An Anonymous Attestation Scheme with Optional Traceability

Jiangtao Li and Anand Rajan

Intel Labs, Intel Corporation
jiangtao.li@intel.com, anand.rajan@intel.com

Abstract. Direct Anonymous Attestation (DAA) is a cryptographic scheme designed for anonymous attestation of a hardware device while preserving the privacy of the device owner. Signatures created by a DAA signer are anonymous and untraceable, i.e., cannot be opened to find out the identity of the signer. To prevent abuse of privacy, DAA has a feature called user-controlled-traceability in which the signer and verifier can negotiate whether or not the signatures from the signer can linked. This feature is a preventive mechanism against corrupted DAA signers because they can be prevented from making multiple anonymous authentications. However, it is not a proactive deterrent against such activity as nobody is able to identify the corrupted signer. In this paper, we introduce a new cryptographic scheme called Optionally Traceable Anonymous Attestation (OTAA), in which the signer and verifier can negotiate whether signatures from the signer are traceable to the issuer instead of just being linkable. In the OTAA scheme, if a corrupted signer has produced a traceable signature or published his private key widely, the issuer can identify the signer and effectively revoke him using the verifier-local revocation. We give a construction of an OTAA scheme from bilinear pairing. Our OTAA scheme is efficient and provably secure in the random oracle model under the strong Diffie-Hellman assumption and the external Diffie-Hellman assumption.

1 Introduction

The concept and a concrete scheme of Direct Anonymous Attestation (DAA) were first introduced by Brickell, Camenisch, and Chen [9] for remote anonymous authentication of a hardware module, called Trusted Platform Module (TPM). The DAA scheme was adopted by the Trusted Computing Group (TCG) [31], an industry standardization body that aims to develop and promote an open industry standard for trusted computing hardware and software building blocks. The DAA scheme was standardized in the TCG TPM Specification Version 1.2 [30] and has recently been adopted by ISO/IEC as an international standard. After DAA was first introduced, it has drawn a lot of attention from the industry in general and cryptographic community in particular, e.g., [17,29,10,25,24] are some of the relevant references in this regard.

A DAA scheme involves three types of entities: an issuer, signers, and verifiers. The issuer is in charge of verifying the legitimacy of signers and of issuing

A. Acquisti, S.W. Smith, and A.-R. Sadeghi (Eds.): TRUST 2010, LNCS 6101, pp. 196–210, 2010.

a membership credential to each signer. A signer can prove membership anonymously to a verifier by creating a DAA signature. The verifier can verify the membership of the signer from the DAA signature but cannot learn the identity of the signer. DAA scheme can be seen as a group signature scheme without the traceability feature, i.e., nobody (not even the issuer) can open a DAA signature to find out the identity of the signer.

DAA signatures can be created using one of the two options: random base option and name base option. In the random base option, DAA signatures are unlinkable. In the name base option, two DAA signatures produced by a signer (using one verifier's basename) are linkable. Yet signatures created by a signer using two different basenames are unlinkable. This feature is later referred as user-controlled-traceability [11] (probably user-controlled-linkability is more appropriate), as the signer and verifier can negotiate whether the signatures from the signer can be linked by choosing the appropriate options. This feature can be used to prevent abuse of privacy by a corrupted signer. For example, if a provisioning server wants to issue a key to each valid signer, the server can mandate the name base option to make sure each signer can only get one key at most.

The user-controlled-traceability feature in DAA is a preventive mechanism against corrupted signers but not a proactive one, as a corrupted signer can be prevented from making multiple anonymous authentications to a verifier but he cannot be identified or revoked per the definition of DAA. Brickell and Li have proposed an extension of DAA called Enhanced Privacy ID (EPID) in which the issuer can revoke signatures from corrupted signers without knowing their private keys or identities [12,14]. Tsang et al. proposed a similar revocation mechanism in [32]. However, such a revocation mechanism requires the signer to perform zero-knowledge proof for each revoked signature. This could be too computationally demanding for small hardware devices such as TPM.

