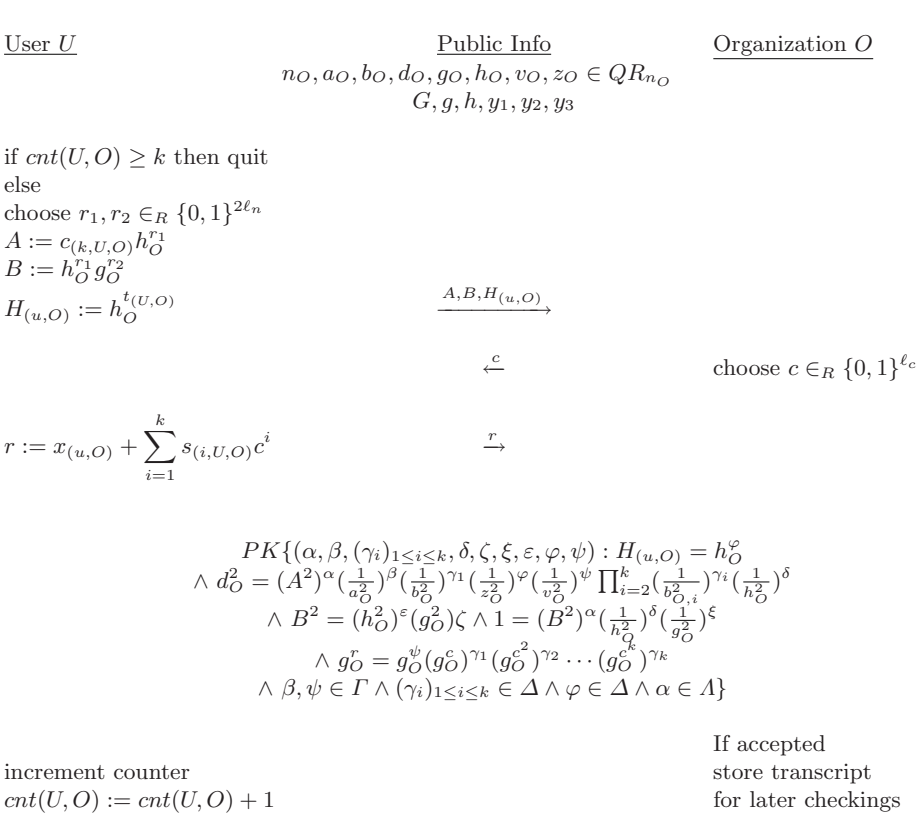


Fig. 6. Showing protocol for revocable k-show credentials

link the user's showings via $H_{(u,O)}$ without actually identifying his pseudonym. In case user U shows his credential beyond the allowed threshold k, then any party can verifiably recover his secret key $x_{(u,O)}$ by interpolating (in the sense of Lagrange) any subset of $(k+1)$ challenge-response pairs (c, r) that were used in the showing protocols conducted by the user. Given $x_{(u,O)}$, organization O computes $Y_{(u,O)} = g^{x_{(u,O)}}$, which will determine user U's pseudonym $N_{(U,O)}$, and enable the revocation of his credentials. The value of $H_{(u,O)}$, retrieved from the incriminating transcripts above, is added to a blacklist of revoked credentials, and used to detect subsequent attempts to show revoked credentials.

3.5 Global User Identification and Credential Revocation

Global revocation allows for the identification of a user by an external referee to the system, denoted CA. The conditions leading to the de-anonymization of the user are negotiated by the verifying organization and the user prior to executing the show protocol. Once the de-anonymization condition is agreed upon, the user may proceed with the showing protocol as follows. The user starts by verifiably encrypting a part of his identity (Y_u) using the CA's public key and sends the

ciphertext, along with a proof of correctness, to the verifying organization. The de-anonymization condition is tied to the encryption, thereby forcing the CA to only decrypt the ciphertext when this condition is met. We denote this de-anonymization condition by L. The verifiable encryption scheme of Cramer and Shoup [CS98] is used to achieve this.

To show a credential to verifying organization V, user U starts by executing the protocol of Figure 7 with V. Both U and V then proceed with the rest of the show protocol as shown in Figure 6.

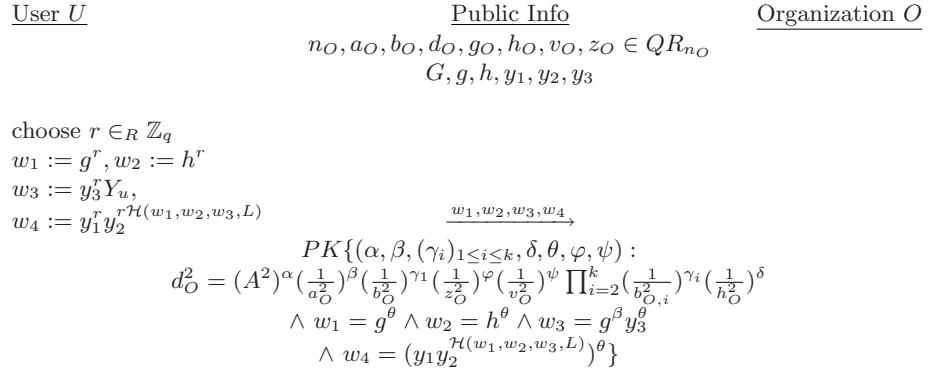


Fig. 7. Verifiable encryption of a user's revocation information

Global revocation. When provided with an encryption (w_1, w_2, w_3, w_4), a de-anonymization condition L, and evidence that L is satisfied, the CA checks whether $w_4 \stackrel{?}{=} w_1^{x_1+x_3\mathcal{H}(w_1,w_2,w_3,L)} w_2^{x_2+x_4\mathcal{H}(w_1,w_2,w_3,L)}$. If this is the case, the CA computes $Y_u = w_3/w_1^{x_5}$ which will determine the identity of user U.

4 Related Work

There have been a number of research efforts on limited-show credentials in the literature. In the following we list the ones that are most relevant to our work. In [LTW05], Lui et al., go around the problem of k-show credentials, by making a user initially fill-up a "wallet" of k coins, each of which can be shown only once. The identity of the user will be revealed as soon as a given coin is shown a second time. The work in [LTW05], also provides a mechanism to recognize and recover lost coins belonging to a given user. In [TFS04, NSN05], the authors propose constructions allowing users to show their credentials, anonymously and unlinkably, up to k times. All additional shows will be linked to a subset of the initial k shows, but the system falls short of identifying the abusers. More recently, Camenisch et al. [CHK+06] proposed a credential system that allows a user to anonymously authenticate at most k times within a pre-defined time period. While the latter represents a significant step towards solving the problem of k-show credentials, it still does not solve it, since users just need to wait until

the beginning of the next time period, before starting to reuse their credentials all the way from scratch.

5 Conclusion

This work extends the Identity Mixer framework to support anonymous k-show credentials, for $k > 1$. The proposed construction allows a user to show his credential up to k times, without the verifying or issuing organizations being able to link the different showings to the identity of the credential holder, or to the instance of the issuing protocol that generated the credential. Similar to the original framework, the credentials proposed in this work, are revocable both locally by the issuing organization, and globally by a trusted third party, upon fulfillment of a pre-defined de-anonymization condition.

Acknowledgement

This work is supported by the IWT SBO ADAPID project (Advanced Applications for e-ID cards in Flanders). Mohamed Layouni was partially funded by a doctoral scholarship from the Universitary Mission of Tunisia in North America.

References

[Bra94] Brands, S.: Untraceable off-line cash in wallet with observers. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 302–318. Springer, Heidelberg (1994)

[Bra00] Brands, S.: Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy. MIT Press, Cambridge (2000)

[Cha85] Chaum, D.: Security without identification: Transaction systems to make big brother obsolete. Commun. ACM 28(10), 1030–1044 (1985)

[CHK+06] Camenisch, J., Hohenberger, S., Kohlweiss, M., Lysyanskaya, A., Meyerovich, M.: How to win the clonewars: efficient periodic n-times anonymous authentication. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, pp. 201–210. ACM Press, New York (2006)

[CL01] Camenisch, J., Lysyanskaya, A.: Efficient non-transferable anonymous multi-show credential system with optional anonymity revocation. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 93–118. Springer, Heidelberg (2001)

[CL04] Camenisch, J., Lysyanskaya, A.: Signature schemes and anonymous credentials from bilinear maps. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 56–72. Springer, Heidelberg (2004)

[CP92] Chaum, D., Pedersen, T.P.: Wallet databases with observers. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 89–105. Springer, Heidelberg (1993)

[CS98] Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (1998)

[Gor98] Gordon, D.M.: A survey of fast exponentiation algorithms. Journal of Algorithms 27, 129–146 (1998)

[Hig07] The Higgins Trust Framework Project. URL functional as of (February 2007) http://www.eclipse.org/higgins/

[Ide07] The Identity Mixer: URL functional as of (February 2007), http://www.zurich.ibm.com/security/idemix/

[LTW05] Liu, J.K., Tsang, P.P., Wong, D.S.: Recoverable and untraceable e-cash. In: Chadwick, D., Zhao, G. (eds.) EuroPKI 2005. LNCS, vol. 3545, pp. 206–214. Springer, Heidelberg (2005)

[NSN05] Nguyen, L., Safavi-Naini, R.: Dynamic k-times anonymous authentication. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 318–333. Springer, Heidelberg (2005)

[TFS04] Teranishi, I., Furukawa, J., Sako, K.: k-times anonymous authentication (extended abstract). In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 308–322. Springer, Heidelberg (2004)

[UPr07] The U-Prove SDK: URL functional as of (February 2007), http://www.credentica.com/uprove_sdk.html

On Partial Anonymity in Secret Sharing

Vanesa Daza and Josep Domingo-Ferrer

Rovira i Virgili University
UNESCO Chair in Data Privacy
Department of Computer Engineering and Mathematics
Av. Països Catalans, 26, E-43007 Tarragona, Catalonia
{vanesa.daza,josep.domingo}@urv.cat

Abstract. Anonymous secret sharing schemes allow a secret to be recovered from shares regardless of the identity of shareholders. Besides being interesting in its own right, this property is especially appealing to guarantee the anonymity of participants when secret sharing is used as a building block of more general distributed protocols (*e.g.* to anonymously share the secret key corresponding to a public key). However, current constructions of anonymous secret sharing schemes are not very efficient (because of the number of shares that every participant must hold) and existing bounds do not leave much room for optimism. In this paper we propose to weaken the anonymity condition to partial anonymity, where by partial anonymity we mean that the identity of the participant is not made public, but he is known to belong to some subset. That is, the search for a participant narrows down to one in a set of possible candidates. Furthermore, we propose a general construction of partial anonymous secret sharing schemes.

Keywords: Privacy, Protocols, Secret sharing.

1 Introduction

Anonymous secret sharing schemes allow a secret to be recovered from a set of shares without knowledge of which participants hold which shares. That is, in such schemes the computation of the secret can be carried out regardless the identities of shareholders. Beyond its intrinsic interest, anonymous secret sharing is particularly attractive to guarantee the anonymity of participants in more general distributed protocols. A typical application is anonymous sharing of the secret key corresponding to a certain public key. Unfortunately, the constructions of anonymous secret sharing schemes in the literature are not very efficient (in terms of the number of shares that every participant must hold) and existing bounds [4,16] do not leave much hope for forthcoming efficient constructions.

Anonymous secret sharing schemes were introduced in 1988 by Stinson and Vanstone [22]. Phillips and Phillips [17] proved that only some specific access structures can yield anonymous secret sharing schemes where the size of the shares given to each participant is equal to the size of the secret (smallest possible size). Later on, Blundo and Stinson [4] gave general constructions of anonymous secret sharing schemes. They also gave lower bounds on the size of the set

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 193–202, 2007.

of shares (as a function of the size of the secret) both for threshold and non-threshold access structures. However, their constructions are not very efficient and their lower bounds preclude substantially improved forthcoming constructions. Since then, some authors have proposed constructions of anonymous secret sharing schemes, but either they are quite inefficient or they are restricted to the particular $(2, n)$ threshold case.

1.1 Contribution and Plan of This Paper

The lack of efficient constructions for anonymous secret sharing motivates us to weaken the anonymity condition in quest of efficiency, measured in terms of the number of shares that must be held by any participant. In that sense, we introduce the notion of partial anonymity with the aim of providing a tradeoff between the level of anonymity achieved by a scheme and its efficiency. Roughly speaking, in partial anonymous secret sharing the identity of the participant is not made public, but he is known to belong to some subset. In other words, the search for a participant narrows down to one in a set of possible candidates. This principle bears some vague resemblance to k-anonymity [18] used for privacy in databases and k-anonymity to preserve privacy in communication protocols [23,24]. On the practical side, we propose an efficient construction of a scheme fulfilling the partial anonymity property.

The rest of the paper is organized as follows. We introduce some basic concepts on secret sharing schemes in Section 2. In Section 3 we review the notion of anonymous secret sharing schemes and we introduce the notion of partially anonymous secret sharing schemes. In Section 4 we provide some constructions of partially anonymous secret sharing schemes. Finally, we conclude in Section 5.

2 Secret Sharing

Secret sharing schemes were independently introduced by Shamir [19] and Blakley [2] in 1979. A secret sharing scheme is a method whereby a special entity D, usually called dealer, distributes a secret s among a set $\mathcal{P} = \{P_1, \ldots, P_n\}$ of n players. The dealer secretly sends to every player P_i his share s_i of the secret s in such a way that only authorized subsets can recover the secret whereas non-authorized subsets obtain no information on the secret s.

A basic principle when designing secret sharing schemes is to minimize the amount of secret material. Therefore, the length of the shares should be as small as possible. In a secret sharing scheme the length of any share of a participant is greater than or equal to the length of the secret. When they are equal, the scheme is called ideal.

The family Γ of the subsets of shares authorized to recover the secret is called access structure. Any access structure is assumed to be monotone, that is, any superset of an authorized subset is also an authorized subset. A particular case is an access structure formed by those sets of players with at least t players, that is,

$$\Gamma = \{A \subset \mathcal{P} \mid\mid A \mid \geq t\}$$

Parameter t is usually called *threshold* and the corresponding access structure is a (t, n) threshold access structure (or t-out-of-n threshold access structure).

Shamir's secret sharing scheme [19] realizes an ideal (t, n) threshold access structure by means of Lagrange polynomial interpolation. Indeed, let $\mathbb{Z}_q$ be a finite field with $q > n$ and $s \in \mathbb{Z}_q$ the secret to be shared. The dealer picks a polynomial $p(x)$ of degree at most $t-1$, with free term the secret s, that is $p(0) = s$. The polynomial $p(x)$ can be written as $p(x) = s + \sum_{j=1}^{t-1} a_j x^j$, where $a_j \in \mathbb{Z}_q$ has been randomly chosen.

Every player P_i is univocally assigned a value $\alpha_i \in \mathbb{Z}_q$. Then, D privately sends to player P_i his share $s_i = p(\alpha_i)$, for $i = 1, \ldots, n$.

In this way, a set $A \subset \mathcal{P}$ of at least t players can recover the secret $s = p(0)$ by interpolating the set of shares they hold:

$$p(0) = \sum_{P_i \in A} s_i \lambda_i^A = \sum_{P_i \in A} s_i \left(\prod_{P_j \in (A \setminus P_i)} \frac{-\alpha_j}{\alpha_i - \alpha_j} \right),$$

where λ_i^A are the Lagrange coefficients.

On the other hand, it is not difficult to prove that less than t players do not have any option better than random guessing to find out the secret.

For the particular case of (n, n)-threshold access structures ($\Gamma = \mathcal{P}$), that is, where all participants must jointly co-operate to recover the secret, more efficient constructions exist. They are not only ideal, but the dealer's computations are simpler. In [15], Karnin, Greene and Hellman proposed the following (n, n)-threshold secret sharing scheme. To share a secret s among a set $\mathcal{P} = \{P_1, \ldots, P_n\}$ of n players, the dealer selects at random $s_i \in \mathbb{Z}_q$, for any player P_i, for $i = 1, \ldots, n-1$ and secretly sends s_i to participants P_i as his secret share. Then, D computes the share of the participant P_n as follows: $s_n = s - \sum_{i=1}^{n-1} s_i \in \mathbb{Z}_q$ and sends s_n to P_n. Note that when all participants join their shares, they recover the secret s by simply adding their shares in $\mathbb{Z}_q$.

Although threshold access structures have been extensively studied and used in the literature not only for secret sharing schemes (see, for example, [3,6]) but also for more general protocols (see, for example, [8,20,7]), they correspond to quite peculiar situations where all players play exactly the same role. On the contrary, what usually happens in real situations is that different players play different roles. For example, some players can have some restrictions and other some special privileges, or players can be divided into different categories depending on some properties. This leads to access structures more general than thresholds. Ito, Saito and Nishizeki [14] proved that for any monotone access structure there always exists a secret sharing scheme realizing it. The main drawback of their construction is that the size of shares is exponential in the number of parties in the access structure.

We describe next a general family of access structures that will be used in our proposal. It is the compartmented access structure, introduced by Simmons in [21]. There is a set of different compartments $C_1, \ldots, C_m$ in such a way that every participant is placed in a compartment, for some $i = 1, \ldots, m$. We

define $\psi : \{1, \ldots, n\} \rightarrow \{1, \ldots, m\}$ as the function assigning to each player P_i a compartment, denoted by $C_{\psi(i)}$. Then, given positive integers $t_1, \ldots, t_m$ and $t \geq \sum_{i=1}^{m} t_i$, the access structure consists of all subsets with at least t_i participants in C_i and a total of at least t participants. That is,

$$\Gamma = \{A \subset \mathcal{P} \mid\mid A \cap C_i \mid \geq t_i, \forall i = 1, \ldots, m \,, \mid A \mid \geq t\}.$$

Brickell [5] proved that there exist ideal secret sharing schemes realizing compartmented access structures. We review next the construction by [11], restricted to the particular case when $t = \sum_{i=1}^{m} t_i$. For the case $t > \sum_{i=1}^{m} t_i$, we refer the reader to [11,5].

The solution is based in Shamir's secret sharing schemes. The main idea is that the dealer shares a secret s by using an $(m-1)$-degree polynomial. Each compartment is related to a share of s. Then, to compute their share, at least t_i players in compartment C_i must join their shares. During the set-up phase, the dealer distributes the share of each compartment C_i among the players in the compartment by using a $(t_i, |C_i|)$-secret sharing scheme. Specifically D randomly selects an $(m-1)-$degree polynomial $P(x) \in \mathbb{Z}_q[x]$, such that $P(0) = s$. Let $s_i = P(\alpha_i)$, for $i = 1, \ldots, m$, where α_i is a public value associated with C_i. Then he distributes s_i among the players in compartment C_i using independent Shamir schemes. That is, he randomly selects a polynomial $F_i(x) \in \mathbb{Z}_q[x]$ of degree $t_i - 1$, for all $i = 1, \ldots, m$. Then, D secretly sends to every player P_i his share $F_{\psi(i)}(\beta_i)$, where β_i is the public value in $\mathbb{Z}_q$ associated to participant P_i.

Compartmented access structures are actually a special case of the more general multipartite access structure [13,9].

3 Anonymity and Partial Anonymity in Secret Sharing

Two kinds of anonymity in secret sharing are described in the literature. On the one hand, those schemes where shareholder identification is not required to successfully recover the secret, but the identity of the shareholder can be derived from the share. On the other hand, those schemes with the additional feature that players cannot be identified even when they show their shares. In other words, nobody can figure out the identity of a participant from the share he holds. The former category of schemes is usually called anonymous secret sharing (sometimes this anonymity is also referred to as submission anonymity). The latter category is usually called cryptographic anonymous secret sharing schemes [10,12]. The cryptographic notion of anonymity for secret sharing schemes is stronger than the submission notion. Some of the schemes satisfying submission anonymity do not offer cryptographic anonymity. This is because each share directly identifies the owner. In spite of this fact, submission anonymity in secret sharing remains especially interesting for example as a building block of some other distributed cryptographic protocols (*e.g.* distributed signature schemes). Then, when a participant publishes his partial information nobody is able to identify him because he does not directly publish his share but some information derived from it.

Several constructions have been published, both for submission anonymity [4,16,22] and for cryptographic anonymity [12]. However they are not very efficient and the most efficient ones have a small threshold ($t = 2$). Motivated by this fact, we introduce the concept of *partial anonymity* for secret sharing schemes. The main idea is to trade off anonymity and efficiency, by relaxing the anonymity condition to obtain more efficient schemes. This idea applies to both kinds of anonymity (submission and cryptographic) described before.

In order to define partial anonymity, we will lean on the definition of (t, n) anonymous threshold secret sharing in [1]. Let Σ be a (t, n) threshold secret sharing scheme on the set of participants $\mathcal{P} = \{P_1, \ldots, P_n\}$. For every secret s we note the set of shares as the vector $(s_1, \ldots, s_n)$, where s_i is the share (not necessarily a single value in $\mathbb{Z}_q$) held by player P_i.

Definition 1. *[Beimel-Franklin] A secret sharing scheme Σ realizing a (t, n) threshold access structure is said to be anonymous if, for every secret s, every vector of shares $(s_1, \ldots, s_n)$ and every permutation $\pi : \{1, \ldots, n\} \to \{1, \ldots, n\}$, the vector $(s_1, \ldots, s_n)$ is a vector of shares for the secret s if and only if the vector $(s_{\pi(1)}, \ldots, s_{\pi(n)})$ is a vector of shares for s.*

The above definition captures the idea that the reconstruction of the secret can be performed from the shares without knowing the identities of the parties holding those shares. That is, if a vector of shares is possible given a secret s, then every possible permutation in the order of the coordinates in this vector is possible given the secret s.

Note that this situation does not happen in Shamir's (t, n) threshold secret sharing scheme described in Section 2 whenever $1 < t < n$. Indeed, the Lagrange coefficients λ_i^A of a participant P_i depend on the set A to reconstruct the secret. However, when $t = n$, Shamir's scheme fulfills Definition 1 (and also Karnin, Greene and Hellman's scheme, see Section 2), although the identity of the participants arises implicitly from the access structure (as it is known that all participants take part in the protocol).

In Definition 1, a permutation is applied to the whole set of shares to guarantee anonymity of the shareholders. We do not propose to apply a permutation to the whole set of shares, but to divide the set of participants into different groups and to fulfill Definition 1 within each group to guarantee anonymity of a participant inside its group.

More specifically, let $G_1, \ldots, G_m \subset \mathcal{P}$ be subsets of participants in such a way that every participant P_i is placed in one and only one subset $G_1, \ldots, G_m$. Let $\psi : \{1, \ldots, n\} \to \{1, \ldots, m\}$ be the mapping that assigns each participant to a group, so that participant P_i will be placed in group $G_{\psi(i)}$. Let n_i be the cardinality of each set G_i and let t_i be a threshold assigned to set G_i, for all $i = 1, \ldots, m$. Then, we require that Definition 1 be satisfied for each of the sets G_i to guarantee that, locally, the participant is anonymous within his group and, globally, the only partial information that is obtained is that the participant is a member of a specific group.

Let Σ_i be a (t_i, n_i) secret sharing scheme for the set of participants in G_i. For every secret s we note the set of shares in Σ_i as the vector $(s^i_{j_1}, \ldots, s^i_{j_{n_i}})$, where s^i_j

is the share (not necessarily a single value in $\mathbb{Z}_q$) held by player P_j and we assume that the set of players in G_i is $P_{j_1}, \ldots, P_{j_{n_i}}$, where $\{j_1, \ldots, j_{n_i}\} \in \{1, \ldots, n\}$ for every j. Then, if at least t_i participants in G_i pool their shares, they can directly recover the secret s and the only information that is leaked is that they are participants in the set G_i.

However, it may be the case that the secret is not recoverable by players in each set G_i or maybe it is not desirable that members of a unique set can recover the secret. For example, in the former situation, to satisfy the secrecy condition, each threshold t_i must be at least t (a general threshold set related to the overall number of participants n) but the cardinality n_i of G_i may be smaller than t. A way to circumvent this problem of small sets is to require a threshold t_i for each set G_i in such a way that the sum of all the thresholds is at least t. In this way, the threshold t_i can be adapted to the size of G_i. Then, the resulting access structure is a $(G_1, \ldots, G_m)$ compartmented access structure (see Section 2 above).

Let Σ be a secret sharing scheme realizing a $(G_1, \ldots, G_m)$ compartmented access structure on the set of participants $\mathcal{P}$ with thresholds $t_1, \ldots, t_m, t = \sum_{i=1}^{m}$. For every secret s we note the set of shares as the vector $(s_1, \ldots, s_n)$, where s_i is the share (not necessarily a single value in $\mathbb{Z}_q$) held by player P_i. Then a $(t_1, \ldots, t_m)$-partially anonymous secret sharing scheme realizing a $(G_1, \ldots, G_m)$ compartmented access structure can be defined as follows:

Definition 2. *[Partial anonymous secret sharing] A secret sharing scheme Σ is said to be $(t_1, \ldots, t_m)$-partially anonymous if, for every secret s, every vector of shares $(s_1, \ldots, s_n)$ and every permutation $\pi : \{1, \ldots, n\} \to \{1, \ldots, n\}$ such that $\pi(G_i) = G_i$ (*e.g.* for every $P_j \in G_i$, $P_{\pi(j)} \in G_i$) for all $i = 1, \ldots, m$, the vector $(s_1, \ldots, s_n)$ is a vector or shares for the secret s if and only if $(s_{\pi(1)}, \ldots, s_{\pi(n)})$ is a vector of shares for s.*

Note that for $m = 1$, the new Definition 2 is equivalent to Definition 1 by [1].

4 Some Constructions

In this section we provide a general construction of $(t_1, \ldots, t_m)$-partially anonymous secret sharing scheme from different anonymous secret sharing schemes. The key point is that anonymous secret sharing schemes used as a building block have smaller thresholds, so *the share length of the $(t_1, \ldots, t_m)$-partially anonymous secret sharing scheme is considerably smaller that the one in a (t, n) anonymous secret sharing scheme* (remember that $t = \sum_{i=1}^{m} t_i$ and $n = \sum_{i=1}^{m} n_i$).

To begin with, let us describe the very particular case $t_1 = \ldots = t_m = 1$. That is, we assume the secret is recovered if at least one participant in each of the compartments G_i pool their shares. Note that in this case we are considering $m = t$. Similar reasonings apply if the threshold considered is less than m (at least $t < m$ participants from t different compartments are required to recover the secret).

Example 1. Let $\mathcal{P} = \{P_1, \ldots, P_n\}$ be the set of players and let $G_1, \ldots, G_m$ be compartments such that any set in $\mathcal{P}$ is placed exactly in one compartment, in such a way that no compartment remains empty. Let us construct a $(1, \ldots, 1)$-partially anonymous secret sharing scheme as follows. The dealer picks at random a polynomial $p(x)$ of degree at most $m-1$, whose free term is the secret s he wants to share. Let $p(x)$ be $p(x) = s + \sum_{j=1}^{m-1} a_j x^j$, where $a_j \in \mathbb{Z}_q$ has been randomly chosen, for $j = 1, \ldots, m-1$.

Every compartment G_i is univocally associated a value $\alpha_i \in \mathbb{Z}_q$. Then, D privately sends to each player in G_i his share $s_i = p(\alpha_i)$, for $i = 1, \ldots, m$.

Therefore, a set $A \subset \mathcal{P}$ with at least one player from every compartment can recover the secret $s = p(0)$ by interpolating the set of shares they hold. The main difference with Lagrange interpolation in the usual Shamir scheme is that now the participants do not have to publish their value α_i but only the compartment G_i they belong to. So, the only information that an outsider can derive is that the participant belongs to G_i.

It is easy to check that such a secret sharing scheme fulfills Definition 2 because the shares of all players in G_i are the same, for any $i = 1, \ldots, m$.

This construction can also use Karnin, Greene and Hellman's scheme instead of Shamir's scheme. □

The above construction can be generalized to obtain a $(t_1, \ldots, t_m)$-partially anonymous secret sharing scheme using as building blocks both (t_i, n_i)-anonymous secret sharing schemes and compartment secret sharing constructions. We detail this idea below.

Theorem 1. *Let $G_1, \ldots, G_m$ be disjoint compartmented subsets of players such that $G_1 \cup \cdots \cup G_m = \mathcal{P}$. Let Σ_i be a (t_i, n_i) anonymous secret sharing schemes on the set of players G_i, for $i = 1, \ldots, m$. Then, there exists a $(t_1, \ldots, t_m)$-partially anonymous secret sharing scheme Σ realizing a $(G_1, \ldots, G_m)$ compartmented access structure. Furthermore, the share length of scheme Σ is lower-bounded by the maximum of the share lengths of schemes Σ_i.*

Proof. In order to prove the theorem, we will explicitly construct the scheme Σ from $\Sigma_1, \ldots, \Sigma_m$. Without loss of generality, we can assume that

$$G_1 = \{P_1, \ldots, P_{n_1}\}$$

$$G_2 = \{P_{n_1+1}, P_{n_1+2}, \ldots, P_{n_1+n_2}\}$$

$$\ldots$$

$$G_i = \{P_{n_1+\ldots+n_{i-1}+1}, P_{n_1+\ldots+n_{i-1}+2}, \ldots, P_{n_1+\ldots+n_{i-1}+n_i}\}$$

$$\ldots$$

$$G_m = \{P_{n_1+\ldots+n_{m-1}+1}, P_{n_1+\ldots+n_{m-1}+2}, \ldots, P_n\}$$

Then, to share a secret s, the dealer chooses at random a polynomial $P(x)$ of degree $m-1$. Let $s^i = P(\alpha_i)$ be the share of compartment G_i, for $i = 1, \ldots, m$,

where $\alpha_i \in \mathbb{Z}_q$ is publicly associated to compartment G_i. Then, to distribute s^i (for every $i = 1, \ldots, m$) among the players in G_i he uses the anonymous secret sharing scheme Σ_i. Let

$$s_{n_1+\ldots+n_{i-1}+1}, s_{n_1+\ldots+n_{i-1}+2}, \ldots, s_{n_1+\ldots+n_{i-1}+n_i}$$

be the set of shares for players

$$P_{n_1+\ldots+n_{i-1}+1}, P_{n_1+\ldots+n_{i-1}+2}, \ldots, P_{n_1+\ldots+n_{i-1}+n_i}$$

respectively. Then, D privately sends $s_{n_1+\ldots+n_{i-1}+j}$ for $j = 1, \ldots, n_i$.

In this way, by construction it is easy to check that Σ realizes a $G_1, \ldots, G_m$ compartmented access structure (see Section 2). Thus, a subset not in the $G_1, \ldots, G_m$ compartmented access structure obtains no information on the secret, even if they join all their shares.

It only remains to prove that Σ is, in fact, a $(t_1, \ldots, t_m)$-partially anonymous secret sharing scheme. To do so, we need to check that Definition 2 is fulfilled. Indeed, for any secret s and any permutation $\pi : \{1, \ldots, n\} \to \{1, \ldots, n\}$ such that $\pi(G_i) = G_i$

$$(s_1, \ldots, s_{n_1}, \ldots, s_{n_1+\ldots+n_{i-1}+1}, \ldots, s_n)$$

is a set of shares for s if and only if

$$(s_{\pi(1)}, \ldots, s_{\pi(n_1)}, \ldots, s_{\pi(n_1+\ldots+n_{i-1}+1)}, \ldots, s_{\pi(n)})$$

is a set of shares of s. This easily follows from the fact that the schemes Σ_i are anonymous secret sharing schemes; that is, $(s_{n_1+\ldots+n_{i-1}+1}, \ldots, s_{n_1+\ldots+n_{i-1}+n_i})$ is a set of shares for s^i if and only if $(s_{\pi(n_1+\ldots+n_{i-1}+1)}, \ldots, s_{\pi(n_1+\ldots+n_{i-1}+n_i)})$ is a set of shares for s^i.

By construction, the length of the shares in Σ is lower-bounded by the maximum length of the shares of Σ_i. □

By [4], the lower bound on the size of the share domain depends multiplicatively on the amount $n - t$ for a (t, n) threshold access structure. Then, by Theorem 1, if $G_1, \ldots, G_m$ and $t_1, \ldots, t_m$ are chosen in a way that $\max_{i=1,\ldots,m} \{n_i - t_i\} < (n - t)$, the domain of shares of the resulting $(t_1, \ldots, t_m)$-partially anonymous secret sharing schemes is lesser than the domain of shares of the (t, n) threshold secret sharing scheme.

5 Conclusion

Anonymous secret sharing schemes allow a secret to be recovered from shares regardless of the identity of shareholders. Beyond their intrinsic interest, anonymous secret sharing allows anonymous participation in more general cryptographic protocols. A typical application is to make it possible for several parties to anonymously share the secret key corresponding to a public key.

Since current constructions of anonymous secret sharing schemes are not very efficient, we have introduced the notion of partial anonymous secret sharing schemes in an attempt to relax anonymity requirements to obtain more efficient constructions. A general construction for such partially anonymous schemes has also been described.

Disclaimer and Acknowledgments

The authors are solely responsible for the views expressed in this paper, which do not necessarily reflect the position of UNESCO nor commit that organization. This work was partly supported by the Spanish Ministry of Education through project SEG2004-04352-C04-01 "PROPRIETAS" and by the Government of Catalonia under grant 2005 SGR 00446.

References

1. Beimel, A., Franklin, M.: Weakly-private secret sharing. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 253–272. Springer, Heidelberg (2007)
2. Blakley, G.R.: Safeguarding cryptographic keys. In: Proceedings of the National Computer Conference, American Federation of Information, Processing Societies Proceedings, vol. 48, pp. 313–317 (1979)
3. Blundo, C., Giorgia Gaggia, A., Stinson, D.R.: On the dealer's randomness required in secret sharing schemes. Designs, Codes and Cryptography 11, 107–122 (1997)
4. Blundo, C., Stinson, D.R.: Anonymous secret sharing schemes. Discrete Applied Mathematics 77, 13–28 (1997)
5. Brickell, E.F.: Some ideal secret sharing schemes. J. Math. Combin. Comput. 6, 105–113 (1989)
6. Carpentieri, M.: A perfect threshold secret sharing scheme to identify cheaters. Designs, Codes and Cryptography 5, 183–188 (1995)
7. Cramer, R., Damgård, I., Nielsen, J.: Multiparty computation from threshold homomorphic encryption. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 280–299. Springer, Heidelberg (2001)
8. Desmedt, Y., Frankel, Y.: Threshold cryptosystems. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 307–315. Springer, Heidelberg (1990)
9. Farràs, O., Martí-Farré, J., Padró, C.: Ideal multipartite secret sharing schemes. In: Eurocrypt 2007. LNCS, Springer, Heidelberg (to appear, 2007)
10. Gehrmann, C.: Topics in Authentication Theory, Ph.D. Thesis, Lund University (1997)
11. Ghodosi, H., Pieprzyk, J., Safavi-Naini, R.: Secret sharing in multilevel and compartmented groups. In: Boyd, C., Dawson, E. (eds.) ACISP 1998. LNCS, vol. 1438, pp. 367–378. Springer, Heidelberg (1998)
12. Guillermo, M., Martin, K.M., O'Keefe, C.M.: Providing anonymity in unconditionally secure secret sharing schemes. Designs, Codes and Cryptography 28, 227–245 (2004)
13. Herranz, J., Sáez, G.: New results on multipartite access structures. IEE Proceedings of Information Security 153-4 (December 2006)
14. Ito, M., Saito, A., Nishizeki, T.: Secret sharing scheme realizing any access structure. In: Proc. IEEE Globecom'87, pp. 99–102 (1987)

15. Karnin, E.D., Greene, J.W., Hellman, M.E.: On secret sharing systems. IEEE Transactions on Information Theory 29(1), 35–41 (1983)
16. Kishimoto, W., Okada, K., Kurosawa, K., Ogata, W.: On the bound for anonymous secret sharing schemes. Discrete Applied Mathematics 121(1-3), 193–202 (2002)
17. Phillips, S.J., Phillips, N.C.: Strongly ideal secret sharing schemes. Journal of Cryptology 5, 185–191 (1992)
18. Samarati, P., Sweeney, L.: Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Tech. rep., SRI International (1998)
19. Shamir, A.: How to share a secret. Communications of the ACM 22, 612–613 (1979)
20. Shoup, V.: Practical threshold signatures. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 207–220. Springer, Heidelberg (2000)
21. Simmons, G.J.: How to (really) share a secret. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 390–448. Springer, Heidelberg (1990)
22. Stinson, D.R., Vanstone, S.A.: A combinatorial approach to threshold schemes. SIAM J. Disc. Math. 1, 230–236 (1988)
23. von Ahn, L., Bortz, A., Hopper, N.J.: k-anonymous message transmission. In: Proc. of the 10th ACM Conference on Computer and Communications Security, pp. 122–130. ACM Press, New York (2003)
24. Xu, S., Yung, M.: k-anonymous secret handshakes with reusable credentials. In: Proc. of the 11th ACM Conference on Computer and Communications Security, pp. 158–167. ACM press, New York (2004)

Anonymous Identification and Designated-Verifiers Signatures from Insecure Batch Verification

Sherman S.M. Chow[1] and Duncan S. Wong[2]

[1] Department of Computer Science
Courant Institute of Mathematical Sciences
New York University, NY 10012, USA
schow@cs.nyu.edu
[2] Department of Computer Science
City University of Hong Kong
Hong Kong, China
duncan@cs.cityu.edu.hk

Abstract. Versatility in cryptography is interesting. Instead of building a secure scheme from another secure one, this paper presents an oxymoron making use of the insecurity of a scheme to give useful feature in another context. We show the insecurity of the batch verification algorithms in Cui *et al.*'s work about an identity-based (ID-based) signature scheme. Following Chow *et al.*'s idea in EuroPKI 2005, we turn such attack into a secure ID-based ring signature scheme. We also show how to add linkability. We present two applications of our scheme, which are a short ID-based strong designated verifier signature scheme and an ID-based ad-hoc anonymous identification scheme, with an extension secure against a concurrent man-in-the-middle attack.

Keywords: Identity-based, ad hoc anonymous identification, strong designated verifier signatures, ring signatures, linkability, bilinear pairings.

1 Introduction

Versatility in cryptography is interesting. One central line of research is to identify which cryptographic primitive can help achieving what security function, often in an inconceivable way. An example in the cryptographic scheme level is using multi-trapdoor commitment scheme [16] to transform any proof of knowledge (or identification) protocol into one which is secure against a concurrent man-in-the-middle attack. From application level, we see electronic voting schemes [21,25] based on ring signature schemes with linkability [21].

Instead of building a secure scheme from another secure one, this paper turns the insecurity of a scheme into an useful feature in another context.

1.1 Linkable Ring Signatures

Ring signature scheme is a group-oriented signature scheme with 1-out-of-n signer anonymity and spontaneous signer group formation. As observed in [9],

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 203–219, 2007.

identity-based (ID-based) ring signature is better than its counterpart in traditional public key infrastructure, in terms of spontaneity. One can even involve members who have not requested for their private key from the key generation centre (KGC).

In Chow *et al.*'s survey of ring signatures [9], one of the major paradigms of ID-based ring signature schemes can be seen as derived from the failure to batch verify the underlying ID-based signatures. Indeed, a recent scheme in [2] can be regarded as a work exploiting the aggregate verification of the standard ID-based signature in [23], although the authors gave no discussion about this issue. Here we use the same methodology to transform our attack on the batch verification of ID-based signatures proposed recently by Cui *et al.* [12] into a secure ring signature scheme, with further modification to remove all unnecessary components from straightforward transformation.

Following the recent trends of providing different level of anonymity in group-oriented signatures (e.g. [1,8,21,25]), we show how to add linkability into the resulting scheme. A detailed comparison of the efficiency with existing schemes will also be made. We remark the recent generic approach to build ID-based signature schemes with special properties [15] is not applicable to ring signatures.

A recent application of ring signature is concurrent signature, which partially solves the fair exchange problem without the help of trusted third party. Our ring signature also fits in the generic construction of concurrent signature in [10].

1.2 Strong Designated Verifier Signatures

The first application of our scheme is identity-based strong designated verifier signature. Similar to ring signature, designated verifier signature (DVS) scheme is a privacy-oriented one in which the signature produced can only be verified by a specific user, but no one else. Its interactive version, designated verifier proof, is introduced in [18]. Apart from providing the apparent restricted-verifiability, it also helps to solve other problems in secure two-party computations.

Ring signature gives a simple method to give DVS. By involving the intended recipient of the signed message as the second member of a 2-persons group, only the recipient can ascertain the message's authenticity, but no one else.

The idea of *strong* designated verifier signature has also been considered in [18]. Informally, *strong* here means the use of the verifier's private key is essential to perform verification. The strong property can be implemented by a chosen-ciphertext-secure (CCA2) encryption [20,24]. However, such approach gives inefficient construction as CCA2 encryption needs more computation than a semantic-secure one in general. Considering the signature size, a ciphertext and possibly a validity check tag are appended to the signature. Two proposals without explicit encryption step are presented in [24]. One is a generic construction based on chameleon hash and the other is a concrete scheme with encryption step integrated with the signing step. Their concrete scheme follows Boneh-Franklin paradigm [5], thus an expensive $MapToPoint$ encoding function [5] is required.

Recently, a short ID-based strong DVS scheme is proposed in [17]. In their scheme, given a signature on a message m that an adversary wants to learn the

signer's identity, signing query of the messages m cannot be made. The deterministic nature gives the short signature size but also imposes this restriction. We propose a probabilistic scheme without this restriction. Our scheme does not use the $MapToPoint$ function, with the ciphertext component integrated into the ring signature. As a result, only a logical semantic-secure encryption is needed, instead of requiring full-blown security from CCA2 encryption.

A final remark is that, since designated verifier signature involves the public key of parties other than the signer, the generic approach for building ID-based signature schemes [15] is not applicable, similar to the case for ring signature.

1.3 Ad Hoc Anonymous Identification

Identification is useful in many scenarios, like access control of a certain resource. In this privacy-oriented age, positive identification of the user should be avoided, if only *membership* of the user in a certain group is what necessary to differentiate one with non-privileged outsiders. This concept can be achieved by anonymous identification that allows one to "anonymously identify" the membership. *Ad hoc* anonymous identification is introduced by Dodis *et al.* [13], in which the group formation can be done without explicit group enrollment procedure, so one may not be even aware that he has been included in a certain group for protecting someone else's anonymity.

ID-based ad hoc anonymous identification is studied by Nguyen [22]. Both of [13] and [22] use Fiat-Shamir heuristic [14] to transform the interactive identification protocol into a non-interactive ring signature scheme. Note that Nguyen made no claim about whether his scheme is secure against concurrent man-in-the-middle attack. An ID-based identification protocol secure against such attack in the standard model has been proposed recently by Kurosawa and Heng [19]; however, no anonymity protection is provided.

This paper gives a simple construction of ad hoc anonymous identification, the interactive counterpart of ring signatures. We will show the honest-verifier zero knowledge (HVZK) property and the special soundness of our identification protocol. Thanks to the similarity between the ID-based secret key extraction algorithm of our ring signature scheme and Boneh-Boyen signature scheme [4], we can use the compiler of Gennaro [16] to yield an ad hoc anonymous identification scheme secure against a concurrent man-in-the-middle attack.

2 Preliminaries

2.1 Bilinear Pairings and Related Complexity Assumption

Let $(\mathbb{G}_1, +)$ and $(\mathbb{G}_2, \cdot)$ be two cyclic groups of prime order q. The bilinear pairing is given as $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$, which satisfies the following properties:

1. *Bilinearity*: For all $P, Q, R \in \mathbb{G}_1$, $\hat{e}(P+Q, R) = \hat{e}(P, R)\hat{e}(Q, R)$, and $\hat{e}(P, Q+R) = \hat{e}(P, Q)\hat{e}(P, R)$.
2. *Non-degeneracy*: There exists $P, Q \in \mathbb{G}_1$ such that $\hat{e}(P, Q) \neq 1$.
3. *Computability*: It is efficient to compute $\hat{e}(P, Q)$ $\forall P, Q \in \mathbb{G}_1$.

Definition 1. *The k-Collision Attack Algorithm (k-CAA) Problem in* $\mathbb{G}_1$ *is defined as follows: For a fixed known integer* k*, given a* $(2k+2)$*-tuple* $(h_1, \cdots, h_k, P, Q = xP\frac{1}{h_1+x}P, \cdots, \frac{1}{h_k+x}P) \in {\mathbb{Z}_q}^* \times {\mathbb{G}_1}^{k+2}$*, output a pair* (A, c) *such that* $A = \frac{1}{c+x}P$ *where* $c \in \mathbb{Z}_p^* \backslash \{h_1, \cdots, h_k\}$*. We say that the* (k, τ, ϵ)*-CAA assumption holds in* $\mathbb{G}_1$ *if no* τ*-time algorithm has advantage at least* ϵ *in solving the k-CAA problem.*

The relation of k-CAA with some other problems can be found in [26].

2.2 Identity-Based Ring Signatures

An ID-based ring signature scheme consists of the following four algorithms: $\mathcal{SETUP}$, $\mathcal{KGEN}$, $\mathcal{SIG}$, and $\mathcal{VER}$.

- $\mathcal{SETUP}$: On an unary string input 1^k where k is a security parameter, it produces the master secret key s and the common public parameters *params*.
- $\mathcal{KGEN}$: On an input of the signer's identity $\mathsf{ID} \in \{0,1\}^*$ and the master secret key s, it outputs the signer's secret signing key S_{ID}. (The corresponding public verification key Q_{ID} can be computed easily by everyone.)
- $\mathcal{SIG}$: On input of a message m, a group of n users' identities $\bigcup\{ID_i\}$, where $1 \leq i \leq n$, and the secret keys of one members S_{ID_s}, where $1 \leq s \leq n$; it outputs an ID-based ring signature σ on the message m.
- $\mathcal{VER}$: On a ring signature σ, a message m and the group of signers' identities $\bigcup\{\mathsf{ID}_i\}$ as the input, it outputs $\top$ for "true" or $\perp$ for "false", depending on whether σ is a valid signature signed by a certain member in the group $\bigcup\{\mathsf{ID}_i\}$ on a message m.

These algorithms must satisfy the standard consistency constraint of ID-based ring signature scheme, i.e. if $\sigma = \mathcal{SIG}(m, \bigcup\{\mathsf{ID}_i\}, S_{\mathsf{ID}_s})$, and $\mathsf{ID}_s \in \bigcup\{\mathsf{ID}_i\}$, we must have $\mathcal{VER}(\sigma, \bigcup\{\mathsf{ID}_i\}, m) = \top$.

2.3 Identity-Based Strong Designated Verifier Signatures

An ID-based strong designated verifier signature scheme consists of the following four algorithms: $\mathcal{SETUP}$, $\mathcal{KGEN}$, $\mathcal{DSIG}$, and $\mathcal{DVER}$.

- $\mathcal{SETUP}$, $\mathcal{KGEN}$: the same as those discussed before.
- $\mathcal{DSIG}$: On input of a message m, the public key of the verifier Q_{ID_v}, and the secret key of the signer S_{ID_s}, it outputs an ID-based strong designated verifier signature σ on the message m.
- $\mathcal{DVER}$: On a signature σ, a message m, the public key of the verifier Q_{ID_s}, and the secret key of the verifier S_{ID_v} as the input, it outputs $\top$ for "true" or $\perp$ for "false", depending on whether σ is a valid signature signed by ID_s.

2.4 Identity-Based Ad Hoc Anonymous Identification

An ID-based ad hoc anonymous identification scheme consists of the following four algorithms: $\mathcal{SETUP}$, $\mathcal{KGEN}$, $\mathcal{P}$, and $\mathcal{V}$. The former two are the same

as those in ID-based ring signature. $\mathcal{P}$ and $\mathcal{V}$ constitutes a PPT challenge-response protocol that implement the prover and verifier, respectively. $\mathcal{P}$'s input is $(S_{\mathsf{ID}}, Q_{\mathsf{ID}}, params)$ and $\mathcal{V}$'s input is $(Q_{\mathsf{ID}}, params)$ only. After their intersection, $\mathcal{V}$ returns a decision in $\{\top, \perp\}$, denoting whether it accepts the proof.

3 Turning an Attack into a Scheme

3.1 Review of Cui *et al.*'s Scheme

Define $\mathbb{G}_1, \mathbb{G}_2$, and $\hat{e}(\cdot,\cdot)$ as in previous section. $H_0 : \{0,1\}^* \to \mathbb{Z}_q^*$ and $H_1 : \{0,1\}^* \to \mathbb{Z}_q^*$ are two cryptographic hash functions. Cui *et al.*'s ID-based signature scheme and the corresponding batch verification [12] are as follows.

$\mathcal{SETUP}$: The KGC chooses $x \in_R \mathbb{Z}_q^*$, keeps it as the master secret. The system parameters are: $\{\mathbb{G}_1, \mathbb{G}_2, q, \hat{e}(\cdot,\cdot), H_0(\cdot), H_1(\cdot), g = \hat{e}(P,P), P, P_{pub} = xP\}$.

$\mathcal{KGEN}$: The user submits $\mathsf{ID} \in \{0,1\}^*$ to KGC. KGC computes the private key S_{ID} by $\frac{1}{x+q_{\mathsf{ID}}}P$, and sends it back via a secure channel, where $q_{\mathsf{ID}} \leftarrow H_0(\mathsf{ID})$.

$\mathcal{SIG}$: To sign a message m with private key S_{ID}, the signer chooses $r \in_R \mathbb{Z}_q$, compute $u = g^r$, $h = H_1(m,u)$ and $V = (h+r)S_{\mathsf{ID}}$. The signature is (u, V).

$\mathcal{VER}$: Accepts the signature $\sigma = (u, V)$ as the one on a message m signed by the one with ID, if and only if $g^{H_1(m,u)} \cdot u = \hat{e}(P_{pub} + q_{\mathsf{ID}}P, V)$ holds.

Cui *et al.* suggested the following ways to verify signatures on different messages from the same signer, and signatures on different messages from different signers. To batch verify $\sigma_1 = (u_1, V_1), \cdots, \sigma_n$ from a single signer ID, firstly $h_i = H_1(m_i, u_i), i \in \{1, \cdots, n\}$ are recovered, accepts all the signature if and only if $g^{\sum_{i=1}^n h_i} \cdot \prod_{i=1}^n u_i = \prod_{i=1}^n \hat{e}(P_{pub} + q_{\mathsf{ID}}P, V_i) = \hat{e}(P_{pub} + q_{\mathsf{ID}}P, \sum_{i=1}^n V_i)$ holds. Similarly, for signatures $\sigma_1, \cdots, \sigma_n$ from signers $\mathsf{ID}_1, \cdots \mathsf{ID}_n$ respectively, firstly compute $h_i = H_1(m_i, u_i), i \in \{1, \cdots, n\}$, accepts all the signatures if and only if $g^{\sum_{i=1}^n h_i} \cdot \prod_{i=1}^n u_i = \prod_{i=1}^n \hat{e}(P_{pub} + q_{\mathsf{ID}_i}P, V_i) = \hat{e}(P_{pub}, \sum_{i=1}^n V_i)\hat{e}(P, \sum_{i=1}^n q_{\mathsf{ID}_i}V_i)$.

3.2 Attack of the Batch Verification Algorithms

We only consider the more general one, i.e. the batch verification the signatures from different signers. A similar attack can be applied to the single-signer case.

$\mathcal{ATTACK}$: Let $\{ID_1, \cdots, ID_n\}$ be the set of n signers. The actual signer, indexed by s, carries out the following steps to launch the attack.

1. Choose $u_i \in_R \mathbb{G}_2$, compute $h_i = H_1(m_i, u_i)$ $\forall i \in \{1, 2, \cdots, n\} \backslash \{s\}$.
2. Choose $r'_s \in_R \mathbb{Z}_q^*$; and choose $V_i \in_R \mathbb{G}_1$ $\forall i \in \{1, 2, \cdots, n\} \backslash \{s\}$
3. Compute $u_s = g^{r'_s} \hat{e}(P_{pub}, \sum_{i \neq s} \{V_i\}) \hat{e}(P, \sum_{i \neq s} \{q_{\mathsf{ID}_i} V_i\}) / \prod_{i \neq s} \{u_i\} / g^{\sum_{i \neq s} \{h_i\}}$.
4. Compute $h_s = H_1(m_s, u_s)$ and $V_s = (h_s + r'_s) S_{ID_s}$.
5. Output $\bigcup_{i=1}^n \{(u_i, V_i)\}$.

The below equations show the correctness of our attack.

$$
\begin{aligned}
& g^{\sum_{i=1} h_i} \cdot \prod_{i=1}^{n} u_i \\
&= g^{h_s} \cdot u_s \cdot g^{\sum_{i \neq s} h_i} \cdot \prod_{i \neq s} u_i \\
&= g^{h_s} \cdot g^{r's} \hat{e}(P_{pub}, \sum_{i \neq s} \{V_i\}) \hat{e}(P, \sum_{i \neq s} \{q_{\mathsf{ID}_i} V_i\}) \\
&= \hat{e}(P,P)^{(h_s + r's)} \hat{e}(P_{pub}, \sum_{i \neq s} \{V_i\}) \hat{e}(P, \sum_{i \neq s} \{q_{\mathsf{ID}_i} V_i\}) \\
&= \hat{e}(P_{pub} + q_{\mathsf{ID}_s} P, V_s)^{(h_s + r's)} \hat{e}(P_{pub}, \sum_{i \neq s} \{V_i\}) \hat{e}(P, \sum_{i \neq s} \{q_{\mathsf{ID}_i} V_i\}) \\
&= \hat{e}(P_{pub}, \sum_{i=1}^{n} V_i) \hat{e}(P, \sum_{i=1}^{n} q_{\mathsf{ID}_i} V_i).
\end{aligned}
$$

Considering the verification to get some intuition behind our attack, we can see that u_i is only "hindered" by a single term of $g^{H_1(m_i, u_i)}$, which gives an ideal place to "store" many public keys. Concretely, by generating only one signature, we can trick the verifier to accept many signatures from many different signers.

3.3 Building a Ring Signature Scheme

Now we turn the above attack into a ring signature scheme. The reason for this to be possible is that, even there is plenty of room to embed many public keys there, there is still a challenge h_s that one needs to use a secret key to "complete" the ring, in Chow *et al.*'s [9] terminology.

In view of this, we only need a single u_s and a single h_s, since other u's and h's are to be cancelled anyway. This is our first optimization. The second hack is that instead of choosing V_i's from $\mathbb{G}_1$ directly, we can choose v_i's from $\mathbb{Z}_q^*$ and compute $V_i = v_i P \in \mathbb{G}_1$ instead. In this way, two pairing operations in signing can be reduced to only one. Our scheme shares the same $\mathcal{SETUP}$ and $\mathcal{KGEN}$ algorithms as that of Cui *et al.*'s scheme.

$\mathcal{SIG}$: Let $L = \{ID_1, ID_2, \cdots, ID_n\}$ be the set of identities of n users. The actual signer, indexed by s (i.e. his/her identity ID_s), carries out the following steps to give an ID-based ring signature on behalf of the group L.

1. Choose $v_i \in_R \mathbb{Z}_q^*$, and compute $V_i = v_i P \ \forall i \in \{1, 2, \cdots, n\} \backslash \{s\}$.
2. Choose $r \in_R \mathbb{Z}_q^*$.
3. Compute $u = g^r \hat{e}(P_{pub}, \sum_{i \neq s} \{v_i P\}) \hat{e}(P, \sum_{i \neq s} \{q_{\mathsf{ID}_i} v_i P\})$ (the logical step) by $u = g^r \hat{e}(P, \sum_{i \neq s} \{v_i (q_{\mathsf{ID}_i} P + P_{pub})\})$ (the concrete step).
4. Compute $h = H_1(m, u, L)$ and $V_s = (h + r) S_{ID_s}$.
5. Output the signature on m as $\sigma = \{u, \bigcup_{i=1}^{n} \{V_i\}\}$.

$\mathcal{VER}$: A verifier can check the validity of a ring signature $\sigma = \{u, \bigcup_{i=1}^{n} \{V_i\}\}$ on the message m signed on behalf of a set of identities L by checking whether $g^{H_1(m,u,L)} \cdot u = \hat{e}(P_{pub}, \sum \{V_i\}) \hat{e}(P, \sum \{q_{\mathsf{ID}_i} V_i\})$ holds.

3.4 Efficiency

Table 1 considers the costly operations which include point addition on $\mathbb{G}_1$ ($\mathbb{G}_1$ Add), point scalar multiplication on $\mathbb{G}_1$ ($\mathbb{G}_1$ Mul), exponentiation on $\mathbb{G}_2$ ($\mathbb{G}_2$ Exp), $MapToPoint$ hash operation in [5] (MapToPoint) and pairing operation (Pairing). The notation x/y means x operations for the signer side and y for the verifier. The most computationally efficient scheme in the literature is [11]. Table 1 shows a summary of comparison. Compared with the proposal in this paper, we save $O(n)$ of $MapToPoint$ operations for both signer and verifier sides, with just one more pairing for the signer side. Our scheme is a quite efficient one. We remark that the most space-efficient among these is Nguyen's scheme [22].

Table 1. Efficiency Comparison of Recent ID-based Ring Signature Schemes

Schemes	$\mathbb{G}_1$ Add	$\mathbb{G}_1$ Mul	$\mathbb{G}_2$ Exp	MapToPoint	Pairing
Nguyen 05 [22]	$2n+8/n+6$	$2n+13/n+9$	6/10	0/0[1]	2/6[2]
Chow *et al.* 05 [11]	$2n-2/2n-1$	$n+1/n$	0/0	n/n	0/2
Au *et al.* 06 [2][3]	$n+4/1$	$2n+2/0$	0/0	$O(n\ell)$[4]	$0/n+3$
This paper	$2n-3/2n-1$	$2n-1/n$	1/1	0/0	1/2

3.5 Linkability

Linkable ring signature [21] can been seen as offering a "lower" level of anonymity. The linkability can let anyone to determine if two ring signatures are signed using the same private key. To add event-oriented linkability (i.e. signatures issued by the same signer for the sake of the same event can be linked, while those issued by the same signer for different events remain unlinkable), we can do so in a similar way as in [8].

$\mathcal{LSIG}$: For brevity, we only list the steps in additional to those in $\mathcal{SIG}$ algorithm, which implement a proof of knowledge to show the linkability tag t's validity.

1. Generate u according to $\mathcal{SIG}$ algorithm.
2. Compute $t = \hat{e}(S_{\mathsf{ID}_s}, H)$, where H is a $\mathbb{G}_1$ element representing the event associated by this ring signature, possibly derived by $MapToPoint(event\_info)$.
3. Randomly generates $r_1 \in \mathbb{Z}_q^*$, compute $T = S_{\mathsf{ID}_s} + r_1 P$.
4. Randomly generates $r_2 \in \mathbb{Z}_q^*$, compute $a = \hat{e}(P, H)^{r_2}$.
5. Instead of $h = H_1(m, u, L)$, compute $h = H_1(m, u, L, a, t, T)$.
6. Use h in the rest of the steps in $\mathcal{SIG}$ algorithm.
7. Include $(t, T, h, z_1 = r_1 + cq_{\mathsf{ID}}, z_2 = r_2 + hr_1)$ in the signature.

