Privacy Enhancements for Hardware-based Security Modules

Vijayakrishnan Pasupathinathan[1], Josef Pieprzyk[1], and Huaxiong Wang[2]

[1] Centre for Advanced Computing – Algorithms and Cryptography,
Department of Computing,
Macquarie University,
Sydney, Australia.
{krishnan,josef}@science.mq.edu.au
[2] Division of Mathematical Sciences,
Nanyang Technological University, Singapore.
HXWang@ntu.edu.sg

Abstract. The increasing growth in the use of Hardware Security Modules (HSMs) towards identification and authentication of a security endpoint have raised numerous privacy and security concerns. HSMs have the ability to tie a system or an object, along with its users to the physical world. However, this enables tracking of the user and/or an object associated with the HSM. Current systems do not adequately address the privacy needs and as such are susceptible to various attacks.
In this work, we analyse various security and privacy concerns that arise when deploying such hardware security modules and propose a system that allow users to create pseudonyms from a trusted master public-secret key pair. The proposed system is based on the intractability of factoring and finding square roots of a quadratic residue modulo a composite number, where the composite number is a product of two large primes. Along with the standard notion of protecting privacy of an user, the proposed system offers *colligation* between seemingly independent pseudonyms. This new property when combined with HSMs that store the master secret key is extremely beneficial to a user, as it offers a convenient way to generate a large number of pseudonyms using relatively small storage requirements.

Key words: pseudonyms, anonymity, hardware-based security

1 Introduction

Dedicated hardware-based co-processors performing cryptographic and mathematically intensive operations have been in existence for a long time. Recently, many companies have begun embedding active hardware-based security modules into various computing devices, such as mobile telephones, GPS devices, personal computers and network equipments, to provide a hardware-based endpoint security. One such implementation that is widely popular is the Trusted Platform Module (TPM). Promoted by the Trusted Computing Group [1], the

Trusted Platform Module (TPM) is a tamper-resistant hardware chip that forms the basis of trusted computing. The TPM is a secure chip that provides a hardware-based approach to manage user authentication, network access and data protection. The hardware chip is bound to the motherboard of the computing device, and stores certificates, keys and performs the necessary cryptographic operations. A TPM also enhances multi-factor authentication and complements biometric readers by securely storing the keys associated with the biometric within the module. Because TPMs can be integrated with other traditional enterprise tools, such as an active directory and a Novell eDirectory, they provide a seamless method of integrating certificate-based, biometric and hardware-based authentication techniques.

The deployment of the TPM raises some valid privacy concerns. Privacy protection in a TPM currently involves two mechanisms, Privacy Certification Authority (CA)-based attestation (TPM v1.1) [2] and Direct Anonymous Attestation (TPM v1.2)[3, 4]. In a Privacy CA-based attestation, the authentication is based on the direct use of the TPM's Endorsement Key (EK). This method will compromise the anonymity of the module, because all transactions performed by the same TPM can be linked. Furthermore, it will compromise the anonymity of the user associated with the module, because the users/TPMs activity can be tracked. Based on the security requirement of a non-revealable master public key in a TPM, Brickell in [3] propose a method for direct anonymous attestation (DAA) that provides anonymity for a user, based on the Camenisch-Lysyanskaya credential system [5]. The current solution for the DAA is a computing intensive construction. To complete all cryptographic calculations in real time, the computation has to be distributed between the TPM and its host, typically, the device to which it is attached. This introduces an obvious security weakness in the system. Now the cryptographic computations are not performed only by the trusted hardware security module and are therefore prone to software-based vulnerabilities in the host system that the TPM is intended to protect against. Also, the scheme [3] does not provide a secret key linkability for the pseudonyms that are generated. Consequently, the TPM must maintain a database of those identities and the associated secret keys. This database can get quite large if the TPM serves a large group of users. Typically, the database would have to be stored outside the chip, thus defeating the purpose for which the TPMs were introduced.

Another issue that is often voiced with respect to TPMs is user privacy when using Digital Rights Management (DRM)-based software. A DRM system, when combined with a TPM introduces a new security layer where the encryption keys and certificates can be bound to a specific platform and copies can be limited to only that specified platform. This has raised concerns from privacy groups about the purchase of such a TPM-based DRM enabled software. For example, a primary concern is that the purchase of music or movie content should remain anonymous, protecting the user's privacy. Because the DRM-based software will now employ information about users and their platforms, there are fears that the owners of the software products can employ a tracking system that records

information about the users and their platform, thus linking them to specific content.

In this paper, we try to resolve these privacy issues by proposing a pseudonym system for active devices. The system provides restricted anonymity and supports colligation between a trusted *high value* secret key and newly-generated pseudonyms. We first provide a brief overview of the pseudonym systems and their related work. We then provide the details on the construction of an anonymous certification system and cryptographic techniques that underpins our construction. We then present our pseudonym system and discus the security of our proposed construction. Finally, we discuss the integration of our proposed pseudonym system in a TPM-based setting.