In practice, we can separate the use of anonymous attestation into two categories: high-value transactions and low-value transactions. For high-value transactions, such as downloading key materials or accessing medical information, the ability to trace and effectively revoke corrupted signers is required. However, for low-value transactions, such as accessing digital library or proving older than certain age, privacy carries more weight than traceability and revocability. End user may feel more comfortable if his anonymous signatures are untraceable most of the time during his day-to-day transactions, but traceable occasionally for the high-value transactions.

In this paper, we propose a new cryptographic scheme called Optionally Traceable Anonymous Attestation (OTAA) with a concrete construction from bilinear maps. OTAA provides the right balance between privacy and traceability. The difference from DAA is that the signer in OTAA can choose whether or not signatures are traceable instead of linkable. The OTAA scheme has the following features:

1. OTAA signatures are anonymous and unlinkable to the verifiers. However the signer and verifier can negotiate whether or not the signatures are traceable to the issuer.

2. The issuer can open a traceable signature and identify the signer. Furthermore, the issuer can revoke the signer in an efficient way. In addition, if a corrupted signer publishes his private key widely, the issuer can revoke the signer as well. Both revocation checks are performed locally by the verifier. This revocation model is known in the literature as verifier-local revocation.

OTAA can be used in trusted computing as an alternative mechanism for anonymous attestation besides DAA. One of the biggest advantages of OTAA is the revocation method: it is more capable than the original definition of DAA [9,11] and is more efficient than the signature based revocation in EPID [12]. We believe OTAA has wider application beyond the TPM usage, such as in e-commerce, digital content protection, and identity card. Our OTAA construction is very efficient. It has similar efficiency as the DAA schemes [14,24]. Our OTAA scheme becomes a group signature scheme if all the signatures are traceable. We show in Section 6 that our OTAA is more efficient than existing pairing-based group signature schemes [7,20,8,27].

Rest of this paper is organized as follows. We first discuss the related work in Section 2. We then give a formal specification of OTAA and present the security requirements in Section 3. We review the definition of pairing and related security assumptions in Section 4. Next we describe the construction of our OTAA scheme in Section 5. We compare our OTAA scheme with several group signature and DAA schemes in Section 6. We conclude our paper and discuss the future work in Section 7.

2 Related Work

OTAA can be seen as a variant of DAA [9,17,29,3,12,11,25,24]. Many concepts of OTAA borrow from DAA, such as negotiation of different privacy level and revocation if a private key gets revealed publicly. In fact, our construction of OTAA builds on top of the recent pairing-based DAA schemes [14,24]. As we mentioned earlier, DAA signatures cannot be opened. Thus a corrupted signer will not be revoked unless he publishes his private key on the Internet. Such revocation capability in the original definition of DAA [9,11] is limited. OTAA provides a better revocation capability without comprising the efficiency.

OTAA is a special group signatures scheme with the optional traceability feature. In group signature definition [5] and constructions [23,22,1,7,27], all the group signatures are traceable. In most of the group signature schemes, the signer encrypts his identity using the tracing manager's public key in a way that the encryption can be verified by the verifier. To open a group signature, the tracing manager uses his private key to decrypt and find the identity of the signer. Thus, it is not difficult to make a group signature scheme optionally traceable, i.e., the signer can choose whether to encrypt his identity. OTAA is unique in that the issuer not only can open a traceable signature but can also revoke the signer using efficient verifier-local revocation.

Boneh and Shacham proposed an efficient verifier-local revocation group signature scheme [8]. It is not difficult to convert their group signature scheme

into an optionally traceable group signature scheme. To make a group signature untraceable in their scheme, the signer simply chooses u and v randomly from G_1 instead of choosing $\hat{u}$ and $\hat{v}$ from G_2. However, such untraceable signatures cannot be revoked in any scenarios, e.g., even if the private key is revealed and widely distributed. Besides, our OTAA scheme is more efficient than the BS group signatures scheme in signature signing, verification, and the revocation check. The revocation check in the BS group signatures scheme requires two pairing operations per revoked key instead of one exponentiation in our OTAA scheme. A pairing operation is about 10 times more expensive than an exponentiation operation. The revocation check in our scheme is fix-base exponentiation which can be further optimized using fast exponentiation technique [16]. Thus the revocation check in our scheme is about two orders of magnitude more efficient than the BS group signatures scheme.