[1] Effort preparing the accumulator of the public keys is included in $\mathbb{G}_1$ Add and Mul.
[2] We assume constant values are pre-computed and included in system parameters.
[3] For fair comparison, the one secure against adaptive chosen message attack is chosen.
[4] $O(\ell)$ of $\mathbb{G}_1$ Add are needed to "hash" the identity into the public key, where ℓ denotes an identity's bit length. The same operation is needed for "hashing" the message.

$\mathcal{LVER}$: We have more inputs in the hash function H_1, specifically, a, L, and T. But a is not included in the signature. The following steps reconstruct it from (t, T, h, z_1, z_2) and check whether h purported in signature is legitimate.

1. Compute $\tilde{a} = \hat{e}(P, H)^{-z_2}(\hat{e}(T, H)/t)^h$.
2. Check whether $h = H_1(m, u, L, \tilde{a}, t, T)$.

$\mathcal{LINK}$: For a given event and a given signer, the linkability tag t is uniquely determined. To check whether two valid ring signatures are signed by the same signer for the same event, one just checks if the corresponding values of t match.

3.6 Security Analysis

We will establish the existential unforgeability of the above construction by proving its interactive version is a HVZK protocol in Section 5. For linkability, we are implementing a subset of the proof of knowledge protocol in [8], its security is thus guaranteed. In addition, it is easy to add escrowed linkability (such that only a designated one can link the signatures), or even distributed escrowed linkability, as in [8].

For anonymity, v_i where $i \neq s$ are uniformly distributed for sure. For v_s, it is blinded by the random factor r introduced. It is true that the random factor is included in the computation of u. We consider below an attempt to test whether ID_x is the actual signer, by first getting the $g^{r'}$ element, and testing whether the r' will make $V_x = (h + r')S_{\mathsf{ID}_x}$ hold with the help of bilinearity, i.e. checking whether $\hat{e}(V_x, q_{\mathsf{ID}_x}P + P_{pub}) = g^{h+r'}$.

Consider a signature that is signed by ID_s, we have

$$g^{r'} = u/\hat{e}(P_{pub}, \sum_{i \neq x} \{v_i P\})\hat{e}(P, \sum_{i \neq x} \{q_{\mathsf{ID}_i} v_i P\})$$

$$g^{r'} = g^r \cdot \hat{e}(P_{pub}, v_x P)\hat{e}(P, q_{\mathsf{ID}_x} v_x P)/\hat{e}(P_{pub}, v_s P)\hat{e}(P, q_{\mathsf{ID}_s} v_s P)$$

$$r' = r + (s + q_{\mathsf{ID}_x})v_x - (s + q_{\mathsf{ID}_s})v_s$$

The second last equation is what one can compute from the signature, while the last equation shows the implicit value of r' obtained.

The second step of the testing is to check whether $V_x = (h + r')S_{ID_x}$ with the help of bilinearity. Note that u in a valid signature is defined by $u = g^{-h}\hat{e}(P_{pub}, \sum\{V_i\})\hat{e}(P, \sum\{q_{\mathsf{ID}_i} V_i\})$.

$$\begin{aligned}
g^{(h+r')} &= g^{r'}\hat{e}(P_{pub}, \sum\{V_i\})\hat{e}(P, \sum\{q_{\mathsf{ID}_i} V_i\})/u \\
&= g^{r+(s+q_{\mathsf{ID}_x})v_x-(s+q_{\mathsf{ID}_s})v_s}\hat{e}(P_{pub}, \sum\{V_i\})\hat{e}(P, \sum\{q_{\mathsf{ID}_i} V_i\})/u \\
&= g^{(s+q_{\mathsf{ID}_x})v_x-(s+q_{\mathsf{ID}_s})v_s}\hat{e}(P_{pub}, V_s)\hat{e}(P, q_{\mathsf{ID}_s} V_s) \\
&= g^{(s+q_{\mathsf{ID}_x})v_x-(s+q_{\mathsf{ID}_s})v_s}\hat{e}(P_{pub} + q_{\mathsf{ID}_s} V_s, v_s P) \\
&= g^{(s+q_{\mathsf{ID}_x})v_x} \\
&= \hat{e}(v_x P, (s + q_{\mathsf{ID}_x})P) = \hat{e}(V_x, q_{\mathsf{ID}_x} P + P_{pub})
\end{aligned}$$

Thus, even ID_x is not the real signer, the above relationship still holds. Indeed, it is the relationship that must exist in any valid ring signature in the distribution. Hence our scheme achieves unconditional anonymity. Of course, our extension with linkability can only achieve computational anonymity instead, assuming the q- decisional Strong Diffie-Hellman problem is hard [8].

4 Short Strong Designated Verifier Signature

We use the idea of 2-persons group to issue a signature that is indistinguishable from signature on the same message that the designated verifier can compute. To make the signature short, we re-use the randomness in the ring signature.

4.1 Proposed Construction

$\mathcal{DSIG}$: Suppose ID_s is the signer and ID_v is the designated verifier, let $L = \{ID_s, ID_v\}$. The signer carries out the following steps to give an ID-based strong designated verifier signature designated for ID_v.

1. Choose $v_2 \in_R \mathbb{Z}_q^*$, and compute $V_2 = v_2(q_{\mathsf{ID}_v}P + P_{pub})$.
2. Choose $r \in_R \mathbb{Z}_q^*$ and compute $u = g^r\hat{e}(V_2, q_{\mathsf{ID}_v}P + P_{pub})$.
3. Compute $\kappa = g^{v_2}$.
4. Compute $h = H_1(m, u, L, \kappa)$ and $V_1 = (h + r)S_{ID_s}$.
5. Output the signature on m as $\sigma = \{u, V_1, V_2\}$.

$\mathcal{DVER}$: A verifier checks the validity of the signature $\sigma = (u, V_1, V_2)$ as follows.

1. Recover $\kappa' = \hat{e}(V_2, S_{\mathsf{ID}_v})$ and compute $h = H_1(m, u, L, \kappa')$.
2. Check whether $g^h \cdot u = \hat{e}(P_{pub}, V_1 + V_2)\hat{e}(P, q_{\mathsf{ID}_s}V_1 + q_{\mathsf{ID}_v}V_2)$ holds.

4.2 Security Analysis

Consistency follows from $\hat{e}(V_2, S_{\mathsf{ID}_v}) = \hat{e}(v_2(q_{\mathsf{ID}_v}P + P_{pub}), (s + q_{\mathsf{ID}_v})^{-1}P) = g^{v_2}$.

By the ambiguity of the underlying ring signature, the verifier can produce a valid signature that is indistinguishable from the one produced by the real signer. The only additional term here is κ. One may think that the designated verifier ID_v may claim the signature is signed by ID_s by surrendering his private key (or just using a proof of knowledge protocol) to show $\hat{e}(v_2, S_{\mathsf{ID}_v}) = \kappa$. However, ID_v can simulate the same thing even he is the one who created the signature. Specifically, suppose ID_v gives a signature σ with $v_2 = (h + r)S_{\mathsf{ID}_v}$, he can produces κ by $\hat{e}(v_2, S_{\mathsf{ID}_v})$ as well, and use this value of κ to produce such ring signature. To conclude, there exists an efficient algorithm for the designated verifier to simulate a signature that looks like a signature signed by the signer.

The signer privacy is guaranteed by the chosen-plaintext security of ID-based encryption scheme in [6]. To support the verification oracle, one may think that the underlying encryption scheme must be CCA2 secure, since we should recover

the value of κ by decryption query. However, the use of random oracle, and the implicit decisional Diffie-Hellman oracle provided by the bilinear pairing, help us to get rid of it. There is no need for us to know the value of κ, since all the signature and the hash value are stored in the table simulating the random oracle. One can verify the signature by checking whether there is some input tuple (containing components constitutes the signature) and the corresponding output tuple (the hash value) satisfying the equation to be checked in the verification. A recent id-based strong multi-designated verifiers signature scheme by Chow [7] also used this technique.

5 Ad Hoc Anonymous Identification

Now we describe our proposed ad hoc anonymous identification scheme, which can be seen as the interactive version of our ring signature scheme.

5.1 Proposed Protocol

Let $L = \{ID_1, ID_2, \cdots, ID_n\}$ be the set of identities of n users, where the prover is ID_s. Below shows a typical run of our proposed ID-based ad hoc anonymous identification protocol, between the prover algorithm $\mathcal{P}$ and the verifier $\mathcal{V}$.

1. $\mathcal{P}$ performs the following in the commitment phase.
   (a) Choose $v_i \in_R \mathbb{Z}_q^*$, and compute $V_i = v_i P \ \forall i \in \{1, 2, \cdots, n\} \backslash \{s\}$.
   (b) Choose $r \in_R \mathbb{Z}_q^*$.
   (c) Compute $u = g^r \hat{e}(P, \sum_{i \neq s} \{v_i(q_{\mathsf{ID}_i} P + P_{pub})\})$.
   (d) Send (u, L) to $\mathcal{V}$.
2. $\mathcal{V}$ chooses a challenge $h \in \mathbb{Z}_q^*$ and sends h to $\mathcal{P}$.
3. $\mathcal{P}$ performs the following in the response phase.
   (a) Compute $V_s = (h + r) S_{ID_s}$.
   (b) Send $\bigcup_{i=1}^{n} \{V_i\}$ to $\mathcal{V}$.
4. $\mathcal{V}$ verifies by checking whether $g^h \cdot u = \hat{e}(P_{pub}, \sum \{V_i\}) \hat{e}(P, \sum \{q_{\mathsf{ID}_i} V_i\})$ holds.

5.2 Security Analysis

We are now going to show our protocol is a HVZK protocol. Correctness follows easily from that of the ring signature scheme. For zero knowledge property, a valid transcript can be easily simulated by first choosing random $\bigcup_{i=1}^{n} \{V_i\}$ with a random h, and compute $u = \hat{e}(P_{pub}, \sum \{V_i\}) \hat{e}(P, \sum \{q_{\mathsf{ID}_i} V_i\}) / g^h$.

Regarding the special soundness, assume there exists an honest verifier adversary $\mathcal{A}$ that breaks our protocol in a k-prover setting, we use it to solve k-CAA problem. For a problem instance $(h_1, \cdots, h_k, P, Q = xP \frac{1}{h_1 + x} P, \cdots, \frac{1}{h_k + x} P) \in {\mathbb{Z}_q}^k \times {\mathbb{G}_1}^{k+2}$, we set $P_{pub} = Q$. The H_0 hash of the identities are mapped to $(h_1, \cdots, h_k)$ in simulating the random oracle. Setting in this way let us to get all user's private keys by just returning $\frac{1}{h_i + x} P$, without knowing the value of x.

Eventually, $\mathcal{A}$ successfully impersonates an user in a ring of user L^* where none of them are compromised by key extraction query. Suppose the protocol transcript is $(u, L^*, h, \bigcup_{i=1}^{n}\{(V_i)\})$, by the standard rewinding argument [3], we get another valid transcript $(u, L^*, h', \bigcup_{i=1}^{n}\{(V_i')\})$ where $h \neq h'$ with non-negligible probability. We will have $V_i = V_i'$ for $i \in \{1, \cdots, n\}\backslash\{i^*\}$ and $V_{i^*} \neq V_{i^*}'$. Since both of them are valid, we have the following equations hold.

$$g^h \cdot u = \hat{e}(P_{pub}, V_{i^*})\hat{e}(P, q_{\mathsf{ID}_{i^*}} V_{i^*})\hat{e}(P_{pub}, \sum_{i \neq i^*}\{V_i\})\hat{e}(P, \sum_{i \neq i^*}\{q_{\mathsf{ID}_i} V_i\})$$

$$g^{h'} \cdot u = \hat{e}(P_{pub}, V'_{i^*})\hat{e}(P, q_{\mathsf{ID}_{i^*}} V'_{i^*})\hat{e}(P_{pub}, \sum_{i \neq i^*}\{V_i\})\hat{e}(P, \sum_{i \neq i^*}\{q_{\mathsf{ID}_i} V_i\})$$

Dividing them gives $g^{h-h'} = \hat{e}(q_{\mathsf{ID}_{i^*}} P + P_{pub}, V_{i^*} - V'_{i^*})$, we thus get $(q_{\mathsf{ID}_{i^*}}, (V_{i^*} - V'_{i^*})^{1/(h-h')})$ as a solution of k-CAA problem.

5.3 Extension Against Concurrent Man-In-The-Middle Attack

We adopt the multi-trapdoor commitment scheme introduced by Gennaro [16], and an on-line/off-line variant of Boneh-Boyen signature [4] proposed by [19], to transform our basic ad-hoc identification protocol into one that is secure against concurrent man-in-the-middle attack, assuming strong Diffie-Hellman assumption [16] and k-CAA assumption.

We need strong one-time signature and multi-trapdoor commitment scheme as building blocks, Multi-trapdoor commitment scheme is integrated into the description below. Strong one-time signature scheme is used as a black box, details of it can be found in the Appendix.

$\mathcal{SETUP}$: The KGC randomly chooses $x \in_R \mathbb{Z}_q^*$ and $s \in_R \mathbb{Z}_q^*$, keeps (x, s) as the master secret key and computes the corresponding public key $P_{pub} = xP$ and $\tilde{P}_{pub} = -sP$. Let Q be another generator of $\mathbb{G}_1$, a random element $\mu \in_R \mathbb{Z}_q^*$ is randomly chosen and $Q_{pub} = \mu Q$ will be used as a common reference string. We also need three collision-resistance hash functions: $H_0 : \{0,1\}^* \to \mathbb{Z}_q^*$ $H_1 : \{0,1\}^* \to \mathbb{Z}_q^*$. and $H_2 : \{0,1\}^* \to \mathbb{Z}_q^*$. The system parameters are:

$\{\mathbb{G}_1, \mathbb{G}_2, q, \hat{e}(\cdot,\cdot), H_0(\cdot), H_1(\cdot), H_2(\cdot), g = \hat{e}(P,P), P, P_{pub}, \tilde{P}_{pub}, Q, Q_{pub}\}$.

$\mathcal{KGEN}$: The user with identity $\mathsf{ID} \in \{0,1\}^*$ submits ID to KGC. KGC sets q_{ID} to be $H_0(\mathsf{ID}) \in \mathbb{Z}_q^*$. For private key, firstly $\beta \in \mathbb{Z}_q^*$ is randomly chosen, then compute $x_{\mathsf{ID}} = -\beta\tilde{P}_{pub}$ and $y_{\mathsf{ID}} = \beta + sq_{\mathsf{ID}}$ and S_{ID} by $S_{\mathsf{ID}} = \frac{1}{x+x_{\mathsf{ID}}}P$. The user's private key is set as $S_{\mathsf{ID}} = (q_{\mathsf{ID}}, x_{\mathsf{ID}}, y_{\mathsf{ID}})$. Finally, KGC sends this private key S_{ID} to the user via a secure channel.

$\mathcal{P}$ and $\mathcal{V}$: Let $L = \{ID_1, ID_2, \cdots, ID_n\}$ be the set of identities of n users, where the prover is ID_s. Below gives a typical run of the extended protocol, between the prover algorithm $\mathcal{P}$ and the verifier algorithm $\mathcal{V}$.

1. $\mathcal{P}$ performs the following in the commitment phase.
   (a) Choose $v_i \in_R \mathbb{Z}_q^*$, and compute $V_i = v_i P$ $\forall i \in \{1, 2, \cdots, n\}\backslash\{s\}$.

(b) Compute $(vk, sk) \leftarrow \mathcal{OGEN}(1^k)$, where $\mathcal{OGEN}$ is the key generation algorithm
of strong one-time signature scheme.
(c) Compute $e = H_2(vk)$, where e is a specific public key of the multi-trapdoor commitment scheme.
(d) Choose $r \in_R \mathbb{Z}_q^*$.
(e) Compute $u = g^r \hat{e}(P, \sum_{i \neq s} \{v_i(q_{\mathsf{ID}_i} P + P_{pub})\})$.
(f) Set $H_e = eQ_1 + Q'_1$.
(g) Choose $\gamma \in_R \mathbb{Z}_q^*$ and computes the commitment $com = H'(u)P + \gamma H_e$.
(h) Send $(y_{\mathsf{ID}}, com, vk)$ to $\mathcal{V}$.

2. $\mathcal{V}$ chooses a challenge $h \in \mathbb{Z}_q^*$ and sends h to $\mathcal{P}$.
3. $\mathcal{P}$ performs the following in the response phase.
   (a) Compute $V_s = (h + r)S_{ID_s}$.
   (b) Compute $sig = \mathcal{OSIG}_{sk}(\mathsf{ID}_s, P_{pub}, P'_{pub}, com, h, u, \gamma, \bigcup_{i=1}^{n}\{V_i\})$.
   (c) Send $(u, \gamma, \bigcup_{i=1}^{n}\{V_i\}, sig)$ to $\mathcal{V}$.
4. $\mathcal{V}$ verifies as follows.
   (a) Compute $x_{\mathsf{ID}_i} = \tilde{P}_{pub}^{y_{\mathsf{ID}_i}} + q_{\mathsf{ID}_i} P'_{pub}$, $\forall i \in \{1, \cdots, n\}$
   (b) Compute $e = H_2(vk)$ and $H_e = eQ_1 + Q'_1$.
   (c) Continue if $com = H'(u)P + \gamma H_e$, abort otherwise.
   (d) Continue if $\mathcal{OVER}_{vk}(\mathsf{ID}_s, P_{pub}, P'_{pub}, com, h, u, \gamma, \bigcup_{i=1}^{n}\{V_i\})$ return $\top$.
   (e) Return $\top$ if $g^h \cdot u = \hat{e}(P_{pub}, \sum\{V_i\})\hat{e}(P, \sum\{x_{\mathsf{ID}_i} V_i\})$ holds.

5.4 Discussion

We note that the use of the on-line/off-line ID-based private key generation procedure in [19] makes our protocol loss one of the benefits of using ID-based system – spontaneous group formation. To include user ID_i in the ring, one needs to get y_{ID_i}, which mean user ID_i must have requested for the user's private key in the ID-based cryptosystem. On the other hand, we note that ID-based cryptosystem is often deployed within a medium-size organization, in contrast with a global system with a huge number of users. Maintaining a publicly available list of y_{ID_i} is possible in common scenarios. We conjecture the security of the above protocol can be followed from the security of the Gennaro's compiling technique [16]; however, a more careful formal investigation should be made due to the subtlety that may arise from the difference of non-anonymous single-user identification settings and anonymous identification settings.

6 Conclusion

We successfully turn a weakness in cryptographic scheme as a useful feature in another context. A new efficient identity-based linkable ring signature scheme is resulted. We show two applications of our scheme – a short strong designated verifier signature scheme and an ad-hoc anonymous identification scheme.

References

1. Au, M.H., Chow, S.S.M., Susilo, W., Tsang, P.P.: Short Linkable Ring Signatures Revisited. In: Atzeni, A.S., Lioy, A. (eds.) EuroPKI 2006. LNCS, vol. 4043, pp. 101–115. Springer, Heidelberg (2006)
2. Au, M.H., Liu, J.K., Yuen, T.H., Wong, D.S.: ID-Based Ring Signature Scheme Secure in the Standard Model. In: Yoshiura, H., Sakurai, K., Rannenberg, K., Murayama, Y., Kawamura, S. (eds.) IWSEC 2006. LNCS, vol. 4266, pp. 1–16. Springer, Heidelberg (2006)
3. Bellare, M., Palacio, A.: GQ and Schnorr Identification Schemes: Proofs of Security against Impersonation under Active and Concurrent Attacks. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 162–177. Springer, Heidelberg (2002)
4. Boneh, D., Boyen, X.: Short Signatures Without Random Oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
5. Boneh, D., Franklin, M.K.: Identity-Based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
6. Chen, L., Cheng, Z.: Security Proof of Sakai-Kasahara's Identity-Based Encryption Scheme. In: Smart, N.P. (ed.) Cryptography and Coding. LNCS, vol. 3796, pp. 442–459. Springer, Heidelberg (2005)
7. Chow, S.S.M.: Identity-Based Strong Multi-Designated Verifiers Signatures. In: Atzeni, A.S., Lioy, A. (eds.) EuroPKI 2006. LNCS, vol. 4043, pp. 257–259. Springer, Heidelberg (2006)
8. Chow, S.S.M., Susilo, W., Yuen, T.H.: Escrowed Linkability of Ring Signatures and Its Applications. In: Nguyen, P.Q. (ed.) VIETCRYPT 2006. LNCS, vol. 4341, pp. 175–192. Springer, Heidelberg (2006)
9. Chow, S.S.M., Lui, R.W.C., Hui, L.C.K., Yiu, S.M.: Identity Based Ring Signature: Why, How and What Next. In: Chadwick, D., Zhao, G. (eds.) EuroPKI 2005. LNCS, vol. 3545, pp. 144–161. Springer, Heidelberg (2005)
10. Chow, S.S.M., Susilo, W.: Generic Construction of (Identity-based) Perfect Concurrent Signatures. In: Qing, S., Mao, W., Lopez, J., Wang, G. (eds.) ICICS 2005. LNCS, vol. 3783, pp. 194–206. Springer, Heidelberg (2005), Corrected version available at `http://eprint.iacr.org/2006/361`
11. Chow, S.S.M., Yiu, S.M., Hui, L.C.K.: Efficient Identity Based Ring Signature. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 499–512. Springer, Heidelberg (2005)
12. Cui, S., Duan, P., Chan, C.W., Cheng, X.: An Efficient Identity-based Signature Scheme and Its Applications. International Journal of Network Security 5(1), 89–98 (2007)
13. Dodis, Y., Kiayias, A., Nicolosi, A., Shoup, V.: Anonymous Identification in Ad Hoc Groups. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 609–626. Springer, Heidelberg (2004)
14. Fiat, A., Shamir, A.: How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)
15. Galindo, D., Herranz, J., Kiltz, E.: On the Generic Construction of Identity-Based Signatures with Additional Properties. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 178–193. Springer, Heidelberg (2006)

16. Gennaro, R.: Multi-trapdoor Commitments and Their Applications to Proofs of Knowledge Secure Under Concurrent Man-in-the-Middle Attacks. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 220–236. Springer, Heidelberg (2004), Full version at `http://eprint.iacr.org/2003/214`
17. Huang, X., Susilo, W., Mu, Y., Zhang, F.: Short (Identity-Based) Strong Designated Verifier Signature Schemes. In: Chen, K., Deng, R., Lai, X., Zhou, J. (eds.) ISPEC 2006. LNCS, vol. 3903, pp. 214–225. Springer, Heidelberg (2006)
18. Jakobsson, M., Sako, K., Impagliazzo, R.: Designated Verifier Proofs and Their Applications. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 143–154. Springer, Heidelberg (1996)
19. Kurosawa, K., Heng, S.-H.: The Power of Identification Schemes. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T.G. (eds.) PKC 2006. LNCS, vol. 3958, pp. 364–377. Springer, Heidelberg (2006)
20. Laguillaumie, F., Vergnaud, D.: Designated Verifier Signatures: Anonymity and Efficient Construction from Any Bilinear Map. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 105–119. Springer, Heidelberg (2005)
21. Liu, J.K., Wei, V.K., Wong, D.S.: Linkable Spontaneous Anonymous Group Signature for Ad Hoc Groups. In: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP 2004. LNCS, vol. 3108, pp. 325–335. Springer, Heidelberg (2004)
22. Nguyen, L.: Accumulators from Bilinear Pairings and Applications. In: Menezes, A.J. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 275–292. Springer, Heidelberg (2005)
23. Paterson, K.G., Schuldt, J.C.N.: Efficient Identity-Based Signatures Secure in the Standard Model. In: Batten, L.M., Safavi-Naini, R. (eds.) ACISP 2006. LNCS, vol. 4058, pp. 207–222. Springer, Heidelberg (2006)
24. Susilo, W., Zhang, F., Mu, Y.: Identity-Based Strong Designated Verifier Signature Schemes. In: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP 2004. LNCS, vol. 3108, pp. 325–335. Springer, Heidelberg (2004)
25. Tsang, P.P., Wei, V.K.: Short Linkable Ring Signatures for E-Voting, E-Cash and Attestation. In: Deng, R.H., Bao, F., Pang, H., Zhou, J. (eds.) ISPEC 2005. LNCS, vol. 3439, pp. 48–60. Springer, Heidelberg (2005)
26. Zhang, F., Safavi-Naini, R., Susilo, W.: An Efficient Signature Scheme from Bilinear Pairings and Its Applications. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 277–290. Springer, Heidelberg (2004)

A Definitions of Security

A.1 Identity-Based Ring Signature Scheme

Definition 2. *An ID-based ring signature scheme is said to satisfy the property of existential unforgeability against adaptive chosen-message-and-identity attacks if no adversary has a non-negligible advantage in the game below.*

Setup: The challenger $\mathcal{C}$ takes a security parameter k and runs $\mathcal{SETUP}$ to generate common public parameters *params* and the master secret key s.

Attack: After obtained *params* from $\mathcal{C}$, the adversary $\mathcal{A}$ can perform a polynomially bounded number of queries described below in an adaptive manner

– Hash functions queries: $\mathcal{A}$ can ask for the values of the hash functions (e.g. $H_0(\cdot)$ and $H_1(\cdot)$ in our proposed scheme) for any input.

- $\mathcal{KGEN}$: $\mathcal{A}$ chooses an identity ID. $\mathcal{C}$ computes $\mathcal{KGEN}(\mathsf{ID}) = S_{\mathsf{ID}}$ and sends the result to $\mathcal{A}$.
- $\mathcal{SIG}$: $\mathcal{A}$ chooses a group of n users' identities $\bigcup\{\mathsf{ID}_i\}$ where $1 \leq i \leq n$, and any message m. $\mathcal{C}$ outputs an ID-based ring signature σ.

Forgery: The adversary $\mathcal{A}$ outputs an ID-based ring signature σ and a group of n users' identities $\bigcup\{\mathsf{ID}_i\}$ where $1 \leq i \leq n$. The only restriction is that $(m, \bigcup\{\mathsf{ID}_i\})$ does not appear in the set of previous $\mathcal{SIG}$ queries and each of the secret keys in $\bigcup\{S_{\mathsf{ID}_i}\}$ is never returned by any $\mathcal{KGEN}$ query. i.e. no private keys in $\bigcup\{S_{\mathsf{ID}_i}\}$ is known. It wins the game if $\mathcal{VER}(\sigma, \bigcup\{\mathsf{ID}_i\})$ is equal to $\top$. The advantage of $\mathcal{A}$ is defined as the probability that it wins.

Definition 3. *An ID-based ring signature scheme is said to have the unconditional signer ambiguity if for any group of n users' identities $\bigcup\{\mathsf{ID}_i\}$ where $1 \leq i \leq n$, any message m and any signature σ, where $\sigma = \mathcal{SIG}(m, \bigcup\{\mathsf{ID}_i\})$; any verifier $\mathcal{A}$ even with unbounded computing resources, cannot identify the actual signer with probability better than $\frac{1}{n}$ ($\frac{1}{n-1}$ if $\mathcal{A}$ is in the signers group).*

A.2 Identity-Based Strong Designated Verifier Signature Scheme

The existential unforgeability is defined like a normal ID-based signature scheme, with an addition verification oracle as verification now requires a private key.

Strong designated verifier property means that there exists a PPT algorithm, which takes the verifier's secret key S_{ID_v}, produces an identically distributed signature that is indistinguishable from the one produced by the $\mathcal{SIG}$ algorithm.

A.3 Identity-Based Ad Hoc Anonymous Identification Scheme

The anonymity can be defined similarly as that in ring signatures. In addition to completeness, we want special soundness (or witness extractability) and HVZK. Special soundness means there exists a probabilistic polynomial-time (PPT) algorithm $\mathcal{E}$ outputting the secret key of the user, given two valid transcripts authenticating that user, using the same commitment in responds with different challenges. HVZK means there exists an PPT algorithm $\mathcal{S}$ such that for all secret keys, it can output a valid transcript with the same distribution as that produced by $\mathcal{P}$ on input of the same secret key and an honest $\mathcal{V}$.

The definition below is extended from the single-prover 3-pass protocol, in public key based and non-anonymous settings [3] to the n-prover ℓ-pass protocol, in ID-based anonymous settings. Here give a formal definition of security against (unrestricted) cheating verifier attacks.

(*Definition of the Adversary.*) An adversary $\mathcal{A} = (\hat{\mathcal{V}}, \hat{\mathcal{P}})$ is a pair of probabilistic algorithms, denoting the cheating verifier and cheating prover, respectively. We consider a game having two phases, the training and impersonation phase.

(*Initialization.*)In the start of training phase, the setup algorithm $\mathcal{SETUP}$ first executed to output the system parameters *parmas*. It is assumed that all

algorithm below will take *params* as one of the input parameters. Then the key generation algorithm $\mathcal{KGEN}$ is executed for n times to give $(Q_{\mathsf{ID}_i}, S_{\mathsf{ID}_i}), i \in \{1, \cdots, n\}$. $\hat{\mathcal{V}}$ is given $\cup_{i\in\{1,\cdots,n\}}\{Q_{\mathsf{ID}_i}\}$,

(*Training Phase.*) $\hat{\mathcal{V}}$ interacts concurrently with different clones of each prover P_i, all of these have independent random tapes and being initialized with $(Q_{\mathsf{ID}_i}, S_{\mathsf{ID}_i})$.

In a more formal way, P_i is a function that takes an incoming message and current state, and returns an outgoing message and updated state. $\hat{\mathcal{V}}$'s message to P_i is in the form of (M, j), where j denotes the clone j of P_i.

M can be a null string ϵ for initiating clone of P_i to start the identification process. In this case, a fresh random tape $R_{i,j}$ is chosen, the initial state $St_{i,j,0}$ of clone j of P_i is set to $(Q_{\mathsf{ID}_i}, S_{\mathsf{ID}_i}, R_{i,j})$, then $(Mout_1, St_{i,j,1}) \leftarrow P(\epsilon, St_{i,j,0})$ is executed and $Mout_1$ is returned to $\hat{\mathcal{V}}$. The updated state $St_{i,j,1}$ is saved by $P_{i,j}$.

Cheating verifier $\hat{\mathcal{V}}$ sends the subsequent messages of the protocol (can be more than one, so our definition makes sense in x-pass identification protocol where x is larger than 3) by issuing a request of the form (Min_ℓ, j), where Min_ℓ is the ℓ-th message (counting the null message ϵ as the "0-th" message) sent by $\hat{\mathcal{V}}$ to the clone $P_{i,j}$. $P_{i,j}$ then computes the output in the form of $(Mout_{\ell+1}, St_{i,j,\ell+1})$ by taking input of $(Min_\ell, St_{i,j,\ell})$.

(*Adversary Compromises Provers.*) $\hat{\mathcal{V}}$ can also issue a special request of the form $(\texttt{compromise}, i)$, to get the secret key corresponding to S_{ID_i}.

(*Concurrency of Attack.*) By saving the state $St_{i,j,\ell}$, prover clone will respond according to the sequence specified by the protocol. These requests of $\hat{\mathcal{V}}$ can be arbitrarily interleaved otherwise, e.g. $\hat{\mathcal{V}}$ can firstly send $(Mout_1, j)$ to $P_{i,j}$, then (ϵ, j') to $P_{i',j'}$, followed by $(Mout_1, j')$ to $P_{i,j'}$.

(*Impersonation Phase.*) Eventually, $\hat{\mathcal{V}}$ outputs some state information $St_{trained}$ and declares the end of the training phase. The impersonation phase then starts, and the cheating prover $\hat{\mathcal{P}}$ is initialized with $St_{trained}$. $\hat{\mathcal{P}}$ outputs a list of user L^* to indicate the choice to impersonate some prover P_{i^*} in this list L, verifier V is initialized with such list L and yet another independent random tape. $\hat{\mathcal{P}}$ and V then interact. We say that adversary $\mathcal{A}$ wins if it has not compromised any prover in the training phase and V accepts in this interaction.

(*Definition of Advantage.*) The advantage $\mathrm{Adv}_{imp-ca}(k)$ of $\mathcal{A}$ in this game is the probability that $\mathcal{A}$ wins, taken over the coins of $\mathcal{K}$, the coins of $\hat{\mathcal{V}}$, the coins of the prover clones and the coins of V. We say that an identification scheme for n-prover is secure if $\mathrm{Adv}_{imp-ca}(k)$ is negligible in k for all adversary $\mathcal{A}$.

B Strong One-Time Signature

Let $\mathcal{S} = (\mathcal{OGEN}, \mathcal{OSIG}, \mathcal{OVER})$ be a public key signature scheme consists of the key generation algorithm $\mathcal{OGEN}$, signing algorithm $\mathcal{OSIG}$ and verification algorithm $\mathcal{OVER}$. $\mathcal{OGEN}$ takes as an input security parameter 1^k and outputs a signing/verification key pair (sk, vk). $\mathcal{OSIG}$ takes sk and a message m as input,

outputs a signature σ. $\mathcal{OVER}$ is a deterministic algorithm taking vk, a message m and a signature σ, outputs $\top$ or $\bot$ depending whether the signature is valid. $\mathcal{S}$ should be correct such that $\mathcal{O}VER_{vk}(\mathcal{S}ig_{sk}(m)) = \top$ for all message m and for all (sk, vk) generated by $\mathcal{O}GEN$. For security, we assume $\mathcal{S}$ is strongly unforgeable (cannot create a new valid signature even for previously-signed messages) under adaptive chosen-message attacks. We refer *one-time signature schemes* as a class of signature schemes with a slightly modified security model that an adversary can only request a signature on a single message.

OpenHSM: An Open Key Life Cycle Protocol for Public Key Infrastructure's Hardware Security Modules*

Jean Everson Martina[1,**], Tulio Cicero Salvaro de Souza[2], and Ricardo Felipe Custodio[2]

[1] University of Cambridge
Computer Laboratory
William Gates Building
15 JJ Thomson Avenue
Cambridge – CB3 0FD – United Kingdom
[2] Laboratório de Segurança em Computação (LabSEC)
Universidade Federal de Santa Catarina (UFSC)
Caixa Postal 476 – 88040-900 – Florianópolis – SC – Brasil
Jean.Martina@cl.cam.ac.uk, salvaro@inf.ufsc.br, custodio@inf.ufsc.br

Abstract. The private keys used in a PKI are its most important asset. Protect these keys from unauthorised use or disclosure is essential to secure a PKI. Relying parties need assurances that the private key used to sign their certificates is controlled and managed following pre-defined statement policy. Hardware Security Modules (HSM) offer physical and logical protection and should be considered for any PKI deployment. The software that manages keys inside an HSM should control all life cycle of a private key. Normally this kind of equipment implements a embedded key management protocol and this protocols are not available to public scrutiny due to industrial interests. Other important issue is that HSMs are targeted in their development to the Bank industry and not to PKI, making some important PKI issues, like, strict key usage control and a secure auditing trail, play a secondary role. This paper presents an open protocol to securely manage private keys inside HSMs. The protocol is described, analysed and discussed.

Keywords: Key management protocol, Hardware Security Modules.

1 Introduction

Key management includes key establishment, rules and protocols for generating keys, and the subsequent handling of those keys. Securely manage cryptographic keys is one of the most important and resource consuming efforts to guarantee the security on public key cryptosystems. It means we must have a rigid control on the life cycle of those keys and this is not a trivial task. Moreover, we can

* Work supported and founded by Rede Nacional de Pesquisa/Brazil.
** Supported by CAPES Foundation/Brazil on grant #4226-05-4.

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 220–235, 2007.

assume that a public cryptosystem can be considered as secure as the keys are secured. Taken this as a premise we should guarantee that a key is strictly secure during all events on its life cycle. A way to achieve this is by designing systems to securely create, manage, copy, and destroy private keys maintaining an audit record of all uses during the key life.

Hardware security modules are specific hardware designed to protect key against any kind of logical and physical tampering or extraction of cryptographic material from its environment. The HSMs are normally hardware that passed by certification procedure. The most widely known are FIPS 140-2 [1], a certification developed by USA's Department of Commerce, and Common Criteria [2], developed by a consortium having in mind the creation of protection profiles for such equipment. Normally these equipments implement their own key management protocols, which due to industrial concerns are not made publicly available for scrutiny, making us reasoning about their true correctness. Another important issue to the actual HSMs is their targeted development to the Bank industry and not to PKI, making some important PKI issues, like, strict key usage control and auditing, play a secondary role in the security context, normally making the HSM just a digital safe where we trow our keys.

Key management life cycle has been studied by many researchers [3,4,5]. Menezes et al. [6] discuss the public key management in a general context, including from user registration and initialization to key revocation.

However, protecting a private key in a CA context was always one of the main concerns in any PKI deployment, and is discussed by Jeff Schiller [7]. He states that protection schemes can be broken into two basic classes: schemes where no human ever has access to the raw private key material and schemes where a human may have access to the raw private key material. In the first, the private key is stored in a hardware device which itself requires a hardware token to operate. He advises that when a key is generated by this kind of devices, special attention should take in account to deploy facilities to recover the key from a failed unit.

Having an open protocol is an important matter when concerning to cryptographic algorithms and to cryptographic protocols. This was stated by Auguste Kerckhoffs[8] in the 19th century and by Claude Shanon[9] in 1948, and our main concern when designing this protocol is the lack of access to the industry owned protocols due to their intent to protect their copyrighted material. This makes us always suspicious when using a so sensitive equipment like a HSM to control keys for a Certification Authority in a PKI environment.

This work presents a cryptographic protocol to manage private keys. Our focus is an open key life cycle protocol for public key infrastructure's Hardware Security Modules which will fit on Schiller's first category. The proposed protocol was embedded in a hardware designed to be a HSM holding all physical tampering countermeasures.

The paper is structured with this introduction section, followed by section 2, where we present all the protocol basic ideas and concepts, as well as the premises we assumed during the protocol development in subsection 2.1. Later on section 3

we present our sub-protocol for initialisation and creation of the administrator group. In section 4 we present our sub-protocol for creation of the operators groups, addressing the problem of no trust between the groups inside the proposed HSM. Then in section 5 we present the sub-protocol for key generation and assignment to an operator group, which is followed by section 6, where we present the sub-protocol responsible to apply the concept of key policies and that will allow the operators group to unwrap a key for usage. Finally in section 7 we address some implementation issues that arose in our work and detail our prototyping environment. In section 8 we present our conclusions and propose future work in the theme.

2 Protocols

To address the problems on key life cycle management, we need to answer some questions on the key during all its life cycle. First, it is important to know who is authorised to create a key. This means that only authorised users can create keys and then delegate the use of the key to other user. In addition, we should guarantee that the key is unique and was generated on a secure and controlled environment, i.e., no one knows or has generated the same key pair before. The system administrator can answer the former questions by using a certified system. The singleness of the key can be achieved by using a true random number generator and the key has always to be stored in the memory of such certified system while not ciphered.

The precise time when the key is generated, used, or destroyed must be logged. The way the key is released to use and when and how many times it was used must be also subject of control and tracing.

To each key can be attributed a policy. The key, for instance, can be released for n uses or for a period of time for an application. Other aspect to consider is the control of how many operational copies there are for a key. As many operational keys exists, much more difficult to manage the life cycle of each copy of the key. Due to the additional difficulties introduced by a high number of operational keys, is advisable to keep as low is possible these number. In some practical situations, like a signature system, is common to keep only one key.

We will be presenting in this paper our proposal to address the problem of private key management in public key infrastructures, specifically in the domain of Hardware Security Modules. Our key ideas in the protocol development were to work under a shared responsibility scheme, were all operations must be done by groups instead of a single person, and they will have one symmetric key Ks which will be used to encrypt the asymmetrical private key Kr material assigned to them. This symmetric key will be shared between the group using a share secret algorithm, like the one proposed by Shamir [10], allowing to reconstruct the secret with a minimum number of parts specified at group creation. This is an important operation in the OpenHSM protocol, on all its uses, because by splitting the ownership of a key through a group, we increase the number

of people necessary to corrupt and to exploit to gain access to the key being shared, increasing in this way the whole security of the system.

Another key feature we present in our protocol is the existence of an internal public key infrastructure, which will have a self-signed certificate issued internally by the HSM that will constitute our trust point. From this certificate, we construct a single certification authority to issue certificates to all people involved in the operations of the proposed protocol. By controlling the access to the private key of this internal CA we can limit administrative operations in our protocol, chaining the administrative task to a successful secret share reconstruction and a subsequent decryption of the private key tied with this certificate.

This paper just take in count the four initial processes of the key management scheme, that are the initialisation and creation of administrator group, the creation of operators group, the application's asymmetric key generation and application's asymmetric key usage. This is done this way due to limits on space in the paper. We also give clues on other important facts to a complete key management scheme, like the necessity of modifying the current groups, change the ownership of an application key, do secure backups to the whole system and enable an audit trail. Also because of space limitations we summarise all the descriptions for principals, message parts/objects and operations in Appendix A.

2.1 Premises

To design the proposed protocol we have established some premises as follow:

- each administrator ADM_I and operator $OPER_I$ hold securely its private keys, respectively Kr_{ADM_I} and Kr_{OPER_I};
- random number generator works perfectly and true randomly as an internal part of the principals generating cryptographic keys;
- NXD is a data storage that should be considered as any other, but its data is flushed on a pre-established basis;
- data storages, like ADS, ODS, CRL, CTL, NXD and KDS, store data as it was sent to them, no additional cryptography is used;
- the secret sharing scheme works perfectly;
- all data is securely deleted by a part that holds it when it knows that it will not be used anymore in the current run of the protocol, and no data can be kept to other runs of the protocol, except the data sent to those principals acting as storage facilities;
- all certificates used in the protocols are checked against their $CRLs$ and their certificate path must be constructed with a certificate contained in CTL as point of trust.

The premises above have the only purpose of delimitating and normally address some implementation related problem that we must assume as solved when considering the protocol.

3 Initialisation and Creation of Administrator Group

We start this process creating a system-wide administrator group by informing two values to feed the shared secret scheme, N and M, where $1 \leq N \leq M$ and they will denote the minimum number of principals in the group to activate it, and the size of the system-wide administrator group. Each individual administrator must know the key pair and the personal information that will compose its certificate that will be issued by the internal PKI in the HSM. Finally, to initiate the protocol run, the HSM can generate in advance its key pair Kr_{HSM}, Ku_{HSM} and a symmetric key Ks_{ADM}, all this initial knowledge is described in Table 1.

Table 1. Principals Initial Knowledge

Principal	**Initial Knowledge**
ADM_I	$N, M, Kr_{ADM_I}, Ku_{ADM_I}, ADM_{ID_I}$ [1]
HSM	$Kr_{HSM}, Ku_{HSM}, Ks_{HSM}$ [1]

3.1 Messages Exchange

The sub-protocol we propose to handle the initialisations, create the basic environment to the key life cycle management, and create the administrator group is as follows:

1. $ADM \longrightarrow HSM : N, M$
2. $HSM \longrightarrow CTL : \{|Ku_{HSM}|\}_{Kr_{HSM}}$
3. $ADM_I \longrightarrow HSM : Ku_{ADM_I}, ADM_{ID_I}$
4. $HSM \longrightarrow ADM_I : \{|Ku_{ADM_I}|\}_{Kr_{HSM}}, \{|Ku_{HSM}|\}_{Kr_{HSM}}$
5. $HSM \longrightarrow HSM : Ks_{ADM_0}||...||Ks_{ADM_M}$
6. $HSM \longrightarrow ADS : \{Ks_{ADM_{0\ldots M}}\}_{Ku_{ADM_{0\ldots M}}}, \{|Ku_{ADM_{0\ldots M}}|\}_{Kr_{HSM}}, \{Kr_{HSM}\}_{Ks_{ADM}}$

In the step 1, an administrator from the proposed set informs the HSM his willing to initialise the process by sending N and M, that are respectively, the minimum amount of administrators to reconstruct the shared secret used to protect their session key and the size of the administrator group. In the transition between step 1 and step 2, the HSM generates a key pair, named Kr_{HSM} and Ku_{HSM}, and will generate a self-signed digital certificate in X509v3 format[11], that will mark our point of trust. To establish the trust in this certificate, in the step 2, the HSM will store this certificate in the CTL, that will be checked to guarantee the validity of every principals certificate present in the subsequent protocols runs.

After this initial trust establishment, every ADM_I from the administrators groups will interact with the HSM, by generating its own key pair Kr_{ADM_I},

[1] Not necessarily an initial knowledge. This knowledge can be generated during the protocol run.

Ku_{ADM_I} and sending Ku_{ADM_I} and ADM_{ID_I} to the HSM as is shown in step 3, where ADM_{ID_I} is the identification needed to issue the ADM_I user certificate. By receiving Ku_{ADM_I} and ADM_{ID_I} the HSM, in step 4 will issue a certificate $\{|Ku_{ADM_I}|\}_{Kr_{HSM}}$ with the information provided, and will send this certificate, together with its own self-signed certificate $\{|Ku_{HSM}|\}_{Kr_{HSM}}$ to the administrator ADM_I that will store them properly.

After interacting with all M administrators informed in step 1, the HSM will run step 5, where it will generate randomly a session key Ks_{ADM}, and will split it using the secret sharing scheme explained in the previous section 2. After having Ks_{ADM} shared, the HSM will encrypt every $\{Ks_{ADM_I}\}$ with the public key Ku_{ADM_I} extracted from the ADM_I's certificate just generated, and will store all M encrypted parts together with $\{Kr_{HSM}\}_{Ks_{ADM}}$ in the Administrators Data Storage (ADS), as show in step 6.

3.2 Key Objectives of the Sub-protocol

As this sub-protocol is meant to initialise the HSM, establish trust points, and to create the administrators, we shall accomplish the following security objectives:

- never disclose Kr_{HSM};
- never disclose Ks_{ADM};
- never disclose Ks_{ADM_I};
- never disclose Kr_{ADM_I};

The requirement of non disclosure of Kr_{HSM} exist because this private key is used to establish all trust inside the HSM by signing all the certificates belonging to administrators, and self-signing the HSM certificate. Ks_{ADM} should never be disclosed because it is used to protect Kr_{HSM} on its storage form. Any Ks_{ADM_I} should never be disclosed because by having N or more parts, independently of order, can lead to the reconstruction of Ks_{ADM}, what will make Kr_{HSM} accessible. According to what was stated on previous sections, Kr_{ADM_I} should never be disclosed during the protocol run, because this can compromise Ks_{ADM_I} and subsequently all the rest of the security objectives of the sub-protocol.

4 Creation of Operators Group

This operation will create the operators groups, which will be responsible to use and retain the guard of the HSM managed keys. This process is started by informing the K and L values, where $1 < K < L$, by the administrator group, respectively the minimum amount of operators needed to reconstruct the shared secret, and the size of the operators group. The purpose of these secret sharing operations is the same as explained in the earlier section 2.

Normally each operators group present inside our HSM implementation will represent a Certification Authority private key being protected inside our cryptographical perimeter. This feature was introduced to let a single HSM being

used independently inside the same PKI environment to represent CA in the same or different levels in the hierarchy depending just on the policy established by the PKI management team.

The creation of an operators group slightly differs from administrators group creation, because it will not have a private key directly assigned to it (in this first moment) and because of the existence of Ks_{OPER*}, that is responsible for trace when the group is valid for administrative operations.

This is necessary because we do not want to establish trust between the administrator group and any of the operators group. When Ks_{OPER*} is deleted for some operator group, no administrative task can be done to this group until the recovery of Ks_{OPER*}. This scheme is possible by the use of XOR properties, and will be also important in the future to define traceability of operational copies in the case of backups of the HSM contents.

To initialise a run of this sub-protocol, we should have some initial knowledge by each player, and this is described in Table 2. Each ADM_I should know its key pair Kr_{ADM_I}, Ku_{ADM_I}, as well as the HSM certificate Kr_{ADM_I}, Ku_{ADM_I}. The administrator group should know in advance the values of $OPER_{ID}, K$ and L, that will be the group identification, the threshold of the secret sharing reconstruction and the size of the operators group being created respectively. The HSM should be able to generate during the run the values of Ks_{OPER}, Ks_{OPER*} and at least N nonce N_I to securely authenticate the administrator group.

Table 2. Principals Initial Knowledge

Principal	**Initial Knowledge**
ADM_I	$OPER_{ID}, K, L, Kr_{ADM_I}, Ku_{ADM_I}, \{\|Ku_{HSM}\|\}_{Kr_{HSM}}$
HSM	$Ks_{OPER}, Ks_{OPER*}, N_I$[4]
$OPER_I$	$Kr_{OPER_I}, Ku_{OPER_I}, OPER_{ID_I}$[4]
CTL	$\{\|Ku_{HSM}\|\}_{Kr_{HSM}}$
ADS	$\{Ks_{ADM_{0\ldots M}}\}_{Ku_{ADM_{0\ldots M}}}, \{Kr_{HSM}\}_{Ks_{ADM}}, \{\|Ku_{ADM_{0\ldots M}}\|\}_{Kr_{HSM}}$

Each operator $OPER_I$ should know in advance its own key pair Kr_{OPER_I}, Ku_{OPER_I} and its personal data $OPER_{ID_I}$ that will be used to issue its certificates. The principals CTL and ADS should be able to provide the data necessary to correctly authenticate the administrators group.

4.1 Messages Exchange

The sub-protocol we propose to handle the creation of operators groups is the following:

1. $ADM \longrightarrow HSM : K, L, OPER_{ID}$
2. $ADS \longrightarrow HSM : \{Ks_{ADM_{0\ldots M}}\}_{Ku_{ADM_{0\ldots M}}}, \{Kr_{HSM}\}_{Ks_{ADM}}, \{\|Ku_{ADM_{0\ldots M}}\|\}_{Kr_{HSM}}$
3. $HSM \longrightarrow ADM_I : \{\{Ks_{ADM_I}\}_{Ku_{ADM_I}}, N_I\}_{Ku_{ADM_I}}$

4. $ADM_I \longrightarrow HSM : \{Ks_{ADM_I}\}_{N_I}$
5. $HSM \longrightarrow HSM : Ks_{ADM_0}||...||Ks_{ADM_N}$
6. $HSM \longrightarrow NXD : Ks_{OPER*}$
7. $CTL \longrightarrow HSM : \{|Ku_{HSM}|\}_{Kr_{HSM}}$
8. $HSM \longrightarrow ODS : \{Ks_{OPER} \oplus Ks_{OPER*}\}_{Ku_{HSM}}$
9. $HSM \longrightarrow HSM : Ks_{OPER_0}||...||Ks_{OPER_L}$
10. $OPER_I \longrightarrow HSM : Ku_{OPER_I}, OPER_{ID_I}$
11. $HSM \longrightarrow OPER_I : \{|Ku_{OPER_I}|\}_{Kr_{HSM}}, \{|Ku_{HSM}|\}_{Kr_{HSM}}$
12. $HSM \longrightarrow ODS : \{Ks_{OPER_{0...L}}\}_{Ku_{OPER_{0...L}}}, \{|Ku_{OPER_{0...L}}|\}_{Kr_{HSM}}$

This process is started with the administrator group informing the HSM an operator group identification $OPER_{ID}$ and the values of K and L in the step 1. K and L values are respectively, the minimum amount of operators to reconstruct the shared secret used to protect their keys and the size of the operators group, and $OPER_{ID}$ is a unique identification for the operators group being created. As we consider this is an administrative operation, we must have administrative authorisation. To start this process, in step 2 the ADS sends to the HSM the values $\{Ks_{ADM_{0...M}}\}_{Ku_{ADM_{0...M}}}$ and $\{Kr_{HSM}\}_{Ks_{ADM}}$, that is all Ks_{ADM_I} ciphered with the Ku_{ADM_I}, plus Kr_{HSM} ciphered with Ks_{ADM}. Following the step 3, the HSM will send to each ADM_I in the administrator group, its ciphered part $\{Ks_{ADM_I}\}_{Ku_{ADM_I}}$ to deciphering, plus $\{N_I\}_{Ku_{ADM_I}}$, that is a freshly generated nonce ciphered with the ADM_I public key extracted from the certificates already sent from ADS. This is done due to two reasons.

First we want to avoid replay attacks in the messages sent by the ADM back to the HSM. Second, we need a shared value to cipher Ks_{ADM_I} sent in step 4 back from ADM to HSM, because this communication normally flows outside the cryptographic barrier of the HSM, and the collection of N part of Ks_{ADM_I} can lead to the reconstruction of Ks_{ADM}. Note that we double cipher the value of $\{Ks_{ADM_I}\}_{Ku_{ADM_I}}$ to avoid the capabilities of a Dolev-Yao attacker [12] in reconstructing messages to replay. After doing this process with at least N administrators, the HSM is able to reconstruct Ks_{ADM} as shown in step 5 and consequently decipher Kr_{HSM}, accomplishing the administrators authentication and enabling the protocol to continue.

Following the protocol, the HSM will generate two symmetric encryption keys, one Ks_{OPER} that will be used to cipher every key belonging to the group being created, and Ks_{OPER*}, that is a key to enable administrative operations on the operators group and will be the base for the separation of trust between the groups. The key Ks_{OPER*}, is stored in the NXD as shown in step 6 to enable administrative operations on the group for a while. In step 7, the CTL sends to the HSM the value $\{|Ku_{HSM}|\}_{Kr_{HSM}}$ that is the self-signed certificate of the HSM. This is done because we will need to cipher the result of the XOR operation between Ks_{OPER} and Ks_{OPER*} with the public key that is on this certificate. This will tie any subsequent operation on the groups with deciphering Kr_{HSM}, that means an administrative authentication. The value of $\{Ks_{OPER} \oplus Ks_{OPER*}\}_{Ku_{HSM}}$ is stored in the Operators Data Storage ODS in step 8.

After this binding to administrative task when dealing with the operator group, we need to share Ks_{OPER} to the L operators belonging to this operator group. This task is very similar with what we have done when sharing Ks_{ADM} to the ADM group in section 3. In step 9 the HSM splits Ks_{OPER} in L parts, then in step 10, each operator will inform to the HSM is own public key Ku_{OPER_I} and $OPER_{ID_I}$ that is all the identification needed to issue the operators X509v3 certificate $\{|Ku_{OPER_I}|\}_{Kr_{HSM}}$. In step 11, the HSM sends the operators certificate, plus the HSM certificate to each operator $OPER_I$ with the message $\{|Ku_{OPER_I}|\}_{Kr_{HSM}}, \{|Ku_{HSM}|\}_{Kr_{HSM}}$.

Finally, the HSM will use the public keys present in each certificate issued to the L administrator and will cipher all the L values Ks_{OPER_I} parts with it and send them to ODS together will all operators certificates $\{|Ku_{OPER_{0\ldots L}}|\}_{Kr_{HSM}}$, accordingly with step 12.

4.2 Key Objectives of the Sub-protocol

As this sub-protocol is meant to create the operators group, that are the groups responsible to use the keys managed by the HSM, and taking in consideration that there is no complete trust between any groups inside the HSM, we should accomplish the following security objectives:

- never disclose Ks_{OPER};
- never disclose Ks_{ADM};
- never disclose Kr_{HSM};
- never disclose Ks_{ADM_I};
- never disclose Kr_{ADM_I};
- never disclose Kr_{OPER_I};
- never disclose Ks_{OPER_I};
- never disclose $\{Ks_{OPER} \oplus Ks_{OPER*}\}$;

As we have access to Kr_{HSM}, we have the same chain of objectives as stated in section 3, so we must protect Ks_{ADM}, Ks_{ADM_I} and Kr_{ADM_I} from disclosure. The additional security objectives we must consider now are the non disclosure Ks_{OPER} that is the symmetric key that will cipher all the keys belonging to the group being created The non disclosure on Ks_{OPER_I}, because with a K amount of them we can reconstruct Ks_{OPER}, and we shall not disclose any operator private key Kr_{OPER_I}. Finally we also must protect $\{Ks_{OPER} \oplus Ks_{OPER*}\}$, because by knowing that this is a XOR compound value, an attacker can access Ks_{OPER*} from NXD and use it with $\{Ks_{OPER} \oplus Ks_{OPER*}\}$ to obtain Ks_{OPER}, that is an already specified security objective.

5 Application's Asymmetric Key Generation

This sub-protocol explain how to create the HSM managed keys. We consider this an administrative process, however, there should exist the operators group to which this key will be delegate and this administrative operation must be

authorised by the operator group with the explicit existence of Ks_{OPER*} in the NXD.

The administrators are able to recover the operators group secret using Ks_{OPER*} key from NXD and $\{Ks_{OPER} \oplus Ks_{OPER*}\}_{Ku_{HSM}}$ from ODS. When the administrators are authenticated, they recover Ks_{ADM} key. Using it, they are able to load $\{Kr_{HSM}\}_{Ks_{ADM}}$ from ADS and after they can decrypt the XOR operation result. Now, using the reversible propriety of XOR operation, the administrator group can recover the operators key applying $\{Ks_{OPER} \oplus Ks_{OPER*}\} \oplus Ks_{OPER*}$. Becoming the operators key Ks_{OPER} known.

Other particularly in this sub-protocol is the KEY_PARAM. It will specify the key properties, like algorithm and size. For example, it could be a 1024 bits RSA key pair.

The table 3 show a summary about the necessary things to execute this sub-protocol.

Table 3. Principals Initial Knowledge

Principal	**Initial Knowledge**
ADM_I	$OPER_{ID}, KEY\_ID, KEY\_PARAM, Kr_{ADM_I}, Ku_{ADM_I}$, $\{\|Ku_{HSM}\|\}_{Kr_{HSM}}$
HSM	Kr_{APP}, KU_{APP}, N_I[4]
NXD	Ks_{OPER*}
ADS	$\{Ks_{ADM_{0\ldots M}}\}_{Ku_{ADM_{0\ldots M}}}, \{Kr_{HSM}\}_{Ks_{ADM}}$
ODS	$\{Ks_{OPER} \oplus Ks_{OPER*}\}_{Ku_{HSM}}$

The administrator group must know in advance $OPER_{ID}$, KEY_ID and KEY_PARAM, that are the operator group identification to which the keys will be delegated, and identification for the keys being generated, and the parameters for the key generation process respectively. Each ADM_I Must know its own key pair, and the HSM must be able to generate the key pair Kr_{APP}, KU_{APP} and at least N nonce N_I to securely authenticate the administrators.

The NXD must have stored Ks_{OPER*} for the group to which the keys will be delegated, showing with this that the operator group was aware of this administrative operation. ADS must have all authentication data for the administrator group stored in it, and ODS must have all authentication data for the operator group informed in $OPER_{ID}$.