1.1 Pseudonymns

The use of pseudonyms have been proposed as a mechanism to hide a user's identity by providing anonymity, while still being suitable to authenticate the holder of the pseudonym in a communication system [6]. David Chaum argues that using pseudonyms provides a way to allow a user to work anonymously with multiple organisations by allowing the users to obtain a credential from one organisation using their pseudonym and obtain services using that credential from another organisation without revealing their true identity [6]. To this end, Chaum and Evertse developed a pseudonym system and propose a RSA-based implementation while relying on a trusted centre that must sign all credentials [7]. Chen extends the scheme from [6] and presents its discrete-logarithm version that relies on a trusted centre [8]. An advantage of these schemes is that they allow the users to generate pseudonyms, giving the users a greater degree of control over their identity. However, these schemes have a common weakness. Although the identity of a user is hidden, the credentials (such as the certificates of their public key) or pseudonyms can be easily shared (unauthorised transfer) with other users.

Based on the security of preserving a high-value (*master*) secret key, Canetti [9] and Lysayanskaya [10] independently propose non-transferable pseudonym systems. Though credentials obtained on pseudonyms can be used anonymously, the authors of [9] assume that the certification authority (CA) grants credentials only when each user has revealed their true identity to them. This makes their scheme prone to collusion between a CA and a verifier, because the real identity associated with the user pseudonym can be deduced. The scheme from [10] protects against an unauthorised transfer of the user credentials by forcing a user to reveal the master secret key should they choose to share their credentials. But the scheme shares the same weakness as in [9] – during the registration phase, users are required to disclose their true identity (master public key) to a CA.

1.2 Scope and Contribution

This paper presents a pseudonym system that is based on the public key cryptosystem. The main idea is to use a single, trusted master secret key with many

matching public keys (pseudonyms). The proposed system gives users the ability to generate multiple pseudonyms (that are independent of the master public key) from a trusted master secret key. An important property of the system is that it provides the users with the ability to generate signatures using the master secret key, which are verifiable using certificates that were issued against pseudonyms.

To consider an example, a TPM contains a certified public-secret key pair. The public key is certified by its manufacturer and recorded on the TPM chip at the time of manufacturing. The certified public key of the chip can be used to authenticate the machine with the TPM. The TPM is used to further certify public keys of users associated with the machine. A verifier can authenticate a user based on the certificate chain consisting of the user certificate, the TPM certificate and the manufacturer's certificate. However, revealing the identity of the machine to every verifier would not only compromise the anonymity of the machine, but also the anonymity of the user(s) of the machine. It is possible to identify a user using their pseudonyms, but the verifier trusts only the TPM chip's certified public key and not the operating system of the machine or any newly-generated pseudonyms. Therefore, we require a system that gives a user the ability to generate and control the use of the multiple identities, based on a trusted master identity (TPM's certified public key). Here, the pseudonyms should not only be independent of the master identity (anonymity), but also there is a relation between all pseudonyms generated[3] and the trusted master secret key stored in the chip (we call this relation *colligation*).

Anonymity and colligation are in some sense contradictory. Anonymity requires that, it is impossible (at least computationally) for an entity with knowledge of a pseudonym, to link that pseudonym with either the master identity or any other generated pseudonym. Whereas, colligation requires that the prover be guaranteed that there is an underlying link that exists between all pseudonyms (that appear to be unrelated to each other) was generated from the trusted master secret key. Previously published proposals like, [11, 10, 5, 8, 9, 6] that achieved anonymity have considered a user's identity that consists of public-secret key pair as a single unified structure. Under such assumption it is unfeasible to obtain both anonymity and colligation. We aim to segregate the structure and provide anonymity to a user but still maintain colligation between pseudonyms generated using the user's master secret key. The implication of this structure is that, a user's master secret key becomes highly valuable, as all his pseudonyms are linked directly to the secret key.

2 Anonymous Certification System

User anonymity and colligation between the master secret key and the user-generated identities is of paramount importance. To provide anonymity to the

[3] To a certifier it is essential that the system provides a guarantee that all pseudonyms from a particular TPM can be traced back to a single secret key; but a verifier needs proof of this binding between the master secret key and only the pseudonym that are currently presented with. We do not make this distinction here.

user-generated identities (pseudonyms), our proposal will make use of an anonymous certification scheme, such as a scheme with blind signatures. An anonymous certification system is necessary to provide anonymity to a user and to prevent collusion between a certifier and a verifier. To this end, we will employ the modified blind signature scheme (refer Section 3.4) proposed by Pointcheval [12]. Note that any anonymous certification scheme that supports non-transferability and the revocation of anonymity can be employed with some necessary modifications. To provide colligation between the generated pseudonyms and the master secret key, we can use any one-way function. In our construction we use a squaring modulo a composite integer. In this section, we first we describe the model of an anonymous certification scheme that will provide certificates for user-generated identities (pseudonyms). In the remainder of this section, we summarise the main cryptographic building blocks used in our constructions.

The anonymous certification system (ACS) represents the certification process of a public key by a certifier who does not know the public key. This is essentially a Chaum blind signature [13] on the public key of the user, that is, it provides anonymity to the receiver[4].