Our OTAA scheme uses verifier-local revocation, a revocation model used widely in all the DAA schemes as well as in some group signature schemes [2,8]. We believe that this revocation model is a practical model, as the revocation lists are only sent to the verifiers. After all, it is in the verifiers' interest to perform the revocation check. This is similar to the revocation model we currently have in the public key infrastructure. Observe that this model adds no burden to the signers. In our OTAA scheme, the revocation check takes n fix-base exponentiations, where n is the size of the revocation list. Using the fast exponentiation technique [16], a verifier can easily process a few thousands revocation checks within one second. Another popular revocation model in the group signatures is to use dynamic accumulators [19,18]. Although dynamic accumulators are very efficient, a limitation of this approach is the infrastructure overhead. Each signer needs to constantly connect to the issuer and update his credentials. This seems to not be practical in the TPM implementation.

3 Specification and Security Requirements of OTAA

In the rest of this paper, we use the following standard notations. Let S be a finite set, $x \leftarrow S$ denotes that x is chosen uniformly at random from S. Let $b \leftarrow A(a)$ denote an algorithm A that is given input a and outputs b. Let $\langle c, d\rangle \leftarrow P_{A,B}\langle a, b\rangle$ denote an interactive protocol between A and B, where A inputs a and B inputs b, in the end A obtains c and B obtains d.

3.1 Specification of OTAA

An OTAA scheme involves three types of entities: an issuer $\mathcal{I}$, platforms $\mathcal{P}$, and verifiers $\mathcal{V}$. There are the following four polynomial-time algorithms Setup, Sign, Verify, and Open, and one interactive protocol Join.

Setup : This setup algorithm for the issuer $\mathcal{I}$ takes a security parameter 1^k as input and outputs a group public key `gpk` and the issuer's private key `isk`. The public key `gpk` includes the global public parameters for the system.

$$(\texttt{gpk}, \texttt{isk}) \leftarrow \mathsf{Setup}(1^k)$$

Join : This join protocol is an interactive protocol between the issuer $\mathcal{I}$ and a platform $\mathcal{P}_i$ and consists of two randomized algorithms: $\mathsf{Join_t}$ and $\mathsf{Join_i}$. The platform $\mathcal{P}_i$ uses $\mathsf{Join_t}$ to produce a pair $(\mathtt{sk}_i, \mathtt{comm}_i)$, where $\mathtt{sk}_i$ is platform's secret key and $\mathtt{comm}_i$ is a commitment of $\mathtt{sk}_i$.

$$(\mathtt{sk}_i, \mathtt{comm}_i) \leftarrow \mathsf{Join_t}(\mathtt{gpk})$$

On input of a commitment $\mathtt{comm}_i$, the group public key $\mathtt{gpk}$, the issuer's private key $\mathtt{isk}$, the issuer $\mathcal{I}$ uses $\mathsf{Join_i}$ to produce $\mathtt{cre}_i$, a membership credential associated with $\mathtt{sk}_i$. The $\mathtt{cre}_i$ includes a unique tracing key $\mathtt{tk}_i$. $\mathcal{P}_i$ receives $\mathtt{cre}_i$ while $\mathcal{I}$ updates its tracing database $\mathtt{dbase}$ by inserting a record of $\mathcal{P}_i$'s identity $\mathtt{id}_i$ and the tracing key $\mathtt{tk}_i$.

$$(\mathtt{cre}_i, \mathtt{dbase}) \leftarrow \mathsf{Join_i}(\mathtt{gpk}, \mathtt{isk}, \mathtt{dbase}, \mathtt{comm}_i)$$