5.1 Messages Exchange

The sub-protocol which creates a new HSM managed key is described below:

1. $ADM \longrightarrow HSM : KEY\_NAME, KEY\_PARAM, OPER_{ID}$
2. $ADS \longrightarrow HSM : \{Ks_{ADM_{0\ldots M}}\}_{Ku_{ADM_{0\ldots M}}}, \{Kr_{HSM}\}_{Ks_{ADM}}$
3. $HSM \longrightarrow ADM_I : \{\{Ks_{ADM_I}\}_{Ku_{ADM_I}}, N_I\}_{Ku_{ADM_I}}$
4. $ADM_I \longrightarrow HSM : \{Ks_{ADM_I}\}_{N_I}$
5. $HSM \longrightarrow HSM : Ks_{ADM_0}||...||Ks_{ADM_N}$

6. $ODS \longrightarrow HSM : \{Ks_{OPER} \oplus Ks_{OPER*}\}_{Ku_{HSM}}$
7. $NXD \longrightarrow HSM : Ks_{OPER*}$
8. $HSM \longrightarrow HSM : Ks_{OPER}$[2]
9. $HSM \longrightarrow OUT : Ku_{APP}$
10. $HSM \longrightarrow KDS : \{Kr_{APP}\}_{Ks_{OPER}}, Ku_{APP}, KEY\_NAME, OPER_{ID}$

This process is started with the administrator group informing the HSM the identifier of the new key KEY_NAME, the key generation parameters KEY_PARAM and the operator group identifier $OPER_{ID}$ (this group must be created before), as shown in the step 1. We set this as an administrative operation, so the administrator group must be validated. The validation is made in the steps 2, 3 and 4, where the ADS load to the HSM the data necessary to the authentication process. Thus, the HSM sends to each ADM_I its ciphered part $\{Ks_{ADM_I}\}_{Ku_{ADM_I}}$, plus a ciphered nonce $\{N_I\}_{Ku_{ADM_I}}$, and each ADM_I send back to the HSM your part ciphered with the nonce the HSM send in the previous step. Finally the administrator secret can be recovered in the 5 when it has more than N deciphered parts of Ks_{ADM}. This validation has been shown in the previous section 4.

Following the protocol, in step 6, the ODS will send to the HSM the result of the XOR operation done when creating the operators group, ciphered with Ku_{HSM}. In the next step, 7, the NXD will send the HSM the authorisation value Ks_{OPER*}, that will be used by the HSM to XOR with $\{Ks_{OPER} \oplus Ks_{OPER*}\}$ and obtain Ks_{OPER}, as shown in step 8.

Between the steps 8 and 9, the application key pair subject to the HSM protection is generated, then the public key is exported in the step 9. In step 10, all necessary information regarding the key is stored into the Key Data Storage KDS including the operators group identifier $OPER_{ID}$, key name KEY_NAME, public key and encrypted private key $\{Kr_{APP}\}_{Ks_{OPER}}, Ku_{APP}$. The operators group will use this information to allow the recovery of the key.

5.2 Key Objectives of the Protocol

The main objectives in this protocol are to securely generate a key pair and delegate its usage and management to an operator group, so the main security objectives are:

- never disclose Ks_{OPER};
- never disclose Ks_{ADM};
- never disclose Kr_{HSM};
- never disclose Kr_{APP};
- never disclose Kr_{ADM_I};
- never disclose Ks_{ADM_I};
- never disclose $\{Ks_{OPER} \oplus Ks_{OPER*}\}$;

[2] This is recovered by applying the XOR properties in the $\{Ks_{OPER} \oplus Ks_{OPER*}\}$.

As we consider this an administrative operation, we must have administrative authorisation to do so. By doing this way we have the same security requirements as other previous sub-protocols, that are the non disclosure of Kr_{HSM} our trust root, Ks_{ADM} its ciphering key, and Kr_{ADM_I} and Ks_{ADM_I} that are the secrets that protect Ks_{ADM} in its shared form.

Additionally, as we deal with the operator group, by doing and administrative operation in its name, we must have access to $\{Ks_{OPER} \oplus Ks_{OPER*}\}$, that when XORed with Ks_{OPER*} gives us Ks_{OPER}. They are security objectives, because they will protect Kr_{APP}, our main security goal, and mean of existence. Ks_{OPER} is important because it will cipher Kr_{APP}, and $\{Ks_{OPER} \oplus Ks_{OPER*}\}$ is important because with it we can derive easily Ks_{OPER}.

6 Application's Asymmetric Key Usage

This sub-protocol will cover the managed keys usage. This usage does not represent specifically to sign or encrypt something with the key. It will load the key and apply the specified policy on its. The operations of signing or ciphering something with the private key subject to protection is no in the scope of this protocol, and should be treated as HSM API.

The policy, represented by KEY_POL, will specify restrictions on loaded keys. The operators group can set a maximum number of operations using the key for signatures or encryptions and/or set a range of time for the key remain loaded. For example, the key can be loaded for three uses and five minutes. The first policy reach will unload the key automatically and this sub-protocol must be executed again to load the key for a next usage.

In the table 4, we can see the items which must be initial knowledge. Basically, the system must be initialised, the administrators, at least one operators group must have been created and one key must also have been created.

Table 4. Principals Initial Knowledge

Principal	**Initial Knowledge**
$OPER_I$	$KEY\_ID, KEY\_POL, Kr_{OPER_I}, Ku_{OPER_I}, \{\|Ku_{HSM}\|\}_{Kr_{HSM}}$
ODS	$\{Ks_{OPER_{0\ldots M}}\}_{Ku_{OPER_{0\ldots M}}}, \{\|Ku_{OPER_{0\ldots L}}\|\}_{Kr_{HSM}}$
KDS	$\{Kr_{APP}\}_{Ks_{OPER}}, Ku_{APP}, OPER_{ID}$
HSM	N_I

6.1 Messages Exchange

1. $OPER \longrightarrow HSM : KEY\_ID, KEY\_POL$
2. $KDS \longrightarrow HSM : \{Kr_{APP}\}_{Ks_{OPER}}, Ku_{APP}, OPER_{ID}$
3. $ODS \longrightarrow HSM : \{Ks_{OPER_{0\ldots L}}\}_{Ku_{OPER_{0\ldots L}}}, \{\|Ku_{OPER_{0\ldots L}}\|\}_{Kr_{HSM}}$
4. $HSM \longrightarrow OPER_I : \{\{Ks_{OPER_I}\}_{Ku_{OPER_I}}, N_I\}_{Ku_{OPER_I}}$
5. $OPER_I \longrightarrow HSM : \{Ks_{OPER_I}\}_{N_I}$
6. $HSM \longrightarrow HSM : Ks_{OPER_0}||...||Ks_{OPER_K}$
7. $HSM \longrightarrow HSM : Kr_{APP}, KEY\_POL$

The process starts, in the step 1, when the operators group send the command to load the key including the identification KEY_ID and a policy KEY_POL for it. This is not an administrative operation, but it is necessary to validate the operators group, as they are the effective owners of the key subject to the HSM protection. The operator group to which the key belong is known because it is loaded from KDS in step 2,together with they ciphered private key $\{Kr_{APP}\}_{Ks_{OPER}}$ and its public part Ku_{APP}. The validation of the operators group will follow the same mechanism followed by the administrator group authentication in other sub-protocols.

First, in step 3 ODS will send all data necessary to authenticate the operator group, that is all ciphered parts $\{Ks_{OPER_{0\ldots L}}\}_{Ku_{OPER_{0\ldots L}}}$, and the operators certificates $\{|Ku_{OPER_{0\ldots L}}|\}_{Kr_{HSM}}$. In step 4, the HSM will send each operator $OPER_I$ it ciphered part $\{Ks_{OPER_I}\}_{Ku_{OPER_I}}$, plus a freshly generated nonce ciphered with its public key $\{N_I\}_{Ku_{OPER_I}}$. Following, each operator $OPER_I$ will send to the HSM its part of the shared secret ciphered with the nonce $\{Ks_{OPER_I}\}_{N_I}$, as described in step 5.

After collecting K parts deciphered by the administrator, the HSM will be allowed to recover Ks_{OPER} as shown in step 6, by the reconstruction of the shared secret. Finally, the policy KEY_POL is applied in the step 7 and the key is loaded, being ready for its usage.

6.2 Key Objectives of the Protocol

The main objectives in this protocol were to securely load a key pair by an operator group following a policy, so the main security objectives are:

- never disclose Ks_{OPER};
- never disclose Ks_{OPER_I};
- never disclose Kr_{APP};
- never disclose Kr_{OPER_I};

As this operation must just have authorisation from operators group, the authorisation must never have disclosure of Ks_{OPER}, the operators group secret and the key that is used to encrypt the managed keys. Kr_{OPER_I} and Ks_{OPER_I} that are the secrets that protect Ks_{OPER} in its shared form.

Additionally, as the main objective of HSM, we must never have disclosure of Kr_{APP}. It must be used just under authorisation and respecting the policy.

7 Implementation Issues

Although we create and manage keys, we do not plan any protocol for key destruction or deletion, because this could be simply accomplished by removing the ciphered parts from KDS, but this is not always that simple and sometimes extremely difficult to achieve. Normally this is covered by the PKI policies, what tend to the destruction of all private key material by physical means, like the

incineration of the HSM itself and everything that could have had contact with these key material.

Other thing that is not covered in the protocol, is the operators group destruction, but this is easily achievable just by destroying X operators privates keys, where $X > N$. This will also render all keys that belonged to the operators groups set being destroyed unusable, as long the NXD part of the group is not available for administrative matters.

7.1 Prototype

The embedded application was developed in C language and involved many technologies, including smart card support, cryptography, data storage, secret sharing and X509v3 certificates. To solve these technology gaps we used known open/free libraries, such: OpenSSL - for cryptographic operations and X509 certificate support, SQLite - a lite database with focus in embedded systems, OpenSC - smart card support, SharedSecret - for sharing secrets and PCSC-LITE - supporting smart card readers and tokens, everything integrated with a FreeBSD system running with embedded customisations we developed.

The hardware was projected to be tamper proof system using a Security Unit(S.U.) to manage all sensors and protection systems, including a Random Number Generation, a separate Clock and a true eraser circuit. The key manager software was integrated using a library to manage the configuration of the S.U. and used a pipe system to receive a problem detection message from the S.U.

8 Conclusions

This work presented a cryptographic protocol to manage private keys in a Certification Authority context, i.e., an application that can be embedded in a Hardware Security Module. It is known the security of a PKI is related to how the keys are generated, used and destroyed. Thus, our protocol was conceived to have a strong control of the keys, a requirement in PKI solutions. The protocol was itself built on an internal PKI, i.e., we have designed a PKI to manage private keys of external Certification Authorities.

To manage the HSM, it was created groups. The administrator group is responsible to create new application keys and the operator groups. The latter group is bound to the private keys of the applications, like a certification authority.

The protocol was coded and embedded in a prototype HSM. The implementation has shown that it is comfortable and secure to manage private keys to Certification Authorities. It also showed that we can use this strict key controls to debug CA software. As future work it is intended to extend the protocol presented by including a backup feature and auditing system. We also propose to do formal analysis on the protocol.

References

1. FIPS: Security requirements for cryptographic modules, FIPS PUB 140-2 (2002)
2. Killmann, W., Leitold, H., Posch, R., Sall é, P.: Protection profile - secure signature-creation device type 3 (July 2001) `http://www.commoncriteriaportal.org /public/files/ppfiles/pp0006b.pdf`
3. Barker, E., Barker, W., Burr, W., Polk, W., Smid, M.: Recommendation for key management part 1: General. Technical Report 800-57, NIST, May 2006, NIST Special Publication (2006)
4. Neumann, P.G.: Crypto key management. Commun. ACM 40(8), 136 (1997)
5. Daemen, J.: Management of secret keys: Dynamic key handling. In: Preneel, B., Rijmen, V. (eds.) State of the Art in Applied Cryptography. LNCS, vol. 1528, pp. 264–276. Springer, Heidelberg (1998)
6. Menezes, A.J., Vanstone, S.A., Oorschot, P.C.V.: Handbook of Applied Cryptography. CRC Press, Boca Raton, FL, USA (1996)
7. Schiller, J.: Protecting a private key in a ca context, A useful discussion of the issues and patterns (2000)
8. Kerckhoffs, A.: La cryptographie militaire. Journal des sciences militaires IX, 5–38 (1883)
9. Shannon, C.E.: A mathematical theory of communication. Bell System Technical Journal 27, 379–423 (1948)
10. Shamir, A.: How to share a secret. Commun. ACM 22(11), 612–613 (1979)
11. X.509, I.T.R.: Information technology - open systems interconnection - the directory: Authentication framework. Technical report, ITU-T (1997)
12. Dolev, D., Yao, A.C.: On the security of public key protocols. IEEE Transactions on Information Theory 29(2), 198–208 (1983)

Appendix A: Conventions

The protocols are subject to conventions in Table 5, Table 6 and Table 7.

Table 5. Description of all Operations Used in the Protocols

Operation	Description
$\{\}$	Data inside brackets is ciphered by subscript key outside brackets.
$\{\|\|\}$	Data inside piped brackets is signed by subscript key outside brackets.
$\|\|$	Data is concatenated or dissociated using a secret sharing scheme.
$\{\oplus\}$	Data inside brackets is logically XOR-ed and the result becomes 1 single object.

Table 6. Description of all Principals of the Protocols

Principal	Description
ADM_I	An administrator of the HSM
HSM	The HSM Crypto-Processor
CTL	Certificate Trust List
ADS	Administrator Data Storage
$OPER_I$	An operator of the HSM
NXD	Non Exportable and Temporary Data Storage
ODS	Operator Data Storage
OUT	External output of HSM.
KDS	Key Data Storage
CRL	Certificate Revocation List.

Table 7. Description of all Objects and Message Parts of the Protocols

Message	Description
M	Size of the administrators group.
N	Minimum amount of administrators to reconstruct a shared secret.
I	Any principal part of a group.
Kr_{HSM}	Private key of the HSM.
Ku_{HSM}	Public key of the HSM.
Kr_{ADM_I}	Private key of one Administrator.
Ku_{ADM_I}	Public key of one Administrator.
ADM_I	Identification for one Administrator.
Ks_{ADM}	Symmetric key used for ciphering Kr_{HSM}.
Ks_{ADM_I}	Shared part of Ks_{ADM} belonging to one Administrator.
N_I	A random value just used once (Nonce).
K	Size of the operators group.
L	Minimum amount of operators to reconstruct a shared secret.
$OPER_{ID}$	Identifier of a operators group.
Kr_{OPER_I}	Private key of one operator.
Ku_{OPER_I}	Public key of one operator.
$OPER_{ID_I}$	Identification for one operator.
Ks_{OPER}	Symmetric key for ciphering private keys owned by operators group.
Ks_{OPER_I}	Shared part of Ks_{OPER} belonging to one Operator.
Ks_{OPER*}	Non exportable and temporary operand that enables administrative operations to an operator group.
KEY_ID	Identifier for the key being generated or being used.
KEY_PARAM	Parameters for the key being generated.
Kr_{APP}	Application protected private key.
Ku_{APP}	Public key related with Kr_{APP}.
KEY_POL	Policies for a key being used.

Two Worlds, One Smart Card: An Integrated Solution for Physical Access and Logical Security Using PKI on a Single Smart Card*

Jaap-Henk Hoepman[1,2] and Geert Kleinhuis[1]

[1] TNO Information and Communication Technology
P.O. Box 1416, 9701 BK Groningen, The Netherlands
jaap-henk.hoepman@tno.nl, geert.kleinhuis@tno.nl
[2] Institute for Computing and Information Sciences
Radboud University Nijmegen
P.O. Box 9010, 6500 GL Nijmegen, The Netherlands
jhh@cs.ru.nl

Abstract. We present a use case of the introduction of a large scale Public Key Infrastructure (PKI) environment in an incumbent telecommunications company in The Netherlands. The main characteristics of the case are the integration of an existing physical access facility with a PKI environment for logical security of the company ICT infrastructure. In fact, both are accessed using a single (smart) company card. The purpose was to implement a high level of security, within the practical constraints at hand, and to reach a level of *reduced sign-on* for company employees. This integration poses numerous challenges. In this article we describe how PKI is actually introduced to support authentication, signing and encryption services for its employees.

18.000 personalised smart cards with PKI were issued, controlling access to over 1500 buildings, fitted with in total more than 6000 smart card readers. The smart cards also controlled access to 14.000 personal workstations both desktops and laptops (each fitted with a contact smart card reader), with access to over a 1000 different applications.

Keywords: PKI, Access control, smart card, reduced sign-on.

1 Introduction

To grant their employees access to office buildings and plants, companies these days issue their employees a (smart) card that is both an identity card as well as an electronic key. Usually, this key can be used without any further authentication to enter the premises. Few companies would require their employees to enter a PIN code as well as presenting their card to open a door, for instance. Such a system for access control to physical objects has been known and in use for quite some time. It grants or denies access to office buildings and sectors within such buildings in a convenient and uniform manner.

* Id: pki-geert.tex,v 1.7 2007/04/16 11:56:59 jhh Exp.

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 236–247, 2007.

However, access control to objects in the digital domain (like computing systems, company applications and information) is usually not handled in the same uniform manner. They often have their own access control mechanism. This is a burden on employees. Consider, for example, the multitude of user names and passwords an office worker may have to enter during the course of a single working week.

This difference can be explained partially by the fact that implementing access control for digital objects is considered more difficult than implementing access control for physical objects. It is also caused by the fact that no single system for uniformly handling authentication and access control is in widespread use today. This is true because Kerberos[NT94], and other methods of single sign-on, largely remain academic exercises, even though (a variant of) Kerberos is part of the Microsoft code base.

This paper describes a use case where all employees of a large telecommunications company in the Netherlands were issued with a *single* smart card to obtain access to both physical and digital objects. Security, authentication and access control in the digital domain is based on a Public Key Infrastructure (PKI) (cf. [AL99, RFC 3280, ES00]).

There were three reasons to use a single smart card for access control in the digital as well as the physical domain.

1. There were high security requirements concerning the general handling of digital information, as well as the authentication of the actor in a workflow.
2. The aim was to arrive at a more user-friendly system of *reduced sign-on*.
3. It was desirable to reduce cost through a simpler and unified access control management organisation.

The latter point could only be achieved through a scalable solution that was usable for a large population of workers with varying skills and technical background. This solution is documented in this paper.

The remainder of this paper is structured as follows. We first discuss the issue of how many keys are needed for physical and logical access control. Section 3 describes the functional architecture. Details on the use case are presented in Section 4. The concrete architecture is given after that. We finish with an example of how an employee is entered into the system (section 6) and conclude with user experience, security issues and conclusions.

2 How Many Keys Do We Need?

A number of international information security organisations have studied the trend that security increasingly crosses the confines of individual objects, towards a more holistic, integrated, approach. They concluded that the convergence of security within (large) enterprises is rapidly emerging and enterprises need to adapt accordingly [Ham05]. In fact, this convergence may cover all the objects within a value chain, and extends through physical as well as informational goods.

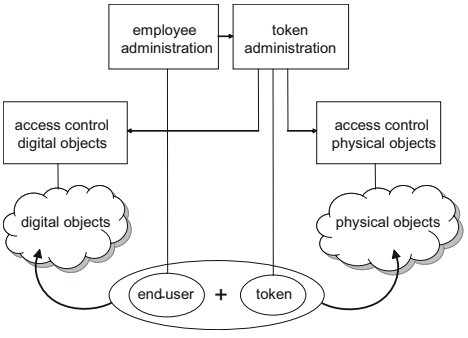


Fig. 1. Our vision on keys

In practise, access control in the physical world is achieved by companies throughout the world using one electronic key. Access to individual doors and entrances is managed by an access control management system, that maintains an access control list (ACL) of all allowed keys for each individual door. Of course, this concept can in principle also be used to manage access to objects in the digital domain. As figure 1 shows, one single token[1] could even suffice to control access to both physical and digital objects. However, there are practical issues that lead us to the conclusion that we need two separate tokens, one for the digital and one for the physical domain (that can, of course, be stored on a single carrier like a smart card).

2.1 Access to Physical Objects

First of all, there already exists, within the company, a huge nationwide up-to-date installed based for physical access to the company premises. A lot of physical readers are installed nationwide, that would need to be replaced in case of technology change. So for costs reasons a change to this installed base should be avoided.

For physical objects, companies may pose, as an extra requirement, that the handling of an access request is handled very quickly. This way, the number of entry doors in a company can be kept low even when many employees enter at the start of the working day. For this reason, entering a PIN code when entering the building is not an option, as it would be prohibitively expensive in terms of time.

Moreover, access to different sectors within a building are usually protected using locked doors that need to be unlocked separately.

2.2 Access to Digital Objects

For digital objects, usually the access rights for all available objects are determined the moment the user is authenticated by the system. That means,

[1] Note that a token refers to a ticket, or access right, not a hardware token. In other words, a token is not the same as a smart card.

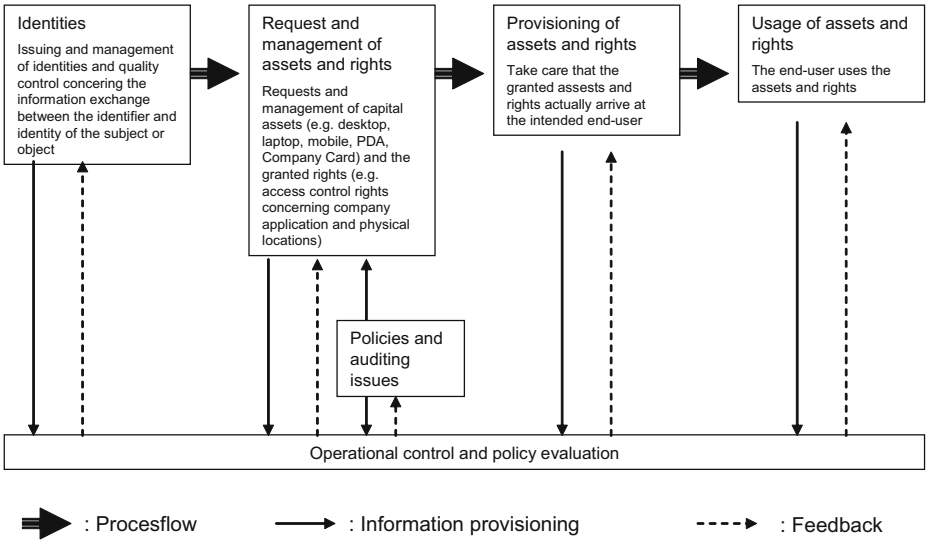


Fig. 2. The four building blocks of the architecture

however, that a more thorough access procedure is in order. If the key grants access to a large collection of objects (which is typically the case) then the applications with the most stringent security requirements determine the minimal requirements for this access procedure. A PKI based solution then appears to be an appropriate choice compared with other forms of authentication (like username/password approaches). Once the decision to base the solution on PKI is made, the smart card containing the key can also be used to store other keys for other applications one wishes to distribute to the employee [GK03].

3 Functional Architecture

At a high level of abstraction we have distinguished four building blocks in our architecture for access management:

- identity management,
- requesting and managing of assets and access rights,
- provisioning (actually delivering) the assets and the access rights, and
- the actual use of the assets and the access rights by the user.

These four building blocks and their interdependence are depicted in figure 2. In the use case, each of the four building blocks is implemented using one or more specific components, as described in section 5.

The four building blocks in figure 2 have been drawn in their logical process order. Each building block provides information to the audit and control layer. Because business more and more need to prove that rules and regulations were

followed within their business processes (think, for example, about the Sarbanes-Oxley legislation), a separate layer addressing policy and auditing issues has been added to the picture. This layer can also support special forms of assigning access rights to business processes (for instance classical function separation, or geographically determined access rights).

Using a common architecture can reduce costs. Becker & Drew [BD05] report on practical experiences with investing in solutions for the building blocks 'requesting and managing assets and access rights' and 'provisioning'. They conclude that in order to obtain an acceptable return-on-investment (ROI), the number of employees using the building blocks for which the investment was made should be larger than 10.000. Applying the same architecture to 5-10 systems simultaneously will considerably increase the ROI.

4 The Use Case

The use case concerns a large incumbent telecommunications company in The Netherlands. In this case, more than 18.000 personalised smart cards with PKI were issued, controlling access to over 1500 buildings, fitted with in total more than 6000 card readers. The smart cards also controlled access to 14.000 personal workstations both desktops and laptops (each fitted with a contact smart card reader), with access to over a 1000 different applications. These numbers made the case a challenging one.

The smart card used actually contains two chips: a contactless Mifare chip[2] to access physical objects (containing the physical access key), and a contact Philips microcontroller with 32K EEPROM, Triple-DES coprocessor and FameX RSA coprocessor for access to digital objects (containing the PKI keys). The latter chip uses the GlobalPlatform[3] Card Specification v2.1.1. Mifare is an industry standard for contactless communication developed by Philips. It is also subsumed by the newer NFC standards[4]. For the sake of completeness, actually the contact microcontroller contains two PKI key pairs. One key pair is used for authentication (which is also used for digital signatures) and one key pair is used for encryption.

The name of the holder and a passport photo are printed on the smart card. For the sake of completeness the smart card was also fitted with a magnetic stripe to remain fully backwards compatible with the installed base of magstripe readers. The smart card is used to authenticate its holder, to grant access to physical buildings and digital applications and information.

Depending on the security level required by the application, authentication is performed through one of the following 3 functions:

1. the smart card with a passport photo that resembles the holder, or
2. the smart card itself (mainly to open physical doors), or

[2] `www.mifare.net.`
[3] `www.globalplatform.org.`
[4] `www.nfc-forum.org.`

3. the smart card together with the pin code unlocking the embedded PKI controller.

The smart card is personalised in a single phase, in which also the PKI key pairs and certificates are being generated. In the use case it was decided that two separate token mechanisms were needed to meet the different requirements regarding authentication, speed and robustness. By integrating these separate mechanisms on a single smart card the total constellation of access control procedures and mechanisms remain manageable both for employees as well as the managers. Once combined, future applications can choose whichever authentication token they wish to use. In the future a PKI based authentication (involving a PIN) could be used to grant access to highly sensitive areas of a building.

Function 1 is used to bind the card to a physical person.

Function 2 is primarily used to grant access to company buildings. Another application that uses function 2 lies on the boundary of the physical and the digital world. It concerns the selection of printers to which documents in a print queue should be printed. Printers only actually print a job in the queue when the owner of the job presents the smart card to the printer. The owner does not select the printer when issuing the print command, but physically when reaching the actual printer and presenting the card.

Function 3 is primarily used to grant access to the personal account in the digital world. The employee presents the smart card to the desktop computer (or the laptop), and is asked for the PIN code.

Moreover, the smart card can also be used to read encrypted mail and digitally sign mails or e-forms (in a legally binding way [1999/93/EC]).

At the end of 2008, most of the company applications can only be accessed through function 3, meaning that employees are less confronted with a plethora of username and password combinations than before, to achieve a form of *reduced sign-on*.

5 Technical Architecture

We will now discuss the building blocks used in the technical architecture, and and how these building blocks relate to the procedures surrounding identity management. In the use case, the building blocks together implement a full-service 'token management' system: all phases in the life cycle of a smart card are supported (such that the end user does not have to worry about smart card production and data processing), such that they can rely on the quality of the authentication offered by the card. Some of these building blocks are offered commercially as a service component (e.g. the token management application)

The existing ICT infrastructure of the use case was taken as point of departure. This guarantees optimal reuse of existing infrastructure, which results in faster implementation and less cost. The most important element to be integrated is PKI. In the global architecture we distinguish the following building blocks (for ease of presentation not all elements that exist in reality are mentioned). See Figure 5 for details.

- Personnel administration (functional block: 'identities')
  - status employee: processes for job-entry (A-0) and exit.
  - core data for smart card and access control (personnel number, name on card, e-identifier (i.e. email address), state, function, manager, organisation code, etc.)
  - self-service interface for employees and managers (function interface X-1)
- Token Management Application (functional block 'request and management')
  - status smart card: processes for issuing and management
  - core smart card date (card id, state, corresponding employee number)
  - flow control for life-cycle smart card
  - self service and signalling functions for employees and managers
  - back office functions and help desk
  - control and audit functions
- Order application for end-user accounts (function-interface X-3; functional block 'request and management')
  - status end-user accounts and processes for delivery and management
- Management application for authorisations on digital objects (function-interface X-4; functional block 'request and management')
- Management application for authorisations on physical objects(functional interface X-2; functional block: 'request and management')
- Passport photo function for passport photo and identity control (functional block 'issuing')
- Production line for smart cards and mailings (functional block 'issuing')
  - processes for production and control
  - scalable issuing (100-10.000 smart cards a week)
  - core data smart cards and mailings
  - direct mail
  - PKI Certificate Authority (functional block 'issuing')
  - PKI local Registration Authority (functional block 'issuing')

As an example of the type of adjustments needed to the existing situation, and as a prime example of the general design philosophy, consider the process of entering identifying information for a new employee. The new employee first needs to be entered into the personnel administration. All items that need to be put at the disposal of the new employee (smart card, end-user account) can only be issued to the end-user after this registration. The order in which these items are issued cannot be guaranteed by a certain order. The smart card needs to contain the end-user account details of the employee. The logical consequence of this fact is that the issuing of email address and account details is shifted from the party normally issuing the end-user accounts to the personnel department. This way the smart card (that contains the PKI certificates and email address) can be produced even before the user owns an end-user account and has a working email address. Only after the employee gets access to his account, his reserved email address is activated. A notice is sent to the personnel administration. As a consequence, the email addresses are no longer under the control of the ICT

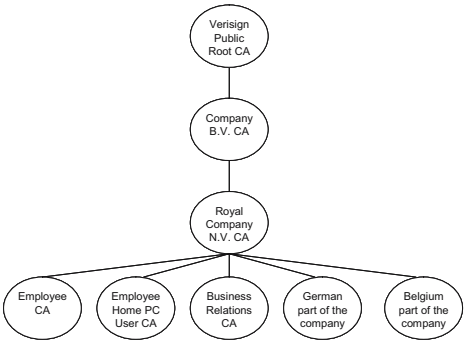


Fig. 3. PKI Hierarchy

environment (those are now controlled by the personnel administration), but the status of the email addresses (whether they are activated or not) still is under the control of the ICT environment.

5.1 End-User Certificates, Certificate Authorities, and PKI Hierarchy

Five different Certificate Authorities (CA's) are available to create end-user certificates. These CA's each cover a different domain and have different policies to issue certificates. The five domains are:

1. Employee CA
2. Employee Home PC User CA
3. Business Relations CA
4. German part of the company
5. Belgium part of the company

At the moment the Employee CA and Business Relation CA are operational. The five CA's are part of a hierarchy. In figure 3 the hierarchy is depicted. The hierarchy is setup this way to possibly create a commercial proposition as well. It is possible to create another 'Company subCA' at the same level as the 'Royal Company N.V. CA'. This can than possibly create its own different subCA's depending on the requirements of that company. Still leaving the 'Company B.V. CA' responsible.

End-user certificates (issued by the Employee CA) and accompanying private keys all reside on a smart card. End-user certificates (issued by the Business Relations CA) and accompanying private keys all reside on a USB stick. No final decision is made concerning the other three CA's concerning the way end-user certificates and accompanying private keys are issued.

5.2 End-User PKI Key Generation and Certificate Creation Process

The process of PKI key generation and certificate creation starts with an input file generated by the Token Management Application (TMA), see figure 4. TMA is responsible for the employee information in the certificate. This information is retrieved from the Company employee information database combined with the information that is provided by the photographers. The smart card prepersonalistion and personalisation is outsourced to a dedicated company with a dedicated smart card production line. This company is responsible for the generation of the signing PKI key pair. The public key of the signing key pair and the employee information with regard to the signing related certificate is than sent to a System Integrator in a secure way. The Key Manager process that resides at the System Integrator generates the encryption PKI key pair and stores the encryption key pair in a secure way. Both the public signing key and the public encryption key together with the employee information ending up in the certificate are used by Verisign[5] (the Employee CA, see also figure 3) to create the final end-user certificates. The certificates together with the private encryption key are sent in a highly secure way to the smart card personalisation system. This information is put on the smart card.

6 How a New Employee Is Entered into the System

The flow 'new employee' can globally be determined from the scheme in Figure 5. The flow starts at A-0, were the new employee 'is born' in the system. As soon as the personnel file contains enough data, the applications in the function block 'request and management' (see Figure 2) can be fed with the relevant data. After that, management actions X-1 up to X-4 can be executed independently from smart card production. These management actions comprise requesting an end-user account and setting access rights to buildings, for example. Smart card production is controlled by step B. The smart card production flow is controlled by the token management application (TMA). In step C-1, the new employee receives a letter instructing him to obtain a passport photo from a passport photographer. The employee is free to choose the photographer from a list of photographers offering the service. In step E the token management application receives the passport photo digitally from the photographer, together with control information regarding the identity paper presented by the employee at the photographer. The control information is necessary for securing this part of the smart card issuing process, to prevent issuing smart cards to the wrong people.

After a quality check the smart card can be produced (step F-1). The card is produced by an external, third, party, see also figure 4. The production process facilitates key recovery for the private decryption key.

[5] `www.verisign.com.`

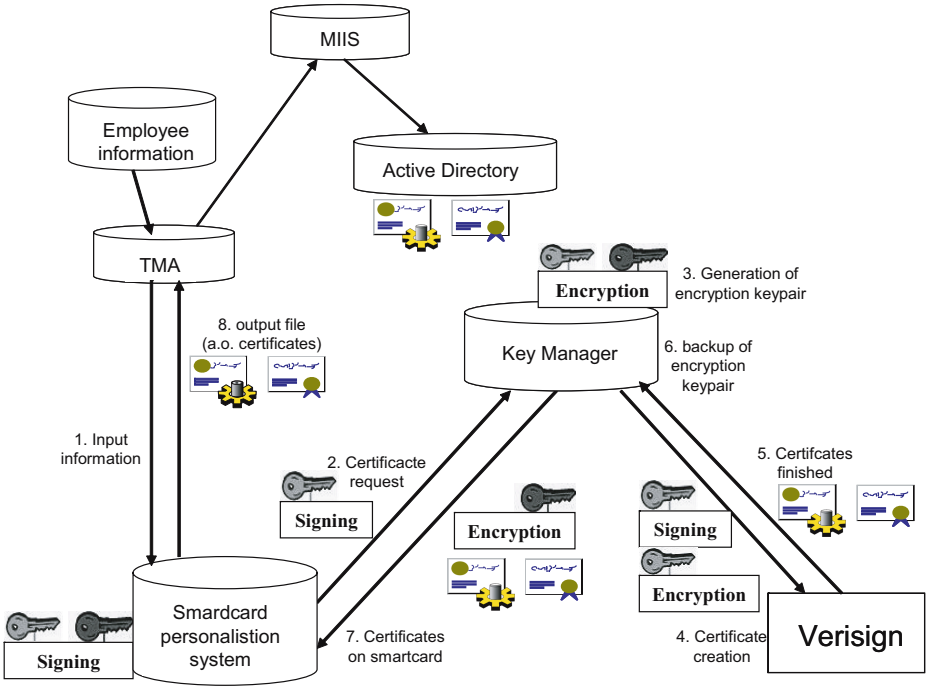


Fig. 4. Card production and key generation

The private authentication key (also signing key) never leaves the smart card (and is also not recoverable). This offers a very high level of trust in the authentication provided through messages signed by this key. Public certificates are generated, signed by the PKI Certification Authority and stored on the smart card (steps G-1 through G-4). From that point onwards both PKI key pairs can be used. The card (and independently the PIN and the PUK (recovery PIN)) are sent to the employee. The token management application receives, through H-2, a signal regarding the state of the smart card request. When the card is produced, applications are notified (J-1, J-2). The employee is also sent a separate activation code. Only after activation (I) the functions for (physical) access are available.

The PKI key recovery function for the private decryption key is triggered whenever an encryption certificate is revoked through the token management application. When that happens, the lost key is provided automatically to the end-user environment of the corresponding user, or his/her manager in case the person is no longer employed by the company.

The large scale roll out of the smart card took place without any significant problems. Because of the close cooperation between all parties involved, startup problems could be resolved immediately.

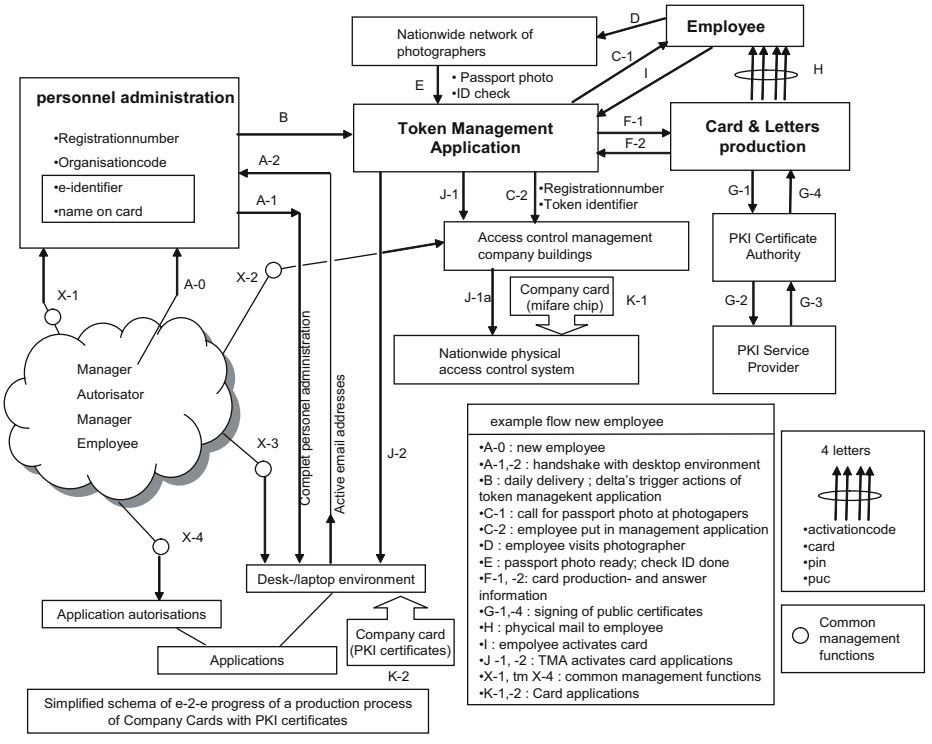


Fig. 5. How a new employee is entered into the system

7 Conclusions

7.1 Security Level

The whole process of issuing smart cards and providing authentication runs at a high security level. Smart card production is performed in a secured facility and data is exchanged through virtual private networks (VPNs). The whole process satisfies the requirements for a Verisign class 3 environment. Because of existing contractual obligations between the telecommunication company and Verisign in the use case Verisign class 2 was chosen.

7.2 User Experiences

User satisfaction concerning the operation of access control to the buildings remained positive after introduction of the new smart card. Regarding access control to the digital objects, results are only known for the test environments. They are also positive. A first point of particular interest is the handling of the smart card by end-users. After the complete changeover, there is no longer a need to change passwords every 6 months. We expect a considerable reduction in calls to the help desk because of this. People that (have to) use the secure

email functionality will notice a change and a slight increase in the complexity of their tasks (related to the management aspects of secure email). We do not have concrete user experience data for this yet.

7.3 Summary

Using available building blocks and by following the process oriented functional architecture, a production line for a smart card supporting several security mechanisms was implemented in a period of 10 months. The combination of physical and digital access on the same smart card offers increased user convenience. With the large scale introduction of PKI an important component to secure digital objects has become available within the company. The smart card was issued to a total of 18.000 people in a matter of four weeks.

References

[1999/93/EC] Community framework for electronic signatures, Directive, 1999/93/EC (1999)

[AL99] Adams, C., Lloyd, S.: Understanding Public-Key Infrastructures. SAMS (1999)

[BD05] Becker, M., Drew, M.: Overcoming the challenges in deploying user provisioning/identity access management backbone. BT Technical Journal 23, 4 (2005)

[ES00] Ellison, C., Schneier, B.: Ten risks of pki: What you're not being told about public key infrastructure. Computer Security Journal 16, 1 (2000)

[GK03] Gelbord, B., Kleinhuis, G.: On the use of pki in a residential gateway environment. In: ICWI 2003, pp. 1125–1128 (2003)

[Ham05] Hamilton, B.A.: Convergence of enterprise security organisations (2005)

[RFC 3280] IETF. RFC 3280, Internet X.509 public key infrasturcture. Tech. rep (2002)

[NT94] Neuman, B., Ts'o, T.: Kerberos: an authentication service for computer networks. IEEE Communications Magazine 32, 9, 33–38 (1994)

On the Robustness of Applications Based on the SSL and TLS Security Protocols

Diana Berbecaru and Antonio Lioy

Politecnico di Torino, Dip. di Automatica e Informatica
Corso Duca degli Abruzzi, No. 24, 10129, Torino (Italy)

Abstract. The SSL and TLS security protocols have been designed and implemented to provide end-to-end data security. This includes data integrity that is the data cannot be modified, replayed or reordered by an attacker without being detected at the receiving endpoint. SSL and TLS however does not provide data delivery integrity, in the sense they do not guarantee that all the sent data will actually arrive at the other side. This is because, for example, SSL/TLS cannot know in advance which is the exact size of the data to be sent over the secured channel. The mosts recent versions (SSLv3 and TLSv1) provide some form of protection against loss of data records by means of sequence numbers and specialized close_notify alert messages to be sent when tearing down the SSL connection. Unfortunately, this is not enough when the last record containing application data together with the closure alert are deleted on purpose, as it happens in the *truncation attacks*. SSLv3/TLSv1 specifications do not indicate what should happen (at the application level) if the close_notify message never arrives at the receiver. Consequently, for applications where it is important to ascertain that the data reached untruncated the other party, it is required to have an additional control at the application level.

In this paper we show (based on practical tests) that some widely-used applications implementing SSLv3 and TLSv1 do not perform further controls on the size of the data to be received, and thus they are vulnerable to truncation attacks. For tests we implemented a specialized MITMSSL tool, used to manipulate the SSL/TLS records exchanged between two communicating parties.

Keywords: security, SSL/TLS, truncation attack, MITM attack.

1 Introduction

The Secure Sockets Layer (SSL) security protocol has been widely implemented and is nowadays the de facto standard(s) for providing secure e-commerce transactions over the Web. SSL was first developed in 1994 by Netscape, when the browser's designers realized that there was no way to guarantee the security of the network through which Web data was transmitted. Consequently, the best way to protect data was to provide encryption and decryption at the connection's endpoints. Since Netscape's designers wanted a unified solution that could

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 248–264, 2007.

be used also with non-HTTP applications, SSL was not incorporated into the browser itself but it was located at a level among the reliable TCP transport layer and the application layer above. In theory, application developers would take advantage of the new layer by replacing all the traditional TCP calls (e.g. *send*, *recv*) with the new SSL calls implemented by SSL libraries (e.g. *SSL_write*, *SSL_read* in OpenSSL library [1]). SSL gained remarkable attention and popularity in short time, and it was further improved in version 2, while version 3 [2] was heavily modified in order to fix some serious security drawbacks of SSLv2.

In the same time, as other commercial vendors, such as Microsoft, started to develop their own security protocols operating on top of the transport layer, the Internet Engineering Task Force was asked (by Netscape and Microsoft) to define a standard for an encryption layer protocol as a compromise to stave off incompatible, vendor-specific solutions. The result was the definition of the Transport Layer Security (TLS) protocol, which took into consideration inputs from multiple vendors for its specification. The first version of TLS, TLS 1.0 [3], is based on SSL version 3 (SSLv3), consequently it is sometimes referred as SSL 3.1. TLS 1.1 [4] instead contains mainly improvements to protect the protocol against the cipher block attacks in CBC mode pointed out by Vaudenay [5]. TLS 1.0 and TLS 1.1 are referred further as TLSv1. TLSv1 and SSLv3 have subtle implementation differences [6], the application developers usually notice only very little differences while the end users would not see any difference at all. Nevertheless, TLSv1 and SSLv3 are not interoperable, the most significant difference being that TLSv1 requires certain encryption algorithms that SSLv3 does not. Most (commercial) products support nowadays SSLv2, SSLv3 and TLS 1.0.

All SSL versions, as well as TLSv1, were subject to various security analyses and studies, aimed mainly to identify the weaknesses of the protocol both for what it concerns its design (theoretical attacks), its implementation (practical attacks) as well as its usage, e.g. attacks related to user's behaviour or to the features of the application based on SSLv3/TLSv1 protocols. Examples of such attacks include: the downgrade and the truncation attacks that are due to weaknesses in the protocol specification (in SSLv2 for example) [7]; the Man In The Middle (MITM) attacks that are mostly due to wrong user's behaviour with respect to server's digital certificate used for authentication; the side channel attacks like timing attacks [8] that exploits the information gained from the physical implementation of the SSL protocol running on a system, rather than the theoretical weaknesses in the protocol itself; Vaudenay's attack (further extended in [9]) exploits instead certain characteristics of the application running on top of SSLv3/TLS 1.0, like for example the fact that the same password is going to be sent several times over the SSLv3/TLS 1.0 protected channel.

In this paper we demonstrate (based on practical tests) that a truncation attack can actually be performed against some widely-used applications that support SSLv3/TLS 1.0, and not only against SSLv2, which has already been known to be vulnerable to such an attack. For this purpose we developed a tool named MITMSSL, whose role is to intercept and manipulate (i.e. delete, modify, copy and replace) the SSLv3/TLS 1.0 records exchanged between the client

and the server. We illustrate the results obtained for testing the robustness of some SSL/TLS-enabled products, including browsers (Firefox, Mozilla, Internet Explorer), software tools (WGet & CUrl), web servers (Apache) and security libraries (OpenSSL).

The paper is organized as follows: Section 2 gives a brief overview both of the phases executed at SSLv3/TLSv1 (further referred also as SSL/TLS) connection establishment and shut down as well as of the fields of protocol records involved in the attack, Section 3 explains our contribution consisting in the description of SSLv3/TLSv1 truncation attack and our MITMSSL tool used to perform it, Section 4 shows the experimental tests performed with MITMSSL tool, and Section 5 provides a short discussion and the future work envisaged in this area.

2 SSL/TLS Protocol Details

2.1 SSL/TLS Connection Phases

Generally speaking, four principal operational phases are executed by one party when establishing and closing an SSL/TLS connection with another communicating party. The phases are illustrated in Fig. 1 and a short description of each phase is given below:

Phase 1: Set up a TCP connection. In this phase, basically a three way handshake is run among the client and the server in order to establish a TCP connection. It is important remind here that SSL/TLS distinguishes clearly between a *connection* and a *session*. In SSL/TLS terms, a connection is a peer-to-peer connection with two network nodes, while a session is an association between a client and a server that defines a set of security parameters such as the encryption algorithms used, the session number etc, and are established during the negotiation of the SSL handshake protocol. Sessions are used to avoid negotiation of new security parameters for each connection. This means that a single SSL/TLS session can actually be shared among multiple connections. However, for simplicity purpose, in Fig. 1 we illustrated a one-to-one correspondence between TCP connections and SSLv3/TLSv1 sessions, which could not be the common case in practice.

Phase 2: SSL/TLS Handshake protocol. Its main role is to establish an SSL/TLS connection. In this phase basically two processes take place. The first one is the key exchange, meaning that the client and the server must securely exchange a shared secret, named *pre-master secret*. The pre-master secret is used next to independently compute one shared key, named *master secret*, which will be used subsequently to derive the keys required in provide integrity and confidentiality of the communication. The second process is the authentication of the communicating parties involved[1]. The authentication is typically done via digital certificates. There exists a full handshake protocol used to initiate a secure SSL/TLS session and a shorter one used to resume a session, which is named abbreviated SSL/TLS handshake.

[1] In practice it is always required for server.

Phase 3: Data transfer. In this phase the application data is exchanged between the client and the server. The SSL/TLS protocol encrypts all application-layer data with an encryption algorithm and a session key negotiated by the handshake protocol. The encryption algorithm used can be either a stream ciphers (e.g. RC4) or a block cipher (e.g. RC2, DES, 3DES) depending on the ciphersuite negotiated in the handshake phase.

Phase 4: Teardown. The main role of this phase is to close a SSL/TLS connection so that the other communicating party is notified that all the application data has been transferred and the connection is going to be teared down. For this purpose messages belonging to the Alert protocol are used. This protocol can be invoked by the application to close a TLS connection, by the handshake protocol to signal an error (e.g an illegal parameter) or by the record protocol to indicate some specific SSL/TLS errors, such as an invalid Message Authentication Code (MAC) value.

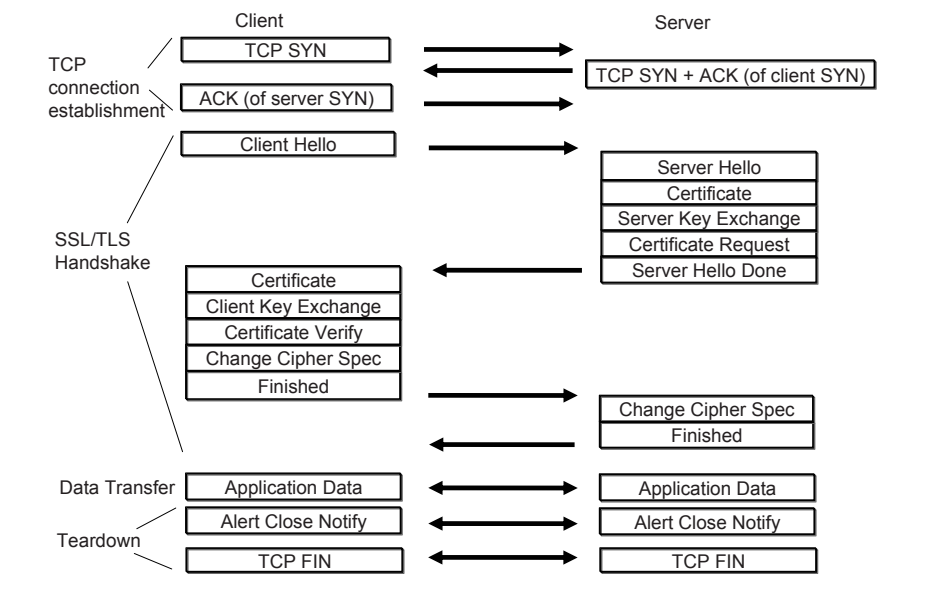


Fig. 1. Phases for establishing and closing an SSLv3 and TLSv1 connection. For simplicity, a one-to-one correspondence between TCP connections and SSL/TLS sessions has been illustrated.

2.2 Overview of SSL/TLS Records

Since our MITMSSL program operates on the SSL/TLS records, we remind first the structure of a SSL/TLS record and explain only the SSL/TLS record's

details that have been actually used or related to the security attack detailed in Section 3, namely the MAC calculation, the sequence numbers and the alert protocol messages.

The SSL/TLS protocol is not just one protocol, but it is rather a set of protocols organized on layers: at a lower level stands the Record layer composed of the Record protocol, while at the upper level there are 4 kinds of protocols: the *Handshake* protocol, the *Alert* protocol, and the *Change Cipher Spec* protocol are used to establish and close an SSL/TLS connection, while the *Application* protocol carries application data messages. The most widely known application protocol used in conjunction with SSL/TLS is HTTP to protect communication channel over the Internet, but in practice SSL/TLS can be used to protect the transmission for any TCP/IP service.

The role of the Record Layer is to provide data security and integrity services to the upper layer protocols. In practice, the Record protocol provides data fragmentation, compression, encryption, and transmission services on the transmitting side, and data reassembly, decompression, decryption and delivery services on the receiving side. It uses the session keys negotiated during the handshake for data encryption and decryption. Thus, data of either of the 4 types passed in from upper layers is encapsulated in SSL/TLS record messages for transmission over the underlying TCP communication protocol. Since each SSL/TLS record can include 16 Kbytes of data or less, the messages passed in from upper layers are fragmented as necessary to meet the size limit. A record begins with a header which includes the version of the protocol, the length of the data in bytes and the type of the message, i.e. one of change cipher spec, alert, handshake or application data. After the header comes the message data section.

MAC calculation and the padding bytes. In all cases excepting the handshake messages prior to the Finished during an initial SSL/TLS connection set up, the data section of a record will include the actual data contents, a digest of those contents, and possibly some padding out to the block size of the encryption algorithm. The digest is a MAC calculated using the MAC key negotiated in the handshake phase and it is appended to the record. When transmitting, the client or the server combines the data fragment, the digest, and the record header and encrypts them with the secret key to produce the completed SSL/TLS record. When receiving, the client or the server decrypts the packet, computes the MAC, and compares it to the received MAC. If the two values do not match, then a bad_error_mac alert message is generated, causing the client or the server to close the connection.

Sequence numbers. SSL and TLS includes a monotonically increasing sequence number in the data that the MAC runs over. In practice, the sequence number are not transmitted but are only included in the MAC. They are used to detect the missing/extra data and to prevent replay or reordering attacks. There is a separate sequence number for records sent in each direction over the SSL connection. The sequence numbers are updated by the sender and the receiver

independently as SSL/TLS records are sent and received. One important clue is to note that alert messages like close_notify also contain a sequence number, in order to detect replay/reordering attacks of the last SSL records containing application data.

Alert messages. The Alert protocol is used by the client and the server to convey session messages associated with the data exchange and functioning of the protocol. Each message in the alert protocol consists of two bytes. The first byte takes a value warning (1) or fatal(2), that determines the severity of the message sent. Sending a message having a fatal status by either party will result in an immediate termination of the SSL session. The second byte of the message contains one of the defined error codes, such as bad_error_mac, which may occur during an SSL communication session.

From the alert messages defined in the SSLv3 and TLS 1.0 specifications, we focus on the close_notify alert message, since this message is handled in MITMSSL tool. This message is used to notify the other party that it will not send any more messages on the current SSL/TLS connection. In our opinion, two issues need to be understood (in the SSL/TLS'specification) before testing the robustness of SSLv3/TLS 1.0 implementations:

1. whether the close_notify is mandatory (to be sent) by either or both the client and the server.
2. what it should happen if the communicating party (client or server) doesn't receive an alert close_notify message from the other party. Should this indicate a possible truncation attack?

On the point 1) above the SSLv3 states that "*..either party may initiate the exchange of closing messages*" but gives no hint on the behaviour if this alert message is not sent. TLS 1.0 states that "*..each party is required to send a close_notify alert before closing the write side of the connection*", while the TLS 1.1 specification states that *"...unless some other fatal alert has been transmitted, each party is required to send a close_notify alert before closing the write side of the connection.."*. Thus, the close_notify messages can be considered obligatory in TLS 1.0 and TLS 1.1 but not in SSLv3.

On the point 2) above the SSLv3 indicates only that the session is not resumable if any connection is terminated without proper close_notify messages, but gives no indication on what to do if the close_notify alert has not arrived in the first place. TLS 1.0 states also that the session is not resumable and additionally it states that when a close_notify arrives, *".. It is required that the other party respond with a close_notify alert of its own and close down the connection immediately, discarding any pending writes"*. Nevertheless, *"..it is not required for the initiator of the close to wait for the responding close_notify alert before closing the read side of the connection"*! In TLS 1.1 the session becomes resumable and it is mentioned that *..The other party MUST respond with a close_notify alert of its own and close down the connection immediately, discarding any pending writes. It is not required for the initiator of the close to wait for the responding close_notify alert before closing the read side of the connection.*

Thus, also here it is not clear what should happen (at the application level) if the SSL/TLS close_notify message never arrives at the receiving party. The tests we've performed indicates that this can result in a truncation attack, which is not detected in some widely used SSL/TLS enabled applications.

3 Truncation Attack

3.1 SSLv3/TLSv1 Truncation Attack

The truncation attack is a security attack that can be applied in phase 4, that is when tearing down an SSL/TLS connection. SSLv2 is subject to the truncation attack (referred subsequently as *SSLv2 truncation attack*), because it allows either side to send a TCP FIN to terminate an SSL connection. In a *SSLv2 truncation attack*, the attacker could make it appear that a message was shorter than it was, by simply forging a TCP FIN. Imagine for example the case of an SSL application that fragments SSL/TLS data records in blocks of 128 bytes; in this case an attacker is able to delete the last part of the following sentence: "Dear Mr. Smith, I decided to take the following action after our discussion hold last Thursday: I will buy 1000 auctions of PRNG", making it appear as "Dear Mr. Smith, I decided to take the following action after our discussion hold last Thursday: I would buy 1000 auctions of PRN". Unless the receiver had some other way of knowing what message length to expect, it would simply believe that it has received a shorter message. To prevent this security problem, SSLv3 introduced a close_notify alert message, whose role is to help systems detect such attacks. If a system received "...buy 1000 auctions of PRN" but did not receive an alert close_notify message, it would recognize that the complete message may not have arrived.

We remind that the close_notify is an SSL message (and therefore secured) but is not part of the data stream itself and so is not seen by the application. No data may be transmitted after the close_notify is sent. As mentioned in the above section as well as in [10], unfortunately not all environments can rely on the close_notify alert messages. Web browsing users, for example, may simply turn off their personal computer after completing a transaction, before that computer has a chance to send a close_notify alert message. More thorough protection requires that applications using SSL security be sensitive to the possibility of premature closures. Web servers that support the HyperText Transfer Protocol (HTTP) for example, include a Content-Length field with each page they send to a client. Clients should verify that the amount of data they receive is consistent with this field's value.

In the following tests, we'll perform basically truncation attacks also for SSLv3 and TLS 1.0. We'll show that even though some SSLv3/TLS 1.0 implementations support and send alert close_notify messages, an attacker will be able to do the same attack as for SSLv2 if he manages to cancel this alert message. To distinguish this attack from the one applying to SSLv2, we call this attack *SSLv3/TLSv1 truncation attack*.

3.2 MITMSSL Tool

A Man In The Middle (MITM) attack is "form of active wiretapping attack in which the attacker intercepts and selectively modifies communicated data in order to masquerade as one or more of the entities involved in a communication association" [11]. Thus, the MITM attacks have mainly two characteristics [12]:

- they represent active attacks.
- they target the associations between the communicating entities, rather than the entities or the communication channels between them.

The MITM attack can be applied at various levels in the TCP/IP stack and for various protocols. For example, the Address Resolution Protocol (ARP) cache poisoning and the TCP connection hijacking are in fact MITM attacks applied to different levels. In a typical setting, the MITM manages to place himself between the communicating entities in a way that he can talk to each of them separately, while each of the communicating entity thinks that he is talking directly to the other one. Thus, neither the client nor the server are aware of the presence of the attacker since it acts like a *forwarder* between them. Cryptography alone could not be sufficient in this attack since the MITM intercepts all the messages and thus he can decrypt and reencrypt all the messages sent back and forth (if he is able to recover the key) or he could simply delete part of the encrypted messages exchanged.

The classical MITM attacks in SSL/TLS communications are due to the fact that the server is not authenticated correctly, for example when the MITM attackers employs tricks like tools for visual spoofing to give the user the impression of being connected to the origin server. Ettercap[2] tool for example generates a self-signed certificate that looks like the original server's certificate. In this particular case however, the application (such as the browsers) would return some warning to the user, e.g. because the CA issuing the server certificate is not configured as trusted in the client's browser.

Our MITMSSL tool does not decrypt/decode SSL traffic nor it can be used for visual spoofing, but instead it modifies the SSL encoded data flow, that is is able to cancel or duplicate entire SSL/TLS Records or even to memorize them in a temporary buffer in order to send them later. For this purpose, the MITMSSL tool is composed of three main modules: the *Networking Module* used for handling the SSL/TLS connections, the *Interface* module used for reading commands for record handling and the *Record analysis and modification* module used for executing the commands on the records. Other characteristics of the tool are:

- the possibility to save two types of log of the entire communication: the first one memorizes all the messages sent by the communicating parties to the MITM, i.e. the original messages; the other one instead is activated when the MITM starts sending data and thus it keeps trace of all SSL/TLS records really sent.