A typical ACS consists of four entities and three protocols. The entities are, a user $\mathcal{U}$, a verifier $\mathcal{V}$, a certifier $\mathcal{C}$ and a trustee (tracer) $\mathcal{T}$. The protocol suites include: a *certification protocol*, where $\mathcal{U}$ interacts with $\mathcal{C}$ to obtain a certified pseudonym, that is, the pseudonym is blindly signed; an *identification protocol*, where $\mathcal{V}$ interacts with $\mathcal{U}$ to authenticate $\mathcal{U}$'s credential and provide services; a *trace protocol*, where $\mathcal{T}$ participates and is invoked to trace the real identity associated with $\mathcal{U}$'s pseudonym.

System setting The user, $\mathcal{U}$, chooses a modulus N_i, such that a $N_i = p_1^{(i)} p_2^{(i)}$, is a product of two distinct large primes each congruent to 3 (mod 4), ($p_1^{(i)}, p_2^{(i)}$ are Blum integers [14]), an element $g \in \mathbb{Z}_{N_i}$ whose order is $\phi(N_i) = (p_1^{(i)} - 1)(p_2^{(i)} - 1)$ and where i is the number of pseudonyms. We also require the modulus for pseudonyms to be different; otherwise anonymity can be compromised trivially by merely maintaining a list of modulus. The user chooses a master secret key $SK_{\mathcal{U}_0} \in \mathbb{Z}_{N_0}$ and publishes the master public key $PK_{\mathcal{U}_0} = g^{SK_{\mathcal{U}_0}} \mod N_0$ (which represents the user's true and public identity). The certifier $\mathcal{C}$ publishes its public key $PK_{\mathcal{C}} = g^{SK_{\mathcal{C}}} \mod N_c$ while keeping the corresponding secret key private. The certifier also publishes the public key of the Trustee $\mathcal{T}$, (for tracing and revocation) which would be of the form $PK_{\mathcal{T}} = g_1^{SK_{\mathcal{T}}} \mod N_T$, where $g_1 \in \mathbb{Z}_{N_T}$. Every user registers with a certification authority to obtain a certificate of the form $\texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_0}\rangle$.

Protocol Certify The certification involves two steps, the certification of the master public key and the certification of the pseudonyms. In an TPM-based

[4] Whereas, group signature schemes as employed by [3] provide anonymity to the source.

setting the master public key is certified by the manufacturer, and the following describes the certification of the pseudonyms.

The user, $\mathcal{U}$, generates pseudonyms of the form $(PK_{\mathcal{U}_1}, \ldots, PK_{\mathcal{U}_l})$ using the identity generation process described in Section 3.3. The users then identify themselves (using the master public key) to the certifier and engages in a *certify* protocol to obtain a certificate for a pseudonym $PK_{\mathcal{U}_i}$. The value of $PK_{\mathcal{U}_i}$ is never revealed to the certifier. We shall express this phase as

$$(PK_{\mathcal{U}_i}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle) \leftarrow Certify(\mathcal{U}, \mathcal{C}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_0}\rangle)$$

that is, ' $\mathcal{U}$ engages in the certify protocol with $\mathcal{C}$ using $\texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_0}\rangle$ to obtain a certificate on $PK_{\mathcal{U}_i}$, $\texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle$'.

Protocol Identify A user $\mathcal{U}$ who wishes to avail services offered by a verifier $\mathcal{V}$, engages in an identification protocol to convince that they possess the necessary credentials. We shall express this phase as

$$\langle \texttt{PROOF}_{\mathcal{U}_i}\rangle \leftarrow Identify(\mathcal{U}, \mathcal{V}, PK_{\mathcal{U}_i}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle, PK_{\mathcal{T}})$$

that is, ' $\mathcal{U}$ engages in an identification protocol with a verifier $\mathcal{V}$ using the psuedonymn $PK_{\mathcal{U}_i}$ and $\texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle$ and which contains the encryption of the identity under the public key $PK_{\mathcal{T}}$'.

Protocol Trace A verifier who needs to trace the identity of the user contacts the trustee $\mathcal{T}$ by providing the transcript from an identification protocol $\langle \texttt{PROOF}_{\mathcal{U}_i}\rangle$. We shall express this phase as

$$(PK_{\mathcal{U}_0}) \leftarrow Trace(\mathcal{V}, \mathcal{T}, PK_{\mathcal{U}_i}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle, \langle \texttt{PROOF}_{\mathcal{U}_i}\rangle)$$

that is, ' $\mathcal{V}$ engages in the tracing protocol with $\mathcal{T}$ using the values $PK_{\mathcal{U}_i}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle$ and proof of identity use $\langle \texttt{PROOF}_{\mathcal{U}_i}\rangle$ to obtain the master identity $PK_{\mathcal{U}_0}$'.

3 Pseudonym System Colligated with Master Secret Key

We first outline our security assumptions and cryptographic tools used, and then present our scheme that consists of four phases, identity generation, certification, identification and trace.

3.1 Assumptions

Our system relies on the following assumptions:

- **Assumption 1** (*Factoring*) For any probabilistic poly-time algorithm $\mathcal{G}$ that on input $1^{|N|}$ produces factors of N, where N is a composite of two prime number, p_1 and q_1, such that for any probabilistic polynomial time algorithm $\mathcal{A}$, the probability that $\mathcal{A}$ can factor N is negligible, that is, the probability of its success is negligible in the length of $\frac{1}{poly(|N|)}$.