The join protocol can be formulated as

$$\langle \mathtt{dbase}, (\mathtt{sk}_i, \mathtt{cre}_i) \rangle \leftarrow \mathsf{Join}_{\mathcal{I},\mathcal{P}} \langle (\mathtt{gpk}, \mathtt{isk}, \mathtt{dbase}), \mathtt{gpk} \rangle$$

Sign : On input of $\mathtt{gpk}$, $\mathtt{sk}_i$, $\mathtt{cre}_i$, a boolean value $\mathtt{tr}$, and a message m, the probabilistic signing algorithm outputs a signature σ. If $\mathtt{tr} = 1$, σ is a traceable signature, otherwise, σ is a non-traceable signature. We often times call $(\mathtt{sk}_i, \mathtt{cre}_i)$ the signing key of $\mathcal{P}_i$.

$$\sigma \leftarrow \mathsf{Sign}(\mathtt{gpk}, \mathtt{sk}_i, \mathtt{cre}_i, \mathtt{tr}, m)$$

Verify : On input of $\mathtt{gpk}$, a message m, a traceable or a non-traceable signature σ, a list of revoked secret keys $\mathtt{sRL}$ (the revocation list for non-traceable signatures), and a list of revoked tracing keys $\mathtt{tRL}$ (the revocation list for traceable signatures), this verification algorithm outputs $\mathtt{valid}$, $\mathtt{revoked}$, or $\mathtt{invalid}$. We shall discuss how to build the revocation lists in the later sections.

$$\mathtt{valid/revoked/invalid} \leftarrow \mathsf{Verify}(\mathtt{gpk}, m, \sigma, \mathtt{sRL}, \mathtt{tRL})$$

Open : On input of $\mathtt{gpk}$, a message m, a signature σ, and the tracing database $\mathtt{dbase}$, the deterministic tracing algorithm outputs $\mathtt{invalid}$ if the signature is not valid, $\mathtt{untraceable}$ if the signature is not traceable, or $(\mathtt{id}_i, \mathtt{tk}_i)$, the identity and tracing key of the signer.

$$\mathtt{invalid/untraceable/}(\mathtt{id}_i, \mathtt{tk}_i) \leftarrow \mathsf{Open}(\mathtt{gpk}, m, \sigma, \mathtt{dbase})$$

Observe that the revocation lists are only sent to the verifiers. This revocation method is known in the literature as verifier-local revocation [8] and has been used in most of the DAA schemes [9,12,10,25,24]. One implication of verifier-local revocation is the signature is selfless-anonymous, i.e., a platform can tell whether he generated a particular signature σ, but if he did not he learns nothing else about the signer of σ.

3.2 Security Requirements of OTAA

A secure OTAA scheme needs to satisfy the following three requirements: correctness, user-controlled-anonymity, and user-controlled-traceability. We borrow the terms "user-controlled-anonymity" and "user-controlled-traceability" from the definition of DAA [11], as the platform and verifier in OTAA can negotiate the privacy level of a signature, i.e., whether the signatures can be traced. However, the security requirements of OTAA are different from those in DAA [9,11]. Roughly speaking, user-controlled-anonymity guarantees that only the issuer is able to identify the actual signer of a traceable signature and nobody can identify the actual signer of a non-traceable signature, except that if the signer is revoked. User-controlled-traceability guarantees that no one except the issuer is able to successfully add a new platform to the group. Our security requirements follow the framework of Bellare et al. definition of group signatures for dynamic groups [5] and Boneh and Shacham's definition of verifier-local revocation group signatures [8].