[2] http://ettercap.sourceforge.net/

– the possibility to configure and save in a file the sequence of operations to be done on the SSL/TLS records, that is deletion, duplication and/or registration of records; the file allows thus to run the test operations automatically.

4 Experiments with SSLv3/TLSv1 Truncation Attack

4.1 Testing Environment

The testing environment was basically composed of two machines: a Window XP machine for the client and a machine running Linux RedHat 7.2 for the (web) server. On the server machine we've started the actual SSL-enabled web servers and other server applications such as OpenSSL's s_server, the MITMSSL tool used to manipulate the exchanged SSL traffic and the SSLDump[3] tool used to display the intercepted SSL/TLS records in a textual form. As for the web software, we've used both an Apache Web Server v1.3.33 with mod_ssl v2.8.22 and Apache v1.3.9 with mod_ssl 2.4.10. On the client machine we've run most-commonly used browsers and other widely used applications, such as Wget, Curl and OpenSSL's s_client application. We remind also that applications typically rely on dedicated libraries for SSL/TLS implementation, e.g. Mozilla uses the Network Security Services (NSS) library[4], while Internet Explorer uses Microsoft CryptoAPI. In all tests the clients tried to download from the server an image file of 68 KB in size, shown in Fig. 2.

In the tests we distinguished between the behaviour at *protocol level*, that is the actual SSL records exchanged as captured and viewed with SSLDump, and the behaviour at *application level*, that is the results (error messages a.s.o.) the user sees into his running application. Obviously, in the tests where the alert close_notify message was deleted with MITMSSL, this record will not be viewed with SSLDump. Each test is denoted with a number (e.g. T1) and the action performed with MITMSSL is described. In the following sections, when writing SSL/TLS we refer to SSLv3 and TLS 1.0.

4.2 SSL/TLS in Mozilla Firefox

For this test we used Mozilla Firefox v1.5.0.7, enabling the options "Use SSL 3.0", "Use TLS 1.0". The messages exchanged on SSL/TLS connection closure for each test performed are shown in Table 1.

In the normal SSL/TLS session, on closing the connection, both the server and the client send a close_notify alert, followed by the closure at a lower level with a TCP FIN message.

T1. Cancel a data record sent by the server, in the middle of the data flow. At protocol level, the connection is closed by the client due to the bad MAC error detected when processing the SSL record following the one that has

[3] http://www.rtfm.com/ssldump/
[4] http://www.mozilla.org/projects/security/pki/nss/

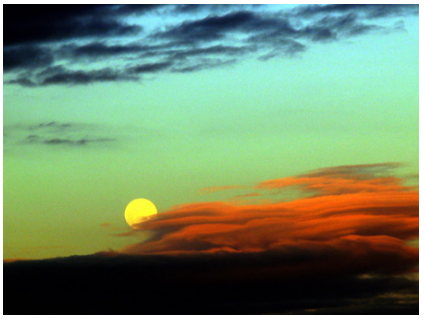


Fig. 2. Full image

Fig. 3. Truncated image

been canceled with the MITMSSL tool. At the application level the result is that the image is not shown and the the browser displays an error message of type: "The image cannot be displayed because it contains errors".

T2. Cancel the last data record sent by the server, but DO NOT cancel the close_notify sent by the server. The behaviour at the protocol level is as follows: the connection is closed due to the bad_record_mac alert generated when processing the close_notify message. Afterwards, the client initiates a handshake again and tries to transfer the data. If also the second time the last data record is canceled but the Alert close_notify is not canceled, the same behaviour as above is obtained at the protocol level and at the application level, but the client does not make another (the third) trial. Finally, the result at application level is that no image is shown and the browser displays an error message of type: "The image cannot be displayed because it contains errors".

T3. Cancel the last data record and the close notify record sent by the server. At protocol level, the SSL connection is closed in the following manner: the server closes the connection with a close_notify (which is deleted with MITMSSL) and at a lower level with TCP FIN; the client sends also an alert close_notify and then it closes the connection at a lower level with TCP FIN. Consequently, in this case the application's implementation on the client side doesn't signal any error and the communication is closed without problems by the browser, even though the last portion of the data sent by the server has been truncated. Thus, at application level no error is signaled by the browser to the user, the image file is transferred to the client but it is INCOMPLETE, that is the truncated image shown in the Fig. 3 is viewed in the browser.

4.3 SSL/TLS in Mozilla

For the client side, two versions of Mozilla web browser have been tested: Mozilla v1.7.8 and Mozilla v1.7.12. We enabled the options "Enable SSL version 3" and "Enable TLS 1.0" and disabled "Enable SSL version 2". For each test performed,

the messages exchanged on SSL/TLS connection closure at protocol level are shown in Table 2.

T1. Normal TLS session. At protocol level, the following behaviour has been observed on SSL/TLS connection closure: after the completion of the handshake and after transferring the data, the server sends an alert close_notify, followed by the closure of TCP connection with a TCP FIN packet. The client sends also an alert close_notify and closes subsequently the connection with a TCP RST.

T2. Cancel a data record sent by the server, in the middle of the data flow. At the protocol level the following behaviour is observed: a MAC error is signaled by the client, at the calculation of the sequence number for the record following the one that has been deleted. The result at the application level is that no image is shown and the browser displays an error message of type "The image cannot be displayed because it contains errors".

T3. Cancel the last data record, but DO NOT cancel the alert close_notify sent by the server. At the protocol level, the behaviour on SSL connection closing is as follows: an error is signaled by the client, because when it receives the server's alert close_notify whose MAC has been computed with the server's sequence number, the client detects the error when comparing it with the one base on his internally computed sequence number. Consequently, the client signals the error by issuing an alert bad_record_mac (fatal level). This alert might not be received by the server because it already closed the connection (with TCP FIN), but the error will be signaled anyhow to the user in the browser. The result at application level is that no image is shown and the browser displays an error message of type "The image cannot be displayed because it contains errors".

T4. Cancel the last data record and the alert close_notify sent by the server. The result obtained is the same as in T3 in the above section. Namely, at protocol level the server closes the connection at a lower level with TCP FIN, after having sent the close_notify alert message, which is deleted with MITMSSL in this test; the client sends independently an alert close_notify and then it closes also the connection at a lower level with TCP FIN. Consequently, the implementation doesn't signal any error and the communication is closed without problems by the browser, even though part of the data sent by the server has been actually deleted. The result at the application level is that no error is signaled by the browser to the user, the image file is transferred, but it is INCOMPLETE, that is the truncated image shown in Fig. 3 is viewed in the browser.

4.4 SSL/TLS in Internet Explorer

In this test there have been used Microsoft Internet Explorer v6.0.26 and Internet Explorer v6.0.29, SP2 as versions of the browser.

T1. Normal TLS sessions. In the normal SSL/TLS connection establishment and closing, the following behaviour is encountered at the protocol level: the

Table 1. Messages exchanged at SSL/TLS connection teardown for tests performed with Mozilla Firefox

Test No.	Messages Exchanged
T1	S>C bad MAC C>S Alert level fatal value bad_record_mac C>S Alert level warning value close_notify C>S TCP FIN S>C bad MAC S>C TCP RST
T2	S>C bad MAC S>C TCP FIN C>S Alert level fatal value bad_record_mac
T3	S>C TCP FIN C>S Alert level warning value close_notify C>S TCP FIN

Table 2. Messages exchanged at SSL/TLS connection teardown for tests performed with Mozilla

Test No.	Messages Exchanged
T1	S>C Alert level warning value close_notify S>C TCP FIN C>S Alert level warning value close_notify C>S TCP RST
T2	S>C bad MAC C>S Alert level fatal value bad_record_mac C>S Alert level warning value close_notify C>S TCP FIN S>C bad MAC S>C TCP RST
T3	S>C bad MAC S>C TCP FIN C>S Alert level fatal value bad_record_mac
T4	S>C TCP FIN C>S Alert level warning value close_notify C>S TCP FIN

Table 3. Messages exchanged at SSL/TLS connection teardown for tests performed with WGet

Test No.	Messages Exchanged
T1	S>C bad MAC S>C TCP FIN C>S Alert level fatal value bad_record_mac New TCP connection ... C>S TCP FIN S>C TCP FIN
T2	S>C TCP FIN C>S Alert level warning value close_notify C>S TCP FIN New TCP connection ...

browser opens two SSL/TLS connections, one for the handshake followed by a new one for the data transfer. In the second SSL/TLS connection the security parameters established in the first one are resumed and the client sends the client certificate (if requested) over the SSL encrypted channel, guaranteeing thus the confidentiality of the data contained in the certificate. In the SSL connection used for the handshake, the server sends a close_notify in the teardown phase and then it closes the TCP connection with a TCP FIN. The client closes at its turn the connection with a TCP FIN. Instead, the client and server don't send alert close_notify messages in the SSL connection used for the data transfer. The communication is simply closed at the TCP level, with TCP FIN messages. The following two tests apply to the SSL connection used for data transfer.

T2. Cancel the penultimate data record. The behaviour at protocol level on SSL/TLS connection closure is as follows: a MAC error is detected, but no alert message (i.e bad_record_mac, fatal level) is generated by the client. The connection is simply closed at a lower level with TCP FIN. The result at application level is that no image is shown.

T3. Cancel the last record; the alert close_notify cannot be canceled because it misses also in normal SSL/TLS sessions. The behaviour at the protocol level when tearing down the SSL connection for data transfer is as follows: no error is signaled. The connection is simply closed at a lower level with TCP FIN. The result at the application level is that no error is signaled by the browser to the user, the file is transferred, but it is INCOMPLETE, i.e. the last part of the image cannot be seen, as shown in the truncated image in Fig. 3. Furthermore, since no fatal alert is sent when the record is deleted, the session can even be resumed when a successive SSL connection is tried to be established.

4.5 SSL/TLS in WGet

In the tests there have been used WGet v1.9 and Wget v1.10.2[5] as software versions. The messages exchanged at SSL/TLS connection teardown for each test performed are shown in Table 3.

In a normal SSL/TLS connection, the behaviour at protocol level on connection closure is as follows: the server sends an alert close_notify message, but the client does not respond with the corresponding close_notify message. It just sends the TCP FIN message at a lower level.

T1. Cancel the last data record, but DO NOT cancel the close_notify sent by the server. The behaviour at the protocol level on connection closure is as follows: a MAC error is signaled at the client, resulting in closing the TCP connection with a TCP FIN. Afterwards, the client reconnects automatically, i.e.

[5] http://www.gnu.org/software/wget/

opens a new connection to get the missing data record, i.e. the last data record. We've measured that the client tries to reconnect automatically 20 times (if the same record as above is deleted each time) and then it gives up.

T2. Cancel the last data record and the server's close_notify message in the initial connection and also at reconnect. The behaviour at the protocol level on connection closure is as follows: each time the client tries to open a new SSL connection toward the server by initiating a full SSL handshake, and it tries to get the remaining of data, that is the last data record. After (re)trying 20 times to get the remaining data, WGet gives up. The final result at the application level is that the file is saved incomplete on disk.

4.6 SSL/TLS in CUrl

In the tests there have been used CUrl v7.11.1 and CUrl 7.15.4[6] with OpenSSL 0.9.8a. The messages exchanged on SSL/TLS connection closure for each test performed are shown in Table 4.

T1. Normal TLS session. The behaviour at protocol level on SSL/TLS connection closure is that both the client and the server sends a alert close_notify message and TCP FIN messages at a lower level.

T2. Cancel the last data record, but DO NOT cancel the alert close_notify sent by the server. The behaviour at protocol level is the same as for Mozilla, namely the client detects the deletion of the last data record on the MAC calculation performed for the received close_notify message. Thus, the client issues an alert bad_record_mac message (fatal level). The server closes the connection by sending a TCP RST at a lower level. At application level curl prints an error message "SSL3_GET_RECORD:decryption failed or bad record mac" and an indication of how many bytes remained to read, e.g. "transfer closed with 3435 bytes remaining to read".

T3. Cancel the last data record and the server's close_notify message. SSLDump shows that at protocol level the server closes the connection with TCP FIN, while client sends a close_notify message (warning level). At the application level CUrl (client side) exists with an error that indicates how many bytes it has't received, e.g. "transfer closed with 3435 bytes remaining to read".

4.7 SSL/TLS in OpenSSL

In this test it has been used the OpenSSL's s_client application together with the corresponding s_server application. The versions tested were: openssl v0.9.7 (as of 31 Dec 2002) and openssl v0.9.8b on cygwin (as of 04 May 2006). The

[6] http://curl.haxx.se/

Table 4. Messages exchanged at SSL/TLS connection teardown for tests performed with CUrl

Test No.	Messages Exchanged
T1	S>C Alert level warning value close_notify S>C TCP FIN C>S Alert level warning value close_notify
T2	S>C bad MAC S>C TCP FIN C>S Alert level fatal value bad_record_mac
T3	S>C TCP FIN C>S Alert level warning value close_notify

Table 5. Messages exchanged at SSL/TLS connection teardown for tests performed with OpenSSL

Test No.	Messages Exchanged
T1	S>C Alert warning close_notify S>C TCP FIN C>S Alert level warning close_notify C>S TCP FIN
T2	S>C bad MAC S>C TCP FIN C>S Alert level fatal value bad_record_mac
T3	S>C TCP FIN C> Alert read:errno=0 SSL3 alert write: warning: close notify

Table 6. Resuming table indicating the current impact at application level of the tests

Test (action done with MITMSSL)	**Mozilla Firefox**	**Mozilla**	**IE**	**WGet**	**CUrl**	**OpenSSL**
Cancel a data record in the middle	detected	detected	detected	detected	detected	detected
Cancel the last data record but NOT the server's close_notify	detected	detected	detected	detected	detected	detected
Cancel the last data record AND the server's close_notify	not detected	not detected	not detected	not detected	detected	not detected

messages exchanged on connection closure (as viewed with SSLDump) for each test performed are shown in Table 5.

T1. Normal session. The behaviour at protocol level on SSL/TLS connection closing is that both the client and the server send alert close_notify messages and TCP FIN messages.

T2. Cancel the last data record but DO NOT cancel the server's close_notify. The behaviour at the protocol level on SSL/TLS connection closure is the same as for Mozilla, namely a MAC error is detected by the client when

receiving the server's close_notify message. At the application level, OpenSSL's s_client terminates indicating an "SSL3 alert write:fatal:bad record mac" error.

T3. Cancel the last data record and the server's close_notify. At protocol level it has been observed that after sending the close_notify message, which is deleted in this test with MITMSSL, the server closes also the connection at a lower level with a TCP FIN; the client sends independently a close_notify alert, but no error at this level is signaled. At the application level s_client application terminates and indicates just a "SSL3 alert write:warning:close notify" warning.

5 Conclusions and Future Work

Despite its wide use for providing end-to-end security, the SSL protocol has its own limitations. One of this limitation is due to the fact that SSL/TLS has been designed with flexibility in mind, being thus totally unaware of the features of the application above, such as the overall size of the data exchanged between the parties. The paper demonstrated that practical truncation attacks can be performed against some widely used applications that are based on SSLv3/TLS 1.0 protocols when the application do not or cannot provide further checks for application data control flow. We believe our work points out an important issue to be taken into consideration by the developers of applications based on SSL/TLS protocols: for scenarios where the data should reach the server untruncated is vital, the application will have to resort to other means to ascertain that. We are currently studying alternative solutions for extending SSL, taking into consideration solutions like the DTLS protocol [13] derived from SSL but targeted to datagram environments, that is environments that do not guarantee the delivery of data.

Acknowledgements

This work has been partly funded by Regione Piemonte (Italy) as part of the GAL-PMI research project.

References

1. OpenSSL library, available at `http://www.openssl.org`
2. SSL 3.0 Specification, available at `http://wp.netscape.com/eng/ssl3/`
3. Dierks, T., Allen, C.: The TLS Protocol Version 1.0. RFC 2246, IETF (January 1999)
4. Dierks, T., Rescorla, E.: The Transport Layer Security (TLS) Protocol Version 1.1. RFC 4346, IETF (April 2006)
5. Vaudenay, S.: Security Flaws Induced by CBC Padding - Applications to SSL, IPSEC, WTLS. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 534–545. Springer, Heidelberg (2002)
6. Oppliger, R.: Security Technologies for the World Wide Web, 2nd edn. Artech House Publishers, Norwood, MA (2003)

7. Wagner, D., Schneier, B.: Analysis of the SSL 3.0 Protocol. In: Proc. of The Second USENIX Workshop on Electronic Commerce Proceedings, November 1996, pp. 29–40. USENIX Press (1996)
8. Brumley, D., Boneh, D.: Remote Timing Attacks are Practical. In: Proc. of 12th Usenix Security Symposium , pp. 1–14 (2003)
9. Canvel, B., Hiltgen, A., Vaudenay, S., Vuagnoux, M.: Password Interception in a SSL/TLS Channel. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 583–599. Springer, Heidelberg (2003)
10. Thomas, S.: SSL and TLS Essentials - Securing the Web. John Wiley & Sons Inc, West Sussex (2000)
11. Shirey, R.: Internet Security Glossary, RFC 2828 (May 2000)
12. Oppliger, R., Hauser, R., Basin, D.: SSL/TLS Session-Aware User Authentication - Or How to Effectively Thwart the Man-in-the-Middle. Computer Communications 29(12), 2238–2246 (2006)
13. Modadugu, N., Rescorla, E.: The Design and Implementation of Datagram TLS. In: Proceedings of ISOC NDSS 2004, February 2004, San Diego, California (2004)

Using WebDAV for Improved Certificate Revocation and Publication

David W. Chadwick and Sean Anthony

Computing Laboratory, University of Kent, UK

Abstract. There are several problems associated with the current ways that certificates are published and revoked. This paper discusses these problems, and then proposes a solution based on the use of WebDAV, an enhancement to the HTTP protocol. The proposed solution provides instant certificate revocation, minimizes the processing costs of the certificate issuer and relying party, and eases the administrative burden of publishing certificates and certificate revocation lists (CRLs).

Keywords: revocation, CRLs, LDAP, HTPP, WebDAV.

1 Introduction

The most common and standardized way of publishing X.509 certificates is in corporate LDAP servers. Several technical problems with the use of LDAP directory servers have been widely documented [1], including: the use ;binary encodings, the lack of a distributed directory capability, and the inability to search for certificates containing specific fields. Other problems are less technical in nature and instead are operational, but they are nevertheless just as inhibiting to successful deployments. For example, most corporate firewalls do not allow the LDAP protocol to pass through them, therefore certificates or certificate revocation lists (CRLs) cannot be easily accessed by external organizations. Finally, we have the complexity of installing and managing LDAP servers, for example, loading the correct schema and setting the correct access rights, which although not insoluble, nevertheless cause frustration and inconvenience especially to small scale deployments with a lack of specialist staff. For these reasons we wanted to find an alternative mechanism to LDAP for publishing X.509 certificates and CRLs that would not suffer from these problems. We wanted a generic solution that would support both public key certificates (PKCs) and attribute certificates (ACs), and that most (ideally all) organizations would already be familiar with. We chose to use Apache servers and the HTTP protocol, since these are ubiquitous. But HTTP on its own is insufficient, since it does not provide a number of useful features, such as the ability to search for specific content. For this reason we investigated (and subsequently implemented) the WebDAV extensions to HTTP [2].

The most common way of revoking public key certificates is through the use of CRLs. However these suffer from a number of well documented operational problems such as the potential for denial of service attacks from lack of availability, or the

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 265–279, 2007.

consumption of too many resources due to their increasingly large size. We will address the whole issue of certificate revocation in the next section, and describe why we have adopted an alternative approach for revocation based on WebDAV.

The rest of this paper is structured as follows. Section 2 re-appraisers the whole issue of certificate revocation and proposes a different approach to addressing this issue. Section 3 describes the WebDAV extensions to HTTP and how they can be used for X.509 certificate and CRL storage and retrieval. Section 4 describes our implementation of WebDAV in our PERMIS authorization infrastructure, in order to store X.509 attribute certificates used for authorization. This mechanism can similarly be used by PKIs to store public key certificates and CRLs. Section 5 discusses our approach, compares it to other work, and concludes.

2 Reappraising Revocation

There are several different approaches that have been taken to the complex issue of revocation of certificates, and of informing remote relying parties when revocation has taken place. A relying party is any Internet based service that consumes the issued certificate (whether it be a PKC or an AC). The primary objective of revocation is to remove a certificate (and all its copies, if any) from circulation as quickly as possible, so that relying parties are no longer able to use it. If this is not possible, a secondary objective is to inform the relying parties that an existing certificate in circulation has been revoked and should not be used or trusted. The latter can be achieved by requiring either the relying parties to periodically check with the certificate issuer, or the certificate issuer to periodically notify the relying parties. Of these, requiring the relying parties to periodically check with the certificate issuer is preferable for two reasons. Firstly, it places the onus on the relying parties rather than on the issuer, since it is the relying parties who are taking the risk of using revoked certificates. Secondly, an issuer may not know who all the relying parties are, so will have difficulty contacting them all, but the relying parties will always know who the certificate issuer is.

The simplest approach to certificate revocation, that used by X.509 proxy public key certificates [6], the Virtual Organisation Membership Service's (VOMS) X.509 attribute certificates [8] and SAML attribute assertions [7] (which are to a first approximation simply XML encoding of attribute certificates), is to never revoke a certificate, and instead to issue short lived certificates that will expire after a short period of time. The certificates are thus effectively and automatically removed from circulation after this fixed period expires. The assumption is that it is unlikely that the certificates will ever need to be revoked immediately after they have been issued and before they have expired due to abuse, therefore the risk to the relying parties is small. Risk (or more precisely risk exposure) is the probability of occurrence multiplied by the loss if the event occurs. Because short lived certificates are only valid for a short period of time, the probability of occurrence is small. Of course, the loss or amount of damage that can be done in a short period of time can be huge, so short lived certificates are not always the best solution where the resulting loss can be high. Consequently SAML attribute assertions have the optional feature of containing a "one time use" field which means that the consuming service can only use the attribute assertion once to grant access, and then it should never be used again. This is

designed to minimise the loss. A similar standardised extension could easily be defined for short lived X.509 public key and attribute certificates, in order to flag them for one time use. But this is not a ubiquitous solution since short lived certificates are not appropriate for every situation, nor is one time use.

An advantage of short lived certificates is that they effectively remove a certificate from circulation after a short period of time, and consequently they mandate that users or service providers must frequently contact the certificate issuer in order to obtain new freshly minted certificates.

The main disadvantage of short lived certificates is knowing how long to issue them for. They should be valid for the maximum time that any user is likely to need them for, otherwise one of the later actions that a user performs may fail to be authenticated or authorised which could lead to a session being aborted and all the processing that has been done so far, being lost. This is a current well known problem with the use of X.509 proxy certificates in grid computing. On the other hand, the longer the certificates are valid, the greater their possibility of misuse without any direct way of withdrawing them from circulation. This has caused some researchers to suggest that proxy certificates should be revocable!

A second disadvantage of short lived certificates is that the bulk of the effort is placed on the issuer, who has to keep reissuing new short lived certificates. This could become a bottleneck to performance. A better solution should put the bulk of the processing effort onto the relying parties, since these are the ones who want to use the issued certificates and the ones who need to minimise their risks. Thus it seems appropriate that they should be burdened with more of the costs.

If the primary objective of removing a certificate from circulation cannot be easily achieved, then the secondary objective of notifying the relying parties when a certificate has been revoked can be achieved by issuing CRLs. A CRL is a digitally signed list of revoked certificates, usually signed by the same authority that issued the original certificates. Revocation lists are updated and issued periodically. X.509 CRLs contain their date and time of issue, and have an optional *next update* time which signifies the latest date by which the next CRL will be issued. Relying parties are urged to obtain the next issue of the revocation list from the issuer's repository before the next update time has expired, in order to keep as up to date as possible. If the next update time is not specified in the CRL, then the frequency of update has to be communicated by out of band means from the issuer to the relying parties. Alternatively, the latest CRL can be sent by the subject along with his certificate, to prove that his certificates has not been revoked. The use of CRLs is standardised in X.509 [4] and the use of LDAP for storing CRLs in RFC 2252 [9]. Revocation lists ensure that relying parties are eventually informed when a certificate has been revoked, no matter how many copies of the certificate there are in circulation, but revocation lists have several big disadvantages. Firstly there is always some delay between a user's certificate being revoked and the next issue of the revocation list appearing. This could be 24 hours or even longer, depending upon the frequency of issuance of the CRLs. Thus, in order to reduce risk to a minimum, a relying party would always need to delay authorising a user's request until it had obtained the latest CRL that was published *after* the user issued his service request, which of course is impractical for most scenarios. If the relying party relies on the current revocation list, then the risk from using a revoked certificate equates, on average, to half that of using

a short lived certificate, assuming the validity period of a short lived certificate is equal to the period between successively issued CRLs. This reduced risk comes at an increased processing cost for the relying party and the issuer.

CRLs can put a significant processing load on both the issuer and the relying party. CRLs have to be issued at least once ever time period, regardless of whether any certificates have been revoked or not during that period. In a large system the lists can get inordinately long containing many thousands of revoked certificates. These have to be re-issued every time period, distributed over the network and read in and processed by the relying parties. In Johnson and Johnson's PKI, their CRL was over 1MB large within a year of operation [10]. To alleviate this problem, the X.509 standard defines delta revocation lists, which publish only the changes between the last published list and the current one. But this increases the processing complexity of the client, and few systems appear to support this feature today.

An alternative approach to notifying relying parties is to use the online certificate status protocol (OCSP) [3]. Rather than a relying party periodically retrieving the latest revocation list from the issuer's repository, the OCSP allows a relying party to ask an OCSP responder in real time if a certificate is still valid or not. The response indicates if the certificate is good, or has been revoked, or its status is unknown. Since most OCSP responders base their service on the latest published CRLs, the revocation status information is no more current than if the relying party had consulted the latest revocation list itself, thus the risk to the relying party is not lessened. But what an OCSP responder does do is reduce the amount of processing that a relying party has to undertake in order to validate a user's certificate. This reduced cost to the relying parties is offset by the cost of setting up and running the OCSP service.

We can see that none of the above approaches to revocation is ideal. Certificates often have a naturally long validity period. For example, authentication certificates are typically issued and renewed annually, whilst some authorisation certificates might require an equally long duration e.g. project manager of a 2 year project. Long lived certificates have traditionally necessitated the use of CRLs but they have several disadvantages. Alternatively we could issue short lived session certificates throughout the duration of the natural validity period, for both authentication and authorisation purposes, without needing to issue CRLs as well, but then there is the inherent conflict between making the short lived certificates long enough for the biggest session and short enough to minimise the risk. Thus we propose a new conceptual model that we believe is superior to short lived certificates, CRLs and OCSP servers.

2.1 A New Model for Revocation

We believe that the optimum approach to certificate issuing and revocation should have the following features:

- A user's certificate should only need to be issued once (and not continually reissued as with short lived certificates) in order to minimise the effort of the issuer.
- The certificate should be valid for as long as the use case requires, which can be a long (measured in years) or short (measured in minutes) duration. Again, this minimises the effort of the certificate issuer (and the delegator, when attribute certificates are used to delegate authority).

- A certificate should be able to be used many times by many different relying parties, according to the user's wishes, without having to be reissued. Of course, the certificate will need to be validated by each relying party each time it is used. But this mirrors the situation today with our plastic credit cards and other similar types of certificate.
- A certificate should be capable of being revoked at any time, and the revocation should be instantaneous. Relying parties should be able to immediately learn about revocations thereby minimising their risks.

All the above features, including instant revocation, can be achieved in the following way. The issuer issues a certificate, giving it its natural validity period, and stores the certificate in a HTTP based repository which is under the control of the issuer. We choose HTTP since this protocol can penetrate firewalls and provide read access to external relying parties. Each certificate is given its own unique URL (called the certificate URL) which points to its location in the repository. This URL is stored in the certificate in a standard extension, in order to strongly bind the repository location to the certificate, so that relying parties know where to go to check if the certificate is still valid. In order to revoke a certificate, the issuer simply deletes the certificate from its repository. The absence of a certificate at its published URL indicates that it is no longer valid. As an added security measure, the issuer may also simultaneously issue a CRL of length 1 and store this at another unique URL (called the revocation URL). The revocation URL, if used, should also be inserted into the original certificate using the existing standard CRL distribution points extension. Relying parties are now able to instantaneously find out the current status of a certificate by contacting the issuer's repository, using either of the URLs embedded in the certificate.

The frequency and method by which a relying party contacts the issuer's repository is determined by its risk mitigation strategy and the optional presence of the revocation URL. The frequency can vary per application or per user request, and is set by the relying party as appropriate, and not by the issuer, which is putting the responsibility and risk where it belongs, with the relying party. In order to minimise risk, a relying party should contact the issuer's repository when a certificate is first validated, and then periodically during the life of the user's session according to its own risk assessment. If the relying party is operating in certificate pull mode, then it must contact the repository anyway at first use in order to pull the certificate, but if the relying party is operating in certificate push mode, contacting the repository is optional from a technical perspective. It must therefore be determined from a risk perspective.

We propose the following procedure for determining the revocation status of a certificate. The relying party periodically issues a HTTP or HTTPS GET command to the certificate URL. (As previously stated, this should be when the certificate is first validated, and then at risk determined intervals.) We will discuss the choice between HTTP and HTTPS later. If the HTTP status code 404 Not Found is returned, the relying party may assume that the certificate has been revoked, and permanently record this in its internal cache along with the time of the request. If the certificate is returned, a simple bitwise comparison of the initial validated certificate with the subsequently retrieved copy of the certificate is all that is needed by the relying party

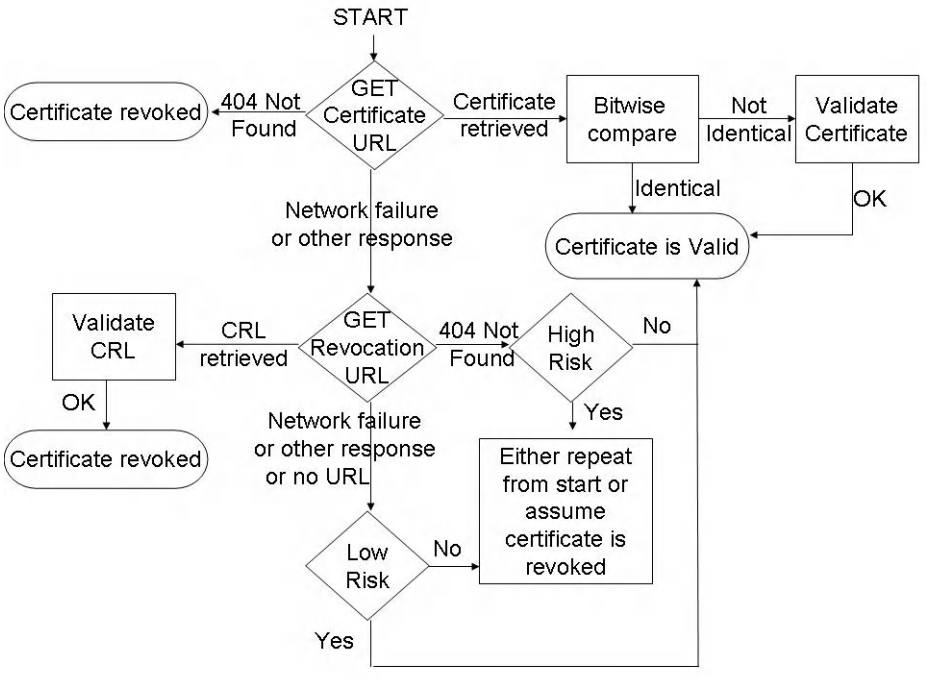


Fig. 1. The revocation procedure

to ensure that the certificate is still identical to the one originally validated. Certificate signature verification is therefore not needed for revocation status checking. The relying party may optionally cache the valid certificate and its time of last retrieval. If the certificate has been updated in the repository during the user's session, then the retrieved certificate will fail the bitwise comparison and will need to be validated again, but this should be a relatively rare occurrence. This procedure is designed to minimise the processing effort of the issuers and the relying parties, whilst maximising the freshness of the revocation information. Issuers do not need to continually mint new certificates or CRLs, and relying parties do not need to process potentially large revocation lists or perform expensive cryptographic operations for the vast majority of their revocation checks.

If on the rare occasion a client is unable to contact the certificate URL and retrieve either a certificate or a Not Found response, it may attempt to contact the revocation URL, providing there is one is in the certificate. If the HTTP status code 404 Not Found is returned from the revocation URL, the relying party may assume that the certificate is still valid (except in high risk cases, where an attacker may be blocking access to the certificate URL and spoofing the revocation URL), and optionally cache this result along with the time of the request. If the CRL is returned instead of Not Found, the signature is validated and the relying party caches the result permanently to ensure the certificate cannot be used again and no further retrieves need be made.

Intermediate caching of the CRL is supported and encouraged, so that if a certificate has been revoked and the CRL successfully retrieved, intermediate web servers can cache the CRL to speed up subsequent queries. Finally, if the relying party is unable to make a connection to the revocation URL, or one does not exist, then the relying party can check its cache to see if a previous request to either URL has succeeded or not. If neither URLs are available, the relying party should use its local risk assessment procedure to decide what to do when there are network problems. For example, if the transaction is low risk, it may decide to treat the certificate as valid. Alternatively it may decide to try contacting the URLs again, or alternatively to treat the certificate as revoked. The flow chart in figure 1 summarises this procedure.

Clients may use either HTTPS or HTTP depending upon their and the issuer's security requirements. HTTP presents a number of security weaknesses compared to HTTPS. Firstly HTTP provides public access to the certificate, which may violate the privacy of the certificate subject. (There is no equivalent privacy leakage for a CRL.) Furthermore intermediate Web servers may cache copies of frequently accessed web pages to improve performance, but this would negate the proposed revocation service. To counteract this, the issuer's Web server must use the no-cache cache-response-directive [15] in the HTTP response of successful certificate requests and Not Found CRL requests, to prevent intermediate servers from caching these responses. This will ensure that all subsequent queries are directed to the authoritative source of the information and that stale cached responses are not received. Finally HTTP is susceptible to redirection, substitution and man in the middle attacks. Consequently, if the certificates are not meant to be publicly available or stronger security is required, then secure access should be provided using HTTP with TLS [5]. This will stop network redirection, substitution attacks and intermediate caching. TLS can also provide confidentiality of the retrieved certificates during transfer, in cases where privacy protection of sensitive certificates is required by the issuer. TLS can also provide strong client side authentication, which will allow access controls to be placed on the WebDAV repository, further protecting the privacy of the subjects' certificates. The privacy of CRLs is less important, and it enhances security if more copies of these are publicly available.

3 The WebDAV Protocol and Its Use with X.509

WebDAV [2] is an Internet RFC that specifies extensions to the HTTP/1.1 protocol so that web content can be managed remotely. WebDAV provides users with the ability to create, remove and query information about web pages, including their contents and properties, such as their creation dates, expiry dates, authors etc. In the context of X.509, a web page will be a single X.509 certificate (either public key or attribute) or a CRL containing a single entry, and their properties can be any fields of the certificate or CRL. WebDAV also provides the ability to create sets of related web pages, called collections, and to retrieve hierarchical membership listings of them. In the context of X.509, a certificate subject can represent a collection, and his/her

certificates can be the collection membership listing. The set of CRLs issued by an issuer can also be a collection membership listing. WebDAV is widely supported, several open source implementations are available including one for Apache, and there is an active community working with it (see http://www.webdav.org/).

WebDAV resources are named by URLs, where the hierarchical names are delimited with the "/" character. The name of a collection ends with /. If we model our X.509 certificate store in the same way as an LDAP directory tree, and name it using the subject DNs to represent collections, this provides us with the ability to retrieve a listing of all the certificates that are owned by a single subject. For example, a public key certificate belonging to the subject whose Distinguished Name is c=gb, o=University of Kent, cn=David Chadwick, might be named in a WebDAV repository with the URL:

> https://server.dns.name/c=gb/o=University%20of%20Kent/cn=David%20Chadwick/pkc=Verisign%20Class1.p7c

Note that the last component pkc=Verisign%20Class1.p7c is the unique name (in terms of the collection) given to the certificate by its issuer. We do not mandate any specific values here, but we recommend using the following file extensions: .p7c for public key certificates, .ace for attribute certificates and .crl for CRLs. A GET request to retrieve all the certificates of David Chadwick would use the URL of the collection, viz:

```
GET /c=GB/o=University%20of%20Kent/cn=David%20
Chadwick/ HTTP/1.1
Host: server.dns.name
```

We can similarly model a CRL store as a collection under its issuer, using the collection name cn=CRLs/, and name each CRL that is issued with the serial number of the certificate that it revokes. This provides us with the ability to retrieve a listing of all the CRLs that have been issued by a single issuer. For example, if David Chadwick is an attribute authority who delegates an attribute certificate with serial number 1234456 to another person in his organization, and then subsequently revokes the AC, the CRL would be located at:

> http://server.dns.name/c=gb/o=University%20of%20Kent/cn=David%20Chadwick/cn=CRLs/serialNumber=1234456.crl

A GET request to retrieve all the CRLs issued by David Chadwick would use the URL of the collection, viz:

```
GET /c=GB/o=University%20of%20Kent/cn=David%20
Chadwick/cn=CRLs/ HTTP/1.1
Host: server.dns.name
```

In order to create a new collection, WebDAV specifies the MKCOL method. The difference between this method and HTTP PUT or POST, is that the latter are allowed to overwrite existing content at the specified URL, whereas MKCOL will fail if there is any existing content at the specified URL. In the context of X.509, this ensures that a certificate issuer cannot unwittingly overwrite existing certificates when creating a

new collection for a subject. This is an important concern when there are several certificate issuers for the same subject (either attribute certificate issuers and/or public key certificate issuers). It is important to ensure that no issuer deletes the certificates issued by another issuer.

In order to create a certificate or CRL or update an existing certificate in an existing collection, the PUT method is used. It is essential that every certificate and CRL has a unique name within a collection, so that updates can overwrite the same certificate and new certificates and CRLs cannot overwrite existing ones. The onus for creating the unique names is with the issuer. We have defined our own algorithms for ensuring this uniqueness is maintained in our implementation, see Section 4 below.

In order to revoke a certificate, the HTTP DELETE command is used by the issuer. This removes the certificate and its properties from the WebDAV server. Simultaneously with this, if CRLs are supported, the issuer should use the HTTP PUT method to create a new CRL containing the serial number of the certificate that has just been revoked. The revocationDate and thisUpdate fields of the CRL should be set to the current time, and the nextUpdate field should be set to sometime after the certificate was due to expire, thereby ensuring that the CRL never needs to be reissued or updated.

Document properties are specified in XML as name/value pairs. Property names must be globally unique and are specified using XML namespaces [11]. Property values should be human readable (in any appropriate character set). Properties can be flagged as live or dead, where live means that the server validates that the values are syntactically correct, and the server may actually set the values of some system known properties, whereas dead means that the client is responsible for validating the syntax and semantics of the property values. For X.509 use, we initially intended to use live properties, set by the certificate issuer, to represent fields of the certificate. We anticipated this would allow easy searching of the certificate store to find certificates with certain properties, for example, find the AC of David Chadwick that has a *manager* role value. The WebDAV protocol does support the PROPFIND method, in which the properties of a resource can be retrieved, but it not possible to specify which property value you require. Only the property types can be specified. Consequently, if we perform a PROPFIND for the "Role" property, then the web server will return an XML encoded message containing all the ACs that contained a property named Role along with their values. Clearly this is not a viable solution. Work on the WebDAV searching and locating capability (DASL) started in 1998, but the work was never completed and the IETF closed the DASL working group some years later. The latest version of the WebDAV Searching and Locating protocol is very recent [17] and several implementations are said to exist, but we were unable to find a usable one. Consequently we have left the search feature for future work. Instead we have implemented a browsing capability in our user agents which allows a user to tree walk through a certificate store and select the certificate that he is looking for. The browse capability is fully scalable, user friendly and meets all the requirements of our use cases. See section 4 and figure 3 for more details.

WebDAV also provides other features that we do not need for X.509 use, such as the ability to copy and move web documents between servers, and the ability to write lock the certificate store when an issuer is performing updates. Consequently, these wont be discussed further.

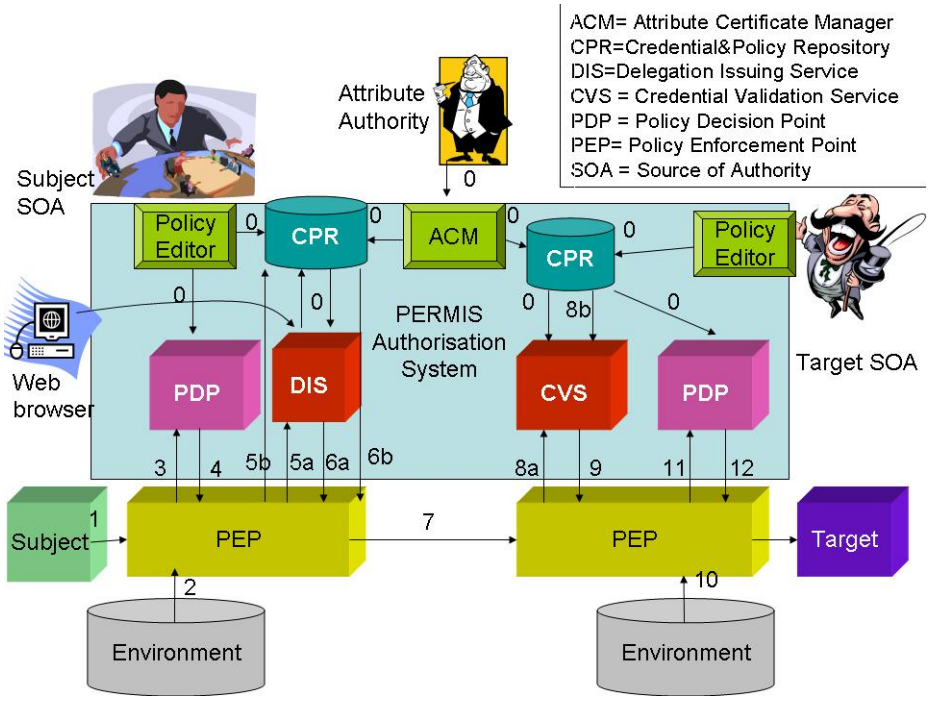


Fig. 2. The PERMIS Authorisation Infrastructure

4 Using WebDAV in PERMIS

PERMIS [12, 13] is an application independent privilege management infrastructure that comprises credential issuing and credential validation functionality as well as policy creation and policy decision making functionality. The components that are important to the current discussion are the Attribute Certificate Manager (ACM) and Delegation Issuing Service (DIS), which both issue X.509 role ACs to holders, and the Credential Validation Service (CVS) which validates the issued role ACs (see Figure 2). In addition, the Policy Editor creates XML policies to control the behaviour of the DIS, CVS and PDP, and each policy can be embedded as a policy attribute in an X.509 AC and digitally signed by its issuer. The policy AC can then be stored in the issuer's collection of ACs, along with his role ACs. Note that the only difference between a role AC and a policy AC is the content of the attribute that is embedded in the AC (although the holder and issuer of a policy AC always contains the same DN). In the original implementation of PERMIS, all the issued X.509 ACs were stored in LDAP directories, in the attributeCertificateAttribute of the entries of their holders. A major disadvantage of this, is that it is impossible to retrieve a single AC of a holder. Instead the entire set of ACs (role and policy ACs) held by a holder has to retrieved as a set. The latest implementation now has the ability to store the ACs in and retrieve them from WebDAV repositories, in which each AC is uniquely identified.

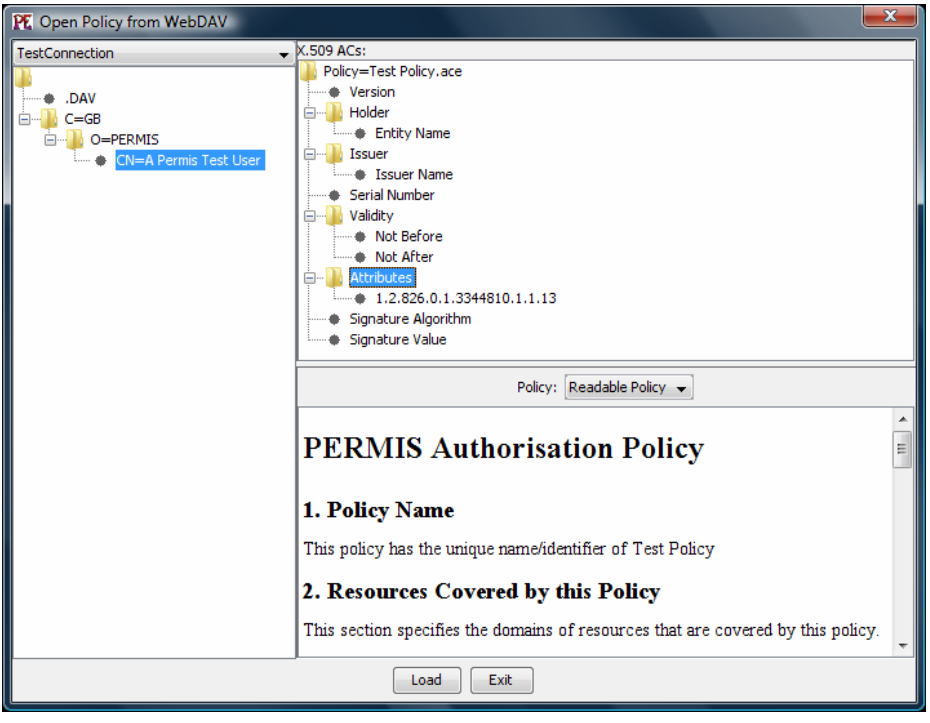


Fig. 3. Retrieving a Policy AC using WebDAV

4.1 Deriving Unique Names for Certificates and CRLs

Because policy and role ACs may be updated by their issuers it is important to have unique names for each of them. Furthermore, all role ACs must have their unique certificate URL and optional revocation URL embedded in extensions so that relying parties can retrieve the contents of either URL to check that the role AC has not been revoked. Rather than allowing the user to specify the names of the ACs or CRLs, we algorithmically create the unique names as follows:

- each AC has the file suffix .ace, whilst each CRL has the file suffix .crl
- the name of a role AC file is created from the contents of its first embedded attribute value plus the serial number of the certificate E.g. a role AC with the embedded attribute type PermisRole with attribute value Project Manager, and certificate serial number of 12345 would create the filename "PermisRole=Project Manager+SN=123456.ace". The serial number provides the uniqueness, whilst the attribute type and value provides user friendliness when the issuer wants to browse WebDAV and retrieve an AC for editing (see Figure 3). As an additional user friendly feature, the WebDAV AC browser displays the entire contents of all the attributes in the bottom window so that the user is sure that he is retrieving the correct role AC.
- the name of a policy AC file is created from the unique name of the embedded XML RBAC policy, which is an XML attribute of the RBAC Policy Element.

E.g. a policy with the name "AstroGridUsers" would produce the name "Policy=AstroGridUsers.ace" (see Figure 3). As an additional user friendly feature, the WebDAV policy browser displays the content of the XML policy attribute in either raw XML or natural language (see lower screen) so that the user is sure that he is retrieving the correct policy.

- the name of a CRL file is created from the serial number of the certificate that it revokes. E.g. a CRL that revokes a certificate with serial number 1234 would produce the filename "serialNumber=1234.crl".

4.2 Certificate Extensions

The optional revocation URL can be placed in the existing standard CRL distribution points extension. The certificate issuer should place the HTTP URL of the future CRL in the uniformResourceIdentifier component of the GeneralName of the DistributionPointName. Note that this URL will not exist until the certificate has been revoked, therefore it is important that the issuer has a deterministic algorithm for creating these URLs, such as the one given in section 4.1 above.

In order to place the certificate URL in an X.509 extension field, we define a new access method for the AuthorityInformationAccess (AIA) extension defined in RFC 3280 [14]. The AIA extension is designed to point to services of the issuer of the certificate in question. One of the standard uses of this extension is to point to the OCSP service provided by the issuer. Since our WebDAV service is replacing the OCSP service, it seems appropriate to use the AIA extension to point to our WebDAV service. We copy below the ASN.1 of the AIA extension, taken from [14] for the convenience of the reader:

```
AuthorityInfoAccessSyntax  ::=
      SEQUENCE SIZE (1..MAX) OF AccessDescription

AccessDescription  ::=  SEQUENCE {
      accessMethod          OBJECT IDENTIFIER,
      accessLocation        GeneralName  }
```

We now define our new accessMethod, webdav, as follows:

```
webdav OBJECT IDENTIFIER ::= { 1.2.826.0.1.3344810.10.2 }
```

When the AIA accessMethod is webdav, then the accessLocation must be a URL pointing to the WebDAV server where the certificate can be found. The URL must point to the exact location of the certificate in the server so that relying parties can download the certificate to compare it to the copy they hold. The absence of the certificate at the URL of this WebDAV server means that the certificate has been revoked.

The Object Identifier in the definition above is one that we have allocated ourselves. However, we propose to take this definition to the IETF PKIX group for standardisation, so that the OID can be replaced by one defined by the PKIX group.

5 Discussion and Conclusions

The OASIS SAML specification has the concept of an artifact that can be obtained from a remote server using an ArtifactResolve message to request the artifact and an ArtifactResponse message to return it [16]. The artifact could be a SAML Attribute Assertion, which is similar in concept to an X.509 attribute certificate, except that it is designed to be short lived and never revoked. The SAML artifact messages are carried over HTTP, therefore will pass transparently through firewalls in the same way as our WebDAV protocol. But there the similarity between the two schemes ends. There are fundamental conceptual differences between the artifact concept in SAML and the certificate publishing and revocation concepts in this paper. Firstly a SAML artifact can only be used once. The SAML specification states "The responder MUST enforce a one-time-use property on the artifact by ensuring that any subsequent request with the same artifact by any requester results in an empty response" [16]. Secondly SAML artifacts are meant to be short lived, quote "The artifact issuer SHOULD enforce the shortest practical time limit on the usability of an artifact, such that an acceptable window of time (but no more) exists for the artifact receiver to obtain the artifact and return it in an <ArtifactResolve> message to the issuer" [16]. In our design, certificates are assumed to be as long lived as required, and used as many times as needed by as many different recipients as the subject desires. We thus believe are system is more flexible and requires less processing resources on the part of both the issuer and relying party.

Our scheme has some similarities with the Netscape Navigator certificate revocation mechanism [18]. In the Netscape scheme, an X.509 extension **netscape-revocation-url** is used to refer to a web location where information about a certificate's status can be found. The actual URL that a relying party should use comprises this extension concatenated with the certificate's serial number (encoded as ASCII hexadecimal digits) e.g. https://www.certs-r-us.com/cgi-bin/check-rev.cgi?02a56c. The document that is retrieved from this URL contains a single ASCII digit, '1' if the certificate is not currently valid, and '0' if it is currently valid. The differences with our scheme are immediately obvious. The revocation URL document always exists, and its content is not digitally signed by the issuer. In comparison our certificate only exists whilst it is has not been revoked and it is a standard certificate digitally signed by the issuer. Optionally our CRL only exists if the certificate has been revoked, and it is a standard CRL containing a single entry signed by the issuer. Our scheme also optionally allows a relying party to find out all the certificates that have been revoked by a particular issuer, by retrieving the WebDAV CRL collection (cn=CRLs/) under the issuer's WebDAV entry.

The one security weakness in our scheme is that it is vulnerable to denial of service attacks, in that if the WebDAV server is not available, relying parties will not be able to tell if a certificate has been revoked or not. But other schemes such as OCSP servers and published CRLs are also equally vulnerable to DOS attacks, and so our scheme is no different in this respect. However, published CRLs do have one advantage in that an old CRL retrieved sometime in the past might still be available to the relying party, and this is better than having no CRL at all, since it does contain some revoked certificates. If this is seen to be a significant benefit, then our optional CRL publishing mechanism is equivalent to it, in that a CRL collection can be

downloaded at any time, just like a conventional CRL. The CRL collection can also be replicated and cached to improve availability. Other well known DOS protection methods, such as overcapacity and server clustering will have to be employed in order to be fully protected against DOS and DDOS attacks, but these are fairly standard techniques that are employed by DNS servers and commercial web sites such as Amazon. Consequently we do not believe that DOS attacks are any more of a significant security threat to our scheme than to existing ones.

The one performance weakness of our scheme is that the latency of certificate validation increases due to network overheads, as compared to that of traditional CRLs, but not to that of OCSP servers. This may be a critical factor to some relying parties such as central servers which process thousands of certificates per second. Central servers benefit from downloading traditional CRLs when they are not busy and storing the results in a local cache for fast lookups when they are validating certificates. The disadvantage of this approach is that the central server still has a vulnerability period between the date and time the latest CRL was published and now, during which all recently revoked certificates will not yet have been incorporated into the latest CRL. This provides an attack window for the holders of the recently revoked certificates. If on the other hand the certificate issuer supports our WebDAV certificate publishing scheme alongside its traditional CRL publishing (or our WebDAV single CRLs scheme) then the central server may check the current status of certificates. It can use the WebDAV GET operation for the certificates of high risk or high value transactions, whilst continuing to use its CRL cache for the certificates of lower risk transactions. In this way the latency penalty of WebDAV lookups is only incurred for a few certificate validations.

In contrast, low throughput relying parties which only process a certificate every few minutes, and where the subject base is very large (and hence so are the traditional CRLs) will benefit greatly from our WebDAV approach of contacting the certificate URL at the time of each certificate validation. This is not too dissimilar from contacting an OCSP server, in terms of processing overheads (HTTPS overheads vs. signed OCSP responses) and latency. The advantages of our WebDAV scheme are that HTTPS and web servers are more ubiquitous than OCSP servers, and where OCSP servers compute their responses on published CRLs and therefore are out of date, WebDAV responses are based on the latest up to date certificate status information.

Finally, comparing our single CRL per certificate scheme against traditional CRLs, we see that there is a trade off between the currency of the revocation information and the overhead of signature creation and validation. A traditional CRL only requires one signature creation per revocation period and one signature validation per relying party per period, whereas our scheme requires one signature creation per revoked certificate and one signature validation per relying party per revocation. The more certificates are revoked per revocation period, the more the processing overhead increases, but so does the risk. Consequently the increased processing overhead has to offset against the risk reduction of instant revocation, but this tradeoff can only be determined on a per-application basis.

To conclude, we have described a new way of publishing and revoking X.509 certificates based on the ubiquitous WebDAV protocol that has a number of distinct advantages over current schemes. We have implemented this in our PERMIS

privilege management infrastructure and performed initial user testing. It has recently been released as open source software along with the existing PERMIS source code, and we will soon expect to obtain operational experiences from users.

References

1. Chadwick, D.W.: Deficiencies in LDAP when used to support a Public Key Infrastructure. Communications of the ACM 46(3), 99–104 (2003)
2. Goland, Y., Whitehead, E., Faizi, A., Carter, S., Jensen, D.: HTTP Extensions for Distributed Authoring – WEBDAV. RFC 2518 (February 1999)
3. Myers, M., Ankney, R., Malpani, A., Galperin, S., Adams, C.: X.509 Internet Public Key Infrastructure: Online Certificate Status Protocol – OCSP, RFC 2560 (1999)
4. ITU-T. The Directory: Public-key and attribute certificate frameworks" ISO 9594-8 (2005) /ITU-T Rec. X.509 (2005)
5. Dierks, T., Allen, C.: The TLS Protocol Version 1.0, RFC 2246 (January 1999)
6. Tuecke, S., Welch, V., Engert, D., Pearlman, L., Thompson, M.: Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile. RFC3820 (June 2004)
7. OASIS. Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0, OASIS Standard (15 March, 2005)
8. Alfieri, R., Cecchini, R., Ciaschini, V., Dell'Agnello, L., Frohner, A., Lorentey, K., Spataro, F.: From gridmap-file to VOMS: managing authorization in a Grid environment. Future Generation Computer Systems 21(4), 549–558 (2005)
9. Wahl, M., Coulbeck, A., Howes, T., Kille, S.: Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions. RFC 2252 (December 1997)
10. Guida, R., Stahl, R., Bunt, T., Secrest, G., Moorcones, J.: Deploying and using public key technology: lessons learned in real life. IEEE Security and Privacy 2(4), 67–71 (2004)
11. Bray, T., Hollander, D., Layman, A.: Namespaces in XML. World Wide Web Consortium Recommendation REC-xml-names-19900114. See http://www.w3.org/TR/1999/REC-xml-names-19990114/
12. Chadwick, D.W., Otenko, A., Ball, E.: Role-based access control with X.509 attribute certificates. IEEE Internet Computing, 62–69 (March-April, 2003)
13. Chadwick, D., Zhao, G., Otenko, S., Laborde, R., Su, L., Nguyen, T.A.: Building a Modular Authorization Infrastructure, UK All Hands Meeting, Nottingham (September 2006). Available from http://www.allhands.org.uk/2006/proceedings/papers/677.pdf
14. Housley, R., Ford, W., Polk, W., Solo, D.: Internet, X.: 509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, RFC 3280 (April 2002)
15. Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., Berners-Lee, T.: Hypertext Transfer Protocol – HTTP/1.1. RFC 2616 (June 1999)
16. OASIS. Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0, OASIS Standard (15 March, 2005)
17. Reschke, J., et al.: Web Distributed Authoring and Versioning (WebDAV) SEARCH. <draft-reschke-webdav-search-11> (9 February, 2007)
18. Netscape Certificate Extensions, Navigator 3.0 Version. Available from http://wp.netscape.com/eng/security/cert-exts.html

Reducing the Computational Cost of Certification Path Validation in Mobile Payment

Cristina Satizábal[1,2], Rafael Martínez-Peláez[1], Jordi Forné[1], and Francisco Rico-Novella[1]

[1] Telematics Engineering Department. Technical University of Catalonia
C/Jordi Girona 1-3, C3, 08034 - Barcelona (Spain)
{isabelcs,rafaelm,jforne,f.rico}@entel.upc.edu
[2] Engineering and Architecture Department. Pamplona University
Km 1 via a Bucaramanga, Pamplona (Colombia)

Abstract. PKI can improve security of mobile payments but its complexity has made difficult its use in such environment. Certificate path validation is complex in PKI. This demands some storage and processing capacities to the verifier that can exceed the capabilities of mobile devices. In this paper, we propose TRUTHC to reduce computational cost of mobile payment authentication. TRUTHC replaces verification operations with hash operations. Results show a better reduction of the cost with ECDSA than with RSA.

Keywords: Public Key Infrastructure (PKI), mobile payment, certification path validation, hash chains.

1 Introduction

The m-payment is defined as the process of exchange money using a mobile device to obtain a product or service [1]. Since the process of payment is achieved without any physical contact between the customer and merchant, and payment information is sent through an open communication, participants can be victims of fraud. Due to these drawbacks, m-payment scheme must offer a high level of security.

The m-payment must consider the following security requirements in order to reduce or avoid frauds: authentication, authorization, confidentiality, integrity and non-repudiation. Different studies about m-payment are centred on the vulnerabilities of its authentication mechanisms and the repudiation problem that affects merchants and financial entities.