- **Assumption 2** (*Square Root*) for any probabilistic polynomial-time algorithm $\mathcal{B}$ that on input N and a, where N is a composite of two prime numbers, p_1 and q_1 and $a \in \mathrm{QR}_N$ is a quadratic residue, the probability that $\mathcal{B}$ can output b, such that $b^2 \equiv a \mod N$ is negligible, that is, the probability of success is smaller than $\frac{1}{poly(|N|)}$.
- **Assumption 3** (*Square Decisional Diffie-Hellmann*) The square decisional Diffie-Hellman (SDDH) problem is the task of distinguishing between the triple (g, g^a, g^{a^2}) from the triple (g, g^a, g^r), where r is random and a uniformly-chosen integer from $\{1, \ldots, N-1\}$. We assume that there exists no probabilistic polynomial-time algorithm $\mathcal{G}$ that can distinguish the triplets with the probability better than $\frac{1}{2} + \frac{1}{poly(|N|)}$.

We also use the Chaum and Pederson construction [15] as a sub-protocol for an interactive proof of knowledge for the discrete log problem (DL-EQ). Their protocol [15] was designed for the case when a group of the exponents has a prime order, whereas, in our protocol, the group of the exponents have a composite order. However, as suggested in [16], the proof of knowledge of a discrete logarithm from different groups (DL-EQ) holds even when working over a cyclic sub-group of $\mathbb{Z}_N^*$. We combine the DL-EQ with the El-Gamal encryption over a composite modulus [17] to encrypt the master identity of the user under the public key of the trustee, verifiable by the certification authority.

3.2 System Setting

The system involves four entities. A user $\mathcal{U}$ who holds a long-term certified public key $PK_{\mathcal{U}_0}$ (we shall call it the master public key), and wishes to hide his identity from a verifier $\mathcal{V}$. The public keys are certified by a certification authority $\mathcal{C}$ and a trustee $\mathcal{T}$ responsible for tracing the pseudonym used by the user.

The $\mathcal{U}$ master public-secret key-pair is generated as in Section 2. $\mathcal{U}$ then obtains a certificate on the master public key $PK_{\mathcal{U}_0}$ from a certification authority $\mathcal{C}$, which represents the $\mathcal{U}$'s true identity.

The public key of the certification authority is $PK_{\mathcal{C}} = g^{SK_{\mathcal{C}}}$ and public key of the trustee is $PK_{\mathcal{T}} = g_1^{SK_{\mathcal{T}}}$, where $SK_{\mathcal{C}}$ and $SK_{\mathcal{T}}$ are the corresponding secret keys for the certification authority and the trustee respectively.

3.3 Identity Generation

$\mathcal{U}$ generates new identities using the following key generation process, which takes the inputs, N_j, g, a counter value i (indicating the total number of new identities being generated), identity level l (number of identities generated previously) and the master secret key $SK_{\mathcal{U}_0}$.

`I-Generation`(g,i,l,$SK_{\mathcal{U}_0}$)
`For` $j = l,\ldots,i$ `do` $PK_{\mathcal{U}_j} = g^{SK_{\mathcal{U}_0}^{2^j}} \mod N_j$ `EndFor`

$\texttt{Return}(PK_{\mathcal{U}_l},\ldots,PK_{\mathcal{U}_j})$

During the first run, the value of the identity level l would be 1 and the counter value i is the number of new identities $\mathcal{U}$ requires. Further calls to the key generation, the identity level would be the counter value that was used during the previous run ($l' = i$). An implicit requirement is that, $\mathcal{U}$ should keep track of the values i and l as long as the master public key remains valid.

We could (and do) treat the identities generated as public keys, that are of the form $(PK_{\mathcal{U}_l},\ldots,PK_{\mathcal{U}_i}) = (g^{SK_{\mathcal{U}_0}^{2^l}},\ldots,g^{SK_{\mathcal{U}_0}^{2^i}})$