Certificates and PKI can be used to provide a good authentication mechanism in m-payment. The use of certificates allows establishing a secure channel for the payment, avoids the repudiation of a transaction and guarantees the integrity and origin of data through digital signature [1].

In spite of the advantages that certificates and PKI offer to m-payment mechanisms, their use is not common, because of the limited resources of mobile devices (processing power, storage capacity and power consumption) and the bandwidth of existing wireless technologies [2]. Additionally, the security infrastructure used for

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 280–296, 2007.

m-payment must support efficiently certificate management (distribution, revocation, and path validation) [3].

Certification path validation is one of the most complex processes of PKI, because it involves: discovering the certification path, retrieving the certificates in the path, verifying the signature of each certificate, and checking the expiration and revocation state of the certificates. This process becomes more complex when the infrastructure grows and also the length of the certification paths, what supposes more work for the verifier and an increase in its storage and processing capacities. Therefore, constrained devices, such as mobile telephones and smart cards, can not always support these requirements.

In this paper, we describe a mechanism to establish an alternative trust relationship between the different entities of a hierarchical PKI using hash chains, what we have called TRUTHC (Trust Relationship Using Two Hash Chains). This contributes to reduce the number of signature verification operations of a path validation process and therefore decreases the verifier's computational cost. This mechanism can be used in some mobile payment scenarios. Section 2 describes the most common mobile payment scenarios, the certification path validation process and the hierarchical architecture. In addition, it defines the hash chains. In section 3, we present some proposals that use certificates for authentication in mobile payment environments. Section 4 shows the operation of TRUTHC. In section 5, we evaluate and compare the computational cost of cryptographic operations in a P2P mobile payment scenario. Finally, section 6 concludes.

2 Background

2.1 Mobile Payment Scenarios

There are different scenarios of mobile payment that can be classified as follows [4]:

- *Real POS (Point Of Sale):* The customers purchase a product on a vending machine using their mobile device. There are two types: in the first type, the customer establishes a communication with the vending machine through a short wireless technology (e.g. Bluetooth or infrared) and all the business transaction is achieved without the participation of the merchant; in the second type, the customer and merchant establish a communication through their mobile device to carry out the payment.
- *Virtual POS:* The merchant gives the option to pay for products using mobile devices. In this scenario participate the following entities: banks (issuer and acquirer), customer, merchant and SP (Service Provider). The customer can pay using a bank account or via phone bill. The payment information is send through UMTS (Universal Mobile Telecommunications System) by a SMS (Short Message Service) or WAP (Wireless Application Protocol).
- *Internet:* The customer makes all the purchase process with his/her mobile device using Internet. The participants are the same like in the virtual POS. The authentication of the merchant before the customer and bank uses digital certificates. The payment information is send through the SP network using WAP.

- *P2P (peer-to-peer):* The customer and merchant have not a defined roll. The same person can be a customer or merchant, it depends on the situation. In this scenario, it is necessary the use of a PG (Payment Gateway) in order to transfer funds between bank accounts. The customer and merchant are authenticated through their certificates.

2.2 Certification Path Validation

A CA's *certification domain* defines the organizational or geographical boundary within which the CA is considered trustworthy. Thus, all the PKI users in a CA's certification domain consider this authority like their trust anchor.

A *trust anchor* is a Certification Authority (CA) that a PKI user explicitly trusts under all circumstances. This is used by the client application as the starting point for all certificate validation. Each user receives the public key of its trust anchor when it is registered in the PKI.

When two users belong to the same certification domain and they want to communicate each other, one can obtain easily the other's public key, since they know the public key of their trust anchor. But when users belong to different certification domains their communication is only possible if there is an uninterrupted chain of trust points between them, which supposes the intervention of several CAs and an agreement among their policies. CAs use cross certification to allow users building trust chains from one point to another, called certification paths.

A *certification path* [5] is a chain of public key certificates through which a user can obtain the public key of another user. The path is traced from the verifier's trust anchor to the CA public key required to validate the target entity's certificate. Thus, the certification path length is equal to the number of CAs in the path plus one: a certificate for each CA and the target entity's certificate.

The primary goal of a path validation process is to verify the binding between a subject and a public key. Therefore, the verifier must check the signature of each certificate in the path in order to trust the public key of the target entity. In general, a path validation process involves the following steps:

- *Discovering a Certification Path:* It is to build a trusted path between the verifier's trust anchor and the target entity based on the trust relationship among the CAs of the PKI.
- *Retrieving the Certificates:* It is to retrieve each certificate in the path from the place(s) where they are stored.
- *Verifying the Digital Signatures*: It is to verify the validity of the digital signature of each certificate in the path. It involves:
  1. Decrypting the signed part of the certificate with its issuer's public key.
  2. Calculating a hash of the certificate's content.
  3. Comparing the results of 1 and 2. If they are the same then the signature is valid.
- *Verifying the Validity of the Certificates:* It is to determine if the certificates have expired or have been revoked.

2.3 Hierarchical Architecture

There are different ways in which CAs might be configured in order to allow PKI users to find the certification paths, called certification architectures. In hierarchical architecture, all the users trust the same root CA (RCA). That is, all the users of a hierarchical PKI begin certification paths with the RCA public key. In general, the root CA does not issue certificates to users but only issues certificates to subordinate CAs. Each subordinate CA may issue certificates to users or another level of subordinate CAs, if it is permitted by policy (Fig. 1).

The certification paths are easy to build in a hierarchical PKI because they are unidirectional and the longest path is equal to the depth of the tree plus one: a CA certificate for each subordinate CA plus the user's certificate. Also, the users of a hierarchy know implicitly which applications a certificate may be used for, based on the position of the CA within the hierarchy.

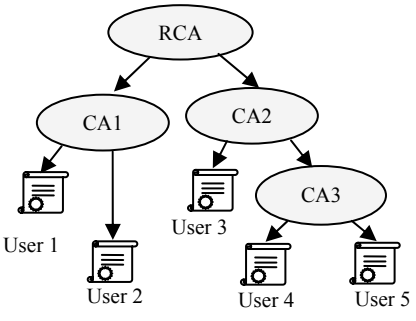


Fig. 1. Hierarchical architecture

The problems of a hierarchical PKI result from the reliance on a single trust point. The compromise of RCA private key results in a compromise of the entire PKI. In addition, transition from a set of isolated CAs to a hierarchical PKI may be logistically impractical because all the users must adjust their trust points.

2.4 Hash Chains

A *hash chain* [6] is a list of values $y_1, y_2, ..., y_m$ linked together cryptographically, where m is the length of the chain. These chains are created by recursively computing a hash function H over a random seed x:

$$y_1 = H(x)$$
$$y_2 = H(y_1)$$
$$\vdots$$
$$y_m = H(y_{m-1})$$

A hash function H is a transformation that takes a variable-size input x and returns a fixed-size string, which is called the hash value.

H must be a one-way function, that is to say:

1. Given a value x, it is easy to compute $H(x)$
2. Given a value y, it is not feasible to compute a value x such that $y = H(x)$

Thus, given a value y_i of the chain, it is unfeasible to compute the previous values.

In addition, H can be collision-free, what means that it is computationally infeasible to find any pair (x, z) such that $H(x)=H(z)$.

3 Related Works

Lee et al. [7] propose a new authentication structure for a P2P m-payment scenario. This is a hierarchical structure, where PAA (Policy Approving Authority) is the root CA, and PCA (Policy Certification Authority), IntCA (Intra Certification Authority) and IntRA (Intra Registration Authority) are subordinated CAs. Thus, the registration process, and the issue, publication and revocation of certificates are done using PKI. Participants are authenticated through certificate exchange and repudiation attack is avoided thanks to digital signatures. In addition, X.509 certificate is modified to obtain a simpler version. In order to reduce computational cost, the authors propose the use of ECC (Elliptic Curve Cryptography) instead of RSA.

Hassinen et al. [3] use the PKI of the Finnish PCR (Population Register Centre). This is a PKI of only one CA, so all the participants trust in the same root CA. In this scheme, the certificates are used to provide authentication, confidentiality and non-repudiation. This paper proposes two scenarios: virtual POS and real POS. In the first scenario, the merchant is authenticated through its certificate. Both, the customer and merchant have an Internet connection. In the other case, the vending machine does not have connection to Internet. The communication between the customer and vending machine is achieved by Bluetooth technology. Both are authenticated through their certificates.

Although the use of ECC as public key algorithm are an interesting alternative to reduce the size of certificates and the runtime of cryptographic operations, the complexity of path validation is still an open topic. TRUTHC contributes to simplify and reduce the computational cost of signature verification process.

4 Trust Relationship Using Two Hash Chains (TRUTHC)

The signature verification pursues two main goals:

1. Ensuring the integrity of the certificates in the path. It is done by hash operations.
2. Verifying the certificates origin or the trust relationship among the entities in the path. It is done by verification operations.

The cost of verification operations is much higher than the cost of hash operations; therefore, the verifier's computational cost can be lower if the number of verification operations during the path validation process is reduced by changing the way in which the trust relationship among the different entities of a hierarchical PKI is established.

TRUTHC establishes an alternative trust relationship among the entities of a hierarchical PKI through two hash chains: one links the secret seeds of the certification authorities and the other links the certificates of each path. This replaces the verification operations by hash operations, what reduces the computational cost of the certification path validation from the verifier's point of view.

In order to describe this proposal, we use the methodology of Karjoth et al. in [8]. Table 1 shows notation used in this paper.

Table 1. Notation

Notation	Meaning
PK_X	Public key of X
SK_X	Private key of X
H(I)	Hash of information I
$CERT_X$	Certificate of entity X
Cnt_X	Content of $CERT_X$
Sig_X	Signature over Cnt_X
s_{RCA}	Random secret seed of the root CA
n_X	Secret seed of authority X
N_X	Encapsulated seed of authority X
L	Certification path length.
h_X	Integrity check value associated with authority X
SN_X	Serial Number of the certificate $CERT_X$
OP_{hash}	Number of hash operations
T_{hash}	Runtime of a hash operation
OP_{en}	Number of encryption operations
T_{en}	Runtime of an encryption operation
OP_{dec}	Number of decryption operations
T_{dec}	Runtime of a decryption operation
OP_{sig}	Number of signature operations
T_{sig}	Runtime of a signature operation
OP_{ver}	Number of verification operations
T_{ver}	Runtime of a verification operation
COST	Computational cost

4.1 Issuing Certificates

TRUTHC extends the typical certificate issuing process [9] in a hierarchical architecture. The chaining relation, encapsulated seed and protocol are:

Chaining Relation

$n_{RCA} = H(s_{RCA})$

$n_{CA1} = H(n_{RCA}, SN_{CA1})$

$n_{CAi} = H(n_{CAi-1}, SN_{CAi}),\ 2 \le i \le L - 1$

$h_{CA1} = H(n_{RCA}, Cnt_{CA1})$
$h_{CAi} = H(h_{CAi-1}, n_{CAi-1}, Cnt_{CAi})$, $2 \le i \le L-1$
$h_U = H(h_{CAL-1}, n_{CAL-1}, Cnt_U)$

Encapsulated Seed:
$N_{CAi} = \{n_{CAi}\}_{PK_{CAi}}$, $1 \le i \le L-1$

Protocol
$CAi \rightarrow CAi+1$: $CERT_{RCA}$, $CERT_{CAi+1}$, h_{CAi+1}, N_{CAi+1}
$CA_{L-1} \rightarrow U$: $CERT_{RCA}$, $CERT_U$, h_U

We assume that the hash function H is collision-free.

The protocol begins when RCA chooses the random secret seed s_{RCA} and obtains n_{RCA}. Next, RCA issues the certificate $CERT_{CA1}$ and computes h_{CA1} and n_{CA1} by using the seed n_{RCA}. Afterwards, RCA sends to CA1: the trust anchor's certificate $CERT_{RCA}$, the certificate $CERT_{CA1}$, h_{CA1} and the seed n_{CA1} encrypted with CA1 public key, so that only CA1 can decrypt it.

Later, CA1 issues the certificate $CERT_{CA2}$ and carries out the following operations:

$$\begin{aligned} n_{CA1} &= \{\{n_{CA1}\}_{PK_{CA1}}\}_{SK_{CA1}} \\ h_{CA2} &= H(h_{CA1}, n_{CA1}, Cnt_{CA2}) \\ n_{CA2} &= H(n_{CA1}, SN_{CA2}) \\ N_{CA2} &= PK_{CA2}(n_{CA2}) \end{aligned}$$

Then, CA1 sends to CA2: $CERT_{RCA}$, $CERT_{CA2}$, h_{CA2}, N_{CA2}

And so on, until the user U receives from its certification authority CA_{L-1}: the trust anchor's certificate $CERT_{RCA}$, its certificate $CERT_U$, and h_U

Thus, it is created a hash chain with the secret seeds (n_{CAi}) and the serial number of the certificates (SN_{CAi}):

$$\begin{aligned} n_{RCA} &= H(s_{RCA}) \\ n_{CA1} &= H(n_{RCA}, SN_{CA1}) \\ n_{CA2} &= H(n_{CA1}, SN_{CA2}) \\ &\vdots \\ n_{CAL-1} &= H(n_{CAL-2}, SN_{CA\,L-1}) \end{aligned}$$

Where: $L-1$ is the number of subordinate CAs in the certification path.

And another hash chain with the integrity check values (h_{CAi}), the secret seeds (n_{CAi}), and the content of the certificates (Cnt_{CAi}):

$$\begin{aligned} h_{CA1} &= H(n_{RCA}, Cnt_{CA1}) \\ h_{CA2} &= H(h_{CA1}, n_{CA1}, Cnt_{CA2}) \\ &\vdots \\ h_{CAL-1} &= H(h_{CAL-2}, n_{CAL-2}, Cnt_{CAL-1}) \\ h_U &= H(h_{CAL-1}, n_{CAL-1}, Cnt_U) \end{aligned}$$

Fig. 2 shows the chaining relation of the certificates.

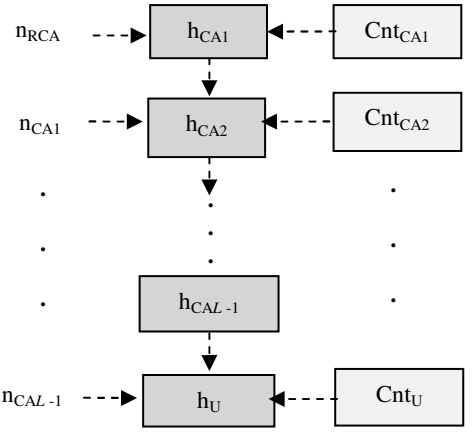


Fig. 2. Chaining relation of the certificates

4.2 Verifying Certificates

We include a new third trust party (TTP) to the PKI called Verification Authority (VA) (Fig. 3). VA verifies the integrity of the certificates and the trust relationship among the entities of a certification path. However, functions of VA can be carried out by RCA, since this authority can compute the seeds of its subordinated CAs.

RCA issues the certificate $CERT_{VA}$, and sends to VA: the trust anchor certificate $CERT_{RCA}$, the certificate $CERT_{VA}$ and the secret seed n_{RCA} encrypted with the public key of VA, so that only VA can decrypt it.

Encapsulated Seed
$N_{VA} = \{n_{RCA}\}_{PK^{VA}}$

Protocol
$CA \rightarrow VA$: $CERT_{RCA}$, $CERT_{VA}$, N_{VA}

If user V wants to verify the signature of a message M sent by user U, they are necessary the following steps:

1. User V retrieves $CERT_U$ and h_U.
2. User V sends VA: $CERT_U$
3. VA retrieves the other certificates of the certification path: $CERT_{CA1}$, $CERT_{CA2}$... $CERT_{L-1}$.
4. VA computes h'_U by using equations of the chaining relation in section 4.1 and the secret seed n_{RCA}. Then, it returns h'_U in a signed response to user V.
5. User V verifies the signature of the VA response. This implies to verify the signature of the VA certificate with PK_{RCA} and then the signature of the VA response.
6. If h_U and h'_U are the same, user V can trust the integrity of the certificate $CERT_U$ and that it is part of the certification path of user U.
7. User V verifies the signature of message M using the public key of user U, PK_U, obtained from $CERT_U$.

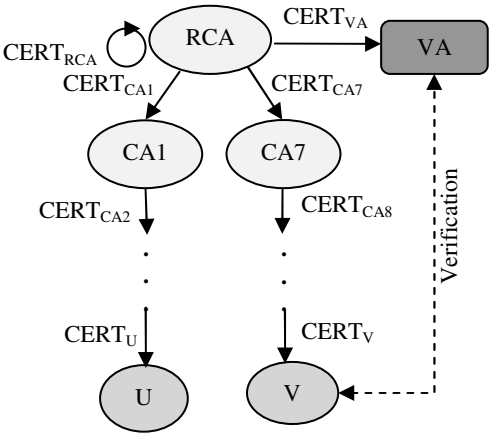


Fig. 3. Verification model

4.3 Integration into X.509 Certificates

The integrity check value h_X can be included as an extension in the X.509 certificates, so TRUTHC is compatible with them. However, it is good to realize that VA has to exclude this extension to compute the hash of content Cnt_{CAi}.

The inclusion of this extension in X.509 certificates increases their size slightly, for example, if SHA-1 is used, the size of the h_X extension will be 20 bytes.

4.4 Security Properties

1. *Seed confidentiality:* Thanks to the properties of public key cryptography, only the authority CAi can decrypt $N_{CAi}=PK_{CAi}(n_{CAi})$. Thus, each CA of the PKI knows its own seed and can compute the seeds of its subordinate CAs but not the seed of its superior authority in the hierarchy.
2. *Integrity of the certificates:* If the attacker changes the content of the certificate $CERT_{CAi}$ in the path, but leaves h_{CAi} intact, to hold the chaining relation with the modified content Cnt'_{CAi}, it is necessary:

$$H(h_{CAi-1},n_{CAi-1},Cnt'_{CAi})=H(h_{CAi-1},n_{CAi-1},Cnt_{CAi})$$

   But this violates the assumption that the hash function H is collision-free. Therefore, it is not possible to modify the content of a certificate in the certification path without modifying the chaining relation of the certificates.
3. *Insertion Resilience:* If the attacker wants to insert a new certificate in a path and the encryption scheme is secure, although he knows h_{CAi}, he will not be able to obtain the value n_{CAi}, needed to compute h_{CAi+1}.
4. *Strong Forward Integrity of the Certificates:* The certification authorities are the unique ones that can compute the integrity check values h_{CAi} of their issued certificates because only they know the seeds n_{CAi}. Therefore, if the value h_U retrieved by

the verifier V is the same as the value h'u returned by VA, the verifier can trust the integrity of the certificates and that they are part of the same certification path.

5. *Integrity of the Hash Chain:* If the attacker modifies the value of some h_{CAi} to carry out a denial of service (DoS) attack, when VA detects a wrong h_{CAi}, it can verify the signature of the implied certificate to ensure the integrity of this value of the chain.

5 Evaluation of Computational Cost

One of the possible application scenarios of TRUTHC is mobile payment. In this section, we evaluate the computational cost of the authentication process in a P2P mobile payment scenario. We define computational cost like the CPU time required to carry out the cryptographic operations involved in a mobile payment.

Let us suppose that there is a hierarchical PKI used for P2P mobile payments. This PKI involves the following entities (Fig. 4):

- *Root Certification Authority (RCA)*: It could be a world famous certification authority, such as Verisign, GlobalSign, etc. RCA issues certificates to CCAs.
- *Credit Card Certification Authority (CCA)*: It is a CA that belongs to some credit card corporation, such as VISA, MasterCard, etc. A CCA issues certificates to BCAs.
- *Bank Certification Authority (BCA)*: It is a CA that belongs to certain bank. BCAs issue certificates to PGAs.
- *Payment Gateway Authority (PGA)*: It is the CA of some e-commerce application service provider. A PG acts like mediator between the customer and merchant, such as PayPal, eBay, etc. PGAs issue certificates to users (customers and merchants).
- *Customer (C)*: It is a user that wants to obtain some object or service from a merchant.
- *Merchant (M)*: It is a user that offers its products or services to the customers.

In this scenario, the customer and merchant verify the certification path of each other during the mutual authentication process. Since, their terminals are limited mobile devices and path validation demands a high computational cost and storage capacity, TRUTHC is suitable for this scenario, because it allows the customer and merchant delegate path validation process in other entity, the VA. In this case, RCA is the VA.

5.1 Number of Cryptographic Operations

If customer C wants to carry out a mobile payment process with the merchant M in Fig. 4, it is necessary a mutual authentication. We suppose they use WTLS protocol.

The messages exchanged in the mutual-authentication are shown in Fig. 5. In a full WTLS handshake, two round-trips are required before the client (customer) and server (merchant) can start exchanging encrypted application data (payment procedure). In the first round-trip, client and server hello messages are exchanged. In

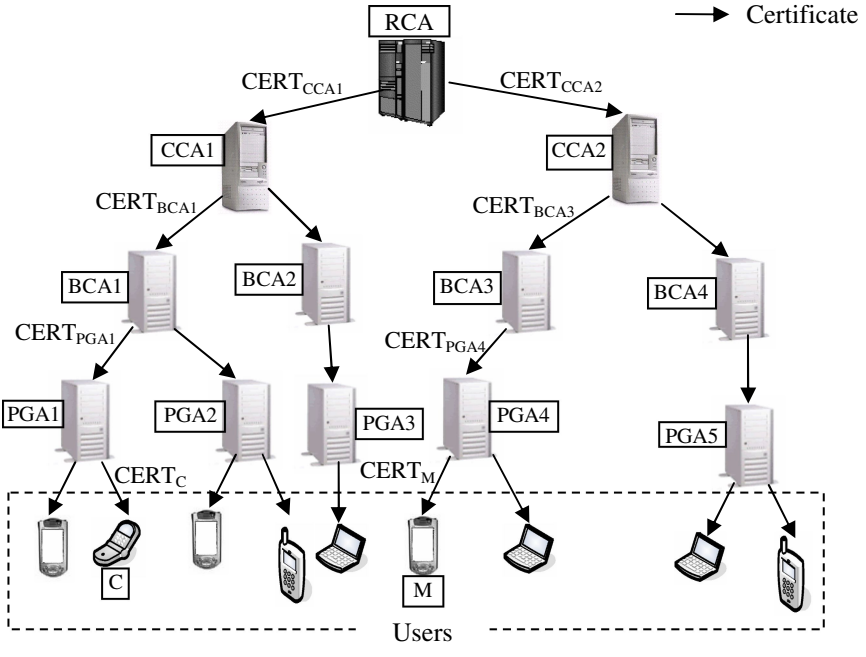


Fig. 4. Hierarchical PKI of P2P scenario

addition, the server sends the client its public-key certificate in the second half of the first round trip [10].

When the client receives the server's certificate, it validates the certificate by verifying the certification path of this certificate. According to Fig. 4, the certification path of the merchant includes certificates $CERT_{CCA2}$, $CERT_{BCA3}$, $CERT_{PGA4}$ and $CERT_M$. In a typical PKI, the customer must verify the signature of these certificates using PK_{RCA} to check the first certificate ($CERT_{CCA2}$).

Client Hello	---------->	
		ServerHello
		Certificate
		CertificateRequest
	<----------	ServerHelloDone
Certificate		
ClientKeyExchange (only for RSA)		
CertificateVerify		
ChangeCipherSpec		
Finished	---------->	
	<----------	Finished
Application Data	<---------->	Application Data

Fig. 5. Mutual-authentication WTLS messages

After the *ServerHelloDone* message is received, the client sends its certificate to the server. The merchant also verifies the certification path of the customer. Fig. 4 shows that this certification path includes the certificates $CERT_{CCA1}$, $CERT_{BCA1}$, $CERT_{PGA1}$ and $CERT_{C}$. Therefore, the merchant carries out the same number of cryptographic operations than the customer to verify the certification path, in a typical PKI as in a PKI with TRUTHC.

Tables 2 and 3 show the number of cryptographic operations carry out by customer and merchant in a typical PKI, using RSA and ECC respectively. On the other hand, Tables 4 and 5 show the number of cryptographic operations carry out by customer, merchant and RCA in a PKI with TRUTHC, using RSA and ECC respectively.

Table 2. Cryptographic operations using RSA in a typical PKI

Entity	OP_{hash}	OP_{en}	OP_{dec}	OP_{sig}	OP_{ver}
Customer	6	1	0	1	4
Merchant	6	0	1	0	5

Table 3. Cryptographic operations using ECC in typical PKI

Entity	OP_{hash}	OP_{en}	OP_{dec}	OP_{sig}	OP_{ver}
Customer	6	0	0	1	4
Merchant	6	0	0	0	5

Table 4. Cryptographic operations using RSA in a PKI with TRUTHC

Entity	OP_{hash}	OP_{en}	OP_{dec}	OP_{sig}	OP_{ver}
Customer	3	1	0	1	1
Merchant	3	0	1	0	2
RCA	16	0	0	2	0

Table 5. Cryptographic operations using ECC in a PKI with TRUTHC

Entity	OP_{hash}	OP_{en}	OP_{dec}	OP_{sig}	OP_{ver}
Customer	3	0	0	1	1
Merchant	3	0	0	0	2
RCA	16	0	0	2	0

5.2 Computational Cost

We assume that the speed benchmarks showed in Table 6 is the runtime of the cryptographic operations carried out by the RCA. They were run on an UltraTM-80, a Sun server equipped with a 450MHz UltraSPARC II processor ECC over RSA increases at higher key sizes[11].

To evaluate the efficiency of our mechanism in different mobile devices, we assume that the merchant's terminal is a PDA and the customer's terminal is a mobile phone. We use the runtime of cryptographic operations of a PDA with StrongARM

Table 6. Runtime of cryptographic operations for UltraTM-80

Algorithm	Runtime (ms/operation)
RSA-2048 Signature	205,5
ECDSA-193 Signature	9,2

Table 7. Runtime of Cryptographic operations for PDA StrongARM 206MHz

Cryptographic Operation	RSA-2048	ECDSA-193
Signature, Decryption	1273,8	39,0
Verification, Encryption	39,1	76,6

Table 8. Runtime of Cryptographic operations in mobile phones using RSA-1937

Device	Runtime Cryptographic Operations (ms/operation)	
	Signature, Decryption	Verification, Encryption
Nokia 6610	74682	2825
Nokia 6600	7125	157
Ericsson P900	3703	109
Siemens S55	883602	30094

Table 9. Runtime of Cryptographic operations in mobile phones using ECDSA-191

Device	Runtime Cryptographic Operations (ms/operation)	
	Signature, Decryption	Verification, Encryption
Nokia 6610	2294	4382
Nokia 6600	860	1266
Ericsson P900	453	843
Siemens S55	18963	35277

Table 10. Computational cost in P2P mobile payment: Typical PKI vs. PKI with TRUTHC

Device	Entity	COST (ms)			
		Typical PKI		PKI with TRUTHC	
		RSA	ECDSA	RSA	ECDSA
PDA	Merchant	1470	423	1353	154
Nokia 6610	Customer	88808	19823	80333	6677
Nokia 6600	Customer	7911	5925	7440	2127
Ericsson P900	Customer	4249	3826	3922	1297
Siemens S55	Customer	1034073	150072	943791	54241
UltraTM-80	RCA	0	0	411	18

206 MHz processor using RSA-2048 bits and ECDSA-193 bits obtained from [11], shown in Table 7. At the same time, we consider four mobile phones as customers: Nokia 6610, Nokia 6600, Ericsson P900 and Siemens S55. The runtime of their cryptographic operations using RSA-1937 and ECDSA-191 [12] is shown in Tables 8 and 9. According with Lenstra and Verheul [13] a key size of 190 bits in ECDSA is

approximately as secure as a key of 1937 bits in RSA. In a practical case, the difference of the runtime between ECDSA-191 and ECDSA-193 is less significant. For that reason, we use these runtimes to compare the development of a PDA and different

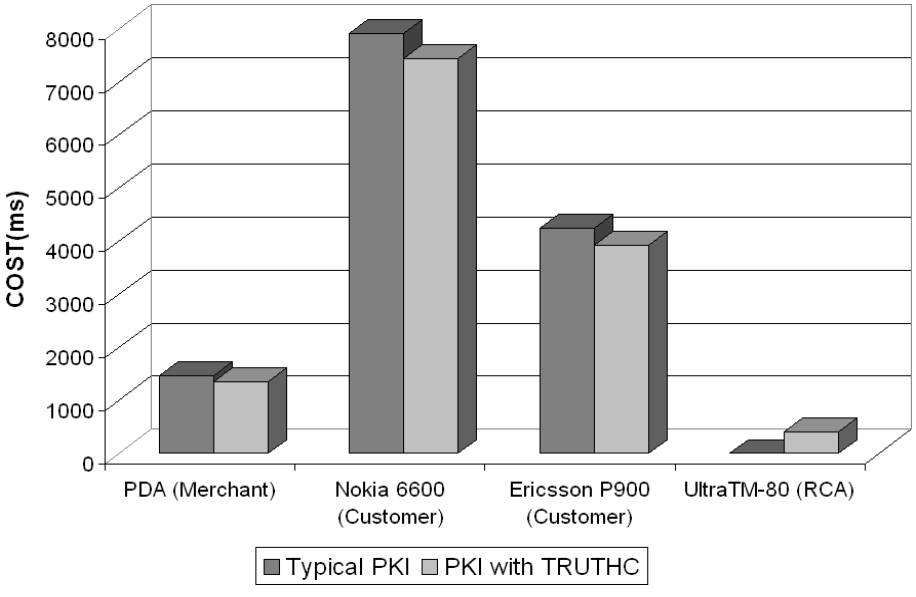


Fig. 6. Computational Cost using RSA: Typical PKI vs. PKI with TRUTHC

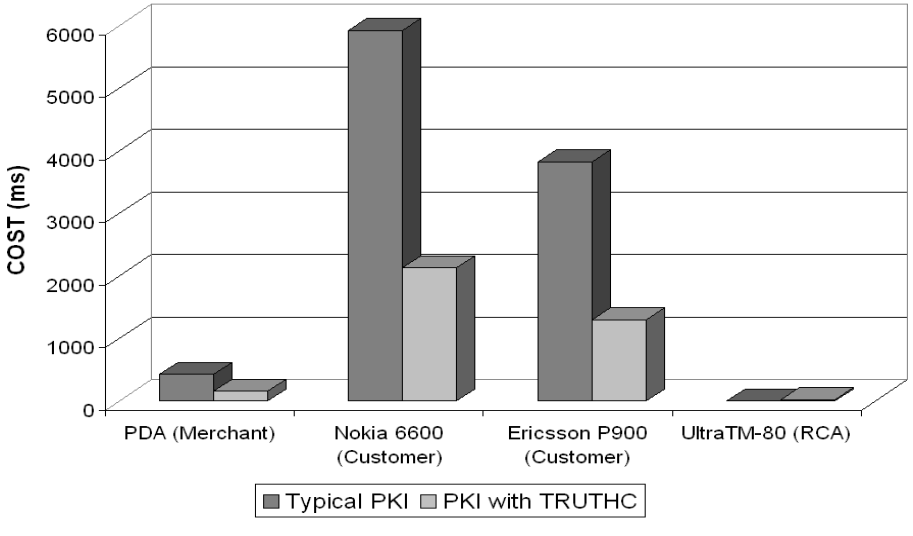


Fig. 7. Computational Cost using ECDSA: Typical PKI vs. PKI with TRUTHC

mobile phones when they carry out TRUTHC. The implementation of RSA-1937 bits is based on the IAIK JCE micro edition, which uses the Chinese Remainder Theorem and Montgomery multiplication.

In addition, we use SHA-1 as hash function and take 0,001ms/operation [14] as the runtime for RCA and 0,19ms/operation [15] as the runtime for customers and merchant.

Equation (1) is used to compute the cost. Table 1 shows the meaning of the notation.

$$COST = (OP_{hash}*T_{hash})+(OP_{en}*T_{en})+(OP_{dec}*T_{dec})+(OP_{sig}*T_{sig})+(OP_{ver}*T_{ver}) \quad (1)$$

Table 10 compares the computational cost of the different entities in a typical PKI and a PKI with TRUTHC.

Figures 6 and 7show the computational cost of the different entities involved in the mobile payment authentication, using RSA and ECDSA, respectively.

6 Conclusions

PKI can provide secure authentication in m-payment scenarios, but its complexity hinder its use in this environment. Certification path validation is one of the most complex processes of PKI and its computational cost is sometimes high, mainly for devices with low processing capacity, such as mobile phones and PDAs. Among the operations carried out by the verifier, cryptographic operations require more processing time than the others, especially verification operations.

In this paper, we propose TRUTHC, a method that establishes an alternative trust relationship among the entities of a hierarchical PKI using two hash chains: one links the secret seeds of the certification authorities and the other links the certificates of each path. TRUTHC reduces the number of verification operations during the signature verification process and therefore the computational cost of the path validation.

TRUTHC allows verifying the integrity of the certificates in a path and checking that each of those certificates belongs to the same path, through simple hash operations.

In addition, we have proved that TRUTHC reduce the number of verification operations carried out by the customer and merchant during mutual authentication. As Tables 2 and 3 show the customer and merchant carry out four and five verification operations respectively during WTLS protocol, in a traditional PKI, while in PKI with TRUTHC (see Tables 4 and 5), the number of verification operations carry out by the customer and merchant is one and two respectively.

Fig. 6 shows that our proposal reduces the computational cost of the mutual authentication process in m-payment around the 8% using RSA. When we use ECDSA algorithm (Fig. 7), the difference between the computational cost of traditional PKI and PKI with TRUTCH is larger than 60%.

The enormous advantage offered by ECDSA algorithm in terms of computational cost and the reduction of verification operations achieved with TRUTHC, motivates the use of certificates in m-payment schemes.

The security of the proposed method depends mainly on the confidentiality of the secret seeds n_{CAi} that must be only known by the certification authorities. Then, they must be stored safely, as if they are the private key of the authority. Thus, the compromise of one of these seeds implies the revocation of the CA certificate and the certificates issued by this authority.

Acknowledgement

This work has been partially supported by the Spanish Research Council (CICYT) under the project SECONNET (TSI2005-07293-C02-01), and graduate scholarship from CONACYT (Mexico).

References

[1] Nambiar, S., Lu, C.-T., Liang, L.R.: Analysis of Payment Transaction Security in Mobile Commerce. In: IEEE International Conference on Information Reuse and Integration (IRI 2004), pp. 475–480 (2004)

[2] Wrona, K., Schuba, M., Zavagli, G.: Mobile Payments – State of the Art and Open Problems. In: Fiege, L., Mühl, G., Wilhelm, U.G. (eds.) WELCOM 2001. LNCS, vol. 2232, pp. 88–100. Springer, Heidelberg (2001)

[3] Hassinen, M., Hyppönen, K., Haataja, K.: An Open, PKI-Based Mobile Payment System. In: Müller, G. (ed.) ETRICS 2006. LNCS, vol. 3995, pp. 86–100. Springer, Heidelberg (2006)

[4] Gao, J., Edunuru, K., Cai, J., Shim, S.: P2P-Paid: A Peer-to-Peer Wireless Payment System. In: Second IEEE International Workshop on Mobile Commerce and Services (WMCS'05), pp. 102–111 (2005)

[5] Housley, R., Polk, W., Ford, W., Solo, D.: RFC3280 - Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. IETF Network Working Group (2002)

[6] Lamport, L.: Password Authentication with Insecure Communication. Communications of the ACM 24, 770–772 (1981)

[7] Lee, B.k., Lee, T.-C., Yang, S.H.: A MEP (Mobile Electronic Payment) and IntCA Protocol Design. In: Yang, L.T., Rana, O.F., Di Martino, B., Dongarra, J.J. (eds.) HPCC 2005. LNCS, vol. 3726, pp. 331–339. Springer, Heidelberg (2005)

[8] Karjoth, G., Asokan, N., Gülcü, C.: Protecting the Computation Results of Free-Roaming Agents. In: Second International Workshop on Mobile Agents (MA'98), pp. 195–207. Springer, London(UK) (1998)

[9] ITU -T. Recommendation X.509: Information Processing Systems - Open Systems Interconnection - The Directory: Authentication Framework (Technical Corrigendum), International Telecommunication Union (2000)

[10] Daswani, N.: Cryptographic Execution Time for WTLS Handshakes on Palm OS Devices, Certicom Public Key Solutions (2000)

[11] Gupta, V., Gupta, S., Chang, S., Stebila, D.: Performance Analysis of Elliptic Curve Cryptography for SSL. In: 3rd ACM Workshop on Wireless Security, pp. 87–94 (2002)

[12] Tillich, S., Grobschädl, J.: A Survey of Public-Key Cryptography on J2ME-Enabled Mobile Devices. In: Aykanat, C., Dayar, T., Körpeoğlu, İ. (eds.) ISCIS 2004. LNCS, vol. 3280, pp. 935–944. Springer, Heidelberg (2004)

[13] Lenstra, A.K., Verheul, E.R.: Selecting Cryptographic Key Sizes. In: Imai, H., Zheng, Y. (eds.) PKC 2000. LNCS, vol. 1751, Springer, Heidelberg (2000)
[14] Long, M., Wu, C.H.: An Intrusion-Resilient and Lightweight Authentication Method Based on Optimum Hash Chain for Wireless Networks (Accessed at: 23/02/2007) http://www.eng.auburn.edu/users/longmen/paper.html
[15] Argyroudis, P.G., Verma, R., Tewari, H., O'Mahony, D.: Performance Analysis of Cryptographic Protocols on Handheld Devices. In: Third IEEE International Symposium on Network Computing and Applications (NCA'04), pp. 169–174 (2004)

Security-by-Contract: Toward a Semantics for Digital Signatures on Mobile Code*

N. Dragoni, F. Massacci, K. Naliuka, and I. Siahaan

Department of Information and Communication Technologies
University of Trento, Italy
surname@dit.unitn.it

Abstract. In this paper we propose the notion of *security-by-contract*, a mobile contract that an application carries with itself. The key idea of the framework is that a digital signature should not just certify the origin of the code but rather bind together the code with a contract. We provide a description of the overall life-cycle of mobile code in the setting of security-by-contract, describe a tentative structure for a contractual language and propose a number of algorithms for one of the key steps in the process, the *contract-policy matching* issue. We argue that security-by-contract would provide a semantics for digital signatures on mobile code thus being a step in the transition from trusted code to trustworthy code.

1 Introduction

Mobile devices are increasingly popular and powerful. Yet, the growth in computing power of nomadic devices has not been supported by a comparable growth in available software: on high-end mobile phones we cannot even remotely find the amount of third party software that was available on our old PC.

One of the reasons for this lack of applications is also the security model adopted for mobile phones. The current security model is exemplified by the JAVA MIDP 2.0 approach and is based on *trust relationships*: mobile code is run if its origin is trusted. This essentially boils down to *mobile code is accepted if it is digitally signed by a trusted party*. The level of trust of the "trusted party" determines the privileges of the code by essentially segregating it into appropriate trust domain.

The problem with trust relationship, i.e. digital signatures on mobile code, is twofold. At first we can only reject or accept the signature. This means that inter-operability in a domain is either total or not existing: an application from a not-so-trusted source can be denied network access, but it cannot be denied access to a specific protocol, or to a specific domain. E.g. if a payment service is available on the platform and an application for paying parking meters is loaded, the user cannot block the application from performing large payments.

The second (and major) problem, is that *there is no semantics attached to the signature*. This is a problem for both code producers and consumers.

From the point of view of mobile code consumers they must essentially accept the code "as-is" without the possibility of making informed decisions. One might well trust

* Research partly supported by the project EU-IST-STREP-S3MS.

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 297–312, 2007.

SuperGame Inc. to provide excellent games and yet might decide to rule out games that keep playing while the battery fells below 20%. At present such choice is not possible.

From the point of view of the code producer they produce code with unbounded liability. They cannot declare which security actions the code will do, by signing the code they essentially declare that they did it. The consequence is that injecting an application in the mobile market is a time consuming operation as SME developers must essentially convince the operators that their code will not do anything harmful.

1.1 Contribution of the Paper

We propose in this paper the notion of *security-by-contract* (as in programming-by contract [1]): the digital signature should not just certify the origin of the code but rather bind together the code with a contract. Loosely speaking, a *contract* contains a description of the relevant features of the application and the relevant interactions with its host platform. A mobile platform could specify platform contractual requirements, a *policy*[1], which should be matched by the application's contract. Among the relevant features, one can list fine-grained resource control (e.g. silently initiate a phone call or send a SMS), memory usage, secure and insecure web connections, user privacy protection, confidentiality of application data, constraints on access from other applications already on the platform.

We provide here a description of the overall life-cycle of mobile code in the setting of security-by-contract, describe a tentative structure for a contractual language and propose a number of algorithms for one of the key steps in the process, namely the issue of *contract-policy matching*.

We argue that security-by-contract would provide a semantics for digital signatures on mobile code thus being a step in the transition from trusted code to trustworthy code.

The research is performed within the limits of the European project "Security of Software and Services for Mobile Systems" (S3MS)[2].

The rest of the paper is organized as follows. In Section 2 we present the security-by-contract framework providing a description of the overall life-cycle of mobile code in this setting. In Section 3 we describe typical security requirements to mobile applications. In Section 4 we focus on contract specification defining also the notion of contract-policy matching. Then in Section 5 we propose an algorithm for contract-policy matching based on the contractual language of Section 4. We end the paper discussing related work and conclusions.

2 The Security-by-Contract Life-Cycle

The framework of security-by-contract for mobile code is essentially shaped by three groups of stake-holders: mobile operator, service provider and/or developer, mobile user. This is shown in Fig. 1.

The mobile code developers are responsible to provide a description of the security behavior that their code provides.

[1] In the sequel we will refer to policy as the security requirements on the platform side and by contract the security claims made by the mobile code.

[2] More information concerning this project can be acquired at http:\\www.s3ms.org.

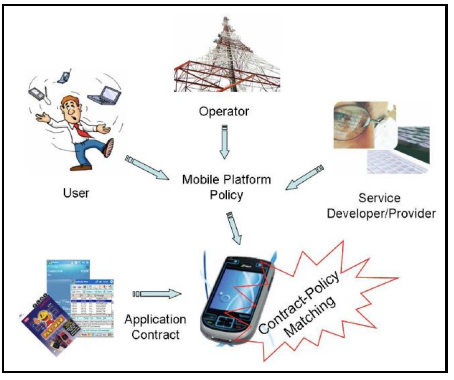


Fig. 1. Key Stakeholders

Table 1. Enforcing Security-by-Contract at Different Stages

Development	Deployment		Execution
(I) at design and development time	(II) after design but before shipping the application	(III) when downloading the application	(IV) during the execution of the application

Definition 1 (Contract). *A contract is a formal complete and correct specification of the behavior of an application for what concerns relevant security actions (Virtual Machine API Calls, Operating System Calls).*

By signing the code the developer certifies that the code complies with the stated claims on its security-relevant behavior.

On the other side we can see that users and mobile phone operators are interested that all codes that are deployed on their platform are secure. In other words they must declare their security policy:

Definition 2 (Policy). *A policy is a formal complete specification of the acceptable behavior of applications to be executed on the platform for what concerns relevant security actions (Virtual Machine API Calls, Operating System Calls).*

A contract should be negotiated and enforced during development, at time of delivery and loading, and during execution of the application by the mobile platform. Fig. 2 summarizes the phases of the application/service life-cycle in which the contract-based security paradigm is present.

In order to guarantee that an application complies with its desired contract or the policy requested on a particular platform we should consider the stage where such enforcement can be done as shown in Table 1.

Enforcing at level (I) can be achieved by appropriate design rules and require developer support; (II) and (III) can be carried out through (automatic) verification techniques. Such verifications can take place before downloading (*static verification* by developers and operators followed by a contract coming with a *trusted signature*) or

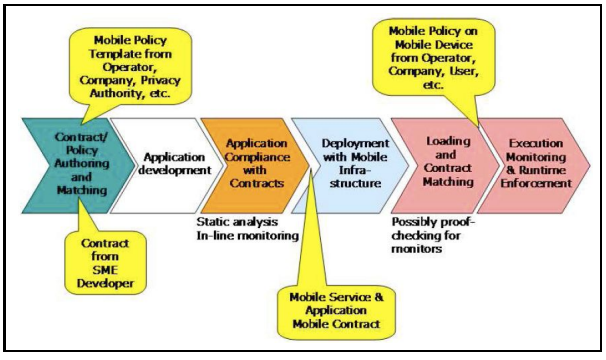


Fig. 2. Application/Service Life-Cycle

as a combination of pre and post-loading operations (e.g., through *in-line monitors* and *proof carrying code*); (IV) can be implemented by *run-time checking*. All methods have different technical and business properties. From an operator's view point:

- working on existing devices would rule out run-time enforcement and favour static analysis, code signing and signature verification on the mobile platform. Monitors may be used (for properties that could not be proved), but on-device proof would then not be possible.
- Operators distrusting the certification process could rely on run-time checks, at the price of upgrading devices' software. Monitors could be used and contracts could be verified on the device itself.
- An operator who wants to be able to run existing applications would prefer run-time enforcement.

The users' perspectives could be different as individuals might care more of privacy, whereas companies might care more of security. Their interest could be on ease of matching the mobile contract against the mobile policy or the combination of policies (e.g. operator, company, user or roaming on another operator's network). The Table 2 shows some of possible strengths and limitations of each different technology.

Contract-Policy Matching. As we can see in Fig. 2, one of the key problems in the overall security-by-contract life-cycle is the *contract-policy matching* issue: given a contract that an application carries with itself and a policy that a platform specifies, is the contract compliant with the policy? Intuitively, matching should succeed if and only if by executing the application on the platform every behavior of the application that satisfies its contract also satisfies the platforms policy. Contract-policy matching represents a common problem in the life-cycle because it must be done at all levels: both for development and run-time operation. To address this issue we need efficient algorithms to match application contracts with device policies. This will be the target of Section 5.

Table 2. Enforcement Technology Strengths and Weaknesses

Criteria	Static Analysis	Monitors	Runtime
Works with existing devices	√	?	×
Works with existing applications	?	×	√
Does not modify applications	√	×	√
Offline proof of correctness	√	√	×
Load-time proof of correctness	×	√	×
May depend on run-time data	×	√	√
Does not affect runtime performance	√	?	×

3 How a Contract Should Look Like?

If contract represents the security behavior of an application the temptation would be to make such contractual claims arbitrarily complex. Since we argue that contract should be matched by mobile device a complex procedure is likely to defy the very spirit of our proposal.

However, a detailed case study [20] in the booming real of Mobile Games shows that detailed contracts are not really necessary. The characteristic feature of these applications is that they need wide access to connectivity to execute correctly. However, the user still wants to control that this connectivity is not abused or misused. Therefore the same permission can be granted or not granted depending, for instance, on previous actions of the applet or some conditions on application environment.

Let us make two examples. Personal information security can be ensured by the following policies:

Example 1. "*No external connections are allowed if the application has accessed the user's personal information*". In this example granting of the permission depends on the applets' previous actions.

Example 2. Using wireless connection can make the device runs out of battery very quickly. To prevent it one can apply the following policy: "*The application is not allowed to use wireless connection if the battery level is below a certain limit*". In this example the permission to run for the application depends on the state of its environment.

Other examples of security policies for mobile devices include:

1. The application sends no more than a number messages in each session.
2. The application only loads each image from the network once.
3. The delay between two periodic invocations of the MIDlet is at least T.
4. The application does not initiate calls to international numbers.
5. The application only uses files whose name matches a given pattern.
6. The application does not send MMS messages.
7. The application connects only to its origin domain.
8. The application does not use the *FileConnection.delete()* function.

9. The application only receives SMS messages on a specific port.
10. The length of a SMS message sent does not exceed the payload of a single SMS message.
11. The application must close all files that it opens.

Notice the difference between policies 1 and 2. The first one specifies the constraint on a single execution (*session*) of the program. The second one puts a restriction on all runs of the application. Policy 3 also requires to make a distinction between multiple sessions of the application. For this reason the contracts must include the constructs that define the *scope* of the obligation. Moreover, such policies as policy 11 are most naturally expressed at the level of separate objects (in this case objects of type *FileConnection*). So three possible scopes of the obligation are:

Multisession – the obligation must be fulfilled by all runs of the application as a whole.
Session – the obligation must be fulfilled by each run of the application separately.
Object – the obligation must be fulfilled by each object of a given type.

Another important issue is the *granularity* of the protected resources. The requirements above show that the field of application of the policy can be limited to particular services (HTTP, SMS, etc.) or even API calls (policy 8). Hence the fine-grained control over the protected resources is an important requirement to the security framework.

4 Contract Specification

A single contract/policy is specified as a *list of disjoint rules* (for instance rules for connections, rules for PIM and so on) instead of one giant specification describing all possible security properties. A rule is defined according to the following grammar:

```
<RULE> :=
    SCOPE [ OBJECT <class> |
        SESSION |
        MULTISESSION ]
        RULEID <identifier>
        <formal specification>
```

Rules can differ both by `SCOPE` and `RULEID`.Scope definition reflects at which scope (`OBJECT`, `SESSION`, `MULTISESSION`) the specified contract will be applied. The tag `RULEID` identifies the *area* of the contract (which security-relevant actions the policy concerns, for example “files” or “connections”).

We assume that `SCOPE` and `RULEID` divide the set of security-relevant actions into *non-interleaving sets* so that two rules with different scopes and `RULEID`s (in the same contract specification) cannot specify the same security-relevant actions. This assumption allows us to perform matching as a number of simpler matching operations on separate rules, as we will show in Section 5.

For `SESSION` and `MULTISESSION` scopes there is a possibility to set `RULEID` equal to *, which means that the contract can include the actions from any area. Still we

recommend to consider such an option carefully before using it because the matching of such contracts and policies can be inefficient.

The `<formal specification>` part of a rule gives a rigorous and not ambiguous definition of the behavior (semantics) of the rule. Since several semantics might be used for this purpose (such as standard process algebras, security automata, Petri Nets and so on), for the limited scope of this paper we abstract from a particular formal specification, identifying the necessary abstract constructs for combining and comparing rules. Moreover, we assume that rules can be combined and compared for matching only if they have the same scope. This assumption allows us to reduce the problem of combining rules to the one of combining their formal specifications, without considering scopes. Therefore the first thing we do when analyzing the specifications is to group rules within one scope together and reason about them separately.

We have identified the following abstract operators (C and P indicate a generic contract and policy respectively):

- [Combine Operator $\oplus$] $\text{Spec} = \oplus_{i=1,\ldots,n}\text{Spec}_i$
  It combines all the rule formal specifications $\text{Spec}_1, \ldots, \text{Spec}_n$ in a new specification Spec.
- [Simulate Operator $\approx$] $\text{Spec}^C \approx \text{Spec}^P$
  It returns 1 if rule formal specification Spec^C simulates rule formal specification Spec^P, 0 otherwise.
- [Contained-By Operator $\sqsubseteq$] $\text{Spec}^C \sqsubseteq \text{Spec}^P$
  It returns 1 if the behavior specified by Spec^C is among the behaviors that are allowed by Spec^P, 0 otherwise.
- [Traces Operator] $\mathcal{S} = \mathsf{Traces}\,(Spec)$
  It returns the set $\mathcal{S}$ of all the possible sequences of actions that can be performed according to the formal specification $Spec$.

We assume that the above abstract constructs are characterized by the following properties:

Property 1. $\mathsf{Traces}\,(\text{Spec}_1 \oplus \text{Spec}_2) = \mathsf{Traces}\,(\text{Spec}_1) \cup \mathsf{Traces}\,(\text{Spec}_2)$

Property 2. $\text{Spec}_1 \sqsubseteq \text{Spec}_2 \Leftrightarrow \mathsf{Traces}\,(\text{Spec}_1) \subseteq \mathsf{Traces}\,(\text{Spec}_2)$

Property 3. $\text{Spec}_1 \approx \text{Spec}_2 \Rightarrow \mathsf{Traces}\,(\text{Spec}_1) \subseteq \mathsf{Traces}\,(\text{Spec}_2)$

Definition 3 (Exact Matching). *Matching should succeed if and only if by executing the application on the platform every trace that satisfies the application's contract also satisfies the platform's policy:*

$$\mathit{Traces}\left(\oplus_{i=1,\ldots,n} \mathit{Spec}_i^C\right) \subseteq \mathit{Traces}\left(\oplus_{i=1,\ldots,m} \mathit{Spec}_i^P\right)$$

Definition 4 (Sound Sufficient Matching). *Matching should fail if by executing the application on the platform there might be an application trace that satisfies the contract and does not satisfies the policy.*

Definition 5 (Complete Matching). *Matching should succeed if by executing the application on the platform every traces satisfying the contract also satisfy the policy.*

Table 3. Examples of Contract/Policy Matching

Contract/Policy Rule	Object can use one type of connection only	Object can use every type of connections
Object can use HTTP connections only	☑	☑
Object can use HTTP and SMS connections	☐	☑

By applying Def. 4 we might reject "good" applications that are however too difficult or too complex to perform. On the other hand, Def. 5 may allow "bad" applications to run but it will certainly accept all "good" ones (and "bad" applications can later be detected, for instance, by run-time monitoring). Examples of matching between contracts and policies are shown in Table 3.

5 Contract Matching Algorithm

In this Section we provide a generic algorithm for contract/policy matching. The algorithm is *generic* since it does not depend on the formal model adopted for specifying the semantics of rules (process algebra, security automata, Petri Nets, and so on). In other words, the algorithm is defined by means of the abstract constructs discussed in Section 4. Therefore, to exploit the algorithm it will be sufficient to have an implementation of these constructs in the formal language adopted for specifying rules. In Section 5.1 we will provide an automata-based implementation of such constructs, giving in this way a complete version of the algorithm for rules formally specified with automata.

As shown in Fig. 3, the generic contract/policy matching algorithm takes as inputs two rule sets $\mathcal{R}^C$ and $\mathcal{R}^P$ representing respectively the contract and the policy to be matched. The algorithm checks if $\mathcal{R}^C$ "matches" $\mathcal{R}^P$.

Algorithm 1 lists the source code of the MatchContracts function, which represents the root function of the whole algorithm. Basically, the algorithm works as follows. First of all, both rule sets $\mathcal{R}^C$ and $\mathcal{R}^P$ are partitioned according to the scope of the rules (lines 1 and 2). This is done by calling the Partition procedure (Algorithm 2) that partitions a generic rule set $\mathcal{R}$ in a sequence of rule sets with the same scope: $\left\langle \mathcal{R}_{SESSION}, \mathcal{R}_{MULTISESSION}, \{\mathcal{R}_{class}\}_{class \in \zeta^C} \right\rangle$. As discussed in Section 4, this partition is necessary because in the S3MS framework comparison of rules starts only within a certain scope. Created two sequences of scope-specific rule sets (one for the contract and one for the policy), the algorithm checks if each rule set in the sequence of the contract matches the corresponding rule set in the sequence of the policy (lines 3-11). In other words, we match rules within the same scope. This is done by calling the MatchRules function (lines 4-6) that we discuss in the next paragraph. If all succeeds (line 11), than the contract matches the policy. Otherwise, matching fails.

Matching Rules with the Same Scope. Matching between rules is performed by the MatchRules function (Algorithm 3). Since the rules of the two input sets $\mathcal{R}^C$ and $\mathcal{R}^P$

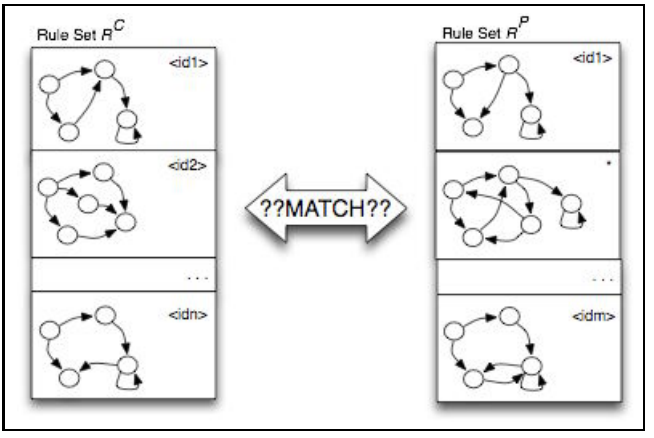


Fig. 3. Contract/Policy Matching Problem

must have the same scope, before doing matching checks the algorithm cleans $\mathcal{R}^C$ and $\mathcal{R}^P$ removing the scope from each rule. As a consequence, two sets L^C and L^P of pairs $\left(\text{ID}^{C/P},\ \text{Spec}^{C/P}\right)$ are built. Now the algorithm is ready to check the contract/policy match. Each pair in L^P is compared with the set L^C by means of the MatchSpec function (line 4). When a match is not found for a pair (line 6), i.e. the MatchSpec function returns 0, that pair is stored in a rule set L^P_{failed} (line 7).

If for all rules in L^P there exists a match with L^C, i.e. the MatchSpec function returns 1 for each pair in L^P so that $L^P_{failed} = \emptyset$, then the match between rules succeeds and the algorithm returns 1 (lines 10-11). Otherwise, if $L^P_{failed} \neq \emptyset$ (i.e. there are no rules in L^C that match with the rules of L^P_{failed}) then the algorithm performs a last "global" check. More precisely, the combination of the rules in L^C is matched with the combination of the rules in L^P_{failed} (line 13). If also this match does not succeed, then the algorithm returns 0, otherwise it returns 1.

Matching Specifications. The MatchSpec function (Algorithm 4) checks the match between a set of pairs $\mathcal{L}^C = \langle \left(\text{ID}^C_1,\ \text{Spec}^C_1\right), \ldots, \left(\text{ID}^C_n,\ \text{Spec}^C_n\right) \rangle$ and a pair $\left(\text{ID}^P,\ \text{Spec}^P\right)$ representing respectively the rules of the contract and a rule of the policy to be matched. The function returns 1 in two situations:

1. there exists a pair $\left(\text{ID}^C,\ \text{Spec}^C\right)$ in L^C that matches with $\left(\text{ID}^P,\ \text{Spec}^P\right)$
2. the combination of all the specifications in L^C matches with $\left(\text{ID}^P,\ \text{Spec}^P\right)$

Otherwise, the function returns 0.