3.4 Certification

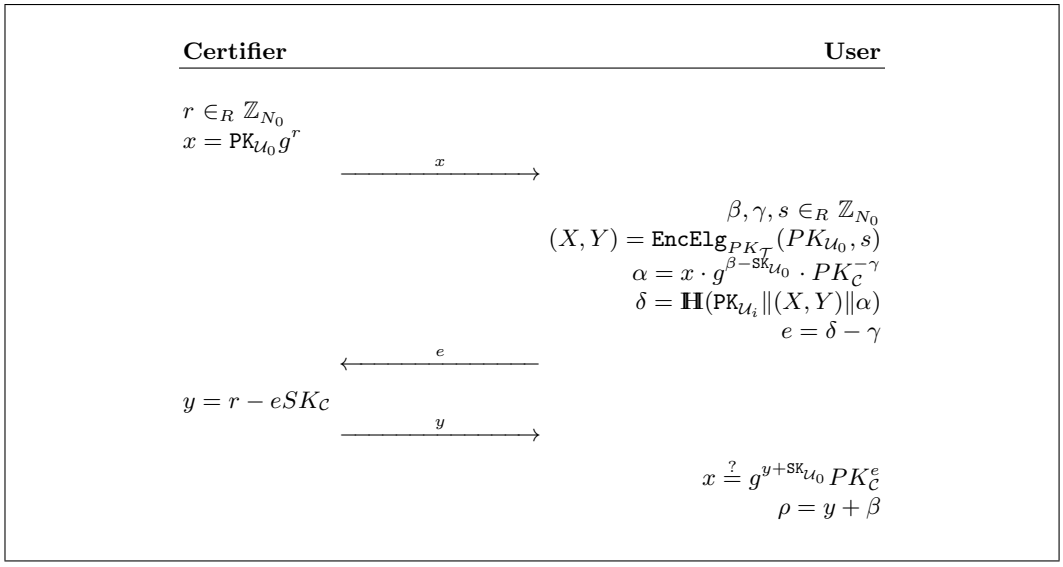

Fig. 1. Modified Blind Certification Protocol of [12]
The signature on $PK_{\mathcal{U}_i}$ is (α, δ, ρ) and a receiver can verify using the relation $\alpha \stackrel{?}{=} g^{\rho} \texttt{PK}_{\mathcal{C}}^{\delta}$

The newly-generated public keys ($PK_{\mathcal{U}_1}$, ..., $PK_{\mathcal{U}_l}$) are required to be certified by $\mathcal{C}$ before they can be used. It is possible to use a normal certification procedure currently employed in public key crypto-systems, where the public key $PK_{\mathcal{U}_i}$ is signed by $\mathcal{U}$ using the master secret key $SK_{\mathcal{U}_0}$ and sent to $\mathcal{C}$ for certification. $\mathcal{C}$ verifies the signature using the master public key $PK_{\mathcal{U}_0}$. On a successful verification, $\mathcal{C}$ digitally-signs using his private key $SK_{\mathcal{C}}$ and sends the certificate to $\mathcal{U}$. This method is quite straightforward, but certain applications (for example, applications based on TPM) require the new identities to be protected even from the certifier. So, we propose a modification to the certification scheme

based on a blind signature scheme using a composite modulus by Pointcheval [12]. The blind signature scheme now includes the master public key of the user that is used by the certifier to form the commitment and is later verified by the user.

The certification process is represented by:

$$(PK_{\mathcal{U}_i}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle)$$

$$\leftarrow Certify(\mathcal{U}, \mathcal{C}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_0}, (X, Y)\rangle)$$

where, $\texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle$ is the valid blind signature $(PK_{\mathcal{U}_i}, \alpha, \delta, \rho)$ by $\mathcal{C}$ on $PK_{\mathcal{U}_i}$ and (X, Y), accomplished by the three-pass protocol depicted in Figure 3.4. The security proof of the modified protocol trivially follows the proof presented in Pointcheval's paper [12].

3.5 Identification

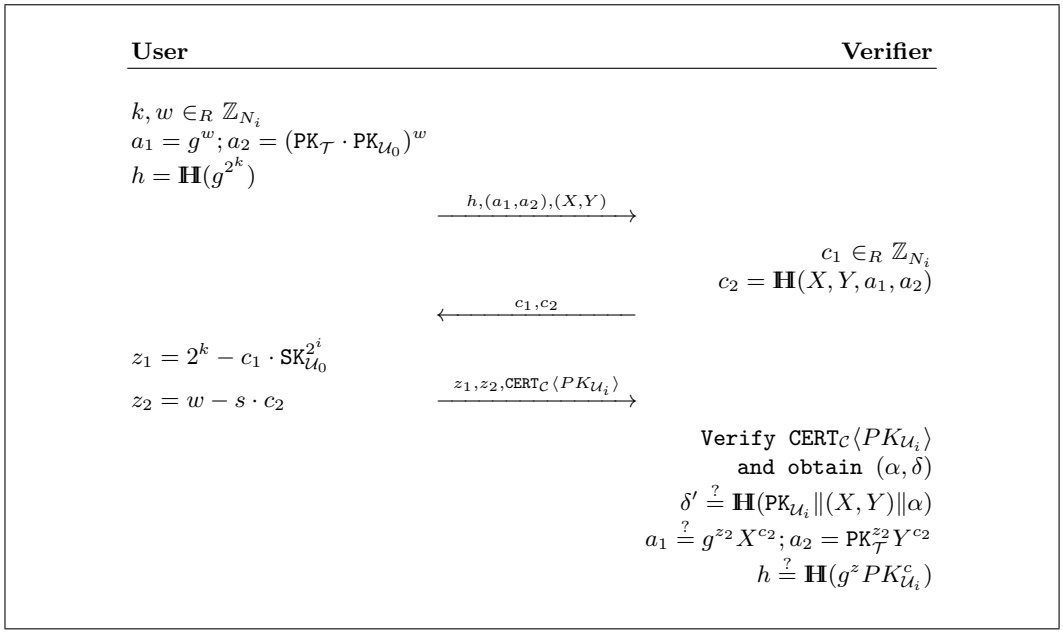

Fig. 2. Identification of Colligated Pseudonyms

The Identification protocol (Figure 2) is based on the Pointcheval optimised identification scheme [12] of Girault's identification scheme [18], but it now also includes the DL-EQ $\log_g X = \log_{\texttt{PK}_{\mathcal{T}}} Y$. In this protocol, a user $\mathcal{U}$ uses his certified pseudonym to identify himself/herself with a verifier $\mathcal{V}$ and at the end of

the protocol the verifier obtains an undeniable proof of $\mathcal{U}$ participation in the protocol. The identification process is represented by

$$\langle \mathtt{PROOF}_{\mathcal{U}_i} \rangle \leftarrow Identify(\mathcal{U}, \mathcal{V}, PK_{\mathcal{U}_i}, \mathtt{CERT}_{\mathcal{C}} \langle PK_{\mathcal{U}_i} \rangle, PK_{\mathcal{T}})$$

3.6 Tracing

The trace protocol (Figure 3) is invoked by a verifier $\mathcal{V}$ after $\mathcal{U}$ has misused a pseudonym and runs between the verifier $\mathcal{V}$ and the trustee $\mathcal{T}$. To trigger the protocol, $\mathcal{V}$ has to provide proof of protocol participation by $\mathcal{U}$. We shall express this phase as

$$(PK_{\mathcal{U}_0}) \leftarrow Trace(\mathcal{V}, \mathcal{T}, PK_{\mathcal{U}_i}, \mathtt{CERT}_{\mathcal{C}} \langle PK_{\mathcal{U}_i} \rangle, \langle \mathtt{PROOF}_{\mathcal{U}_i} \rangle)$$

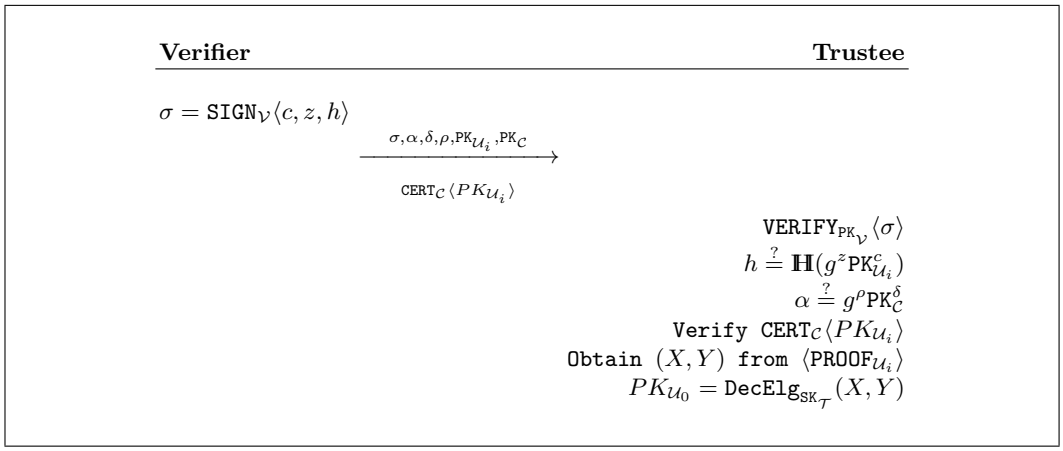

Fig. 3. Tracing Protocol

4 Security

4.1 Adversary Goals

We assume an active adversary $\mathcal{A}$, who is capable of eavesdropping and injecting messages in the communication medium. We also assume that an adversary may also be a legitimate (but dishonest) participant in a protocol, that is, either the certifier or the verifier or both may be dishonest.

As in [11, 10], we want our pseudonym system to be secure against the following attacks, that is, an adversary's goal is to mount any of following attacks:

- *Pseudonym forgery* – An adversary tries to forge a pseudonym for some user, possibly in association with other participants, including the certifier. That is the attack can be either:
  1. An adversary in possession of a valid proof tuple $(PK_{\mathcal{U}_i}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle)$ issued to another user or for a tuple of the form $(PK_{\mathcal{U}_i}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{A}}\rangle)$ is successfully able to execute an identification protocol with a verifier identifying as $\mathcal{U}_i$.
  2. An adversary successfully identifying himself/herself by executing an identification protocol with a tuple of the form $(PK_{\mathcal{A}}, \texttt{CERT}_{\mathcal{C}}\langle PK_{\mathcal{U}_i}\rangle)$.
- *Identity compromise* – An adversary in association with other participants tries to obtain information regarding the user's master public-secret key-pair, that is, an adversary with the knowledge of all user-generated public keys $(PK_{\mathcal{U}_1}, \ldots, PK_{\mathcal{U}_l})$, it should be computationally infeasible for an adversary to obtain the master public key $PK_{\mathcal{U}_0}$.
- *Pseudonym linking and colligation* – An adversary tries to obtain information that links a pair of pseudonyms to the same user or to a user's master public key. The goal is that even with the knowledge of all user generated public keys $(PK_{\mathcal{U}_1}, \ldots, PK_{\mathcal{U}_l})$, it should be computationally infeasible for an adversary to prove that any of the PK's in the set $(PK_{\mathcal{U}_1}, \ldots, PK_{\mathcal{U}_l})$, are related.