Specification matching is verified as follows. If there exists a pair $\left(\text{ID}^C,\ \text{Spec}^C\right)$ in L^C such that ID^C is equal to ID^P (line 1), then the algorithm checks the hash values of the specifications Spec^C and Spec^P. Matching succeeds if they have the same value (line 2). Otherwise, the algorithm checks if Spec^C simulates Spec^P (line 4). If this is

Algorithm 1. MatchContracts Function

Input: rule set $\mathcal{R}^C$, rule set $\mathcal{R}^P$
Output: 1 if $\mathcal{R}^C$ matches $\mathcal{R}^P$, 0 otherwise

$\left\langle \mathcal{R}^C_{SESSION}, \mathcal{R}^C_{MULTISESSION}, \{\mathcal{R}^C_{class}\}_{class \in \zeta^C} \right\rangle \Leftarrow \mathsf{Partition}(\mathcal{R}^C)$
$\left\langle \mathcal{R}^P_{SESSION}, \mathcal{R}^P_{MULTISESSION}, \{\mathcal{R}^P_{class}\}_{class \in \zeta^P} \right\rangle \Leftarrow \mathsf{Partition}(\mathcal{R}^P)$
if $\mathsf{MatchRules}(\mathcal{R}^C_{SESSION}, \mathcal{R}^P_{SESSION})$ **then**
  if $\mathsf{MatchRules}(\mathcal{R}^C_{MULTISESSION}, \mathcal{R}^P_{MULTISESSION})$ **then**
    for all $class \in \zeta^P$ **do** // *for all classes in policy*
      if $\mathsf{MatchRules}(\mathcal{R}^C_{class}, \mathcal{R}^P_{class})$ **then** // *if* $class \notin \zeta^C$, *then* $\mathcal{R}^C_{class} = \emptyset$
        skip
      else
        return(0)
      end if
    end for
    return(1)
  end if
end if
return(0)

Algorithm 2. Partition Procedure

Input: rule set $\mathcal{R}$
Output: $\left\langle \mathcal{R}_{SESSION}, \mathcal{R}_{MULTISESSION}, \{\mathcal{R}_{class}\}_{class \in \zeta} \right\rangle$

$\mathcal{R}_{SESSION} \Leftarrow \{r \in \mathcal{R} \mid \mathrm{Scope}(r) = \mathrm{SESSION}\}$
$\mathcal{R}_{MULTISESSION} \Leftarrow \{r \in \mathcal{R} \mid \mathrm{Scope}(r) = \mathrm{MULTISESSION}\}$
for all $class \in \zeta$ **do** // *for all classes in contract/policy*
  $\mathcal{R}_{class} \Leftarrow \{r \in \mathcal{R} \mid \mathrm{Scope}(r) = \mathrm{OBJECT} < class >\}$
end for

Algorithm 3. MatchRules Function

Input: rule set $\mathcal{R}^C$, rule set $\mathcal{R}^P$
Output: 1 if $\mathcal{R}^C$ matches $\mathcal{R}^P$, 0 otherwise

$L^C \Leftarrow \{(\mathrm{ID}^C,\ \mathrm{Spec}^C) \mid \langle \mathrm{scope}, \mathrm{ID}^C, \mathrm{Spec}^C \rangle \in \mathcal{R}^C\}$
$L^P \Leftarrow \{(\mathrm{ID}^P,\ \mathrm{Spec}^P) \mid \langle \mathrm{scope}, \mathrm{ID}^P, \mathrm{Spec}^P \rangle \in \mathcal{R}^P\}$
for all $(\mathrm{ID}^P,\ \mathrm{Spec}^P) \in L^P$ **do**
  if $\mathsf{MatchSpec}(L^C, (\mathrm{ID}^P,\ \mathrm{Spec}^P))$ **then**
    skip
  else // *may return* $\emptyset$ *for efficiency*
    $L^P_{failed} \Leftarrow L^P_{failed} \cup (\mathrm{ID}^P,\ \mathrm{Spec}^P)$
  end if
end for
if $L^P_{failed} = \emptyset$ **then**
  return(1)
else
  return$\left(\mathsf{MatchSpec}\left(\left(*,\ \oplus_{(\mathrm{ID}^C,\ \mathrm{Spec}^C) \in L^C}\right), \left(*,\ \oplus_{(\mathrm{ID}^P,\ \mathrm{Spec}^P) \in L^P_{failed}}\right)\right)\right)$
end if

Algorithm 4. MatchSpec Function

Input: $L^C = \langle (\mathrm{ID}_1^C,\ \mathrm{Spec}_1^C), \ldots, (\mathrm{ID}_n^C,\ \mathrm{Spec}_n^C) \rangle$, $(\mathrm{ID}^P,\ \mathrm{Spec}^P)$
Output: 1 if L^C matches $(\mathrm{ID}^P,\ \mathrm{Spec}^P)$, 0 otherwise

```
1: if ∃ (ID^C, Spec^C) ∈ L^C ∧ ID^C = ID^P then
2:    if HASH(Spec^C) = HASH(Spec^P) then
3:       return(1)
4:    else if Spec^C ≈ Spec^P then
5:       return(1)
6:    else if Spec^C ⊑ Spec^P then
7:       return(1)
8:    else // Restriction: if same ID then same specification must match
9:       return(0)
10:   end if
11: else
12:   MatchSpec( (∗, ⊕_{(ID^C, Spec^C) ∈ L^C}), (∗, Spec^P) )
13: end if
```

the case, then the matching succeeds, otherwise the more computationally expensive containment check is performed (line 6). If also this check fails, the algorithm ends and matching fails (because the rules with the same ID must have the same specification).

If there exists no pair in L^C such that ID^C is equal to ID^P (line 11) then the algorithm checks the match between the combination of all the specifications in L^C and $(\mathrm{ID}^P,\ \mathrm{Spec}^P)$ (line 12).

5.1 Applying the Generic Matching Algorithm to Automata-Based Rule Specifications

In this Section we show how the matching algorithm can be used when the behavior of rules (`<formal specification>`) is specified by means of finite state automata (FSA). In this way we provide a complete algorithm for matching contracts with FSA-based rule specifications. As already remarked at the beginning of Section 5, we just need to provide an implementation of the $\oplus$, $\sqsubseteq$ and $\approx$ operators used in Algorithms 3 and 4. For the sake of clarity, we briefly introduce FSA. Then we provide algorithms for implementing the abstract constructs.

FSA are widely used as a powerful formalism both for system modeling and for specification of system properties. Basically, FSA consists of finite numbers of states; transitions between states are performed through *actions*. A subset of states is selected to be *accepting states*. If after performing a sequence of actions (a *run*) the FSA arrives in an accepting state then the automaton is said to *accept* this sequence of actions.

FSA that represents a model of the system can be extracted directly from the control-flow graph of the program. This automaton specifies *actual behavior* of the system. An automaton that specifies the *desired behavior* can be either built directly or from other specification language. For example, FSA for a temporal logic specification can be constructed using the *tableaux method* [11].

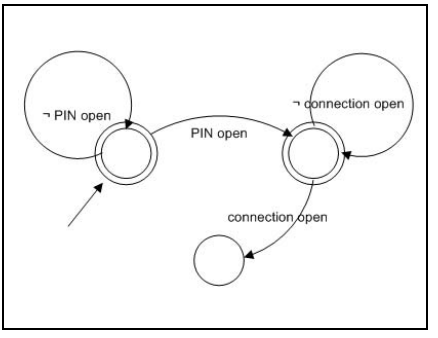


Fig. 4. The Automata-Based Specification of the Policy from Example 1 (circles with double borders denote accepting states, arrows represent the transition, where labels denote actions)

Example 3. To illustrate how FSA can be used for property specifying let us show the automaton for the policy from Example 1. The automaton has two accepting states. It remains in the first (initial) one until the action "PIN open" occurs. This action brings the automaton in the second accepting state. All actions in this state preserve it save the action "connection open". If "connection open" occurs the automaton is brought to the last (non-accepting) state, from which there is no outgoing transition. In this case the automaton terminates in a non-accepting states, which means that the automaton does not accept this run. The resulting automaton is presented at Fig. 4.

Combining Automata-Based Rules. The exploitation of automata for formally specifying rules allows a straightforward implementation of the combine operator $\oplus$: rules are combined by simply making the synchronous product of the related automata.

Automata Matching as Language Inclusion. Given two automata Aut^C and Aut^P representing respectively a rule formal specification of a contract (Spec^C) and of a policy (Spec^P), $\text{Spec}^C \sqsubseteq \text{Spec}^P$ when $\mathcal{L}_{\text{Aut}^C} \subseteq \mathcal{L}_{\text{Aut}^P}$, i.e. the language accepted by Aut^C is a subset of the language accepted by Aut^P. Informally, each behavior of Aut^C is among the behaviors that are allowed by the policy Aut^P. Assuming that the automata are closed under intersection and complementation, then the matching problem can be reduced to an emptiness test [2]:

$$\mathcal{L}_{\text{Aut}^C} \subseteq \mathcal{L}_{\text{Aut}^P} \Leftrightarrow \mathcal{L}_{\text{Aut}^C} \cap \overline{\mathcal{L}_{\text{Aut}^P}} = \emptyset \Leftrightarrow \mathcal{L}_{\text{Aut}^C} \cap \mathcal{L}_{\overline{\text{Aut}^P}} = \emptyset$$

In other words, there is no behavior of Aut^C that is disallowed by Aut^P. If the intersection is not empty, any behavior in it corresponds to a counterexample. Algorithm 5 lists the basic steps for implementing the $\sqsubseteq$ operator for FSA-based rule specifications.

Automata Simulation. Several notions of simulation relations for automata have been introduced in literature ([3,5,4,9] to mention only a few) and discussing each of them is outside the scope of the paper. Intuitively, we can say that a state q_i of an automata

Algorithm 5. FSA-Based Implementation of $\sqsubseteq$

Input: two FSA-based rule specifications Aut^C and Aut^P
Output: 1 if $A_C \sqsubseteq A_P$, 0 otherwise
1: Complement the automaton Aut^P
2: Construct the automaton Aut^I that accepts $\mathcal{L}_{\text{Aut}^C} \cap \mathcal{L}_{\overline{\text{Aut}^P}}$
3: **if** Aut^I is empty **then**
4: return(1)
5: **else**
6: return(0)
7: **end if**

Algorithm 6. FSA-Based Implementation of $\approx$

Input: two FSA-based rule specifications Aut^C and Aut^P
Output: 1 if $A_C \approx A_P$, 0 otherwise.
1: Construct the automaton $\text{Aut}^U = \text{Aut}^C \cup \text{Aut}^P$
2: Build the parity game graph $G^*_A(q_0^{\text{Aut}^C}, q_0^{\text{Aut}^P})$
3: Compute the winning nodes W using Jurdzinski algorithm
4: **if** $(q_0^{\text{Aut}^C}, q_0^{\text{Aut}^P}) \in W$ **then**
5: return(1)
6: **else**
7: return(0)
8: **end if**

A "simulates" a state q_j of an automata B if every "behavior" starting at q can be mimicked, step by step, starting at q_j, i.e. $q_i \approx q_j \Rightarrow L(A[q_i]) \subseteq L(A[q_j])$.

The main approach for determining simulation relations among automata consists of reducing the simulation problem to a simulation game, i.e. to the problem of searching the winning nodes of a parity game graph [4]. Algorithm 6 summarizes the basic operations needed for implementing a simulation construct following this approach [5]. Basically, a parity game graph is constructed starting from two automata A and B and according to a well specific notion of simulation relation (the $*$ in the algorithm indicates a generic simulation relation). Then the Jurdzinski algorithm [10] is used for determining the set of winning nodes.

6 Related Work

Four main approaches to mobile code security can be broadly identified in literature: *sandboxes* limit the instructions available for use, *code signing* ensures that code originates from a trusted source, *proof-carrying code (PCC)* carries explicit proof of its safety, and *model-carrying code (MCC)* carries security-relevant behavior of the producer mobile code.

Sandbox Security Model. This is the original security model provided by Java. The essence of the approach [6] is that a computer entrusts local code with full access to

vital system resources (such as the file system). It does not, however, trust downloaded remote code (such as applets), which can access only the limited resources provided inside the sandbox. The limitation of this approach is that it can provide security but only at the cost of unduly restricting the functionality of mobile code (e.g., the code is not permitted to access any files). The sandbox model has been subsequently extended in Java 2 [7], where permissions available for programs from a code source are specified through a security policy. Policies are decided solely by the code consumer without any involvement of the producer. The implementation of security checking is done by means of a run-time *stack inspection* technique [19].

In .NET each assembly is associated with some default set of permissions according to the level of trust. However, the application can request additional permissions. These requests are stored in the application's *manifest* and are used at load-time as the input to policy, which decides whether they should be granted. Permissions can also be requested at run-time. Then, if granted, they are valid within the limit of the same method, in which they were requested. The set of possible permissions includes, for instance, permissions to use sockets, web, file IO, etc.

Cryptographic Code-Signing. Cryptographic code-signing is widely used for certifying the origin (i.e., the producer) of mobile code and its integrity. Typically, the software developer uses a private key to sign executable content. The application loading the module then verifies this content using the corresponding public key.

This technique is useful only for verifying that the code originated from a trusted producer and it does not address the fundamental risk inherent to mobile code, which relates to mobile code behavior. This leaves the consumer vulnerable to damage due to malicious code (if the producer cannot be trusted) or faulty code (if the producer can be trusted). Indeed, if the code originated from an untrusted or unknown producer, then code-signing provides no support for safe execution of such code. On the other hand, code signing does not protect against bugs already present in the signed code. Patched or new versions of the code can be issued, but the loader (which verifies and loads the executable content and then transfers the execution control to the module) will still accept the old version, unless the newer version is installed over it. [12] proposes a method that employs an executable content loader and a short-lived configuration management file to address this software aging problem.

Proof-Carrying Code (PCC). The PCC approach [14] enables safe execution of code from untrusted sources by requiring a producer to furnish a proof regarding the safety of mobile code. Then the code consumer uses a proof validator to check, with certainty, that the proof is valid (i.e., it checks the correctness of this proof) and hence the foreign code is safe to execute. Proofs are automatically generated by a certifying compiler [15] by means of a static analysis of the producer code. The PCC approach is problematic for two main reasons [16]. A practical difficulty is that automatic proof generation for complex properties is still a daunting problem, making the PCC approach not suitable for real mobile applications. A more fundamental difficulty is that the approach is based on a unrealistic assumption: since the producer sends the safety proof together with the mobile code, the code producer should know all the security policies that are of

interest to consumers. This appears an impractical assumption since security may vary considerably across different consumers and their operating environments.

Model-Carrying Code. This approach is strongly inspired by PCC, sharing with it the idea that untrusted code is accompanied by additional information that aids in verifying its safety [17]. With MCC, this additional information takes the form of a model that captures the *security-relevant behavior* of code, rather than a proof. Models enable code producers to communicate the security needs of their code to the consumer. The code consumers can then check their policies against the model associated with untrusted code to determine if this code will violate their policy. Since MCC models are significantly simpler than programs, such checking can be fully automated. This model has been mainly proposed for bridging the gap between high-level policies and low-level binary code, enabling analyses which would otherwise be impractical (as for PCC).

For policy specification other languages can be used as well. For instance, temporal logic formulae are widely applied for this purpose [8]. Also there is a number of XML-based languages for specification of access control policies, such as XACML [13]. However, while these languages suit well for describing security policies, they are less convenient for formal specification of the whole system, and in our framework it is essential to cover both these aspects. Therefore we chose a FSA-based language, which is suitable for specification of contracts as well as policies. However, it is worth noting that there is a mapping from temporal logic to FSA, which enables translating policies written as logic formulae into our FSA-based language.

7 Conclusion

In this paper we have proposed the notion of *security-by-contract*, a mobile contract that an application carries with itself. The key idea of the approach is that a digital signature should not just certify the origin of the code but rather bind together the code with a contract. From this point of view, our framework essentially makes more concrete some ideas behind MCC. In particular, we use a high level specification language with features that simplify contract/policy matching and allow expressing realistic security policies. Also our matching algorithm is improved for efficiency as it is intended for use on such resource-critical devices as mobiles. For this reason we first perform easier checks of sufficient criteria before performing a complete check (see Alg. 4). Other optimizations are also discussed. These features also differentiate our approach from other frameworks for modeling resource contractualisation, such as [18].

The contributions of the paper are threefold. First, we have proposed the *security-by-contract* framework providing a description of the overall life-cycle of mobile code in this setting. Then we have described a tentative structure for a contractual language. Finally, we have proposed a number of algorithms for one of the key steps in the life-cycle process: the issue of *contract-policy matching*.

The main novelty of the proposed framework is that it would provide a semantics for digital signatures on mobile code thus being a step in the transition from trusted code to trustworthy code.

References

1. Building bug-free O-O software: An introduction to Design by Contract(TM). Availabe at http://archive.eiffel.com/doc/manuals/technology/contract/
2. Clarke, E.M., Grumberg, O., Peled, D.A.: Model Checking. MIT Press, Cambridge (2000)
3. Dill, D.L., Hu, A.J., Wong-Toi, H.: Checking for Language Inclusion Using Simulation Relations. In: Larsen, K.G., Skou, A. (eds.) CAV 1991. LNCS, vol. 575, pp. 329–341. Springer, Heidelberg (1992)
4. Etessami, K.: A hierarchy of polynomial-time computable simulations for automata. In: Brim, L., Jančar, P., Křetínský, M., Kucera, A. (eds.) CONCUR 2002. LNCS, vol. 2421, pp. 131–144. Springer, Heidelberg (2002)
5. Etessami, K., Wilke, T., Schuller, R.: Fair Simulation Relations, Parity Games, and State Space Reduction for Buchi Automata. In: Orejas, F., Spirakis, P.G., van Leeuwen, J. (eds.) ICALP 2001. LNCS, vol. 2076, pp. 694–707. Springer, Heidelberg (2001)
6. Gong, L.: Java Security: Present and Near Future. IEEE Micro 17(3), 14–19 (1997)
7. Gong, L., Ellison, G.: Inside Java(TM) 2 Platform Security: Architecture, API Design, and Implementation. Pearson Education (2003)
8. Havelund, K., Rosu, G.: Efficient monitoring of safety properties. Software Tools for Tech. Transfer (2004)
9. Henzinger, T., Kupferman, O., Rajamani, S.: Fair Simulation. In: Mazurkiewicz, A., Winkowski, J. (eds.) CONCUR 1997. LNCS, vol. 1243, pp. 273–287. Springer, Heidelberg (1997)
10. Jurdzinski, M.: Small Progress Measures for Solving Parity Games. In: Reichel, H., Tison, S. (eds.) STACS 2000. LNCS, vol. 1770, pp. 290–301. Springer, Heidelberg (2000)
11. Kesten, Y., Manna, Z., McGuire, H., Pnueli, A.: A decision algorithm for full propositional temporal logic. In: Courcoubetis, C. (ed.) CAV 1993. LNCS, vol. 697, pp. 97–109. Springer, Heidelberg (1993)
12. Michener, J.R., Acar, T.: Managing System and Active-Content Integrity. IEEE Computer 33(7), 108–110 (2000)
13. Moses, T.: eXtensible Access Control Markup Language (XACML) version 1.0. Technical report, OASIS (2003)
14. Necula, G.C.: Proof-Carrying Code. In: POPL '97: Proceedings of the 24th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, New York, NY, USA, 1997, pp. 106–119. ACM Press, New York (1997)
15. Necula, G.C., Lee, P.: The Design and Implementation of a Certifying Compiler. SIGPLAN Not. 39(4), 612–625 (2004)
16. Sekar, R., Ramakrishnan, C.R., Ramakrishnan, I.V., Smolka, S.A.: Model-Carrying Code (MCC): a New Paradigm for Mobile-Code Security. In: NSPW '01: Proceedings of the 2001 Workshop on New security paradigms, New York, NY, USA, 2001, pp. 23–30. ACM Press, New York (2001)
17. Sekar, R., Venkatakrishnan, V.N., Basu, S., Bhatkar, S., DuVarney, D.C.: Model-Carrying Code: a Practical Approach for Safe Execution of Untrusted Applications. ACM SIGOPS Operating Systems Review 37(5), 15–28 (2003)
18. Le Sommer, N.: Towards Dynamic Resource Contractualisation for Software Components. In: Emmerich, W., Wolf, A.L. (eds.) CD 2004. LNCS, vol. 3083, pp. 129–143. Springer, Heidelberg (2004)
19. Wallach, D.S., Felten, E.W.: Understanding Java Stack Inspection. In: IEEE Symposium on Security and Privacy (1998)
20. Zobel, A., Simoni, C., Piazza, D., Nuez, X., Rodriguez, D.: Business case and security requirements. Public Deliverable D5.1.1, EU Project S3MS (October 2006), Report available at http://www.s3ms.org

Applicability of Public Key Infrastructures in Wireless Sensor Networks

Rodrigo Roman and Cristina Alcaraz

Computer Science Department,
University of Malaga, Spain
{roman,alcaraz}@lcc.uma.es

Abstract. Wireless Sensor Networks (WSN) are becoming a key technology in the support of pervasive and ubiquitous services. The previous notion of "PKC is too expensive for WSN" has changed partially due to the existence of new hardware and software prototypes based on Elliptic Curve Cryptography and other PKC primitives. Then, it is necessary to analyze whether it is both feasible and convenient to have a Public Key Infrastructure for sensor networks that would allow the creation of PKC-based services like Digital Signatures.

Keywords: Wireless Sensor Networks, Public Key Cryptography, Public Key Infrastructure.

1 Introduction

Wireless Sensor Networks [1] can be considered as a key technology to support pervasive and ubiquitous services. They can be applied to a wide number of areas: such as farmland monitoring, smart office, detection of out-of-tolerance environmental conditions, emergency medical care, wearable smart uniforms, etc. However, these networks are quite difficult to protect, because every node becomes a potential point of logical and physical attack.

In this context, it would be extremely useful to have a cryptographic primitive such as Public Key Cryptography (PKC) in order to create services such as Digital Signatures. The use of PKC in sensor networks has been usually considered as "nearly impossible", but at present some studies [4] have started to consider the possibility of utilizing PKC in a highly-constrained networks. It is then the purpose of this paper to review the state of the art of PKC for sensor networks, and to analyze if it is both feasible and convenient to have a working Public Key Infrastructure in a sensor network environment.

The rest of this paper is organized as follows: In section 2 the architecture of a wireless sensor network is explained, alongside with how PKC could influence on solving some major security problems. In section 3, the major PKC primitives that could be applied to constrained environments such as sensor nodes are presented and studied. Finally, in section 4, there is a deep analysis of the applicability of Public Key Infrastructures to a sensor network environment, and in section 5, the conclusions are presented.

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 313–320, 2007.

2 Wireless Sensor Networks

A Wireless Sensor Network, as a whole, can be seen as the "skin" of a computer system, since it is able to provide any physical information of a certain region or element to any external system. The ability of measuring their environment is not the only benefit of these networks: thanks to the wireless capabilities and the limited computational power of their elements, they are easy to set up, are capable of self-configuring themselves, and are relatively inexpensive. The main elements of a sensor network are the sensor nodes and the base station - the "cells" of the system and its "brain".

Sensor nodes are small and inexpensive computers that have limited computational and wireless capabilities: a typical sensor node uses a microcontroller of 8Mhz with 4KB of RAM and 128KB of ROM, and incorporates a transceiver compliant to low-power, low duty standards such as IEEE 802.15.4. On the other hand, the base station is a powerful, trusted device that acts as an interface between the user of the network and the nodes. Regarding their internal configuration, the nodes of the network can group themselves into clusters where all the organizational decisions inside a cluster are made by a single entity called "cluster head" (hierarchical configuration), or all the nodes can participate in both the decision-making processes and the internal protocols (flat configuration).

In a sensor network, amongst other issues, it is extremely important to provide certain basic security mechanisms and protocols in order to avoid attacks from malicious adversaries [3]. It was recently when Public Key Cryptography (PKC) started to be considered as a viable solution for this purpose. Since, in most cases, a node does not know in advance who will be on its neighborhood, PKC can be used for both authenticating such nodes and for allowing the secure exchange of pairwise keys. Any procedure that requires the participation of the base station can also take advantage of these primitives. For instance, it is possible to securely distribute new code to the nodes of the network if it has been previously signed by the base station. Lastly, there are many other services that can effectively use PKC: authenticated broadcast, data source authentication in data aggregation, privilege delegation, etc.

3 Public Key Cryptography Primitives for Sensor Networks

3.1 Existing PKC Primitives

The computational requirements of PKC primitives are quite expensive in comparison with other cryptographic primitives, such as Symmetric Key Encryption (SKE). For instance, the most popular algorithm for public key encryption, RSA [5], is quite inefficient when implemented in sensor nodes. However, there exists other PKC approaches based on various mathematical problems that can be specially useful in highly-constrained environments. The first example is the Rabin signature algorithm [6], proposed by Michael Rabin in 1979. It is very similar to

RSA, but its main advantage is the speed of its encryption and signature verification operations, which are based on a simple squaring operation. A disadvantage is the signature size, though: a single signature requires 512 bits.

One of the most suitable asymmetric cryptography primitives for WSN, the Elliptic Curve Cryptography cryptosystem (ECC) [7], was discovered in 1985. ECC is based on algebraic concepts related with elliptic curves over finite fields $\mathbb{F}_p$ or $\mathbb{F}_{2^m}$. The ECC's security is based in computing the equation $a^b = c$ given a and c, known as the discrete logarithm problem, and the main ECC primitive operation is the scalar point multiplication. The major benefit of ECC is the size of its keys (160 bit against 1024 bit in RSA [10]) and its speed while optimized.

The asymmetric algorithm NTRUEncrypt [8], and its associated signature scheme NtruSign, is based on arithmetic operations in a polynomial ring $R = \mathbb{Z}(x)/((x^N - 1), q)$. Its security is based on the hardness of resolving the Shortest Vector Problem (SVP) and the Closest Vector Problem (CVP). NTRUEncrypt is much faster both for encrypting and for verification operations than RSA, since it uses simple polynomial multiplications. On the other hand, it also shares Rabin's scheme weakness: its signature requires 1169 bits.

The most recent asymmetric approach is the *multivariate public-key cryptosystem*, also known as $\mathcal{MQ}$-schemes [9]. Its security is centered in resolving $w = V^{-1}(z) = (\omega_1, ..., \omega_n) \in K^n$ given a quadratic polynomial map $V = (\gamma_1, ..., \gamma_m) : K^n \rightarrow K^m$. Its signature operations are extremely fast, but there is a significative storage cost for restricted environments due to the size of the keys in RAM. Concretely, it is necessary to book 879 bytes for the private key and 8680 bytes for the public key.

3.2 HW and SW Prototypes

A summary of the different HW prototypes for PKC in sensor networks can be seen in table 1. In 2005, Gautbatz et. al. proposed in [13] several PKC hardware implementations of Rabin's scheme, NtruEncrypt and ECC primitives. All of theses implementations work with an operational frecuency of 500KHz, and they were designed having the node's hardware limitations in mind. Other ECC prototypes for more powerful nodes were developed by Wolkerstorfer et. al. in [14] and Kumar and Paar in [15]. Finally, in 2006, Batina et. al. in [16] improved

Table 1. Summary of Hardware prototypes for PKC

	ECC				NTRU	Rabin	$\mathcal{MQ}$
	Wolkerstorfer	Kumar & Paar	Gaubatz	Batina	Gaubatz	Gaubatz	Yang
Gates	23000	12000	18720	12000	3000	17000	17000
Frequency	68.5MHz	13.5Mhz	500khz	500kHz	500kHz	500kHz	100kHz
Point Mult.	9.98ms	18ms	$\sim$ 400ms	115ms	—	—	—
Encryption	—	—	—	—	58ms	2.88ms	—
Decryption	—	—	—	—	117ms	1.089s	—
Signing	—	—	—	—	234ms	1.089s	44ms
Verifying	—	—	—	—	58ms	2.88ms	—

Table 2. Software implementations of ECC

	TinyECC		WMECC		TinyWMECC	
	Micaz	Telosb	Micaz	Telosb	Micaz	Telosb
Test - ROM size	28266	26048	57982	46156	29734	25774
Test - RAM size	2306	2327	1685	1657	1643	1599
ECC init	1.837s	-	1.809s	1.744s	1.809s	1.744s
ECDSA init	3.550s	5.225s	0s	0s	0s	0s
Pub. Key Gen.	1.788s	-	1.261s	1.425s	1.261s	1.425s
Signature	1.916s	4.361s	1.348s	1.498s	1.348s	1.498s
Verification	2.431s	5.448s	2.017s	2.207	2.019s	2.209s

the previous ECC implementations, and Yang et. al. in [17] proposed an implementation for Multivariate public key cryptosystem.

Regarding software implementations, as of 2007 the most important ECC libraries for sensor networks are TinyECC by Liu and Ning [12], and WMECC by Wang and Li [11]. These libraries work over the micaz and telosb motes in the "de-facto" standard Operative System for WSN, TinyOS, and their performance can be seen in table 2. Both libraries have different implementation approaches, although it is noteworthy that TinyECC has an improved SHA-1 function that allows it to have a reasonable code size compared with WMECC. Fortunately, taking advantage of the component capabilities of TinyOS and the optimized SHA-1 function in TinyECC, it was possible for us to improve the existing WMECC library by changing its SHA-1 function. This improvement, named TinyWMECC, completely solves the code size problem, and has been included recently into the main WMECC code branch.

4 Public Key Infrastructures in Sensor Networks

4.1 Adapting PKI for Sensor Networks

The use of PKC alone is not enough for protecting a WSN: it is necessary to have a Public Key Infrastructure that can be able to establish a trusted identity, amongst other things. The major components of a PKI, according to the PKIX model [2], are the following: the clients, which are the users of a PKI certificate; the Certification Authority (CA), which establishes identities and creates digital certificates; the Registration Authority (RA), which is responsible for the registration and initial authentication of the clients; and the Repository, which stores the certificates and the Certification Revocation Lists (CRLs). In order to provide the services of a PKI, such as initialization and certification, these components and their functionality must be mapped to the entities of a wireless sensor network.

It is not trivial to apply a PKI to a wireless sensor network, though. The architecture of these types of networks have several distinctive features regarding its initialization and maintenance. For example, all nodes have to be configured

by the base station in a secure environment before their final deployment in the network. Also, the architecture of the network is highly decentralized, where the autonomous sensor nodes collaborate towards a common goal, but all the critical information produced by the network must be sent to the Base Station.

Although a sensor network is highly decentralized by nature, it is easy to notice that there is a central system, the base station, that takes the role of initializing the nodes of the network and interacting with the data provided by all these nodes. Therefore, it is clear that the base station can be considered as the Certification Authority. It is the base station, then, the entity responsible for creating the digital certificates that associate the identity of a node with its public/private key pair. Moreover, the base station can also take the role of Registration Authority, since it is in charge of assigning the identity of all the nodes of the network before the deployment of the network. As a side note, the base station can also create the public/private key pair of a node, as it is not efficient for a node to create its own key, and the base station is trustworthy.

Although the base station may also act as the Certificate Repository, this is not practical for sensor networks. Since most sensor nodes need to route their information through other nodes in order to send information to the base station, and the costs of doing so are quite high in terms of energy and time, it is better to adopt a decentralized solution for retrieving the certificates. As a result, every node will have its own certificate, and will provide it to any neighbor that requests it. This exchange can be done in the first steps of the lifetime of the network.

In order to deploy a PKI, it is also obligatory to select an appropriate hierarchy model. Fortunately, in most cases the architecture of a sensor network is extremely simple: one base station that serve as an interface to hundreds or thousands of sensor nodes, which only know and can communicate with the nodes belonging to the same network. Therefore, it is safe to consider that a sensor network will use a simple hierarchical PKI architecture, with only one root CA.

The basic functionality of a PKI, that is, registration, initialization, key generation, certification, and certification retrieval, are performed in the following way: The base station creates the public/private key pair of a sensor node, assigns an unique identification to it, and creates the digital certificate that links that unique identification with its public key. Later, it initializes the contents of the sensor node (such as configuration data and internal programming), including the certificate of the node and the certificate of the root CA (i.e. the base station itself). Later, when a node retrieves the certificate of one of its neighbors, it will be able to check its validity using the root CA certificate.

4.2 Other PKI Functions in Sensor Networks

Thanks to the characteristics and peculiarities of the architecture of wireless sensor networks, it is possible to map the entities required by a PKI in the elements of a sensor network, providing as a result some of the basic functions of a PKI. However, there are still other PKI functions whose applicability must be discussed, such as Key Pair Recovery, Key Update, Cross Certification, and

Key Revocation. Some of these functions are not required for a sensor network context, whereas other functions could be important in certain scenarios.

For example, the issues of key archival and key pair recovery are simple to solve. Since the base station is considered as a fully trusted entity, all keys pairs can be stored inside it, or in another secure server that interacts with it for redundancy purposes. On the other hand, the issue of cross certification is a bit more complicated. For a typical sensor network, with only one base station that behaves as the root CA, it is not necessary to use cross-certificates. However, there are some scenarios where more than one base station can be able to control the network. Moreover, as seen in section 2, there are some hierarchical infrastructures where a set of "cluster heads" control a cluster of nodes.

The additional base stations can be static, also serving as an interface to the functionality of the network, or mobile, directly querying the nodes about their status. Mobile base stations can behave as any other node inside the network, except that they should have a short-lived certificate signed by the main base station, with enough privileges to influence over the nodes' operations. Regarding static base stations, there are usually only a few of them, thus it can be possible to simply preload their certificates, signed by the root CA, into all nodes. Finally, it is not necessary to consider a cluster head as a CA, since it has no need to either produce or sign any certificate of the other members of its cluster. As a conclusion, there is no need to use cross-certificates, even in these complex scenarios.

Regarding Key Revocation and Key Update, there may be some situations in which it is important to use these services. For example, if one node is subverted by an adversary but is discovered by the network, the base station may choose to revoke its certificate. Furthermore, the base station can introduce a new node into the network with a new certificate that replaces the malicious one. Updating the certificate of a certain node is an easy task, since the human administrator of the network has to physically obtain the node for putting inside the new certificate, alongside with the private key associated with it.

Alerting the nodes about the revocation of the previous certificate is not easy, though. It is prohibitive for the nodes to retrieve a Certificate Revocation List from the base station (pull model), since querying the base station is a time-consuming and energy-consuming process. A better solution would be to use a online revocation notification mechanism (push model), where the base station alerts the nodes of the network that a certain certificate has been revoked. Upon receiving this authenticated message, the nodes of the network can then request the public key of the node that had its certificate revoked. A malicious node will not be able to provide a valid certificate, whereas the certificate of a legitimate node will be accepted.

An aspect related to node revocation, and mentioned in mobile base stations, is the existence of a validity period inside all certificates. Nevertheless, for short-lived networks, the context of the application ("deployment") is more important than the expiration date. For instance, a short-lived network may measure the level of ambient noise in a certain area for a week or more (*Deployment A*),

but later the same nodes from that network can be reutilized in another area (*Deployment B*). It should be then more efficient to identify the deployment rather than the expiration date, and discard any certificate that belongs to a previous deployment (e.g. a node belonging to the *Deployment B* does not accept any certificate that was created during *Deployment A*). The root CA, then, has to assign new certificates to all the nodes before deploying them in a new area.

In long-lived networks, such as a network that monitors the overall conditions of a vineyard for an entire year, this notion of "deployments" may not be enough, since the nodes will be continuously monitoring an environment for a long period of time. Nevertheless, the expiration date of the certificates used in these networks should not allow a situation where the nodes are not able to use the PKI services. What expiration date should be chosen is unknown at present due to the lack of long-lived real-world deployments, but it is safe to assume that there is no danger in configuring the certificates of the network to have no expiration date. If there is no external influence, the network will function properly all its lifetime. And if there is any malicious influence, such as the destruction of the base station, the owner of the network can "reboot" the whole network, reconfiguring it and labelling it as a new "deployment".

5 Conclusions

From "Public-key cryptography is right out" to "Public-key is no big deal" [18], it is clear that there is a possibility to incorporate in the near future public key-based services such as Digital Signatures in wireless sensor networks. Therefore, as explained in this paper, the inclusion of a Public Key Infrastructure for sensor networks should be seriously considered. This is an immature area that is full of interesting research problems, like the coexistence of a PKI with other Public-key based schemes such as Homomorphic Encryption [19] and Identity-Based Cryptography [20].

References

1. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: Wireless sensor networks: a survey. Computer Networks: The International Journal of Computer and Telecommunications Networking 38(4), 393–422 (2002)
2. Public-Key Infrastructure (X.509) (pkix) Charter `http://www.ietf.org/html.charters/pkix-charter.html`
3. Walters, J.P., Liang, Z., Shi, W., Chaudhary, V.: Wireless Sensor Network Security: A Survey. In: Xiao, Y. (ed.) Security in Distributed, Grid, and Pervasive Computing, Auerbach Publications, CRC Press, Boca Raton, USA (2006)
4. Lopez, J.: Unleashing Public-Key Cryptography in Wireless Sensor Networks. Journal of Computer Security 14(5), 469–482 (2006)
5. Rivest, R., Shamir, A., Adleman, L.: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM 21(2), 120–126 (1978)

6. Rabin, M.O.: Digitalized Signatures and Public Key Functions as Intractable as Factorization. Technical Report MIT/LCS/TR-212, Massachusetts Institute of Technology (1979)
7. Blake, I., Seroussi, G., Smart, N.P.: Elliptic Curves in Cryptography. Cambridge University Press, Cambridge (2000)
8. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: a Ring based Public Key Cryptosystem. In: Buhler, J.P. (ed.) Algorithmic Number Theory. LNCS, vol. 1423, Springer, Heidelberg (1998)
9. Wolf, C., Preneel, B.: Taxonomy of Public Key Schemes Based on the Problem of Multivariate Quadratic Equations. Cryptology ePrint Archive, Report 2005/077
10. National Institute of Standards and Technology. Recommended Elliptic Curves for Federal Government Use (August 1999)
11. Wang, H., Li, Q.: Efficient Implementation of Public Key Cryptosystems on MICAz and TelosB Motes. Technical Report WM-CS-2006-07, College of William & Mary (October 2006)
12. Liu, A., Kampanakis, P., Ning, P.: TinyECC: Elliptic Curve Cryptography for Sensor Networks (Version 0.3) (February 2007) `http://discovery.csc.ncsu.edu/software/TinyECC/`
13. Gaubatz, G., Kaps, J.-P., Öztürk, E., Sunar, B.: State of the Art in Ultra-Low Power Public Key Cryptography for Wireless Sensor Networks. In: Proceedings of the 2nd IEEE International Workshop on Pervasive Computing and Communication Security (PerSec 2005), March 2005, Hawaii (USA) (2005)
14. Wolkerstorfer, J.: Scaling ECC Hardware to a Minimum. In: ECRYPT workshop - Cryptographic Advances in Secure Hardware - CRASH 2005, September 2005, Leuven (Belgium), Invited talk (2005)
15. Kumar, S., Paar, C.: Are standards compliant elliptic curve cryptosystems feasible on RFID? In: Proceedings of Workshop on RFID Security, July 2006, Graz (Austria) (2006)
16. Batina, L., Mentens, N., Sakiyama, K., Preneel, B., Verbauwhede, I.: Low-Cost Elliptic Curve Cryptography for Wireless Sensor Networks. In: Buttyán, L., Gligor, V., Westhoff, D. (eds.) ESAS 2006. LNCS, vol. 4357, Springer, Heidelberg (2006)
17. Yang, B.-Y., Cheng, C.-M., Chen, B.-R., Chen, J.-M.: Implementing Minimized Multivariate Public-Key Cryptosystems on Low-Resource Embedded Systems. In: Clark, J.A., Paige, R.F., Polack, F.A.C., Brooke, P.J. (eds.) SPC 2006. LNCS, vol. 3934, Springer, Heidelberg (2006)
18. Wagner, D.: The Conventional Wisdom About Sensor Network Security... Is Wrong. IEEE Security & Privacy 2005, invited panelist, Security in Ad-hoc and Sensor Networks (May 2005)
19. Myketun, E., Girao, J., Westhoff, D.: Public Key Based Cryptoschemes for Data Concealment in Wireless Sensor Networks. In: IEEE International Conference on Communications (ICC 2006), May 2006, Istanbul (Turkey) (2006)
20. Oliveira, L.B., Aranha, D., Morais, E., Daguano, F., Lopez, J., Dahab, R.: TinyTate: Identity-Based Encryption for Sensor Networks. Cryptology ePrint Archive, paper, 2007/020 `http://eprint.iacr.org/2007/020.pdf`

Spatial-Temporal Certification Framework and Extension of X.509 Attribute Certificate Framework and SAML Standard to Support Spatial-Temporal Certificates

Ana Isabel González-Tablas Ferreres, Benjamín Ramos Álvarez, and Arturo Ribagorda Garnacho

Computer Science Department, Universidad Carlos III de Madrid (Spain)
{aigonzal,benja1,arturo}@inf.uc3m.es

Abstract. The recent development of location-based services has originated a set of new security services that address their particular security problems. Spatial-temporal certification services are among these new services. They have as main goal the generation of evidences about an entity's spatial-temporal information and, in general, their life-cycle support. Currently there is still a lack of a general framework for spatial-temporal certification services. In this work it is presented such a framework and an extension of the X.509 attribute certificate framework and the SAML standard to represent spatial-temporal certificates.

Keywords: Spatial-temporal certification, X.509 AC, SAML.

1 Introduction

Last decade has witnessed the development and commercial deployment of location-based services. As some authors have pointed out, security is a major challenge in location-aware computing [PMP03]. Trust (authenticity and attestation) and privacy of location information stand out as main security requirements. Several mechanisms have been proposed to address trust of location information, mainly location authentication protocols and spatial-temporal attestation services, which include spatial-temporal certification services. A brief survey on mechanisms that address trust of location information can be found in [GKRR05]. A survey on mechanisms to protect location privacy in pervasive computing can be found in [GTH05].

This work focuses on spatial-temporal certification services. Although several authors have proposed spatial-temporal certification models and mechanisms, there is still a lack of a general framework that defines their goals, model and requirements. This work presents a basic spatial-temporal certification framework and an extension of the X.509 attribute certificate framework [ITU05] and the SAML standard [OAS05] to represent spatial-temporal certificates.

Related work. During the last decade some spatial-temporal certification models and mechanisms have been proposed in [ZKK01, Bus04], but none of them

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 321–329, 2007.

addresses the definition of a general spatial-temporal framework, instead they focus on specific application scenarios. Zugenmaier, Kreutzer and Kabatnik propose a model and a mechanism to provide location stamps for subscribers of the GSM mobile network. Bussard defines a type of privacy-enhancing certificates which he proposes to use, among other applications, in location- and time-stamping. Furthermore, neither Zugenmaier *et al.* nor Bussard do specify the structure of the spatial-temporal certificates using any of the current attribute certificate standards. Within IETF GEOPRIV WG, a location object format has been defined for carrying location information on the Internet [IET05]; digital signatures have been proposed to protect the integrity of this location object but it is not meant to be a proper certificate. Besides, GEOPRIV, in collaboration with the Open GIS Consortium [OGC06], is currently working on the definition of an interoperable geodetic representation worth of taking into account.

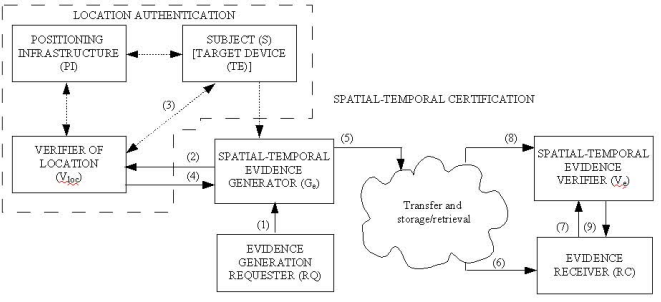
Paper outline. Section 2 presents the basic spatial-temporal certification framework and Section 3 the proposed extensions of the X.509 AC framework and the SAML standard. Section 4 presents the conclusions and future work that have been identified from this research.

2 Spatial-Temporal Certification Framework

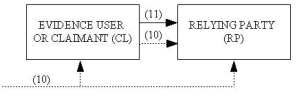
2.1 Goal and General Model

Similar to the definition for non-repudiation services in [ISO97], spatial-temporal certification services are defined as *those services that generate, collect, maintain, make available and validate evidences concerning the spatial-temporal information of an entity.* Spatial-temporal certification services must be provided, as well, within the context of a security policy. Among their applications stand access control to services or resources based on the location of the requester entity. For example, an on-line gambling site may require that, in order to grant access to the site, their clients must be located within some specific geographic area, or a shopping centre may desire to grant privileges depending on users' visiting history. Another application is found in non-repudiation scenarios, e.g., to provide non-repudiation and accountability in the tracking of entities and assets, such as mobile workers, vehicles, ships, hazardous materials or valuable assets. In addition, spatial-temporal evidences can be used to provide non-repudiation and accountability in location-based billing, as in automatic toll collection systems, for highway usage or for entrance in certain areas (high populated urban areas or preserved environmental zones such as biosphere reserves).

Several entities performing a number of roles may be involved in the provision of spatial-temporal certification services (see Figure 1). First, the *evidence generation requester* (RQ) is who requests the generation of a spatial-temporal evidence. The *spatial-temporal evidence generator* (G_e), is in charge of generating the evidences, and probably also collects, maintains and makes them available. The *evidence receiver* (RC) is who obtains the spatial-temporal evidence after it



(a) Phases: certificate generation; certificate transfer, storage and retrieval; and certificate verification



(b) Phase: certificate use

Fig. 1. General model of spatial-temporal certification services (dispute resolution phase is not shown)

has been issued. Evidence receivers should be able of verifying the evidence; the *spatial-temporal evidence verifier* (V_e) performs this task.

The entity which the evidences refer to is the *subject* (S) of the evidence, that is, the spatial-temporal information asserted in the evidence refer to the subject. The subject must be, at least, a positionable device; in addition, the subject of the evidence may also refer to the user controlling the target device. Subjects should be uniquely identifiable according to some identification scheme. It is assumed that the spatial-temporal information of the subject is securely verified or authenticated before the evidence is generated. The *verifier of location* (V_{loc}) performs this verification in collaboration with a positioning infrastructure (PI); this process is done by executing a location authentication protocol (see a description and an analysis of this kind of protocols in [GKRR05]). Note that considering the user controlling the target device as part of the subject would require to verify also the proximity of this particular user to that target device. It is assumed that G_e trusts V_{loc} and PI to obtain authentic spatial-temporal information about the subjects under certain security model.

The use of the evidence must be done within the context of the policy under which the evidence has been issued. The *evidence user* or *claimant* (CL) is who makes use of the spatial-temporal evidence to obtain some benefit (e.g., access to some resource or some tax payment). The *relying party* (RP) is the entity that provides some benefit to the claimant based on the evidence and maybe other auxiliar information. An entity may assume several of the presented roles. Some of them may be performed by trusted third parties (TTP) or trusted platform modules (TPM). Other TTPs may also be involved in the service provision.

Spatial-temporal certification services comprise mostly the same phases as non-repudiation services do [ISO97] (see Figure 1 for the numbers in brackets):

- *Certificate generation.* RQ asks G_e to generate a spatial-temporal certificate on certain subject S (step 1). G_e verifies the request and asks V_{loc} to locate subject S at that moment (step 2). V_{loc}, in collaboration with PI, verifies or authenticates the location of the subject (step 3) and returns to G_e this information (step 4). Finally, G_e generates the spatial-temporal certificate.
- *Certificate transfer, storage and retrieval.* G_e may store the evidence in a repository or transfer it to the receiver entity RC (steps 5 and 6). RC may also retrieve the certificate from the repository by himself afterwards.
- *Certificate verification.* In this phase, RC requests V_e to verify the evidence (step 7), who may need to retrieve the evidence or some additional information (step 8). The result of the verification is returned to RC (step 9).
- *Certificate use.* The evidence should have been transferred either to CL or to RP (step 10) and it is used by CL to obtain some benefit from RP (step 11). RP should verify the evidence before deciding to grant any benefit.
- *Dispute resolution.* If CL and RP do not agree regarding the benefit granting, both parties may leave the decision to an adjudicator, who will resolve the dispute taking into account the available evidences and the policies under which the evidences have been issued. This phase is not always needed.

2.2 Requirements

Establishment of trust on the evidence. Users of the evidence must be able to establish trust on the information certified in the evidence; therefore:

R1.1. Evidences must bind a subject to certain spatial-temporal information.
R1.2. It must be possible to verify who is the evidence author (data origin authentication) and that the evidence has not been modified (data integrity).
R1.3. It should be able to determine the source of the spatial-temporal information asserted in an evidence and the method used to obtain it.
R1.4. It must be possible to determine the temporal validity of an evidence.
R1.5. It should be able to determine the accuracy of the spatial-temporal information asserted in the evidence.
R1.6. It may be able to bind a particular subject with several spatial-temporal information tokens.

Spatial-temporal certification policy. As in non-repudiation services [ISO97], spatial-temporal certification services must be provided within the context of a particular spatial-temporal certification policy; therefore:

R2.1. It must be possible to determine the security policy under which a spatial-temporal evidence has been issued.

Protection of spatial-temporal information privacy. Location information is considered by many legislation corpus as personal data if it can be direct or

indirectly associated to an identified or identifiable entity [Dir02]. It is usually required that affected users consent the processing of their personal data after having been informed of the characteristics of its processing. Spatial-temporal evidences can be considered personal data as spatial-temporal information is bound to some particular subject. These requirements have been identified:

R3.1. Users must be able to control the circumstances under which the spatial-temporal information of their target devices is processed.
R3.2. As a consequence of requirement R3.1, confidentiality of spatial-temporal information must be guaranteed according to user's preferences.
R3.3. It should be able to determine which user's privacy preferences concern a certain spatial-temporal evidence.
R3.4. Spatial-temporal certificates may support use of privacy-enhancing certificates (such as the ones proposed in [Bus04]).
R3.5. Users may be able to specify bounds on spatial-temporal resolution to be used in the spatial-temporal information included in evidences.

Supporting functionalities

R4.1. Support functionalities must be provided to generate, store, retrieve, verify, show and delete spatial-temporal evidences.
R4.2. Support may be provided to automatize some spatial-temporal processes such as evidence generation.
R4.3. Support may be provided to automatize the enforcement of users' privacy preferences.

2.3 Mechanisms to Provide Spatial-Temporal Certification Services

Digital signatures are one of the most common mechanisms used to generate digital evidences. In particular they have been standardized as the mechanism to bind attributes to some entity in the ITU-T X.509 attribute certificate framework [ITU05] and the mechanism to protect the integrity and the issuer authentication of the assertions defined in the OASIS SAML standard [OAS05]. The verification of certificates based on digital signatures usually consists in verifying the signature's correctness and validity. Both X.509 attribute certificates (or its PKIX profile [IET02]) and SAML attribute assertions can be used as baseline to define spatial-temporal certificates.

To fulfill requirement blocks 1 and 2, a spatial-temporal certificate generated with a digital signature mechanism should contain the elements in the first column of Figure 2. Spatial-temporal attributes should allow the specification of certain spatial information and its resolution, certain temporal information and its resolution, the identifier of the spatial-temporal information provider and the method used to obtain the position and time asserted in the attribute. Instead of this classical certificate structure, some of the new privacy-enhancing certificate format recently proposed may be used (e.g., [Bus04]). Privacy-enhancing certificates address in an elegant way some of the requirements in block 3, but

other simpler solutions may also be used. For example, privacy can be provided with access control mechanisms or encryption of spatial-temporal attributes; these mechanisms can be complemented with the binding of the user's privacy preferences to the certificates. To provide support to this and future issues, it is advisable that spatial-temporal certificates allow arbitrary extensions, signed and unsigned. Finally, requirement block 4 should be provided by a certificate management infrastructure as the ones proposed for the X.509 AC framework, its PKIX profile or for SAML assertions.

3 Extension of X.509 Attribute Certificate Framework and SAML Standard

In this Section, two basic spatial-temporal certificate structures are defined to address requirement blocks 1 and 2. The X.509 attribute certificate framework and the SAML standard are used as base. Most of the elements composing a (classic) spatial-temporal certificate (as defined in Section 2.3) can find an equivalent element in the X.509 attribute certificate and the SAML attribute assertion structures. Figure 2 presents these pairs of equivalent elements and points out which ones have no equivalent (the cells are shown with grey background). To obtain a complete spatial-temporal certificate structure, X.509 attribute certificate and SAML assertion have to be extended in order to provide a solution for the greyed elements. Both certificate structures defined in this work limit subject's location representation to geodetic information.

Elements of spatial-temporal certificate	Correspondence to element	
	In X.509AttributeCertificate	In SAML<Assertion>
Version	Attribute.Certificate.version	Version
Serial number	AttributeCertificateInfo.serialNumber	ID
Time of generation	----------	IssueInstant
Issuer identifier	AttributeCertificateInfo.issuer	<Issuer>
Subject identifier	AttributeCertificateInfo.holder	<Subject>
Spatial-temporal attributes (sequence of)	AttributeCertificateInfo.Attributes (1)	<AttributeStatement> (1) (sequence of)
Validity period	AttributeCertificateInfo.attrCertValidityPeriod	<Conditions>.NotBefore <Conditions>.NotOnOrAfter
Spatial-temporal policy	----------	----------
Extensions	AttributeCertificateInfo.extensions	----------
Signature Information	AttributeCertificateInfo.signature	<ds:Signature>.<ds:SignatureInfo>
Signature Value	SIGNED{AttributeCertificateInfo}	<ds:Signature>.<ds:SignatureValue>

Fig. 2. Correspondence between elements of spatial-temporal certificate and elements of X.509 attribute certificate and SAML attribute assertion. Greyed elements do not have equivalent element. Those marked with (1) have an equivalent element but it must be extended to fulfil spatial-temporal certificate requirements.

Basic X.509-based spatial-temporal certificate. In this case, the time of generation may be expressed as a `timeGeneration` extension element defined as `GeneralizedTime`. The `certificatePolicies` field defined in X.509 public-key certificate framework can be used in this case to specify the spatial-temporal

certificate policy. X.509 attribute framework defines a general attribute certificate structure but application specific attributes must be defined as needed. A spatial-temporal attribute may be defined as shown in Figure 3(a). Note that a naïve spatial information element has been specified using a latitude-longitude-altitude tuple expressed in decimal degrees and meters but it should be desirable to use a generalized spatial representation (such a representation may be found in the ISO/TC 211 Geographic information/Geomatics standards).

Basic SAML-based spatial-temporal certificate. In this case, the spatial-temporal policy element may be defined as an XML attribute of a new SAML assertion (<`SpatialTemporalAssertion`>). Then, a new SAML attribute statement (<`SpatialTemporalStatement`>) is defined to contain the spatial-temporal attribute. Furthermore, four new SAML attributes are defined to express the location, the time, the spatial information source (provider and positioning method) and the temporal information source (see Figure 3(b)). Spatial and temporal elements have been defined using types from the GML 3.1.1 standard [OGC04]. Note that the elements belonging to the name space associated with the <`SpatialTemporalAssertion`> element are prefixed with '`sta:`'; types from GML are prefixed with '`gml:`' and types from SAML are prefixed with '`saml:`'. Future extensions of the SAML-based spatial-temporal certificate may be specified with new attribute statements (this approach may also be used in the X.509-based structure).

```
spatialTemporalAttribute ATTRIBUTE ::= {
  SYNTAX SpatialTemporalAttibuteSyntax
SpatialTemporalAttibuteSyntax ::= SEQUENCE {
  locationInfo LocationInfo,
  timeInfo TimeInfo,
  locationSource LocationSource OPTIONAL,
  timeSource TimeSource OPTIONAL,
LocationInfo ::= SEQUENCE {
  location Location,
  locationAccuracy LocationAccuracy OPTIONAL}
Location ::= SEQUENCE {
  latitude REAL,
  longitude REAL}
  altitude REAL}
LocationAccuracy ::= SEQUENCE {
  latitudeAccuracy REAL,
  longitudeAccuracy REAL}
  altitudeAccuracy REAL}
TimeInfo ::= SEQUENCE {
  time GeneralizedTime,
  timeAccuracy REAL OPTIONAL}
LocationSource ::= SEQUENCE {
  locationProvider ProviderName,
  locationMethod OBJECT IDENTIFIER}
TimeSource ::= SEQUENCE {
  timeProvider ProviderName,
  timeMethod OBJECT IDENTIFIER}
ProviderName ::= SEQUENCE {
  baseCertificateID IssuerSerial,
  entityName GeneralNames}
```

(a) X.509 AC extension

```
<xs:element name='LocationInfo' type='saml:AttributeType'/>
<xs:complexType name='LocationInfoType'>
<xs:sequence>
<xs:element ref='sta:LocationValue'/>
<xs:element ref='sta:LocationAccuracy' minOccurs='0'/>
</xs:sequence>
</xs:complexType>
<xs:element name='TimeInfo' type='saml:AttributeType'/>
<xs:complexType name='TimeInfoType'>
<xs:sequence>
<xs:element ref='sta:TimeValue'/>
<xs:element ref='sta:TimeAccuracy' minOccurs='0'/>
</xs:sequence>
</xs:complexType>
<xs:element name='LocationSource' type='saml:AttributeType'/>
<xs:element name='TimeSource' type='saml:AttributeType'/>
<xs:complexType name='InfoSourceType'>
<xs:sequence>
<xs:element ref='sta:Provider'/>
<xs:element ref='sta:Method' minOccurs='0'/>
</xs:sequence>
</xs:complexType>
<xs:element name='LocationValue' type='gml:PointPropertyType'/>
<xs:element name='LocationAccuracy'
type='gml:absoluteExternalPositionalAccuracyType'/>
<xs:element name='TimeValue' type='gml:TimeInstantPropertyType'/>
<xs:element name='TimeAccuracy' type='gml:TimeIntervalLengthType'/>
<xs:element name='Provider' type='saml:NameIDType'/>
<xs:element name='Method' type='xs:anyURI'/>
```

(b) SAML extension

Fig. 3. Main extension elements to support basic spatial-temporal certificates

4 Conclusions and Future Work

Frameworks are important instruments for the security research community. A framework for a security service usually defines its goals, its general provision

models and the requirements it should fulfill. Security frameworks may also describe specific mechanisms that allow the service provision. Recent development of location-based services has originated a set of new security services that address specific security problems for this context. Spatial-temporal certification services stand among these new services. In the last decade several authors have proposed spatial-temporal certification models and mechanisms [ZKK01, Bus04], but none of them addresses the definition of a general spatial-temporal framework. This work addresses the definition of such a spatial-temporal certification framework. A brief discussion on the mechanisms that may be used to provide spatial-temporal certification services is also presented. Authors do not have addressed an exhaustive definition of the framework, instead an initial but grounded baseline is presented in order to be used as discussion starting point. Furthermore, two specific spatial-temporal certificate formats are also proposed, based respectively on the X.509 AC framework and the SAML standard.

Lots of open issues remain unaddressed. A more general format of spatial information, able of representing different geographic places and semantic locations and taking into account interoperatibility issues, is needed. Both structures have to be extended to address requirement block 3, including in the framework privacy-enhancing certificates. Besides, integration with location authentication protocols should be properly analyzed. Finally, implementations of spatial-temporal certification service demonstrators must be developed. We are currently working on such an implementation, which will issue, at first, SAML-based certificates; privacy requirements are being addressed with an access control system based on generalized role-based access control model [MA01].

Acknowledgment

The authors would like to thank the referees. Authors are supported by "Dirección General de Investigación del M.E.C." under contract SEG2004-02604.