We now present our claims on the security of our proposal.

Claim. If the Square Decisional Diffie-Hellman (SDDH) problem is hard, then public keys generated from the master public key are indistinguishable.

The public keys generated are of the form $g^{SK^{2^i}}$ where, $i \in 0, \ldots, l$. For an adversary $\mathcal{A}$ to distinguish between a newly generated public key from a master or another newly-generated public key, $\mathcal{A}$ should solve the square Diffie-Hellman decision problem, that is, efficiently distinguish between two distributions of the form (g, g^{SK}, g^{SK^2}) and (g, g^{SK}, g^c), which is assumed to be hard.

Claim. It is computationally infeasible to obtain the master public key of a user by an adversary even with the knowledge of all newly-generated public key.

Proof (Sketch) For an adversary to obtain the master public key $(\texttt{PK}_{\mathcal{U}_0})$ from a pseudonym $(\texttt{PK}_{\mathcal{U}_i})$ presented, the adversary needs to solve, first, the discrete log problem to obtain $\texttt{SK}_{\mathcal{U}_i}$ and then the square root problem to obtain the value i. This violates our security assumptions. It is also a well-known fact that assuming the factoring of Blum Integers is intractable, the function $f_N = \texttt{SK}_{\mathcal{U}_0}^{2^i} \mod N_i$ is a trapdoor (one-way) permutation [19].

Claim. It is computationally infeasible to obtain the master public key of a user by a verifier or a certifier even if the certifier and verifier collude.

Proof (Sketch) Both $\mathcal{C}$ and $\mathcal{V}$ have knowledge of the public parameters. In addition, $\mathcal{C}$ has the knowledge of the user's master public key $\texttt{PK}_{\mathcal{U}_0}$, whereas, a verifier has the knowledge of the cipher-text obtain from the El-Gamal encryption (X, Y), the pseudonym of the user $\texttt{PK}_{\mathcal{U}_i}$, the signature value on both the pseudonym and the cipher-text $(\alpha, \rho, \delta, \texttt{PK}_{\mathcal{C}})$.

For a dishonest certifier $\widehat{\mathcal{C}}$ and a dishonest verifier $\widehat{\mathcal{V}}$ to obtain the master public key $\mathtt{PK}_{\mathcal{U}_0}$ of a user, either independently or in collusion, any one of the following cases needs to be satisfied.

Case 1 The blind signature protocol during the certification process leaks information about the identity of the user.

Case 2 A verifier is able to deduce the master identity from the pseudonym presented during the identification protocol.

Case 3 The certifier and a verifier with their combined knowledge are able to identify the colligation that exists between a pseudonym and the master secret key.

The security of Case 1 trivially follows the proof of security of the blind signature protocol by Pointcheval in [12]. For Case 2, a verifier can obtain the master public key if the proof of DL-EQ $\log_g X = \log_{\mathtt{PK}_\mathcal{T}} Y$ leaks any information regarding the master public key. A way of proving the security of the scheme is via the oracle replay technique formalised by Pointcheval and Stern [20]. In particular, the Schnorr signature with composite modulus has been proved secure in the random oracle model [21] by Poupard and Stern [22]. They show that if an adversary is able to forge a signature under an adaptively-chosen message attack, then the adversary is able to compute the discrete logarithms in G.

The security of Case 3 is based on the inability of $\widehat{\mathcal{V}}$ and $\widehat{\mathcal{C}}$ to obtain any information that links the user when they interact in the identification and the certification protocols. There are only two possibilities that can identify that a user participated in both the protocols. (a) The pseudonym leaks value about the true identity and, (b) the El-Gamal cipher-text (X, Y) that is used in both the certification and identification protocols can be linked to $\mathtt{PK}_{\mathcal{U}_0}$. If $\widehat{\mathcal{C}}$ and $\widehat{\mathcal{V}}$ in collusion are able to identify that the same (X,Y) that was presented in the identification protocol was used in the certification protocol, then they can positively establish a connection between the pseudonym presented during the identification protocol with the master identity used in the certification protocol.

From Theorem 4.1, we can conclude that it is computationally infeasible for $\widehat{\mathcal{V}}$ or $\widehat{\mathcal{C}}$ to obtain the master identity from a given pseudonym. As for possibility 2, the hash value δ computed with the cipher-text (X, Y), the pseudonym $\mathtt{PK}_{\mathcal{U}_i}$ and the value α as inputs, is blindly signed and never revealed to the certifier.

Claim. If the El-Gamal encryption is secure, then only the corresponding trustee can obtain information about the user from the encrypted cipher-text.

The justification of this claim directly follows the proof in [17]. The authors showed that the security of the composite El-Gamal reduces to computing the quadratic residue over a composite modulus that is a product of two primes. And because the master public key is encrypted using the pubic key of the trustee, only the trustee can successfully decrypt the cipher-text.

Remark 1. The protocol also provides guarantees of the honest participation of a user. The cipher-text containing the master public key is signed (blindly) by

the certifier. A verifier computes the hash value of the cipher-text (X, Y), the pseudonym and α again to verify against the signed hash value in the blind certificate, thus confirming that the user has performed an El-Gamal encryption over the same values that were used during the certification process.

5 Conclusion

In this chapter, we presented a pseudonym system by using the property of preserving a high-value secret key. Such systems are ideally suited for active-hardware security modules because they can protect the high-value secret keys and provide a high level of assurance for its security to the end users. The proposed system not only provides restricted anonymity, but also supports colligation between a trusted *high-value* secret key and the generated pseudonyms, therefore protecting the privacy of the user associated with the hardware module. Compared to other pseudonym schemes, our scheme has an efficient identification protocol where the computation can be carried out on a device that is constrained in processing power. As opposed to other TPM-based schemes [3, 4] that require the computation to be distributed between the TPM and the host computer, the computations in our scheme can be performed on the module itself. Our scheme is also ideally suited for storage-constrained devices, because there are no new secret keys to be generated for each pseudonym, only counter values of the pseudonym. Thus there is no appreciable increase in the storage requirement even when the number of pseudonyms required is high. Finally, in terms of anonymity, in our proposal, not only the applications on a single computer can be associated with a different pseudonym, but also every web-based application used by a user can be associated with a pseudonym.

6 Acknowledgment

Josef Pieprzyk was supported by the Australian Research Council grant DP0987734 and Vijayakrishnan Pasupathinathan was supported by Australian Postgraduate Award.

References

1. Trusted computing group. https://www.trustedcomputinggroup.org/ (2008)
2. TCG: Trusted computing group main specification v1.1 (2001)
3. Brickell, E., Camenisch, J., Chen, L.: Direct anonymous attestation. In: 11th ACM Conference on Computer and Communications Security, ACM Press (2004)
4. TCG: Trusted computing group main specification v1.2 (2007)
5. Camenisch, J., Lysyanskaya, A.: Dynamic accumulators and application to efficient revocation of anonymous credentials. Advances in Cryptology -CRYPTO'02 **LNCS 2442** (2002) 101–120
6. Chaum, D.: Security without identification: transaction systems to make big brother obsolete. Communications of the ACM **28**(10) (1985) 1030 – 1044

7. Chaum, D., Evertse, J.H.: A secure and privacy–protecting protocol for tranmitting personal information between organisation. In: Advances in Cryptology – CRYPTO'86, Springer–Verlag (1986) 118–167
8. Chen, L.: Access with pseudonyms. In Dawson, E., Golic, J., eds.: Cryptography: Policy and Algorithms. Number 1029, Springer–Verlag (1995) 232–243
9. Canetti, R., Charikar, M.S., Rajagopalan, S., Ravikumar, S., Sahai, A., Tomkins, A.S.: Non-transferable anonymous credentials. Patent No: 7222362 (2000)
10. Lysyanskaya, A., Rivest, R.L., Sahai, A., Wolf, S.: Pseudonym systems (extended abstract). Selected Areas in Cryptography'99 **LNCS 1758** (1999) 184–199
11. Damgard, I.: Payment systems and credential mechanisms with provable security against abuse by individuals. Advances in Cryptology - CRYPTO'88 **LNCS 403** (1988) 328–335
12. Pointcheval, D.: The composite discrete logarithm and secure authentication. In Imai, H., Zheng, Y., eds.: International Workshop on Practice and Thoery in Public Key Cryptography - PKC'2000. Volume LNCS 1751., Melbourne, Australia, Springer-Verlag (2000) 113–128
13. Chaum, D.: Blind signatures for untraceable payments. Advances in Cryptology -CRYPTO'82 (1982) 199–203
14. Blum, L., Blum, M., Shub, M.: A simple unpredictable pseudo random number generator. SIAM J. Computing **15**(2) (1986) 364–383
15. Chaum, D., Pedersen, T.: Transferred cash grows in size. Advances in Cryptology -EUROCRYPT'92 **LNCS 658** (1992) 390–407
16. Camenisch, J., Michels, M.: Separability and efficiency for generic group signature schemes. Advances in Cryptology -CRYPTO'99 **LNCS 1666** (1999) 413—430
17. Franklin, M., Haber, S.: Joint encryption and message-efficient secure computation. Advances in Cryptology -CRYPTO'93 **LNCS 773** (1993) 266 – 277
18. Girault, M.: Self-certified public keys. In: Advances in Cryptology – EUROCRYPT'91. Volume LNCS 547., Springer–Verlag (1991) 490–497
19. Goldreich, O.: Modern Cryptoraphy, Probabilistic Proofs and Pseudo-randomness. Springer (1999)
20. Pointcheval, D., Stern, J.: Security proofs for signature schemes. Advances in Cryptology -EUROCRYPT'96 **LNCS 1070** (1996) 387–398
21. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. ACM Conference on Computer and Communications Security'93 (1993) 62–73
22. Poupard, G., Stern, J.: Security analysis of a practical "on the fly" authentication and signature generation. Advances in Cryptology -EUROCRYPT'98 **LNCS 1403** (1998) 422–436