References

[Bus04] Bussard, L.: Trust Establishment Protocols for Communicating Devices. PhD thesis, Institut Eurécom, Télécom Paris (2004)

[Dir02] Directive 2002/58/EC of the European Parliament and of the Council of 12 July 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector (July 2002)

[GKRR05] González-Tablas, A.I., Kursawe, K., Ramos, B., Ribagorda, A.: Survey on location authentication protocols and spatial-temporal attestation services. In: Proc. of IFIP Intl. Symposium on Network-Centric Ubiquitous Systems (2005)

[GTH05] Görlach, A., Terpstra, W.W., Heinemann, A.: Survey on location privacy in pervasive computing. In: Proc. of the Workshop on Privacy, Security and Trust within the Context of Pervasive Computing, Kluwer, Dordrecht (2005)

[IET02] IETF (Internet Engineering Task Force). An Internet Attribute Certificate Profile for Authorization (RFC 3281) (2002)

[IET05] IETF (Internet Engineering Task Force). A Presence-Based GEOPRIV Location Object Format (RFC 4119) (2005)

[ISO97] ISO/IEC. ISO/IEC 10181-4. Information technology - OSI - Security frameworks in open systems - Part 4: Non-repudiation framework (1997)

[ITU05] ITU-T. RECOMMENDATION X.509 - The Directory: Public-key and attribute certificate frameworks (2005)

[MA01] Moyer, M.J., Ahamad, M.: Generalized role-based access control. In: Proc. of Intl.Conf. on Distributed Computing Systems, IEEE Computer Society Press, Los Alamitos (2001)

[OAS05] OASIS. Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) Version 2.0. OASIS Standard (2005)

[OGC04] OGC (Open Geospatial Consortium Inc.). OGC 03-105r1: OpenGIS Geography Markup Language (GML) Implementation Specification (February 2004)

[OGC06] OGC. OGC 06-142: GML 3.1.1 PIDF-LO Shape Application Schema for use by the Internet Engineering Task Force (IETF) (December 2006)

[PMP03] Patterson, C.A., Muntz, R.R., Pancake, C.M.: Challenges in location-aware computing. IEEE PervasiveComputing 2(2), 80–89 (2003)

[ZKK01] Zugenmaier, A., Kreutzer, M., Kabatnik, M.: Enhancing applications with approved location stamps. In: Proc. of IEEE Intelligent Network Wks. (2001)

Electronic Payment Scheme Using Identity-Based Cryptography

Son Thanh Nguyen and Chunming Rong

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

Abstract. Online electronic wallet with decentralized credential keepers is an architecture allowing users to leave most of the content of his electronic wallet at the security of his residential electronic keeper, while traveling with his mobile phone. This paper proposed a new security scheme for mobile payment using such architecture. The approach differs from the previous work in that it uses identity-based encryption for securing payment between the payer and payee, which takes full advantage of public-key cryptography while simplifies the authenticity requirements of the public keys.

Keywords: Identity-based cryptography, electronic wallet.

1 Introduction

The global mobile market has increased dramatically in recent years thanks to the technology development, network infrastructure availability and good tariff policy. According to OECD [14], the ratio of mobile phones per individual is close to one hundred percent in many European countries.

Mobile phones become indispensable devices with many people since they play as communication, entertainment and even business-assistant devices. The service providers, in the meantime, offer more value added services. For example, GSM service providers have launched a pilot program to enable global money transfer using mobile phones [7]. The program can even support transactions for people not having bank accounts. The near future use of one's mobile phone as a special wallet to pay bills at the restaurant, to buy tickets at the train station, or to do shopping is possible.

In [12], the authors proposed an electronic wallet architecture (e-wallet) using mobile phones as payment devices, which enables users using their mobile phones to access different kinds of credentials required for specific tasks (payment information, authentication information etc).

Our paper revisits the security issues in [12] by using identity-based encryption for securing the wireless transmission between the user's mobile phone and the merchant.

The rest of the paper is organized as follow. Section 2 reviews previous works, including the proposed architecture for online e-wallet system with decentralized

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 330–337, 2007.

credential keepers. The previous related works about identity-based encryption is also presented here. Section 3 is our solution using identity-based encryption for securing the mobile payment. Finally, section 4 draws conclusions.

2 Previous Works

2.1 Existing Proposals for E-Wallet

The electronic wallet concept was introduced in Chaum-Pedersen's work [3] and subsequently was revisited in the CAFE project [1], which developed the concept and prototype for electronic payments via short-range radio links or over the Internet. However, the payment concept in the project did not integrate with mobile communication technology. In addition, the project, and other proposals alike, faced with a problem of multi-issuers [12] in which, smart cards from different service providers are not compatible and their information cannot be shared with one another.

To ease the burden of bringing multiple smart cards and remembering multiple credentials, authors in [12] proposed a model in which all the credentials are kept away from the card's owner and can be securely accessed with using his mobile phone through GPRS links.

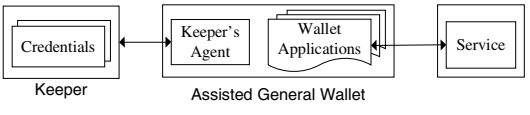


Fig. 1. The online wallet general architecture [12]

Fig.1 represents the general architecture for online electronic wallet. In this model, the keeper is a place where the owner keeps his electronics credentials. This is normally one or more servers with attached card readers and is located in a safe place.

The assisted general wallet is the personal device (e.g. mobile phone) with internet connection (GPRS) plus additional software to enable credential retrieval. When a person needs to do a transaction, (e.g. with a merchant, here denoted by "Service"), he will contact his keeper via network to obtain his credentials (e.g. credit cards information). Having contacted with and authenticated by the keeper, the appropriate credentials are securely sent to his mobile phone through internet connection.

Fig.2 shows an example of a payment system using online wallet derived from [12]. In this figure, the buyer is denoted *b* and is represented by a mobile phone *p*. His counterparts are the merchant *m*, which he will pay for goods, and the trusted server *s*, which he will contact for his appropriate credentials. The buyer *b* owns a mobile phone *p* with a private key SK_p. He also holds the trusted server's public key PK_s. Similarly, the trusted server *s* has its own private key SK_s as well as the mobile's public key PK_p. In addition, the trusted server connects to a database of the buyer's credentials. Since the buyer *b* uses the mobile *p* to do shopping and payment, we use terms mobile phone *p* and buyer *b* interchangeably.

The working principle in Fig.2 is as follow:

1. The buyer *b*, using the mobile phone *p*, connects to the merchant *m* via short-range radio communication (e.g. Bluetooth) to request shopping payment.
2. The merchant *m*, once accepts the shopping request, sends the mobile phone *p* its public key information PK_m and a bill. The merchant *m* also sends the mobile *p* a payment method agreed by both parties as well as its corresponding ID (e.g. account number)
3. The mobile phone *p* checks items and the paid amount, encrypts the information and sends to the trusted server *s* through a secure channel, using public key cryptography. Information sent to the trusted server includes the buyer's access credential, the session number between the buyer and merchant, the amount of paid money, the payment method and the appropriate merchant's ID information.
4. The trusted server *s* obtains such information, decrypts it and locates appropriate information from its database and plug-in modules, which hold the buyer payment credentials (e.g. credit card number).
5. The trusted server sends the required information to buyer *b* (e.g. credit card number, expiration date etc.) The information sent is encrypted with the buyer's public key PK_p.
6. The buyer decrypts and receives the required information.
7. The buyer combines such information with the other information from the merchant, encrypts using the merchant's public key PK_m and sends to the merchant.
8. The merchant clears the payment by sending the information to his appropriate financial service provider. Using the buyer's ID, the financial service provider locates related information of the buyer and informs the merchant if the transaction successes.

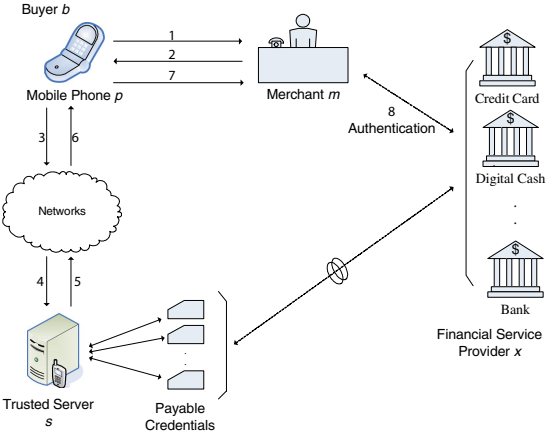


Fig. 2. The transactions of payment system using an online e-wallet

There have been several works related to this model. While the work in [6] proposed identification protocol, the works in [8] and [16] proposed the software architecture for online electronic wallet.

In this paper, we also refer to this model but approach in a different method. While in their original work, the authors exemplified either Bluetooth Security Mode 3 or public-key cryptography to secure the link between the mobile device and the service, we propose to use identity-based encryption to simplify the secure connection between the buyer *b* (wallet) and the merchant *m* (service).

2.2 Identity-Based Encryption

Identity-based cryptography was first proposed by Shamir [15] and then was made complete solutions by Boneh and Franklin [2] and Cock [4].

Similar to public-key cryptography, which uses a publicly known key for encryption and the other related secret key for decryption, identity-based cryptography uses uniquely known identification of an entity as its public key. In this model, there is a Private Key Generator (PKG), which plays a role of trust center for the users. Instead of signing the certificate for each user's public key, the PKG issues a public parameter and broadcasts to all interested parties (i.e. users). Each user, once needs to start a communication, derives the encryption key from the identity information of its counterpart and the public parameter. The recipient uses his appropriate secret key, created by the PKG, to decrypt the encrypted message.

Since a user's identity information is implicitly known by all other parties, the use of identity-based cryptography eases the burden of public-key authenticity associated with public-key cryptography while still takes full advantages of the public-key cryptography such as digital signature and non-repudiation. Another advantage of identity-based cryptography is that the receiver of an encrypted message does not need to have the decryption key in advance.

Fig.3 represents the identity-based encryption and signature scheme. The figure is self-explanatory.

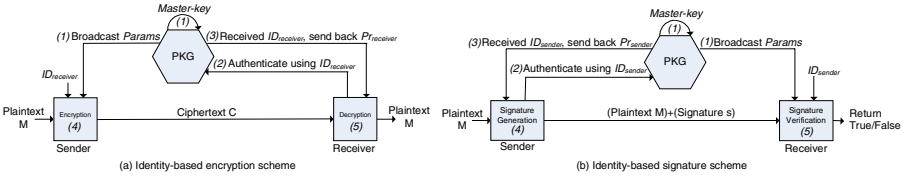


Fig. 3. Identity-based encryption and signature schemes

Since the invention of identity-based cryptography, there have been many related works in both theory and application. There are several application products using identity-based encryption to date. Voltage Security [17] has developed a solution for email security with identity-based encryption. A user's email is used as the public key to encrypt emails destined to him. Since a user's email is unique and publicly known (or easily verified), it can be used as the user's public key without the need of a certificate to prove its authority. Another work using identity-based encryption to

secure email is by HP Trusted System Laboratory [13]. In this work, the authors developed software plug-in to secure email using role-based identity-based encryption, which allows a group of users, sharing the same roles, to decrypt and read emails. The encryption keys here are the role descriptions.

In smart card market, Gemplus [5] (now Gemalto) proposed a solution to apply identity-based encryption for securing smart card messages. The encryption keys here can be the recipient's name, phone number or email address. Other related works involving securing SMS using identity-based encryption are from Hwu ([9], [10]), in which, SMS is sent securely to a mobile user using his mobile phone number as the encryption key.

With its certificateless characteristic, identity-based encryption is simple and very convenient for the end-users. It is, therefore, a promising technology for security applications.

3 An Identity-Based Encryption for Online E-Wallet System

In this section, we use identity-based encryption for securing the electronic payment between the mobile user *b* and merchant *m* as described in the Fig.2. In the original paper, the authors claimed to use public-key cryptography for securing transactions. Our proposal uses identity-based encryption for simplicity and performance. Simplicity results from the use of user's identity as the encryption key. Better performance results from the use of elliptic curve cryptography [11], which is faster and more efficient with the same level of security.

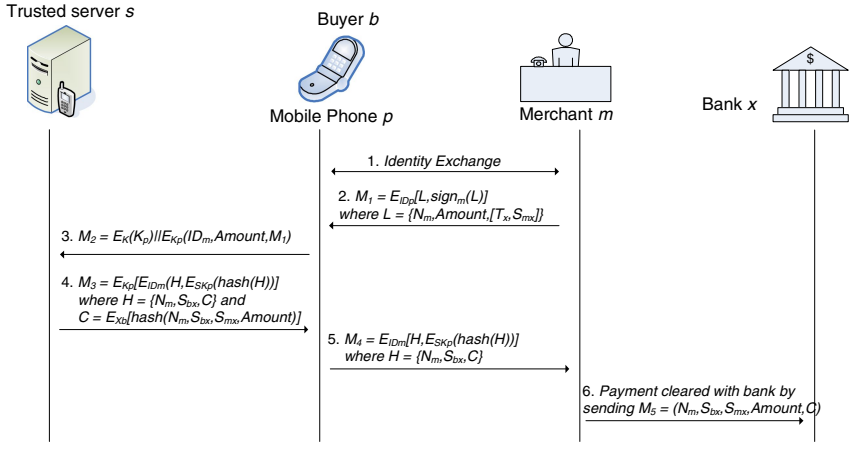


Fig. 4. Transactions in mobile payment scenario

Fig.4 represents an online e-wallet scheme using identity-based encryption. In this scenario, the payment is made between the buyer *b* (represented by his mobile phone *p*) and the merchant *m*, with the assistance of the trusted server *s* and bank *x*.

The involving parties use their own identification as the public key. The identity information of the buyer and the merchant are ID_p and ID_m respectively, where ID_p might be the mobile phone number. Their decryption keys are SK_p and SK_m. The trusted server s is built personally by the buyer. It shares a symmetric master key K with the user's mobile phone.

In order for the buyer and the merchant to use identity-based encryption, they need to agree on public parameters. This requirement can be fulfilled by registering to the same service provider or the alliance of service providers who can exchange their domain public parameters. For the illustration below, we assume they use the same public parameter, *Params*.

The transactions in Fig.4 are presented as follow:

1. The buyer b, using his mobile phone p, exchanges the identity information with the merchant m. The buyer then has the merchant's ID, ID_m, and vice versa. The ID exchange can be verified directly since they meet in person.
2. The merchant sends a billed list to the buyer including the amount to be paid, *Amount*, the payment method preferred by the buyer, T_x, the merchant's appropriate payment information, S_{mx} (account number), and a random transaction number, N_m. The billed list L is in the form $L = \{N_m, Amount, [T_x, S_{mx}]\}$

   The merchant signs this information using his secret key, $sign_m(L)=E_{SKm}(hash(L))$, encrypts the message plus the signature with ID_p and sends to the buyer. The information sent from the merchant is $M_1 = E_{IDp}[L, sign_m(L)]$
3. The buyer might need to check for the integrity of the information from the merchant. However, in our proposal, we move that task to the server bearing in mind that the mobile phone has limited computing power. The buyer then forms a message $D = \{ID_m, Amount, M_1\}$, where *Amount* is the paid amount added for confirmation. The ID_m is sent to the server so that the server can check the integrity of message from the merchant, and then encrypts the message back for the merchant, thus prevent the buyer's mobile phone from doing another resource-intensive encryption task. We also assume that the server holds the *Params* in advance, through the buyer's setting, so that the buyer does not need to send the *Params* every time he contacts the server.

   The buyer also generates a random session key K_p to encrypt the message D and uses the shared master key K to encrypt the session key. The buyer then sends both the encrypted message and the encrypted session key to the server. Since the session key is quite short and changes over time, cryptanalysis is difficult.

   The message $M_2 = E_K(K_p) || E_{Kp}(ID_m, Amount, M_1)$ is sent to the trusted server via communication link. Here the $E_K()$ and $E_{Kp}()$ are the symmetric encryption using K and K_p as the keys, respectively.
4. Having received M_2, the trusted server first decrypts to find the session key, and then uses the session key to decrypt the rest of the message. The trusted server also checks the message integrity from the merchant by looking at his signature and informs the buyer if violation appears. Once all the verifications success, the trusted server locates the user payment information from the plug-in modules (e.g. card readers) and obtains user account information S_{bx} for the payment type T_x. S_{bx} is uniquely known to the financial service provider x, which allows x to locate user

b in its system (e.g. account number). The trusted server then computes payment credential as $C = E_{Xb}(hash(N_m, S_{bx}, S_{mx}, Amount))$

where X_b is the user's master credential at the service provider *x* (e.g. credit card number) and $E_{Xb}()$ plays as the user signature for that payment. This payment credential works like a digital cheque where the owner indicates the payer, the payee, and the amount to be paid and finally signs the cheque with his signature. The trusted server forms the message $H = \{N_m, S_{bx}, C\}$ to be delivered to the merchant. The trusted server then signs *H* with S_{Kp}, encrypts with ID_m and encrypts again with the symmetric session key shared with the buyer, and sends the information to the buyer's mobile phone as message M_3. In this proposal, we use the same session key for both directions. Alternatively, there might be different keys for different directions.

$M_3 = E_{Kp}[E_{IDm}(H, E_{SKp}(hash(H)))]$

5. The mobile phone receives and decrypts M_3 using its decryption key K_p, then forwards the decrypted message to the merchant as M_4.

$M_4 = E_{IDm}(H, E_{SKp}(hash(H)))$

6. The merchant receives M_4, decrypts it using the merchant decryption key SK_m, checks for the integrity of the message using ID_p, and clears the payment by sending $M_5 = (N_m, S_{bx}, S_{mx}, Amount, C)$ to the financial service provider *x*. The service provider will check with the buyer's bank for money transfer and informs the merchant if the transaction is successful. Only when received the payment success from the appropriate financial service provider, does the merchant send out the purchased products.

Security and Performance Analysis

In this model, we propose to use identity-based encryption for securing transactions in the link between the merchant and the buyer to simplify the system deployment. The link between the buyer and his trusted server is secured using symmetric encryption. This link can also use the identity-based encryption. However, we recommend using the symmetric encryption for better performance. It is also easy to set up since the trusted server is personally built and under control of the user.

When the merchant and buyer exchange their identity information, the information may be exposed to man-in-the-middle attack. The attacker can replace the merchant identity information with his and obtains the user payment information subsequently. However, this situation is not likely to happen since the contact between the buyer and the merchant is in short range and thus, the chance that someone disrupts the transaction is small. In addition, to provide better security, the identity information of the merchant can be made public to all the buyers (e.g. publicly posted on the cashier counter) and the buyer can confirm this information on his mobile screen on received.

A variation of this payment model is using identity-based encryption for securing SMS ([9], [10]). This case is different from the one above in that there is no trusted server, the mobile user can retrieve credentials from his financial service providers using secure SMS. In this scenario, the mobile user contacts different financial service providers for appropriate credentials. However, there is no need to build up and maintain a trusted server.

4 Conclusion

This paper proposed a scheme to secure payment for online electronic wallet by using identity-based encryption. The proposal takes full advantages of the decentralized credential keeper architecture by solving the multi-issuer problem of hardware credentials, while reduce the complexity of such system by using identity-based cryptography. In this proposal, we use a mixed scheme with identity-based cryptography in one link and symmetric key cryptography in the other link to get the optimal performance.

References

1. Jean-Paul, B., et al.: The ESPRIT project CAFE – High Security Digital Payment System. In: Gollmann, D. (ed.) Computer Security - ESORICS 94. LNCS, vol. 875, pp. 217–230. Springer, Heidelberg (1994)
2. Boneh, D., Franklin, M.: Identity-based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
3. Chaum., D., Pedersen, T.: Wallet Databases with Observers. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 89–105. Springer, Heidelberg (1993)
4. Cocks, C.: An Identity-based Encryption Scheme Based on Quadratic Residues. In: Proceeding of 8th IMA International Conference on Cryptography and Coding, pp. 26–28 (2001)
5. Gemplus. http://www.gemplus.com
6. Gjerde, M., et al.: Decentralized Credentials. In: Proceedings of the Seventh Nordic Workshop on Secure IT Systems, NORDSEC, pp. 151–159 (2003)
7. GSM Association. www.gsmworld.com
8. Hernandez, R., Mjølsnes, S.F.: E-wallet Software Architecture with Decentralized Credentials, Norsk Informatikkonferanse, NIK (2003)
9. Hwu, J.-S., et al.: An Efficient Identity-based Cryptosystem for End-to-End Mobile Security. IEEE Transactions on Wireless Communications 5(9), 2586–2593 (2006)
10. Hwu, J.-S, et al.: End-to-End Security Mechanisms for SMS. International Journal of Security and Networks 1(3/4), 177–183 (2006)
11. Menezes, A.J.: Elliptic Curve Public Key Cryptosystems. Springer, Heidelberg (1993)
12. Mjølsnes, S.F, Rong, C.: On-Line E-Wallet System with Decentralized Credential Keepers. Mobile Networks and Applications 8, 87–99 (2003)
13. Mont, M.C, et al.: A Flexible Role-based Secure Messaging Service: Exploiting IBE Technology in a Health Care Trial. In: Mařík, V., Štěpánková, O., Retschitzegger, W. (eds.) DEXA 2003. LNCS, vol. 2736, pp. 432–437. Springer, Heidelberg (2003)
14. Organization for Economic Co-operation and Development. OECD Key ICT Indicators (2006) www.oecd.org
15. Shamir, A.: Identity-based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
16. Steinsholt, V.: Software Architecture for On-line Electronic Wallet, Master Thesis, Department of Telematics, Norwegian University of Science and Technology (2003)
17. Voltage Security. http://www.voltage.com/

Undeniable Mobile Billing Schemes*

Shiqun Li[1,2], Guilin Wang[2], Jianying Zhou[2], and Kefei Chen[1]

[1] Dep. of Computer Science and Engineering, Shanghai Jiaotong University, Shanghai 200240, China
[2] Institute for Infocomm Research, 21 Heng Mui Keng Terrace, Singapore 119613

Abstract. An undeniable mobile billing system allows a mobile network service provider to bill its subscribers with trustworthy evidences. Chen, Jan and Chen proposed such a billing system by introducing a trusted third party – Observer and exploiting a hash chain mechanism. In their system, the Observer provides call time evidence to both cellular carriers and subscribers for billing. In this paper, we first identify some vulnerabilities in their mobile billing system. Then, we propose an undeniable billing scheme based on a proper combination of digital signature and hash chain mechanism. The proposed scheme can achieve authentication, non-repudiation, and fairness, which are desirable security requirements for an undeniable mobile billing system.

1 Introduction

In the traditional GSM billing system, both the billing management and the billing information are processed by the Mobile Network Service Provider (MNSP) alone. From the subscribers' point of view, the above method may be not a good solution. Therefore, Chen et al. [3] proposed a mobile billing scheme (CJC scheme, for short) to provide undeniable billing evidences for call services in GSM. They introduced a TTP – Observer and used hash chain to provide billing information. The Observer is in charge of authentication and evidence provision.

In this paper, we first identify some vulnerabilities in the CJC system. Then we propose a new undeniable billing scheme, which is based on a proper combination of digital signature and a hash chain mechanism. It is very lightweight and suitable for the GSM mobile phone users.

The rest of the paper is organized as follows. Section 2 briefly introduces existing mobile billing systems. Section 3 reviews the CJC scheme and analyzes its security. Section 4 presents the proposed mobile billing systems which is based on hash chain technique and digital signatures. Section 5 evaluates the proposed scheme in aspects of security and efficiency. Section 6 draws a conclusion.

* Project supported by the National Nature Science Foundation of China key project(No.90104005) and Specialized Research Fund for the Doctoral Program of Higher Education(No. 20050248043). The primary author's work was done during his attachment to the Institute for Infocomm Research under its sponsorship.

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 338–345, 2007.

2 Mobile Billing Systems

To provide undeniable evidences for mobile network services, several schemes were proposed. The undeniable billing system in mobile communication [6] proposed an efficient solution to undeniable billing when a mobile user roams into foreign networks. This scheme adopted public key cryptographic algorithm to provide authentication and non-repudiation evidences, which is complicated for the current GSM mobile terminals. The Secure Billing for Mobile Information Services [2,4], provided a secure billing scheme for value-added information services using micropayment mechanism. It also requires public key operations for the mobile terminal which is applicable for UMTS mobile users but not the current GSM mobile users.

The CJC scheme [3] introduced an Observer as the TTP and used hash chain mechanism to provide billing information. It is a very efficient for mobile users, since the MSU is not required to perform any asymmetric cryptographic operation. However, our analysis shows that the CJC mobile billing system has some vulnerabilities so that it is not applicable in practice.

Our main purpose in this paper is to propose a new mobile billing scheme such that it is secure and as efficient as the CJC scheme. That is, we do not require the user's MSU do any public key operation (so our work is different from [2,4,6]). On the other hand, as in [6] we also employ the hash chain technique to determine the duration of a call service.

3 CJC Scheme and Its Vulnerabilities

3.1 Review of the CJC Scheme

The CJC mobile billing system [3] is illustrated in Fig. 1. As shown in Fig. 1, the Observer acts as the agent of a subscriber's MSU and shares a hash chain with it. To generate the bill evidence for a call, the MSU will first be authenticated by the MNSP and the Observer. Then the MNSP and the Observer sign the start time and end time of a valid call. Thus, by exploiting the hash chain technique and digital signature mechanism, both the MNSP and the subscriber cannot forge or deny the valid billing records. Note that here the Observer acts as a TTP and is in charge of providing call evidences to both the mobile subscriber and the MNSP. For more details about the CJC scheme, please refer to [3].

The authors claimed that their system satisfies the requirements of a fair mobile billing system. However, our analysis below will show that the CJC scheme cannot provide practicability and non-repudiation as supposed.

3.2 Vulnerabilities in the CJC Scheme

In this part, we show some vulnerabilities in the CJC mobile billing scheme [3]. Some of them are security flaws, and others are about implementation weaknesses.

First of all, the CJC scheme is *not fair* for both the MNSP and the mobile users. We now show two attacks on the fairness of the CJC scheme.

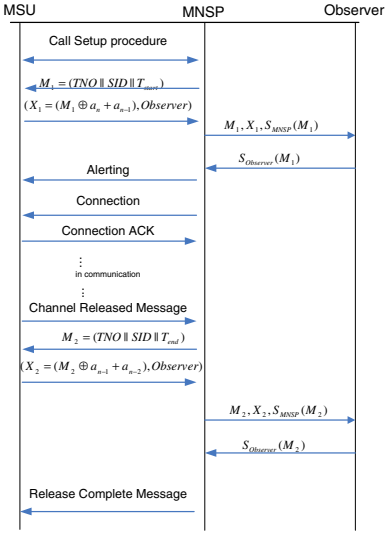


Fig. 1. The CJC Mobile Billing System

– **Attack 1.** In the CJC scheme, if the call is set up successfully but terminated abnormally later, then the MNSP cannot get the signature on T_{end}. In practice, a call may be terminated abnormally because of power failure, operation error or the caller's deliberate cheating (e.g., shutting down the device or unplugging the battery suddenly).

According to the specification in [3], if such abnormal interruptions in Attack 1 happen, the MNSP and the user's MSU cannot exchange M_2) and X_2. However, without proper X_2 the MNSP cannot get $S_{Observer}(M_2)$ from the Observer. So, the MNSP only has $S_{MNSP}(M_1)$ but does not have $S_{MNSP}(M_2)$. Consequently, with just one piece of non-repudiation evidence the MNSP cannot charge the users properly according to the CJC scheme. Moreover, if a called party does not pick up or deny the caller's call request, the connection would not be set up. According to Fig. 1, however, the MNSP and the Observer already calculated and exchanged $S_{MNSP}(M_1)$ and $S_{Observer}(M_1)$, although the *Alerting*, *Connection* and *Connection ACK* messages are not exchanged between the MSU and the MNSP.

Naturally, a subscriber should pay in the first case but needs not to pay in the second case. However, the Observer cannot tell which of those two cases occurred. So the MSU can deny a successful call if an abnormal termination happens. Thus, the CJC mobile billing system is unfair for the MNSP.

On the other hand, the following Attack 2 shows that the CJC scheme is nor fair for the mobile users, since the MNSP can maliciously overcharge them.

– **Attack 2.** At some time T_{start}, a mobile user wants to make a call, so proper messages M_1 and X_1 are exchanged between the user's MSU and

the MNSP. Then, the MNSP gets $S_{Observer}(M_1)$ from the Observer but it tells the user's MSU that this call cannot be set up due to some reason. In this scenario, the user may wait and re-call at T'_{start}. Now, the MNSP uses the same transaction number TNO used in M_1 to generate M'_1, i.e., $M'_1 = (TNO \| SID \| T'_{start})$. Upon receiving X'_1 from the user's MSU, the MNSP directly makes a connection for this MSU without contacting the Observer. Once the MSU ends its call, the MNSP can properly get $S_{Observer}(M_2)$ from the Observer, where $M_2 = (TNO \| SID \| T_{end})$. Therefore, the MNSP can charge the user on this call over the time interval of $T_{end} - T_{start}$, instead of the correct one $T_{end} - T'_{start}$.

The second problem of the CJC scheme is about *synchronization*. It employs the hash chain to realize the mutual authentication between the user's MSU and the Observer. However, the authors of do not provide how to maintain the synchronization of the hash chain between the MSU and the Observer. If a connection is terminated due to any abnormal reason, the MSU and the Observer may lose synchronization of the current state of hash value a_i, and then it would be impossible for the Observer to authenticate later valid calls.

4 The Proposed Scheme

In this section, we propose a secure undeniable mobile billing scheme that satisfies the security and practicability requirements described in Section 1. In our new scheme, the MNSP and the Observer sign the start time of a call. Those signatures serve the first evidence for the non-repudiation of billing. Then, the MSU periodically releases chained hash values during the call. Finally, the MNSP retains the last chained hash value from the MSU as the second non-repudiation evidences for billing.

In the proposed scheme, we assume that the MSU and the Observer share a secret key K_{MO} in advance. A keyed hash $H(M, K_{MO})$ is an ideal "one-time MAC" [1] known by the MSU and the Observer. A charge unit L is a value agreed by the MSU and the MNSP. L can also be a variable value set as a system parameter that can be chosen by the MSU in each call. The proposed scheme with five steps is illustrated in Fig. 2, and explained in detail below.

- *Step 1.* The MSU and the MNSP pass the authentication and begin a call connection. Once the connection is established, the MNSP sends $M_1 = (TNO \| SID \| T_{start} \| L \| etc)$ to the MSU, where L is the pre-defined time unit and etc contains some related information.
- *Step 2.* Upon receiving message M_1, the MSU checks its validity, such as the validity of T_{start} and L etc. If the check passes, the MSU generates a random number a and calculates m chained one-way hash values according to equation (1).

$$H^i(a) = H(H^{i-1}(a)), \quad i = 1, 2, ..., m. \tag{1}$$

Then, the MSU computes a keyed-hash $MAC = H(M_1, m, H^m(a), K_{MO})$ and sends $(M_1, m, H^m(a), Observer, MAC)$ to the MNSP.

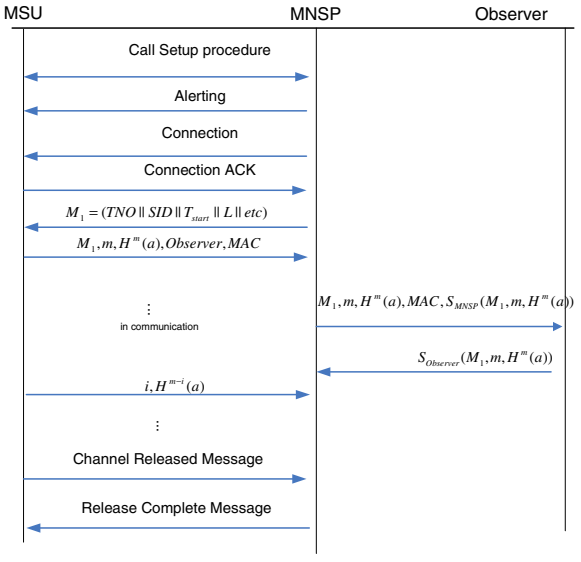


Fig. 2. The Proposed Mobile Billing Scheme

- *Step 3*. Upon obtaining $(M_1, m, H^m(a), Observer, MAC)$ from the MSU, the MNSP signs $(M_1, m, H^m(a))$ and sends $(M_1, m, H^m(a), MAC, S_{MNSP}(M_1, m, H^m(a)))$ to the Observer.
- *Step 4*. After the Observer receives the message from the MNSP, it verifies the keyed-hash MAC by checking $MAC = H(M_1, m, H^m(a), K_{MO})$, where K_{MO} is the shared key between the Observer and the user with identity SID that is specified in M_1. If this is correct, the Observer signs $(M_1, m, H^m(a))$ and sends $S_{Observer}(M_1, m, H^m(a)))$ to the MNSP as the evidence of making a call.
- *Step 5*. The MSU can continue the call by releasing $(i, H^{m-i}(a))$ to the MNSP at the pre-defined interval L during the service. The MNSP will check

$$H^{m-(i-1)}(a) \stackrel{?}{=} H(H^{m-i}(a)), \quad i = 1, 2, ..., m.$$

If it is true, the MNSP overwrites the former pair $(i-1, H^{m-(i-1)}(a))$ with $(i, H^{m-i}(a))$ in its cache. If it is false, it will send a warning and then cut off the call connection. The MNSP retains $(M_1, m, H^m(a), S_{Observer}(M_1, m, H^m(a))$ and the last chained hash value $(i, H^{m-i}(a))$ as *non-repudiation evidences* of a call.

For a given period, the MNSP submits the billing information (including all non-repudiation evidences of those calls) to the Observer and gets the payment from a mobile user through the Observer.

If a user has doubts over a bill provided by the MNSP, she can get the related non-repudiation evidences from the Observer or the MNSP and then

check if each call is correctly charged. For a call with non-repudiation evidences $(M_1, S_{Observer}(M_1, m, H^m(a)), m, H^m(a), i, H^{m-i}(a))$, the user re-calculates the corresponding fee as follows.

- Check the format of M_1: whether the SID and L etc are correct.
- Checks $S_{Observer}(M_1, m, H^m(a))$ is a valid signature of the Observer on message $(M_1, m, H^m(a))$.
- Validate the hash chained values by checking whether $H^m(a) \equiv H^i(H^{m-i}(a))$.
- If all of above checks pass, compute the corresponding fee according to the call time $(m - i) \times L$ and the charge rate. Otherwise, this call should not be charged to the user.

5 Analysis of the Proposed Scheme

In this section, we evaluate the proposed scheme in terms of security and efficiency. The security properties include authentication, non-repudiation, and fairness as discussed in Section 1.

5.1 Security

Firstly, the authentication between the MSU and the MNSP is completed in the call setup phase provided by the GSM system itself. The authentication of the MSU to the Observer is achieved via the keyed hash $MAC = H(M_1, m, H^m(a), K_{MO})$ by using the shared key K_{MO}, which is known by the user and the Observer only. Before a call connection is established, the Observer calculates the value of MAC and checks whether it is the same as that one forwarded by the MNSP. According to the results of keyed hash authentication given in [1], only the right MSU with the key K_{MO} can properly calculate the value of MAC. Therefore, the authentication property is satisfied.

For the non-repudiation property, as described in Section 4, the billing information collected by the MNSP retains sufficient evidences which make a bill undeniable and clear. Namely, the MNSP keeps $S_{Observer}(M_1, m, H^m(a))$ and the last chained hash value $(i, H^{m-i}(a))$ as the non-repudiation evidences of a call. We naturally require the Observer employs an existentially unforgeable signature scheme, so a valid signature $S_{Observer}(M_1, m, H^m(a))$ must be signed by the Observer itself. Moreover, we assume that the Observer is a trusted party, so it issues this signature if and only if it received a message $(M_1, m, H^m(a))$ authenticated by MAC from someone else. However, only the user or the Observer can generate correct MAC, since a secret key K_{MO} is needed. Once more, due to the Observer is a trusted party, it is concluded that $(M_1, m, H^m(a))$ must be approved by the user. Consequently, this implies that the user requested a call that is indexed by message M_1 and hash chain $(m, H^m(a))$. Therefore, the user cannot deny the fact that she made a call defined by non-repudiation evidence $(M_1, m, H^m(a), S_{Observer}(M_1, m, H^m(a)), i, H^{m-i}(a))$.

After receiving $S_{Observer}(M_1, m, H^m(a))$ from the Observer, the MNSP is ensured that the call request is from a specific MSU, since the Observer is a

trusted party. During a call session, the MSU has to release a pair $(i, H^{m-i}(a))$ to the MNSP in each time interval L, while the MNSP can check the validity of such a pair timely. The billing information collected by the MNSP is bounded by the Observer's signature and a hash value released by the MSU. Neither the MNSP nor the MSU can forge or deny a bill record. If any dispute happens later, the user or any third party can check whether a call is correctly charged according to the procedure specified in Section 4. Both evidences cannot be forged, so the scheme satisfies the non-repudiation requirement.

Now we discuss the fairness of our mobile billing scheme. Fairness means that the billing method should provide objective evidences accepted by both the MNSP and mobile users such that neither the MNSP nor the mobile user can cheat the other. First, the duration of each call is clearly and objectively determined by $(m - i) \times L$, if the corresponding non-repudiation evidence is $(M_1, S_{Observer}(M_1, m, H^m(a)), m, H^m(a), i, H^{m-i}(a))$. Second, the MNSP cannot forge valid evidences for a call that is not made by the user since the MNSP cannot forge the Observer's signature $S_{Observer}(M_1, m, H^m(a))$; the MNSP also cannot overcharge a call to the user since beside the user nobody cannot release further pre-images of the hash chain. Finally, a mobile user cannot cheat the MNSP too. The reason is that to get the MNSP's service, a user should be first authenticated by the Observer. Otherwise, the user's call will not be connected by the MNSP. So, only legal users can be served by the MNSP via using their proper keys shared with the Observer. Moreover, during the call session the user has to periodically release new hash values to continue a call. In particular, note that the proposed new scheme is immune to the two attacks presented in Section 3.2, since we do not use two signatures to determine the duration of a call at all.

5.2 Efficiency

To design a practical mobile billing system, the limitations of the computation capability, storage capability and power capability of the mobile terminal should be considered. Generally, the efficiency of a system is mainly determined by the computation complexity and communication complexity. So, in Table 1 we make an efficiency comparison between the CJC scheme and our new solution.

Table 1. Efficiency Comparison

	Communication Steps	Public Key Operations
CJC Scheme [3]	8	8
Our Scheme	5	4

As shown in Table 1, the new scheme has fewer communication steps than the CJC scheme. For the public key operations, the CJC scheme requires 4 signature generation operations and 4 signature validation operations to generate the undeniable evidences. While in the proposed mobile billing mechanism, only 2 signature generation operations and 2 signature validation operations are needed.

Thus our newly proposed scheme is more efficient. On the other hand, as same in the CJC scheme the proposed scheme also does not employ public key algorithm for the MSU and the certificate revocation issue is also avoided without using public key certificate for the MSU. Although a hash value needs to be released periodically in our scheme, the computation and communication overheads are very lightweight. Moreover, note that the hash chain can be pre-computed before a call setup to improve the efficiency. Thus the proposed undeniable billing scheme is more efficient than the CJC scheme and can be integrated into the current GSM systems.

6 Conclusion

In this paper, we first re-examined the security requirements of a secure and undeniable mobile billing system for current mobile systems. Then, we analyzed the CJC mobile billing system [3] and identified some weaknesses in the CJC mobile billing system. In particular, two attacks were demonstrated to show that the CJC scheme is not fair for both the mobile user and the service provider. Finally, we proposed a new scheme by combining digital signature mechanism and the technique of gradually releasing the chained hash values. The scheme satisfies the authentication, non-repudiation, and fairness requirements of a secure undeniable mobile billing system. In our scheme, the user's mobile device just need to perform hash operation without doing any public key operation during call procedures. Therefore, the proposed scheme is very efficient and could be applicable to the current GSM systems.

References

1. Bellare, M., Rogaway, P.: Minimizing the use of random oracles in authenticated encryption schemes. In: Han, Y., Quing, S. (eds.) ICICS 1997. LNCS, vol. 1334, pp. 1–16. Springer, Heidelberg (1997)
2. Chen, L., Hitz, H.J., Horn, G., Howker, K., Kessler, V., Knudsen, L., Mitchell, C.J.: The use of trusted third parties and secure billing in umts. In: Proceedings of ACTS Mobile Telecommunications Summit, pp. 493–499, Granada (1996)
3. Chen, Y.-Y., Jan, J.-K., Chen, C.-L.: A fair and secure mobile billing system. Computer Networks 48(4), 517–524 (2005)
4. Martin, K.M., Preneel, B., Mitchell, C.J., Hitz, H.-J., Horn, G., Poliakova, A., Howard, P.: Secure billing for mobile information services in umts. In: Campolargo, M., Mullery, A. (eds.) IS&N 1998. LNCS, vol. 1430, pp. 535–548. Springer, Heidelberg (1998)
5. Shenker, S., Clark, D., Estrin, D., Herzog, S.: Pricing in computer networks: reshaping the research agenda. SIGCOMM Comput. Commun. Rev. 26(2), 19–43 (1996)
6. Zhou, J., Lam, K.-Y.: Undeniable billing in mobile communication. In: MobiCom '98: Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking, New York, USA, 1998, pp. 284–290. ACM Press, New York (1998)

Universally Composable Signcryption

Kristian Gjøsteen and Lillian Kråkmo

Dept. of Mathematical Sciences, NTNU

Abstract. One of the challenges within public-key based cryptosystems is providing the user with a convenient interface, while retaining security. In the universal composability framework, we propose an ideal functionality for secure messaging, with a user-friendly interface. We also propose an ideal functionality for signcryption, and we show that, given a public key infrastructure and a secure signcryption protocol, we can construct a protocol that securely realizes the secure messaging functionality. Moreover, we show that a signcryption protocol realizes the signcryption functionality if and only if the corresponding signcryption scheme is secure.

Keywords: Secure messaging, universal composability, signcryption.

1 Introduction

Signcryption was first proposed by Zheng [7] as a primitive for achieving both confidentiality and authenticity of message delivery/storage in a logically single step, with the aim of reducing the cost compared to the standard "sign-then-encrypt" method. Regarding security definitions for signcryption schemes, several approaches have been taken. An overview of the different models is provided in [5].

In general, composing several (possibly identical) protocols into a larger protocol may not preserve security. Universally composable security is a framework proposed by Canetti [3] as a way to define security for protocols such that security-preserving composition is possible. This allows for a modular design and analysis of protocols.

For each cryptographic task, an *ideal functionality* can be defined, which incorporates the required properties of a protocol for the task and the allowed actions of an adversary. A protocol is said to *securely realize* the ideal functionality if, loosely speaking, any effect caused by an adversary attacking the protocol can be obtained by an adversary attacking the ideal functionality. When designing complex protocols, one can allow the involved parties to have secure access to ideal functionalities. Then, when implementing the protocol, each ideal functionality is replaced by a protocol securely realizing the functionality. The *composition theorem* then guarantees security. We refer to [3] for a complete overview of this framework.

In Sect. 2 of this paper, we review the properties of a signcryption scheme and define what it means for a signcryption scheme to be secure. Based on these security requirements, we construct an ideal functionality for signcryption, which

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 346–353, 2007.

is defined in Sect. 3. This section also presents a natural ideal functionality for secure messaging, which is a suitable model for applications such as secure email and secure instant messaging. Given functionalities for public key infrastructure and signcryption, we construct a protocol that integrates these services and securely realizes the secure messaging functionality.

Finally, in Sect. 4 we claim that a signcryption scheme satisfies our security definitions if and only if the corresponding protocol securely realizes the signcryption functionality. We note that our results are only valid in the static corruption case. This is discussed further in Sect. 5.

Proofs are omitted due to space limitations, but will appear in the full version of this paper.

2 Signcryption

Our definition of a signcryption scheme is identical to the one given in [5].

Definition 1. *A signcryption scheme $\mathcal{SC}$ is a 5-tuple of algorithms $(\mathcal{C}, \mathcal{K}_s, \mathcal{K}_r, \mathcal{S}, \mathcal{U})$ with the following properties:*

- *$\mathcal{C}$ is a probabilistic algorithm, taking as input a security parameter τ (encoded as 1^τ) and returning the global information I needed by users of the scheme.*
- *$\mathcal{K}_s$ is a probabilistic algorithm, taking as input the global information I and returning a pair (sk^s, pk^s) of secret and public keys for the sender.*
- *$\mathcal{K}_r$ is a probabilistic algorithm, taking as input the global information I and returning a pair (sk^r, pk^r) of secret and public keys for the receiver.*
- *$\mathcal{S}$, the signcryption algorithm, is probabilistic. Its inputs are a sender's private key sk^s, a receiver's public key pk^r and a plaintext m, and its output is a ciphertext c.*
- *$\mathcal{U}$, the unsigncryption algorithm, is deterministic. Its inputs are a sender's public key pk^s, a receiver's secret key sk^r and a ciphertext c. Its output is a plaintext m or the symbol $\perp$, indicating that the signcryption is invalid.*

It is required that $\mathcal{U}(pk^s, sk^r, \mathcal{S}(sk^s, pk^r, m)) = m$ for all plaintexts m and all key pairs (sk^s, pk^s) and (sk^r, pk^r) output by $\mathcal{K}_s$ and $\mathcal{K}_r$.

Our security model for signcryption schemes is similar to the ADR model presented in [5]. Since non-repudiation is not always required, we do not consider it here due to space constraints. Therefore we need only consider unforgeability between honest users. We need two experiments described in Fig. 1.

The first experiment $\mathbf{Exp}^{\text{ind-cca2}}_{\mathcal{SC},A}$ concerns privacy of messages, and adapts the notion IND-CCA2 from public-key encryption. In the beginning of the experiment, the adversary A is given two public keys pk^s and pk^r belonging to the target sender and the target receiver, respectively. A is composed of a *find*-stage algorithm A_1 and a *guess*-stage algorithm A_2. A_1 finds two messages m_0 and m_1 of the same length, while A_2 is given a challenge ciphertext c and guesses whether c is a signcryption of m_0 or m_1.

$\mathbf{Exp}^{\text{ind-cca2}}_{\mathcal{SC},A}(\tau)$

1. $I \leftarrow C(\tau)$.
2. $(sk^s, pk^s) \leftarrow \mathcal{K}_s(I)$.
3. $(sk^r, pk^r) \leftarrow \mathcal{K}_r(I)$.
4. $(m_0, m_1, state) \leftarrow A_1^{\mathcal{O}_\mathcal{S}, \mathcal{O}_\mathcal{U}}(pk^s, pk^r)$.
5. $b \leftarrow \{0,1\}$.
6. $c \leftarrow \mathcal{S}(sk^s, pk^r, m_b)$.
7. $b' \leftarrow A_2^{\mathcal{O}_\mathcal{S}, \mathcal{O}_\mathcal{U}}(pk^s, pk^r, m_0, m_1, c, state)$.
8. If $b' = b$ then return 1, otherwise return 0.

$\mathbf{Exp}^{\text{ext-cma}}_{\mathcal{SC},A}(\tau)$

1. $I \leftarrow C(\tau)$.
2. $(sk^s, pk^s) \leftarrow \mathcal{K}_s(I)$.
3. $(sk^r, pk^r) \leftarrow \mathcal{K}_r(I)$.
4. $c \leftarrow A^{\mathcal{O}_\mathcal{S}, \mathcal{O}_\mathcal{U}}(pk^s, pk^r)$.
5. If $\mathcal{U}(pk^s, sk^r, c) \neq \perp$ then return 1, otherwise return 0.

$\mathbf{Exp}^{\text{ror-cca2}}_{\mathcal{SC},A}(\tau)$

1. $I \leftarrow C(\tau)$
2. $(sk^s, pk^s) \leftarrow \mathcal{K}_s(I)$
3. $(sk^r, pk^r) \leftarrow \mathcal{K}_r(I)$
4. $b \leftarrow \{0,1\}$
5. $b' \leftarrow A^{\mathcal{O}_\mathcal{S}, \mathcal{O}_\mathcal{U}, \mathcal{O}^b_{\text{ror}}}(pk^s, pk^r)$
6. Return b'.

Fig. 1. Experiments for security definitions

Both A_1 and A_2 have access to a flexible signcryption oracle $\mathcal{O}_\mathcal{S}$, which performs signcryption under the fixed key sk^s and an arbitrary key $pk^{r'}$, and a flexible unsigncryption oracle $\mathcal{O}_\mathcal{U}$, which performs unsigncryption under an arbitrary key $pk^{s'}$ and the fixed key sk^r. A_2 is not allowed to query $\mathcal{O}_\mathcal{U}$ with the ciphertext c and the sender key pk^s.

A is said to win if the experiment returns 1. We define the advantage of A in breaking $\mathcal{SC}$ with respect to IND-CCA2 as

$$\mathbf{Adv}^{\text{ind-cca2}}_{\mathcal{SC},A}(\tau) = \left| 2 \cdot \Pr\left[\mathbf{Exp}^{\text{ind-cca2}}_{\mathcal{SC},A}(\tau) = 1\right] - 1 \right|.$$

The scheme $\mathcal{SC}$ is said to be secure with respect to IND-CCA2 if the advantage $\mathbf{Adv}^{\text{ind-cca2}}_{\mathcal{SC},A}(\tau)$ is negligible in τ, whenever A's runtime and number of oracle queries are polynomially bounded in τ.

The second experiment $\mathbf{Exp}^{\text{ext-cma}}_{\mathcal{SC},A}(\tau)$ concerns unforgeability of messages, and adapts the notion EXT-CMA from digital signatures. This experiment starts with the adversary A being given the target sender's public key pk^s and the target receiver's public key pk^r. A's job is to produce a ciphertext c such that c is a valid ciphertext with respect to the target sender and the target receiver. A has access to the oracles $\mathcal{O}_\mathcal{S}$ and $\mathcal{O}_\mathcal{U}$ described above. It is required that c was not output by $\mathcal{O}_\mathcal{S}$ on input of pk^r.

A is said to win if the experiment returns 1. We define the success rate of A in breaking $\mathcal{SC}$ with respect to EXT-CMA as

$$\mathbf{Succ}^{\text{ext-cma}}_{\mathcal{SC},A}(\tau) = \Pr\left[\mathbf{Exp}^{\text{ext-cma}}_{\mathcal{SC},A}(\tau) = 1\right].$$

The scheme $\mathcal{SC}$ is said to be secure with respect to EXT-CMA if the success rate $\mathbf{Succ}^{\text{ext-cma}}_{\mathcal{SC},A}(\tau)$ is negligible in τ, whenever A's runtime and number of oracle queries are polynomially bounded in τ.

We now present a second notion for privacy of messages, adapting the notion "real-or-random" for symmetric encryption given in [1]. The idea is that no adversary should be able to distinguish a signcryption of a known message from a signcryption of a hidden random string. In the experiment $\mathbf{Exp}^{\text{ror-cca2}}_{\mathcal{SC},A}(\tau)$, the adversary A has access to the oracle $\mathcal{O}^b_{\text{ror}}$ (initialized with a hidden bit b) which takes as input a message m. If $b = 0$, it outputs a signcryption of a randomly chosen string of length $|m|$ under sk^s and pk^r. A new random string is chosen for each query. If $b = 1$, it outputs a signcryption of m under sk^s and pk^r. A's challenge is to guess the hidden bit b. As before, the adversary has access to $\mathcal{O}_{\mathcal{S}}$ and $\mathcal{O}_{\mathcal{U}}$.

In this experiment we require that A does not query $\mathcal{O}_{\mathcal{U}}$ with any of the ciphertexts output by $\mathcal{O}^b_{\text{ror}}$ together with pk^s and pk^r.

We define the advantage of A in breaking $\mathcal{SC}$ with respect to ROR-CCA2 as

$$\mathbf{Adv}^{\text{ror-cca2}}_{\mathcal{SC},A}(\tau) = \left|\Pr\left[\mathbf{Exp}^{\text{ror-cca2}}_{\mathcal{SC},A}(\tau)=1|b=1\right] - \Pr\left[\mathbf{Exp}^{\text{ror-cca2}}_{\mathcal{SC},A}(\tau)=1|b=0\right]\right|.$$

The scheme $\mathcal{SC}$ is said to be secure with respect to ROR-CCA2 if the advantage $\mathbf{Adv}^{\text{ror-cca2}}_{\mathcal{SC},A}(\tau)$ is negligible in τ, whenever A's runtime and number of oracle queries are polynomially bounded in τ.

The following theorem is a straight-forward adaption of a theorem in [6].

Theorem 1. *A signcryption scheme $\mathcal{SC}$ is secure with respect to IND-CCA2 if and only if it is secure with respect to ROR-CCA2.*

3 Secure Messaging in the UC Framework

In this section, we define an ideal functionality $\mathcal{F}_{\text{SC}}$ for signcryption, based on the security definitions given in the previous section. With applications such as email and instant messaging in mind, we also define an ideal functionality $\mathcal{F}_{\text{SM}}$ for secure messaging, which arises naturally from the required properties of such applications. We also show that, given a public key infrastructure and a secure signcryption protocol, we can construct a protocol π_{SM} that securely realizes $\mathcal{F}_{\text{SM}}$ in the $(\mathcal{F}_{\text{CA}}, \mathcal{F}_{\text{SC}})$-hybrid model, where $\mathcal{F}_{\text{CA}}$ is an ideal functionality providing a public key infrastructure.

We point out that in the definitions of the ideal functionalities, we only consider the case of static corruption. In other words, when executing a protocol, it is known from the start which parties are corrupted and which are uncorrupted.

The ideal certification authority functionality $\mathcal{F}_{\text{CA}}$ is defined in Fig. 2, and is similar to the one given by Canetti in [2]. Next, in Fig. 3, the ideal signcryption functionality $\mathcal{F}_{\text{SC}}$ is defined, and then, in Fig. 4, the ideal secure messaging functionality $\mathcal{F}_{\text{SM}}$ is defined. The protocol π_{SM} is described in Fig. 5.

We can prove the following result.

Functionality $\mathcal{F}_{\mathrm{CA}}$

$\mathcal{F}_{\mathrm{CA}}$ proceeds as follows, with parties $P_1, \ldots, P_n$ and an ideal adversary S.

CA.Register

Upon receiving the first message $(\mathbf{CA.Register}, sid, v)$ from some party P_i, send $(\mathbf{CA.Register}, sid, P_i, v)$ to S. Upon receiving $(\mathbf{Ok}, sid, P_i)$ from S, record the pair (P_i, v), and output $(\mathbf{CA.Registered}, sid, v)$ to P_i.

CA.Retrieve

Upon receiving a message $(\mathbf{CA.Retrieve}, sid, P_i)$ from party P_j, send $(\mathbf{CA.Retrieve}, sid, P_i, P_j)$ to S, and wait for an $(\mathbf{Ok}, sid, P_i, P_j)$ from S. Then, if there is a recorded pair (P_i, v) output $(\mathbf{CA.Retrieved}, sid, P_i, v)$ to P_j. Otherwise output $(\mathbf{CA.Retrieved}, sid, P_i, \perp)$ to P_j.

Fig. 2. The ideal certification authority functionality $\mathcal{F}_{\mathrm{CA}}$

Functionality $\mathcal{F}_{\mathrm{SC}}$

$\mathcal{F}_{\mathrm{SC}}$ proceeds as follows, with parties $P_1, \ldots, P_n$ and an ideal adversary S.

Upon receiving a message from a corrupted party, $\mathcal{F}_{\mathrm{SC}}$ forwards the message to S, and when S replies to this message, $\mathcal{F}_{\mathrm{SC}}$ forwards the reply to the corrupted party.

SC.KeyGen

Upon receiving the first message $(\mathbf{SC.KeyGen}, sid)$ from some party P_i, send $(\mathbf{SC.KeyGen}, sid, P_i)$ to S. Upon receiving $(\mathbf{SC.Key}, sid, P_i, (pk_i^s, pk_i^r))$ from S, output $(\mathbf{SC.Key}, sid, (pk_i^s, pk_i^r))$ to P_i and record $(P_i, (pk_i^s, pk_i^r))$.

SC.Encrypt

Upon receiving $(\mathbf{SC.Encrypt}, sid, pk^r, m)$ from P_i, do:

1. If $pk^r = pk_j^r$ for some j and P_j is uncorrupted, then send $(\mathbf{SC.Encrypt}, sid, pk_i^s, pk^r, |m|)$ to S. Otherwise send $(\mathbf{SC.Encrypt}, sid, pk_i^s, pk^r, m)$ to S.
2. Upon receiving $(\mathbf{SC.Ciphertext}, sid, pk_i^s, pk^r, c)$ from S such that there is no recorded entry (pk_i^s, pk^r, m', c) for any m', output $(\mathbf{SC.Ciphertext}, sid, pk^r, m, c)$ to P_i. If $pk^r = pk_j^r$ for some j and P_j is uncorrupted, then record the entry (pk_i^s, pk^r, m, c).

SC.Decrypt

Upon receiving $(\mathbf{SC.Decrypt}, sid, pk^s, c)$ from P_j, do:

1. Send $(\mathbf{SC.Decrypt}, sid, pk^s, pk_j^r, c)$ to S. Upon receiving $(\mathbf{SC.Plaintext}, sid, pk^s, pk_j^r, m'/\perp, c)$ from S, continue.
2. If an entry (pk^s, pk_j^r, m, c) is recorded, then output $(\mathbf{SC.Plaintext}, sid, pk^s, m, c)$ to P_j.
3. Otherwise, if $pk^s = pk_i^s$ for some i and P_i is uncorrupted, then output $(\mathbf{SC.Plaintext}, sid, pk^s, \perp, c)$ to P_j.
4. Otherwise, output $(\mathbf{SC.Plaintext}, sid, pk^s, m'/\perp, c)$ to P_j.

Fig. 3. The ideal signcryption functionality $\mathcal{F}_{\mathrm{SC}}$

Functionality $\mathcal{F}_{\text{SM}}$

$\mathcal{F}_{\text{SM}}$ proceeds as follows, with parties $P_1, \ldots, P_n$ and an ideal adversary S.

Upon receiving a message from a corrupted party, $\mathcal{F}_{\text{SM}}$ forwards the message to S, and when S replies to this message, $\mathcal{F}_{\text{SM}}$ forwards the reply to the corrupted party.

SM.Register

Upon receiving the first message (**SM.Register**, sid) from some party P_i, send (**SM.Register**, sid, P_i) to S. Upon receiving (**Ok**, sid, P_i) from S, output (**SM.Registered**, sid) to P_i and record P_i.

SM.Encrypt

Upon receiving (**SM.Encrypt**, sid, P_j, m) from some party P_i, do:

1. If there is a recorded entry P_i, then continue. Otherwise, output (**SM.Encrypt.Error**, sid, P_i *not registered*) to P_i.
2. If P_j is uncorrupted, then send (**SM.Encrypt**, $sid, P_i, P_j, |m|$) to S. Otherwise, send (**SM.Encrypt**, sid, P_i, P_j, m) to S.
3. Upon receiving (**SM.Ciphertext**, sid, P_i, P_j, c) from S such that there is no recorded entry (P_i, P_j, m', c) for any m', check if there is a recorded entry P_j. If there is, then continue. Otherwise, output (**SM.Encrypt.Error**, sid, P_j *not registered*) to P_i.
4. Output (**SM.Ciphertext**, sid, P_j, m, c) to P_i. If P_j is uncorrupted, then record the entry (P_i, P_j, m, c).

SM.Decrypt

Upon receiving (**SM.Decrypt**, sid, P_i, c) from some party P_j, do:

1. If there is a recorded entry P_j, then continue. Otherwise, output (**SM.Decrypt.Error**, sid, P_j *not registered*) to P_j.
2. Send (**SM.Decrypt**, sid, P_i, P_j, c) to S.
3. Upon receiving (**SM.Plaintext**, $sid, P_i, P_j, m'/\perp, c$) from S, check if there is a recorded entry P_i. If there is, then continue. Otherwise, output (**SM.Decrypt.Error**, sid, P_i *not registered*) to P_j.
4. If an entry (P_i, P_j, m, c) is recorded, then output (**SM.Plaintext**, sid, P_i, m, c) to P_j.
5. Otherwise, if P_i is uncorrupted, then output (**SM.Plaintext**, $sid, P_i, \perp, c$) to P_j.
6. Otherwise, output (**SM.Plaintext**, $sid, P_i, m'/\perp, c$) to P_j.

Fig. 4. The ideal secure messaging functionality $\mathcal{F}_{\text{SM}}$

Theorem 2. *The protocol π_{SM} securely realizes $\mathcal{F}_{SM}$ in the $(\mathcal{F}_{CA}, \mathcal{F}_{SC})$-hybrid model.*

4 Securely Realizing $\mathcal{F}_{\text{SC}}$

The protocol $\pi_{\mathcal{SC}}$ given in Fig. 6 is constructed in a natural way from the signcryption scheme $\mathcal{SC}$. We can prove the following result.

Theorem 3. *Let $\mathcal{SC}$ be a signcryption scheme. $\pi_{\mathcal{SC}}$ securely realizes $\mathcal{F}_{SC}$ if and only if $\mathcal{SC}$ is secure with respect to both IND-CCA2 and EXT-CMA.*

Protocol π_{SM}

π_{SM} proceeds as follows, with parties $P_1, \ldots, P_n$ and an adversary A.

SM.Register

1. Upon the first input (**SM.Register**, *sid*), P_i sends (**SC.KeyGen**, *sid*) to $\mathcal{F}_{\mathrm{SC}}$.
2. Upon receiving (**SC.Key**, *sid*, (pk_i^s, pk_i^r)) from $\mathcal{F}_{\mathrm{SC}}$, P_i sends (**CA.Register**, *sid*, (pk_i^s, pk_i^r)) to $\mathcal{F}_{\mathrm{CA}}$.
3. Upon receiving (**CA.Registered**, *sid*, (pk_i^s, pk_i^r)) from $\mathcal{F}_{\mathrm{CA}}$, P_i outputs (**SM.Registered**, *sid*).

SM.Encrypt

1. Upon input (**SM.Encrypt**, *sid*, P_j, m), P_i checks if he has received (**CA.Registered**, *sid*, (pk_i^s, pk_i^r)). If he has, then he sends (**CA.Retrieve**, *sid*, P_j) to $\mathcal{F}_{\mathrm{CA}}$. Otherwise, he outputs (**SM.Encrypt.Error**, *sid*, P_i *not registered*).
2. If P_i receives (**CA.Retrieved**, *sid*, P_j, (pk_j^s, pk_j^r)) from $\mathcal{F}_{\mathrm{CA}}$, then P_i sends (**SC.Encrypt**, *sid*, pk_j^r, m) to $\mathcal{F}_{\mathrm{SC}}$. Otherwise, if P_i receives (**CA.Retrieved**, *sid*, P_j, $\perp$) from $\mathcal{F}_{\mathrm{CA}}$, then P_i outputs (**SM.Encrypt.Error**, *sid*, P_j *not registered*).
3. Upon receiving (**SC.Ciphertext**, *sid*, pk_j^r, m, c) from $\mathcal{F}_{\mathrm{SC}}$, P_i outputs (**SM.Ciphertext**, *sid*, P_j, m, c).

SM.Decrypt

1. Upon input (**SM.Decrypt**, *sid*, P_i, c), P_j checks whether he has received (**CA.Registered**, *sid*, (pk_j^s, pk_j^r)). If he has, then he sends (**CA.Retrieve**, *sid*, P_i) to $\mathcal{F}_{\mathrm{CA}}$. Otherwise, he outputs (**SM.Decrypt.Error**, *sid*, P_j *not registered*).
2. If P_j receives (**CA.Retrieved**, *sid*, P_i, (pk_i^s, pk_i^r)) from $\mathcal{F}_{\mathrm{CA}}$, then P_j sends (**SC.Decrypt**, *sid*, pk_i^s, c) to $\mathcal{F}_{\mathrm{SC}}$. Otherwise, if P_j receives (**CA.Retrieved**, *sid*, P_i, $\perp$) from $\mathcal{F}_{\mathrm{CA}}$, then P_j outputs (**SM.Decrypt.Error**, *sid*, P_i *not registered*).
3. Upon receiving (**SC.Plaintext**, *sid*, pk_i^s, pk_j^r, $m/\perp$, c) from $\mathcal{F}_{\mathrm{SC}}$, P_j outputs (**SM.Plaintext**, *sid*, P_i, $m/\perp$, c).

Fig. 5. The secure messaging protocol π_{SM}

5 Concluding Remarks

We have proposed ideal functionalities for signcryption and secure messaging, and described a protocol π_{SM} that securely realizes $\mathcal{F}_{\mathrm{SM}}$ in the ($\mathcal{F}_{\mathrm{CA}}$, $\mathcal{F}_{\mathrm{SC}}$)-hybrid model. In addition, we have proved that a signcryption protocol securely realizes $\mathcal{F}_{\mathrm{SC}}$ if and only if the corresponding signcryption scheme is secure with respect to IND-CCA2 and EXT-CMA. This provides some evidence that IND-CCA2 and EXT-CMA are the correct security notions for signcryption.

We have proved our results only with static corruption, since it seems impossible to do better. However, it is conceivable that some kind of non-committing encryption can be used to get security against adaptive adversaries. Since $\mathcal{F}_{\mathrm{SC}}$

Protocol $\pi_{\mathcal{SC}}$

$\pi_{\mathcal{SC}}$ proceeds as follows, with parties $P_1, \ldots, P_n$ and an adversary A.

SC.KeyGen

Upon the first input (**SC.KeyGen**, sid), P_i runs the algorithms $\mathcal{K}_s$ and $\mathcal{K}_r$. He obtains a sender's key pair (sk_i^s, pk_i^s) and a receiver's key pair (sk_i^r, pk_i^r), and outputs (**SC.Key**, sid, (pk_i^s, pk_i^r)).

SC.Encrypt

Upon input (**SC.Encrypt**, sid, pk^r, m), P_i obtains $c = \mathcal{S}(sk_i^s, pk^r, m)$ and outputs (**SC.Ciphertext**, sid, pk^r, m, c).

SC.Decrypt

Upon input (**SC.Decrypt**, sid, pk^s, c), P_i obtains $m/\perp = \mathcal{U}(pk^s, sk_i^r, c)$ and outputs (**SC.Plaintext**, $sid, pk^s, m/\perp, c$).

Fig. 6. The signcryption protocol $\pi_{\mathcal{SC}}$

can be securely realized using ideal functionalities for public-key encryption and digital signatures, it may be possible to replace the encryption functionality (providing security against static corruption) with an ideal functionality for non-committing encryption (providing security against adaptive corruption) [4], and get $\mathcal{F}'_{\mathrm{SC}}$ with security against adaptive corruption.

References

1. Bellare, M., Desai, A., Jokipii, E., Rogaway, P.: A Concrete Security Treatment of Symmetric Encryption. In: FOCS '97: Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS '97), Washington, DC, USA, pp. 394–403, IEEE Computer Society (1997)
2. Canetti, R.: Universally Composable Signature, Certification, and Authentication. Cryptology ePrint Archive, Report, 2003/239 (2004). Available at `http://eprint.iacr.org/2003/239`
3. Canetti, R.: Universally Composable Security: A New Paradigm for Cryptographic Protocols. Cryptology ePrint Archive, Report, 2000/067 (2005). Available at `http://eprint.iacr.org/2000/067`
4. Canetti, R., Halevi, S., Katz, J.: Adaptively-Secure, Non-Interactive Public-Key Encryption. Cryptology ePrint Archive, Report, 2004/317 (2004) `http://eprint.iacr.org/`
5. Malone-Lee, J.C.: On the Security of Signature Schemes and Signcryption Schemes. PhD thesis, University of Bristol (2004)
6. Rogaway, P.: Symmetric Encryption. ECS 227 - Modern Cryptography - Winter 99 (1999). Available at `http://www.cs.ucdavis.edu/ rogaway/classes/227/winter99/`
7. Zheng, Y.: Digital Signcryption or How to Achieve Cost(Signature & Encryption) $\ll$ Cost(Signature) + Cost(Encryption). In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 165–179. Springer, Heidelberg (1997)

Chord-PKI: Embedding a Public Key Infrastructure into the Chord Overlay Network*

Agapios Avramidis, Panayiotis Kotzanikolaou, and Christos Douligeris

Department of Informatics, University of Piraeus,
Karaoli & Dimitriou 80, 185 34 Piraeus, Greece
{agapios,pkotzani,cdoulig}@unipi.gr

Abstract. Our goal in this paper is to provide authentication, encryption and non-repudiation services for nodes within Peer-to-Peer networks, in an efficient and scalable way. To accomplish this, we propose a distributed Public Key Infrastructure model, suitable for Peer-to-Peer networks and more particularly for the Chord protocol. Our solution integrates the PKI infrastructure within the Chord architecture. We use well known cryptographic techniques as building blocks, such as threshold cryptography and proactive updating.

1 Introduction

Peer to peer (P2P) networks have received considerable attention in the last few years. In particular, one class of P2P networks, namely structured overlays [1,2,3] seems a very attractive choice for building large scale systems. Almost all structured overlay networks utilize a *Distributed Hash Table* (DHT) abstraction. The DHT uses a consistent hash function (*e.g.* a cryptographic hash function such as SHA-1) in order to assign identifiers to nodes and keys[1]. Moreover, the DHT allows the lookup operations (*get* and *put*) to be performed with logarithmic cost in terms of communication messages. DHTs offer a desirable set of properties for distributed applications such as load balancing, decentralization and scalability.

Until recently, the main focus of research for DHTs was targeted to the performance of the lookup protocols, the topology of the overlay, load balancing and search issues (such as range queries, multi-attribute and aggregation queries) [4]. Recently, research for DHTs has also focused on security issues, *e.g.* [5,6,7].

Towards this direction, we propose the *Chord-PKI*, a distributed Public Key Infrastructure (PKI) embedded into the Chord [1] overlay network. Our system provides certification to the Chord nodes through a synergetic protocol that enables the collaboration of the nodes themselves, without the need for an external

* Research funded by the General Secretariat for Research and Technology (GSRT) of Greece under a PENED grant.

[1] These keys correspond to indices to objects such as files and are not keys in the cryptographic sense.

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 354–361, 2007.

PKI. Chord-PKI provides authentication, encryption and non-repudiation services for the nodes, in an efficient and scalable way. The system uses well known cryptographic techniques as building blocks, such as threshold cryptography [8] and proactive updating [9] and guaranties certain resistance to distributed attacks through redundancy. The rest of the paper is organized as follows. Section 2 presents Chord-PKI as well as its basic functions. Section 3 discusses performance and security issues, while section 4 concludes this paper.

2 The Chord-PKI

Our goal is to build a distributed PKI for the Chord structured overlay network. The use of an external PKI in a P2P environment (such as an external Certification Authority) is not an efficient solution, due to the high communication and management costs involved [10]. Moreover, the use of a traditional PKI would impose additional dependencies with external Trusted Third Parties, which in not always acceptable for decentralized and large-scale applications. Generally, a PKI solution for P2P networks, should achieve the following basic requirements:

- Scalability. Distributed Hash Tables are designed to support very large number of participants (internet scale). Moreover, a basic characteristic of P2Ps is high churn rates (frequent joins and leaves). A scalable PKI model for P2P network must not be affected by these characteristics.
- Efficiency. The certification, revocation, certificate storage and certificate retrieval must not impose heavy computation and communication burden into peer nodes. Traditional PKI models usually imply high computation and communication needs.
- Resiliency to compromised nodes. The trust infrastructure must be resilient to attacks. For example, a hierarchical PKI suffers from a single point of failure (the Root CA).

2.1 A High-Level Description of Chord-PKI

A basic solution for a Chord-based PKI is to empower some peer nodes with certification functionality. However, in this case, each of these certification nodes would be a single point of failure. An enhanced solution would be to partition the overlay network into a number of areas, so that each certification node would serve a single area. In that case, if a certification node were compromised, only one area would be affected. However, the adversary would only have to compromise one certification node in each partition.

Our model is resilient in such attacks, by employing threshold cryptography and it minimizes the burden of public key cryptography, by distributing the cryptographic functionality within the peers. Our solution also minimizes the storage and retrieval requirements for the public keys, by exploiting the distributed storage and retrieval functionality of the Chord protocol. The certificate directory is evenly distributed among the system nodes as a Chord *put* operation, thus balancing the storage cost. Moreover, the lookup of a certificate also exploits Chord functionality and it is implemented through a simple Chord *get* operation.

2.2 Setup

Before the operation of the Chord-PKI a setup period is executed, during which the initial nodes partition the network into virtual *segments*, and generate and certify a public/secret key pair for each segment. We assume that during setup, the initial nodes are trusted. We also assume that after the setup period, the public segment keys will be known to any incoming node joining the network.

During the setup, the initial nodes partition the Chord identifier (id) space into s segments, as shown in figure 1. Thus, if the id space of a Chord overlay network is $[0, 2^m - 1]$, the id space of each segment is $2^m/s$. The segments have a continuous identifier space, *i.e.* $SEG_i = [(i-1)\cdot(2^m/s),\ i\cdot(2^m/s)-1]$, $i \in [1,\ s]$. The value s is a system parameter. After the partitioning, in each segment SEG_i there exist (at least) n_i nodes. These nodes are called the certification nodes. Each segment $SEG_i,\ i \in [1, s]$ must be assigned with a unique public/secret key pair PK_i, SK_i of a public key cryptosystem (such as RSA, ElGamal or DSA). The secret key of each segment will be used for the certification of all the nodes that will later join the network inside the segment, as described in section 2.3. The segment public keys will be used for the verification of the nodes' certificates.

The secret key SK_i is generated by one or more of the n_i certification nodes, is shared among the n_i nodes within the segment and then it is deleted. The key sharing method should not enable any single node to use the secret key SK_i. Moreover, since the certification service must constantly be available, it must be possible to make use of the key SK_i with the participation of only a subgroup of the certification nodes. For these reasons, the key SK_i is shared with a (t_i, n_i) threshold signature scheme [8]. In this way, any subset of t_i out of the n_i key holders is able to generate a signature with the key SK_i.

In order to protect shares of each segment secret key, the certification nodes proactively update their key shares [9]. With proactive update each key share is periodically updated, while the secret key itself does not change. This is achieved by adding a sharing of zero with their previous shares. In this way, an active adversary that compromises key shares must succeed in compromising at least t_i shares within an update period. Otherwise, shares of different update periods cannot be combined to produce SK_i.

The public key of each segment must be universally known to all the nodes within the Chord network. This can be implemented by embedding the public segment keys into the latest release of the software installed by a node in order to join the network, as it happens in many web browsers which are pre-installed with the certificates of trusted Certification Authorities. Each key PK_i is self-certified and the certificate[2] $CERT_i = SIG_{SK_i}(i, PK_i)$ is then pre-installed into the nodes, when they install the latest release of the software. Additionally, each segment certificate $CERT_i$ is stored at the node that corresponds to the Chord identifier $SHA(CERT_i)$, where SHA is the secure hashing algorithm used by the Chord implementation for node and chord-key assignment [1].

[2] The certificates (segment or node certificates) may be formatted following the ITU-T X.509 standard, also containing other attributes such as certification time, issuer certificate etc. For simplicity we omit these values.

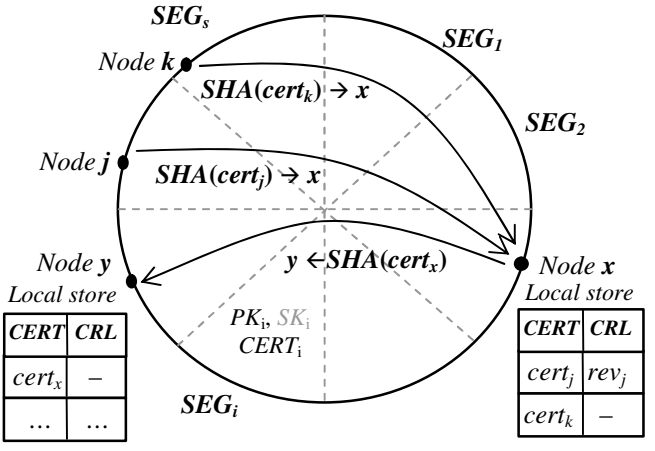


Fig. 1. Partitioning, storage and retrieval in Chord-PKI

2.3 Node Certification

Suppose that an incoming node j with a Chord id $ID_j \in SEG_i$ requests a certificate. The node generates a pair of public/secret keys $(pk_j,\ sk_j)$ and sends its public key to one of the certification nodes of the segment SEG_i it belongs, along with a certification request and a proof of knowledge of the corresponding secret key sk_j (*e.g.* under the PKCS 10 certificate request format).

If the certification node that receives the request trusts the requesting node (based on criteria discussed in section 2.7), then it can act as a combiner of a threshold signature and generate a certificate for the incoming node. The combiner generates a partial signature[3] $SIG_{SK_i^1}(ID_j, pk_j)$. Additionally, the combiner requests partial signatures from $t_i - 1$ other certification nodes of the key SK_i. After the combiner has received the partial signatures $SIG_{SK_i^2}(ID_j, pk_j), ..., \ SIG_{SK_i^{t_i}}(ID_j, pk_j)$, the combiner is able to produce a valid certificate for the node j. By combining the t_i partial signatures, the combiner generates a valid certificate $cert_j = SIG_{SK_i}(ID_j, pk_j)$ for the node j. At the end of this process the combiner verifies the signature of the certificate and if it is not valid, it repeats the procedure with another subset of certification nodes.

2.4 Certificate Revocation

If a node is accused for misbehavior from a number of certified nodes (based on criteria discussed in section 2.7), then its certificate is revoked by the certification nodes of its segment. Again, any subset of at least t_i certification nodes will sign a revocation message $rev_j = SIG_{SK_i}(cert_j, time)$ of the public key of the

[3] For simplicity and without loss of generality, we assume that the contacted certification node has the share SK_i^1 of the key SK_i, although it can be any certification node with a valid share of the key $SK_i^k,\ 1 \leq k \leq n_i$

misbehaving node, where *time* is the time of the revocation. The revocation message must then be stored in a certificate revocation list (CRL).

2.5 Certificate and CRL Storage

The certificate and the CRL storage is distributed among all the chord nodes, as shown in figure 1. Each node has a local certificate and CRL store and and is responsible to retain a number of certificates of other nodes in its local stores. After the generation of a valid certificate $cert_j$ of a node j, the certificate is stored in the local certificate store of the node x that corresponds to the Chord identifier $SHA(cert_j)$. Thus, any node that requires to verify a claimed certificate of another node, knows where the received certificate should be stored. The same strategy was followed for the storage of the segment certificates.

If a $cert_j$ of a node j has been revoked, the revocation message rev_j is also stored in the local CRL of the node x that corresponds to the Chord identifier, $SHA(cert_j) \rightarrow x$. Thus, any node that requires to verify a certificate $cert_j$ will check the local certificate and CRL store of the node x. For redundancy, the certificate $cert_j$, can also be stored in the following r nodes that succeed node x in the Chord identifier space.

2.6 Certificate and CRL Lookup

As explained above, a certificate and a CRL retrieval, is the same as a Chord lookup. Any node of the network, regardless of the segment it belongs, is able to lookup for a certificate (and for a possible revocation message of the certificate), by applying the SHA hash function to the certificate $cert_j$. Then, it will query the certificate and CRL store of the node x that corresponds to the Chord identifier $SHA(cert_j)$ for the particular certificate.

2.7 Trust and Reputation Models

An important issue that needs to be addressed is under what conditions a new node becomes trusted to acquire a certificate and under what conditions a certificate is revoked. The certification policy can be based on trust and reputation strategies similar to [11,12] where nodes are separated into two distinct sets, the trusted and the untrusted. In this case, initially the trusted nodes are only the original nodes of the setup phase, which share the secret key of each segment. When a new node enters the system for the first time it becomes a member of the untrusted set. In order to be promoted to the trusted set, the new node must behave correctly (follow the application protocol) for a certain period and then it reaches the desired reputation level to become a member of the trusted set. Communication protocols between trusted and untrusted nodes are defined in [11,12] as well a *promotion-demotion* protocol for moving nodes between thetwo sets.

The revocation policy, can rely on various "trust & reputation" models, such as [13,14,15]. In these models, each node maintains a list of trust values for every other node it knows. The list of trust values is used for making the decisions

that would lead to distrust, some node and thus revoke its certificate. Note that although in our system the ordinary nodes cannot revoke a certificate of another node, they can sign "distrust" messages for misbehaving nodes.

3 Analysis of the Chord-PKI

3.1 Performance Issues

Due to the additional exchanged messages and the cryptographic computations, Chord-PKI decreases the performance of the ordinary Chord protocol. However, Chord-PKI maximizes the functionality of the Chord protocol itself and in this way it minimizes the additional costs. Since Chord-PKI integrates the functionality of the certification, revocation, storage and retrieval, no communication is required with an external PKI service. This minimizes the required communication messages as well as additional delays imposed by the communication with external parties.

The computational cost for a certification or revocation, is proportional to the threshold t_i of a Chord segment. Note that this threshold may vary within different segments. Thus, it is possible to achieve an acceptable balance for each segment, according to the security requirements of each segment. Moreover, it is possible to modify the threshold set in one or more segments from (t, n) to (t', n') [16] and in this way achieve the required balance between efficiency and security. The computational cost for certificate verification requires at maximum two signature verifications, one for the certificate itself and one for the segment certificate. The model is almost flat, and there are no long chains of certificates.

The communication overhead for certificate and CRL retrieval is simply an ordinary Chord lookup. This is a basic advantage of the proposed system, since it uses the most important functionality of the Chord protocol, the Chord lookup function, which can performs a lookup with a logarithmic cost. Finally, the Chord-PKI performs very well in respect to the storage costs. The certificate and the CRL storage is distributed almost evenly among the nodes. The storage of a certificate $cert_j$ (or of its revocation message rev_j) is performed by using the consistent hash function of Chord, as ordinary Chord objects, to the node x indicated by the value $SHA(cert_j)$. Thus each node will maintain, as a separate Chord table, its local certificate store containing a limited number of certificates and the local CRL corresponding to the stored certificates.

3.2 Security Analysis

By dividing the Chord network into a number of segments and by using a different certification key in each segment, the system is protected from single point of failure attacks. Since there is no root CA in our system, it is not possible for an adversary to affect the whole system by compromising one signature key.

Additionally, the secret signature key SK_i of each segment i is shared among n_i certification nodes with a threshold sharing. In this way, it is not possible for an adversary to forge a node certificate, even if he compromises up to $t_i - 1$

certification nodes of a segment. Moreover, by periodically updating proactively the key shares, the adversary cannot combine key shares that he has collected in different update periods. This makes even harder to forge a certificate, since a successful attack would require compromising t_i nodes within a single update period.

In order to protect the integrity of the distributed certificate and CRL stores, only the certification nodes have the ability to store valid signed certificates or revocations to the local stores of the network nodes. Additionally, the local certificate and CRL storage can be configured as an append-only area, so that it will not be trivial for a malicious node to delete the local certificate stores of other nodes, but only to append a new certificate or a new revocation message.

Finally, the availability of the system is achieved by using redundancy in the certificate and CRL stores. Since a certificate is also stored in the r successor nodes of the node indicated by the first storage node, with high probability the certificate of a node will be available for the other nodes. This redundancy however requires a synchronization method during the storage for consistency reasons.

4 Conclusions and Future Work

In this paper we propose Chord-PKI, a decentralized PKI that exploits the characteristics of the Chord protocol in order to meet the requirements of P2P networks, namely scalability, efficiency and resilience to compromised nodes. Our system provides certification to the Chord nodes through a synergetic protocol that enables the collaboration of the nodes themselves, without the need for an external PKI. By relying on cryptographic techniques such as threshold cryptography our system can tolerate a number of malicious nodes within each segment. Moreover, by segmenting the network, a possible damage will be restrained within one segment. By exploiting the natural load balance property of the consistent hash function (SHA) of Chord the certificates and CRL storage is evenly distributed among the system nodes. The lookup for a certificate (or CRL) is accomplished in $logN$ steps, where N is the number of nodes in Chord-PKI. An important issue that needs to be addressed is fine-tuning trust and revocation models. Currently, we are investigating reputation models for P2P systems [13,14,15]. Unfortunately, many of these proposals have open scalability issues when the number of nodes increases. Thus, our future work involves the creation of a model that specifically meets the scalability requirements and exploits the advantages of Chord-PKI.

References

1. Stoica, I., Morris, R., Liben-Nowell, D., Karger, D., Dabek, F., Balakrishnan, H.: Chord: a scalable peer-to-peer lookup protocol for internet applications. IEEE/ACM Transactions on Networking 11(1), 17–32 (2003)
2. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Schenker, S.: A scalable content-addressable network. In: SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, New York, NY, USA, pp. 161–172. ACM Press, New York (2001)

3. Rowstron, A.I.T., Druschel, P.: Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In: Middleware '01: Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms, pp. 329–350. Springer, Heidelberg (2001)
4. Risson, J., Moors, T.: Survey of research towards robust peer-to-peer networks: Search methods. Elsevier Computer Networks 50(17), 3495–3521 (2006)
5. Sit, E., Morris, R.: Security considerations for peer-to-peer distributed hash tables. In: Druschel, P., Kaashoek, M.F., Rowstron, A. (eds.) IPTPS 2002. LNCS, vol. 2429, pp. 261–269. Springer, Heidelberg (2002)
6. Castro, M., Druschel, P., Ganesh, A., Rowstron, A., Wallach, D.S.: Secure routing for structured peer-to-peer overlay networks. SIGOPS Oper. Syst. Rev. 36, 299–314 (2002)
7. Wallach, D.S.: A survey of peer-to-peer security issues. In: Okada, M., Pierce, B.C., Scedrov, A., Tokuda, H., Yonezawa, A. (eds.) ISSS 2002. LNCS, vol. 2609, pp. 42–57. Springer, Heidelberg (2003)
8. Desmedt, Y., Frankel, Y.: Threshold cryptosystems. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, Springer, Heidelberg (1990)
9. Herzberg, A., Jakobsson, M., Jarecki, S., Krawczyk, H., Yung, M.: Proactive public key and signature systems. In: CCS '97: Proceedings of the 4th ACM conference on Computer and communications security, New York, NY, USA, pp. 100–110. ACM Press, New York (1997)
10. Wolfl, T.: Public-key-infrastructure based on a peer-to-peer network. In: In: HICSS '05: Proceedings of the 38th Annual Hawaii International Conference on System Sciences, vol. 7, pp. 200–201. IEEE Computer Society, Washington, DC, USA (2005)
11. Brampton, A., MacQuire, A., Rai, I.A., Race, N.J.P., Mathy, L.: Stealth distributed hash table: unleashing the real potential of peer-to-peer. In: CoNEXT'05: Proceedings of the 2005 ACM conference on Emerging network experiment and technology, pp. 230–231. ACM Press, New York (2005), doi:10.1145/1095921.1095955
12. Heinbockel, W., Kwon, M.: Phyllo: a peer-to-peer overlay security framework. In: NPSec 2005: 1st IEEE ICNP Workshop on Secure Network Protocols, November 2005, pp. 43–48 (2005)
13. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The eigentrust algorithm for reputation management in p2p networks. In: WWW '03: Proceedings of the 12th international conference on World Wide Web, New York, NY, USA, pp. 640–651. ACM Press, New York (2003), doi:10.1145/775152.775242
14. Yu, B., Singh, M., Sycara, K.: Developing trust in large-scale peer-to-peer systems. In: IEEE First Symposium on Multi-Agent Security and Survivability, 2004, August 2004, pp. 1–10 (2004)
15. Datta, A., Hauswirth, M., Aberer, K.: Beyond web of trust: enabling p2p e-commerce. In: CEC 2003. IEEE International Conference on E-Commerce, June 2003, pp. 303–312 (2003)
16. Desmedt, Y., Jajodia, S.: Redistributing secret shares to new access structures and its applications. Technical Report ISSE TR-97-01, George Mason University (July 1997)

Privacy Protection in Location-Based Services Through a Public-Key Privacy Homomorphism

Agusti Solanas and Antoni Martínez-Ballesté

CRISES Research Group
UNESCO Chair in Data Privacy
Dept. Computer Science and Mathematics
Rovira i Virgili University
{agusti.solanas,antoni.martinez}@urv.cat

Abstract. Location-Based Services (LBS) can be accessed from a variety of mobile devices to obtain value added information related to the location of the user. Most of the times, these services are provided by a trusted company (*e.g.* a telecommunications company). However, the massive use of mobile devices pave the way for the creation of ad hoc wireless networks that can be used to exchange information based on locations. In the latter case, these LBS could be provided by an untrusted party. Sending the location to an untrusted LBS provider could put the privacy of the user in jeopardy. In this paper we propose a novel technique to guarantee the privacy of users of LBS. Our technique consists of several modules, but the highest degree of security is achieved thanks to the use of a public-key privacy homomorphism. Unlike the existing approaches, our proposal does not need any trusted third party to anonymise the users and only makes use of a public-key infrastructure.

Keywords: location privacy, public-key privacy homomorphism.

1 Introduction

Location-Based Services (LBS) allow users to receive highly personalised information. These services can be accessed by using a variety of mobile devices that can utilise a plethora of localisation technologies. Mobile devices have become ubiquitous and services related to the current position of the users are growing fast. Some examples of these LBS are tourist information service, router planners, emergency assistance, etc. For a given user of LBS, sending her location could put her privacy in jeopardy.

An LBS basically consists of an LBS provider delivering location-based information and a set of users asking for this information. Mobile devices have a variety of ways for determining their approximate location. Thus, we assume in this paper that the utilised devices have this capability (*i.e.* they can determine their longitude and latitude).

In this scenario, a user u asks the LBS provider P for some information, sending the message $\{ID_u, query, long, lat\}$. Upon this request, P seeks the desired information in its database and returns an appropriate answer to u. Note that, if u sends her exact location to an untrusted LBS provider P_u, it can misbehave

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 362–368, 2007.

because it can relate the real location '$long, lat$' with the unique identifier of u, ID_u, for instance: (i) P_u is able to know if u is in front of certain shops or places, so it can flood her with undesired advertisements; (ii) P_u can track u so it knows where she has been and when; (iii) P_u can send the identifier of u along with her location to a spammer, and the later can send undesired location-based advertisements to the user.

In order to avert these possible misbehaviour of the LBS provider, two main solutions are possible:

- *Hiding the position within other users.* By using this technique inspired in the well-known k-anonymity approach [1,8], P is not able to distinguish u among a set of k users because they share the same fake location. This indistinguishability makes difficult the tracking and habits inference of the user.
- *Giving an inaccurate position to the LBS provider.* The position should be accurate enough so the information received by u is still useful. However, since the locations collected by P are not exact, they become useless to a spammer[1].

To the best of our knowledge, all previous proposals related to privacy in LBS rely on a trusted third party (TTP). One of them is the so-called *anonymizer*, a TTP used for anonymising locations by means of a mediation between users and LBS providers. The *anonymizer* can behave (i) by deleting personal information from the queries of the users before sending them to the LBS providers, or (ii) by hiding the exact position of the user (*i.e.* modifying it).

In the second case, the *anonymizer* hides the real location of the user under a *cloaked region* (*i.e.* a spatial region containing k users) so that each user becomes k-anonymous (*i.e.* not distinguishable among $k-1$ other users). According to [4], the cloaked region must fulfil the requirements of k-anonymity, but must also consider a spatial cloaking. In that sense, the cloaking algorithm considers a minimum and maximum area sizes and the *anonymizer* uses the requests from other users (and the location data contained in them) to compute a *masked* location, taking into account the value of k and the area requirements. The masked location can be computed as the centroid of the current locations of the users in the cloaked area.

In [2], an efficient algorithm for cloaking is presented. Similar approaches are also presented in [3] and [6].

1.1 Contribution and Plan of This Paper

In this paper we present a novel location privacy technique based on a Public-Key Infrastructure (PKI) and a public-key privacy homomorphism. Unlike the existing proposals, our technique does not rely on the LBS server acting as a TTP but on the collaboration of the users and an LBS server certified by a certification authority.

[1] Note that, although this seems to be a strong assumption, it is reasonable to believe that the user, who really knows her location, could make a proper use of the information given by the provider. On the contrary, the provider has access to the fake location only.

Algorithm 1. Our basic protocol scheme

1 u_a Finds $k-1$ companions under some area constraints.
2 For each companion u_i
3 u_a Requests the location information to u_i.
4 u_i Sends the information to u_a.
5 u_a Computes the centroid $\bar{U}$.
6 u_a Sends the masked location $\bar{U}$ to the LBS provider and to her companions.

The rest of the paper is organised as follows. Section 2 describes the simplest variant of our proposal. Section 3 presents an evolution of the simplest version based on a public-key privacy homomorphism and briefly elaborates on its security properties. Finally, Section 4 concludes the paper and points out some future work.

2 TTP-Free Location Privacy

In this section we present the basis of our proposal for providing LBS users with location privacy without using a TTP to anonymise them. The main actors of the model are: (i) An untrusted LBS provider P_u whose main task is to give information to users (since it is untrusted, users do not want to share their real location with it); (ii) An LBS user who wants to be anonymised u_a (she has to collaborate with other users in order to get her location anonymised); (iii) A set of k users $U = \{u_1, u_2, \ldots, u_k\}$ consisting of u_a and $k-1$ companions[2]. We assume that a given LBS user is able to obtain her location and to find $k-1$ companions in her cover range[3]. For the sake of brevity we will not discuss this issue in this paper.

Our proposal is based on the computation of a centroid among u_a and her $k-1$ companions so that (i) the position given to the LBS provider is inaccurate but useful enough and (ii) the k users may use the same centroid with the LBS provider so that they become k-anonymous. The scheme of the privacy location protocol without TTP is given in Algorithm 1.

As we have explained, we assume that LBS users are able to interact with $k-1$ companions (Step 1 of Algorithm 1). In the next sections we discuss how users can collaborate to compute a common centroid $\bar{U}$ in order to achieve anonymity.

2.1 Computing the Centroid

Once the user u_a has received the location information of her companions, she computes the centroid $\bar{U}$ by using Equation 1,

$$\bar{U} = \left(\frac{\sum_{i=1}^{k} x_i}{k}, \frac{\sum_{i=1}^{k} y_i}{k} \right) \quad (1)$$

[2] We use the term companion to define a user $u_i \in U | u_i \neq u_a$ collaborating with u_a to anonymise her location.

[3] In the case in which the user cannot perform this task, a variety of methods based on the collaboration with other users could be used.

where (x_i, y_i) is the location of each user in U. After computing the centroid $\bar{U}$, u_a sends it to all her companions and to the LBS provider. Using $\bar{U}$, u_a identifies herself whilst her real position remains hidden. Note that, as we assume that the companions U are in the same cloaking region, the centroid is accurate enough to let u_a obtain useful information from P_u.

This approach is easy and computationally cheap. However, it cannot be really applied because all the messages are sent to u_a in plain text, thus, u_a knows the exact location of all her companions. In this case, if u_a is a malicious user, the location of all the companions is in jeopardy.

This approach has the problem that all the companions must trust u_a because they send her their real locations. Although there are real scenarios in which this situation is possible, in many situations, users may prefer to hide their real location from the others.

2.2 Masking the Locations

In order to prevent u_a from knowing the location of her companions, we extend the previous centroid computation scheme by the addition of Gaussian noise with null average $\sim N(0, \sigma)$. By using a Gaussian pseudo-random numbers generator, each companion u_i can obtain a pair of numbers N_i^x and N_i^y following the desired distribution. Then, these values are added to the real location as Equation 2 shows:

$$(\hat{x}_i, \hat{y}_i) = (x_i, y_i) + (N_i^x, N_i^y) \tag{2}$$

Once the real locations of the users are masked, they can be freely sent to u_a in order to let her compute the centroid $\bar{U}$.

$$\bar{U} = \left(\frac{\sum_{i=1}^{k}(x_i + N_i^x)}{k}, \frac{\sum_{i=1}^{k}(y_i + N_i^y)}{k} \right) = \left(\frac{\sum_{i=1}^{k} x_i}{k}, \frac{\sum_{i=1}^{k} y_i}{k} \right) + \left(\bar{N^x}, \bar{N^y}\right)$$

Finally, u sends her masked location $\bar{U}$ to the LBS provider and to her companions. Note that $\bar{N^x} \approx 0$ and $\bar{N^y} \approx 0$ when k is big enough. Thus, the final centroid is properly computed, although the real locations of the users are masked with noise. Unlike the plain approach, the one described in this section is robust against a malicious user because she does not actually know the location of the other users but the masked ones.

Unfortunately, this approach has a limitation. Due to the use of Gaussian noise with null average, users should not use the same technique repeatedly without changing their real location because of the cancellation of the added noise (*i.e.* the average masked location of these users tends to the real location).

3 Location Privacy Based on a Public-Key Privacy Homomorphism

To avoid the limitation of the aforementioned approach, it is necessary to prevent u_a from knowing the $(\hat{x}_i, \hat{y}_i)$ values of their companions, whilst allowing the

computation of a valid centroid $\bar{U}$. To that end, we make use of a public-key privacy homomorphism.

Privacy homomorphisms (PH) were formally introduced in [7] as a tool for processing encrypted data. Basically, they are encryption functions $E_k : T \rightarrow T'$ which allow to perform a set F' of operations on encrypted data without having knowledge of the decryption function D_k. The security gain is obvious because classified data can be encrypted, processed by an unclassified computing facility, and the result decrypted by the classified level.

The PH used must be additive[4]. This results in a third party being able to add values but not being able to know which values are being added. In addition, the PH used must be public-key (only the owner of the secret key is able to retrieve the result of the addition) and probabilistic (the encryption algorithm E_k randomly chooses the encrypted value from a set of possible values). For instance, the Okamoto-Uchiyama [5] public-key cryptosystem is probabilistic and has an additive homomorphic property.

3.1 Our Proposal

Let us assume that there is a PKI supplying to LBS providers public keys of a public-key probabilistic and additive privacy homomorphism. This privacy homomorphism has both encryption and decryption functions $E_{pk}(\cdot)$ and $D_{sk}(\cdot)$. The operation mapping the addition of the encrypted values is $\Gamma(\cdot)$. Let P_u be an untrusted LBS provider whose secret and public keys in the PKI are sk_{P_u} and pk_{P_u} respectively.

Algorithm 2 details all the steps to be taken in order to hide the location of u_a by means of a public-key privacy homomorphism.

By using our protocol we allow the companions of u_a to securely send several times their locations to her, even when they remain in the same location. Assuming that u_a cannot decrypt the locations sent by her companions, she cannot see the locations. Moreover, thanks to the probabilistic property of the utilised privacy homomorphism, a malicious user u_a is not able to track a static companion. Our protocol is robust against the collusion of u_a and P_u when the companions are in movement.

3.2 Security

We next briefly summarise the security of our proposals. To do so, we check the proposed protocols in two different scenarios with malicious users:

– **Scenario with a malicious user u_a.** Generally, the companions of u_a do not trust her enough to give her their location in plain text. Thus, if they protect their location by adding Gaussian noise with null average, they cannot take part into several anonymity procedures if they do not change their

[4] A privacy homomorphism is additive when one of the operations of F' maps the addition of the unencrypted values.

Algorithm 2. The complete protocol scheme

1 User u_a initiates a location request with each of her companions $u_i \in U$, sending the message $\{Id_{P_u}, u_a\}$, where Id_{P_u} is the identifier of P_u.
2 Upon receiving the message, each companion u_i makes use of the PKI to request the public key of the LBS provider Id_{P_u} to a certification authority.
3 By using the PKI, u_i gets pk_{P_u}, signed by a certification authority.
4 u_i Checks the validity of pk_{P_u} and sends to u_a the message $\{E_{pk_{P_u}}(\hat{x_i}), E_{pk_{P_u}}(\hat{y_i})\}$, where $\hat{x_i}$ and $\hat{y_i}$ is the location of u_i masked according to Expression 2.
5 u_a Makes use of the PKI to request pk_{P_u} and she encrypts her masked location.
6 u_a Performs the operation which results in the addition of the unencrypted values, although she is not able to see them:

$$\bar{U}^{\Gamma} = \left(\Gamma_{i=1}^{k} E_{pk_{P_u}}(\hat{x_i}), \Gamma_{i=1}^{k} E_{pk_{P_u}}(\hat{y_i})\right) = \left(E_{pk_{P_u}}(\Sigma_{i=1}^{k}\hat{x_i}), E_{pk_{P_u}}(\Sigma_{i=1}^{k}\hat{y_i})\right)$$

7 u_a Sends to P_u the message $\{\bar{U}^{\Gamma}, k\}$. She also sends the message to her companions, so they can use the same message to identify themselves.
8 Finally, P_u decrypts $\bar{U}^{\Gamma}$, obtains the values $\Sigma_{i=1}^{k}\hat{x_i}$ and $\Sigma_{i=1}^{k}\hat{y_i}$ and divides them by k to obtain the centroid. Note that P_u is the only one able to perform this operation because it is the only one who knows sk_{P_u}.

location (because of the cancellation of the added noise). If a given companion is not changing her location during several anonymity procedures she must use the privacy homomorphism proposed in order to prevent u_a from seeing the masked locations and to track her. In this scenario, the plain sending of locations is not secure enough. It is necessary to mask the locations with some Gaussian noise and the companions must be in movement (cf. Section 1). If the companions are static, they have to use the proposed privacy homomorphism to guarantee they location privacy (cf. Section 2).

– **Scenario with a malicious user u_a colluded with P_u.** In this scenario, u_a is colluded with P_u and, thus, she knows the keys pk_{P_u} and sk_{P_u}. A particular case of this scenario is the one in which P_u acts as u_a. Hence, u_a is able to decrypt the messages sent by her companions and she can see the masked locations. In this case, the companions must be in movement in order to guarantee their location privacy. Although the collusion of u_a and P_u is unlikely to happen, if P_u has been found to be colluded with u_a, pk_{P_u} could be revoked and P_u could be put in a blacklist. In this scenario, using a privacy homomorphism is not enough because u_a has access to sk_{P_u}. Thus, it is necessary to be in movement to avert the revelation of the real location due to the noise cancellation.

4 Conclusions and Future Work

In this paper we have proposed a new method for providing the users of LBS with location privacy. Our method is based on a public-key privacy homomorphism

and, unlike the existing proposals, it averts the use of a trusted third party. Our method relies on a basic scheme, consisting in the computation of the average location of a set of k users, and improves it in the sense that it guarantees the location privacy of the users and the location exchange among them by using a public-key privacy homomorphism. Although privacy homomorphisms are widely used in a variety of areas, we believe that a goal of this paper is to propose an algorithm for using them to solve the privacy aspects related to LBS. Future work includes improving the proposed method to allow static users to take part in several anonymity procedures and studying different collusion scenarios, with more than one user colluded with the LBS provider.

Disclaimer and Acknowledgements

The authors are solely responsible for the views expressed in this paper, which do not necessarily reflect the position of UNESCO nor commit that organisation. This work was partly supported by the Government of Catalonia under grants 2002 SGR 00170 and 2005 SGR 00446, by the Spanish Ministry of Education and Science under project SEG2004-04352-C04-01 PROPRIETAS. The authors thank the insightful comments of the reviewers, which helped in improving the article although it has been shortened.

References

1. Domingo-Ferrer, J., Sebé, F., Solanas, A.: A polynomial-time approximation to optimal multivariate microaggregation. In: Computer and Mathematics with Applications (in press, 2007)
2. Gedik, B., Liu, L.: A Customizable k-Anonymity Model for Protecting Location Privacy (2004)
3. Gruteser, M., Grunwald, D.: Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking. In: Proceedings of the First International Conference on Mobile Systems, Applications, and Services, pp. 31–42 (2003)
4. Mokbel, M.F.: Towards Privacy-Aware Location-Based Database Servers. International Workshop on Privacy Data Management, PDM (April 2006)
5. Okamoto, T., Uchiyama, S.: A new public-key cryptosystem as secure as factoring. Lecture notes in computer science, pp. 308–318
6. Bertino, E., Cheng, R., Zhang, Y., Prabhakar, S.: Preserving user location privacy in mobile data management infrastructures. In: Proceedings of Privacy Enhancing Technology Workshop (PET) (2006)
7. Rivest, R.L., Adleman, L., Dertouzos, M.L.: On data banks and privacy homomorphisms. Foundations of Secure Computation, 169–178 (1978)
8. Sweeney, L.: k-anonymity: A model for protecting privacy. Fuzziness and Knowledge Based Systems 10(5), 557–570 (2002)

A Critical View on RFC 3647

Klaus Schmeh

cv cryptovision gmbh, Munscheidstr. 17,
45886 Gelsenkirchen, Germany
klaus.schmeh@cryptovision.com

Abstract. A Certification Practice Statement (CPS), as well as one or several Certificate Policies (CP) are important parts of a Public Key Infrastructure. The by far most important source of information for writing a CPS or CP was developed by an IETF working group and was published as RFC 3647 [1]. RFC 3647 can be thought of as a generic instruction set for creating a CPS and a CP. Yet, experience shows that working with RFC 3647 can be quite difficult. This is due to some fundamental issues, but also due to some shortcomings and faults in the standard. In addition, it is difficult to use RFC 3647 for a CPS/CP that is used outside the US. This paper names the main problems that a CPS/CP author has to face when following RFC 3647. It discusses possible solutions and reveals why the development of a new standard would be appropriate.

Keywords: PKI, Certification Practice Statement, CPS, Certificate Policy, CP.

1 Introduction

RFC 3647 is the successor of RFC 2527 [2] and can therefore be regarded as a second version document. RFC 3647 is an Informational RFC, which implies that it has no official standard status. Anyway, it will be called a standard in this document, because it is a de-facto standard (obviously, for a paper document the existence of a well-specified standard is less important than for a software solution). Virtually every CPS/CP author uses RFC 3647 as a source of information. To my knowledge there are no relevant alternatives to RFC 3647 today.

The core of a CPS or CP according to RFC 3647 is a so-called „Set of Provisions“. RFC 3647 lists about 58 provision names that may or may not be used by a CPS/CP author. The provisions are grouped into nine chapters. In addition to the provision names, RFC 3647 also describes details about how a provision might look like in an actual CPS/CP. Yet, RFC 3647 doesn't list any provisions itself, nor does it require that certain provision types are present in a CPS/CP. RFC 3647 therefore can be thought of as a toolkit for CPS/CP documents with the provisions as generic building blocks.

Most CPS/CP authors don't keep exactly to the structure proposed in RFC 3647. Usually, RFC 3647 is only used as a rough guideline and as a general source of information. In my view, this is an unlucky situation, because it would have major advantages to have CPS/CP documents that are structured in the same way and that

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 369–374, 2007.

have an analogous content. This would make CPS/CP comparisons a lot easier, and, most of all, would provide for easy policy mapping like it is described in the X.509v3 and the PKIX standards.

One reason for this undesired situation is, of course, the complexity of the PKI topic itself. Yet, RFC 3647 is in my view more complex than necessary, and it has a number of serious disadvantages. The scope of this paper is to name the problems that arise when using RFC 3647. All the problems mentioned have a close relation to practice. This paper is based on experience gathered in more than 20 PKI projects with about a dozen CPS/CP documents.

2 Structure Problems

The structure RFC 3647 suggests for a CPS/CP document is not optimal. In my view, an RFC 3647 set of provisions doesn't provide an easy, intuitive way to find the information the reader is looking for. I see the following reasons for this:

- It is usually considered best-practice to group the description of an IT system into three parts: components, roles, and processes. In many cases one or two additional parts, like "Introduction" or "Policy Issues", may be useful. Yet, RFC 3647 follows a completely different approach. It uses a nine chapter structure: "Introduction", "Publication and Repository Responsibilities", "Identification and Authentication", "Certificate Life-Cycle Operational Requirements", "Facility, Management, and Operational Controls", "Technical Security Controls", "Certificate, CRL, and OCSP Profiles", "Compliance Audit and Other Assessment", and "Other Business and Legal Matters". In my view, it is a lot harder to find the desired information in this structure than it is in a conventional components-roles-processes document.
- Some of the nine chapters recommended in RFC 3647 are in practice very short or even empty. E.g., chapters 2 and 8 don't have subchapters and are therefore usually quite short. Chapter 9 is usually very short, too, because most of its provisions are not relevant in practice. On the other hand, chapters 3, 4, 6 and 7 may grow quite comprehensive. A components-roles-processes structure would avoid such differences in length.
- As there is no dedicated chapter for processes, it is not possible to get an overview on the PKI processes specified by a RFC 3647 document at first sight. Instead, process descriptions are spread to several chapters (4, 5 and 6), which is not very intuitive.
- The situation with roles is similar as with processes. It is not intuitively clear, where roles are described. To be precise, RFC 3647 not even mentions the word "role". Instead, it uses the term "participants". Participants can be described as a part of the introduction subchapter. I consider this a major shortcoming, because roles are important in a PKI and should not be presented as introductory information, but as a substantial part of a CPS/CP.
- In RFC 3647, PKI components are covered even worse than PKI processes and PKI roles. There is simply no provision at all, that is designed to contain a description of all relevant components. Thus, a CPS/CP author has to hide the complete PKI architecture description in the introduction subchapter named "Overview", which doesn't grant this topic the importance it deserves.

- The most important process in a PKI is user enrolment. As the security and usability of a PKI heavily depend on a well-designed user enrolment, it is clear that a CPS/CP document should cover this issue in detail. Unfortunately, RFC 3647 doesn't mention an enrolment process as such. Instead, it separates the steps of an enrolment and spreads it into four subchapters ("Certificate Application", "Certificate Application Processing", "Certificate Issuance" and "Certificate Acceptance"). Again, this is by no means intuitive. In addition, this situation is hard to handle for a CPS/CP author, because enrolment is basically one process, which very often cannot be easily cut into four pieces.
- Another non-intuitive part of the RFC 3647 CPS/CP structure is the renewal of a certificate or key. Three possible renewal processes are mentioned: Certificate Renewal (certificate content and key stay the same), Certificate Re-key (key changes, certificate content stays the same), and Certificate Modification (certificate content changes, key stays the same). Yet, no exact definition of these processes is given. Especially, it is not clear, if it is allowed to change the validity date and the serial number at a Certificate Re-Key (if it is allowed, the definition is wrong; if it is not allowed, it doesn't make sense).
- RFC 3647 requires a separation of the CP and CPS documents. Yet, my experience is that most PKI operators prefer to have only one document containing all necessary policy information. Yet, RFC 3647 doesn't support such a single policy document.
- When writing a CPS/CP document, it is always an important question, if a part of the CPS/CP or even the whole document shall be confidential. RFC 3647 proposes that confidential information should be swapped out into an additional document named "PKI Disclosure Statement" (PDS). It is clear, that such a document adds additional complexity to the whole project. Most PKI operators prefer to have only non-confidential information in a CPS/CP. Confidential issues should be treated in an operational statement, which is an internal document.

3 Structure Problems

RFC 3647 is not only mal-structured, but also has a number of serious shortcomings:

- The expressions Certification Practice Statement and Certificate Policy themselves were not optimally chosen by the RFC 3647 creators. The problem is that the abbreviations of the two words look very similar. The plural of CP is CPs, which is the same sequence of letters as CPS. This is confusing for laymen and even for experts. A better choice would have been CAP (CA Policy) instead of CPS, and DCP (Digital Certificate Policy) instead of CP.
- Some important parts of a well-designed CPS/CP are missing in RFC 3647. As already mentioned, no chapters for components and roles are provided. In addition, an RFC 3647 CPS/CP doesn't feature a literature list and a management summary. Another minor shortcoming: The abbreviation list of RFC 3647 is not complete; it lacks the expression PDS (PKI Disclosure Statement) mentioned above.
- It is a basic requirement that a CPS always should mention the CPs that are supported by the respective CA. Amazingly, RFC 3647 doesn't specify a place in a CPS, where such a link can be set.

- The well-known software PGP and the certificates it provides (PGP-certificates) are not mentioned in RFC 3647. Of course, this is somehow logical, as PGP-certificates by definition don't belong into an X.509 CPS/CP. Yet, most PKI operators don't care about the separation between X.509 and PGP, and wish for a CPS/CP that mentions both.

I hope that most of the shortcomings will be corrected in the successor of RFC 3647. Yet, I don't expect that CPSs and CPs will be renamed, because it would lead to too much confusion.

4 Outdated Content

RFC 3647 was published in 2003. As PKI is a field that changes continuously, there are quite a few topics that came up in the last three years and that are naturally not mentioned in the RFC. Here are the most important ones:

- Many PKI operators require that private keys be stored on a server as so-called "roaming keys". The concept of roaming keys has substantially gained importance in the last few years and must therefore be considered in many CPS/CP documents. Yet, RFC 3647 doesn't even mention roaming keys. This is a shortcoming, because roaming keys impose some major security problems that should not be ignored by CPS/CP authors.
- Short-lived certificates, which need no revocation checking, are another topic that has become more and more popular in the recent past. The concept of short-lived certificates is not mentioned in RFC 3647.
- One of the currently hottest issues in the field of Public Key Infrastructures is Identity Management (IM). Virtually all vendors of lM solutions either offer their own PKI tools (e.g. Microsoft and IBM) or have a partner selling an integrated PKI solution (e.g. Novell). PKI-vendors like Entrust position themselves as suppliers of integrated PKI-IM solutions. Many analysts and IT experts even consider PKI functionality more a feature of IM than a topic of its own. This development is currently changing the PKI market and influences PKI products, as well as PKI projects. Because of this situation, it must be considered a disadvantage that RFC 3647 does not even mention Identity Management, let alone describe how an IM integration can influence processes like enrolment and revocation.
- Another expression not even mentioned in RFC 3647 is "Smart Card". This must be considered a shortcoming, because Smart Cards play an increasingly important role in the field of PKI. Several countries are conducting national card projects with PKI enabled Smart Cards. Companies and authorities setting up a PKI get more and more interested in Smart Cards, because they are more secure and allow for additional non-PKI applications. One problem in this context is that Smart Cards require additional components (e.g. card printer, card management software) and additional processes (e.g. unblocking a card after three wrong PINs have been entered). RFC 3647 doesn't address any of these aspects and must therefore be regarded as a framework for cardless PKIs.

It is clear that most of the mentioned shortcomings can easily be corrected in the next version of RFC 3647. Yet, it might last another two to three years, before a RFC 3647 successor will be published. This arises the question, if an RFC is at all a suitable medium for publishing instructions on CPS/CP writing. It might be better to realise such an instruction set as a document that is updated regularly. While such an approach might lead to confusion and incompatibilities in case of a software, it sounds like the most appropriate way to publish a best-practice document.

5 Problems with Non-english CPS/CP Documents

When using RFC 3647 for writing a non-English CPS/CP document, there are a a few more problems to deal with. Of course, these problems cannot be blamed to the RFC 3647 creators, because they had to focus on international issues. Anyway, the problems exist. Here are the most important ones:

- All RFC 3647 provisions are only available in English. There are no official translations to other languages I am aware of. This is a problem for virtually any CPS/CP writer in countries like Germany, France or Japan, because the legal situation or other constraints require a CPS/CP in the respective language. Of course, it is no problem to find an appropriate translation for a provision name; the problem is that there is no standard and that PKI users and operators therefore have to deal with a confusing variety of provision names.
- There is a similar problem regarding the definitions made in RFC 3647. All of them are English; there are no standard translations. For example, there are at least three different German words for a CA.
- RFC 3647 doesn't mention the European Signature Directive [ESig]. This is a problem, as the EU Signature Directive is currently the most important law of its kind in the world.

It is obvious that RFC 3647 has a focus on the US and on English-speaking countries. Of course, it is not possible to change this situation completely, as a standard minding German, French, Japanese and other language and legal requirements would be not feasible. Yet, it is important to note that writing a CPS/CP in a language different than English can be a major challenge. In my view, it is an important task for national IT security agencies and industry organizations to work out CPS/CP standards that fit with the local requirements.

6 Conclusion and Possible Solutions

The concept of RFC 3647 is by far not optimal. RFC 3647 CPS/CP documents are structured in a non-optimal way, and they lack some important content. In addition, RFC 3647 misses a few up-to-date topics, and it is not well-suited for CPS/CP authors writing in other languages than English. When a simple CPS/CP document is desired, which is very often the case in practice, RFC 3647 is too complex.

For the reasons mentioned, there is a lot of fundamental work to be done in the near future. Of course, RFC 3647 needs to be updated in the next few years in order to

fix some of the shortcomings mentioned in this paper and to update it with recent developments. Issues like Smart Cards, roaming keys and Identity Management have to be addressed.

Yet, updating RFC 3647 will only solve a part of the problems. Therefore, additional work will have to be done by national IT security organizations (authorities or industry organizations) in order to adapt RFC 3647 to local requirements. This includes compulsory translations of the provision names and of the main technical expressions. In addition, national laws have to be considered, when assembling standards for the local use of CPS/CP documents. Most of all, the European Signature Directive, which is valid for about 460 Million people, should be minded.

Apart from this, I see the necessity for an alternative CPS/CP standard, which could be called Simple CA Policy (SCAP). An SCAP would be document that can be used instead of a CPS/CP and would be much simpler. I expect that more PKI operators would be interested in such an SCAP than in an RFC 3647 CPS/CP. SCAP should not be published as RFC, because RFCs are static documents that don't change for years. Instead, SCAP should be treated as a document that is updated regularly.

As a CPS/CP author can't wait until the mentioned changes will come, he or she must find a strategy to live with the current situation. I usually write CPS/CP documents like this:

- There is only one document containing CPS and CP information.
- In addition to the CPS/CP document there is a statement describing the whole PKI architecture with all processes and roles. This statement is confidential. The CPS/CP document only contains information that can be published to the PKI users.
- Although I don't like the RFC 3647 structure, I keep to it, in order to enable easily comparable documents.
- I always list components in the "Overview" subchapter of chapter 1.
- I describe roles in the "Participants" subchapter.
- I usually add a literature table in the "Definitions and Acronyms" subchapter.
- I define a user enrolment process as one piece. Then I cut it into four pieces to match the RFC 3647 structure.
- ·My CPS/CP documents always contain a lot of empty chapters. In the introduction, I mention, why this is the case.

With this strategy, I usually manage to write CPS/CP documents that are readable, although they adhere to the RFC 3647 structure. Anyway, I hope that a better standard will be available one day.

References

1. Chokhani, S., Ford, W.: Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework. RFC 2527 (1999)
2. Chokhani, S., Ford, W., Sabett, R., Merrill, C., Wu, S.: Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework. RFC 3647 (2003)
3. Schmeh, K.: Cryptography and Publuc Key Infrastructure on The Internet. John Wiley, Hoboken (2003)

Author Index