Correctness. The correctness requirement states that every signature, no matter traceable or not, generated by a platform can be verified as valid, except when the platform is revoked. It can be formally stated as follows

$$
\begin{aligned}
&(\mathtt{gpk}, \mathtt{isk}) \leftarrow \mathsf{Setup}(1^k)\\
&\langle \mathtt{dbase}, (\mathtt{sk}_i, \mathtt{cre}_i)\rangle \leftarrow \mathsf{Join}_{\mathcal{I},\mathcal{P}}\langle(\mathtt{gpk}, \mathtt{isk}, \mathtt{dbase}), \mathtt{gpk}\rangle\\
&\sigma \leftarrow \mathsf{Sign}(\mathtt{gpk}, \mathtt{sk}_i, \mathtt{cre}_i, \mathtt{tr}, m)\\
&\mathsf{Verify}(\mathtt{gpk}, m, \sigma, \mathtt{sRL}, \mathtt{tRL}) = \mathtt{true} \iff\\
&\qquad ((\mathtt{tr} = 1) \wedge (\mathtt{tk}_i \notin \mathtt{tRL})) \ \vee \ ((\mathtt{tr} = 0) \wedge (\mathtt{sk}_i \notin \mathtt{sRL}))
\end{aligned}
$$

User-Controlled-Anonymity. An OTAA scheme satisfies the user-controlled-anonymity property if no polynomial-time adversary can win the anonymity games. In the anonymity game, the goal of the adversary is to determine which one of two platforms was used in generating a signature. As mentioned earlier, given a signature and a signing key, the adversary could determine whether the signature was generated using the signing key. If the signature is a traceable signature, then the adversary should not be given access to the tracing key of either platform. Otherwise if the signature is non-traceable, the adversary should be given the secret keys of the platforms. The anonymity game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ is defined as follows.

1. Setup. $\mathcal{C}$ runs $(\mathtt{gpk}, \mathtt{isk}) \leftarrow \mathsf{Setup}(1^k)$ and sends $\mathtt{gpk}$ and $\mathtt{isk}$ to $\mathcal{A}$.
2. Queries. $\mathcal{A}$ can make the following queries to $\mathcal{C}$.
   (a) Join. $\mathcal{A}$ requests for creating a new platform $\mathcal{P}$ by choosing one of the following two types:
      i. $\mathcal{C}$ runs the join protocol as the platform $\mathcal{P}$ by interacting with $\mathcal{A}$ as the issuer. In the end, $\mathcal{C}$ obtains the signing key $(\mathtt{sk}, \mathtt{cre})$, while $\mathcal{A}$ only learns the credential $\mathtt{cre}$.
      ii. $\mathcal{C}$ runs the join protocol locally and generates a signing key $(\mathtt{sk}, \mathtt{cre})$.

(b) Sign. $\mathcal{A}$ requests a signature on a message m with a traceability option $\mathtt{tr}$ for a platform $\mathcal{P}$. $\mathcal{C}$ finds $\mathcal{P}$'s signing key $(\mathtt{sk}, \mathtt{cre})$ and computes $\sigma \leftarrow \mathsf{Sign}(\mathtt{gpk}, \mathtt{sk}, \mathtt{cre}, \mathtt{tr}, m)$. $\mathcal{C}$ returns σ to $\mathcal{A}$.
(c) Corrupt. $\mathcal{A}$ requests the signing key of a platform $\mathcal{P}$. $\mathcal{C}$ responds with $(\mathtt{sk}, \mathtt{cre})$ of $\mathcal{P}$ to $\mathcal{A}$.

3. Challenge. $\mathcal{A}$ outputs a message m, a traceability option $\mathtt{tr}$, and two platforms $\mathcal{P}_0$ and $\mathcal{P}_1$. $\mathcal{A}$ must have not made a corruption query on either $\mathcal{P}_0$ or $\mathcal{P}_1$. If $\mathcal{A}$ has a membership credential of either $\mathcal{P}_0$ or $\mathcal{P}_1$, the requesting $\mathtt{tr}$ must be 0, i.e., $\mathcal{A}$ can only request a non-traceable signature. $\mathcal{C}$ chooses a random bit $b \leftarrow \{0, 1\}$, computes a signature $\sigma \leftarrow \mathsf{Sign}(\mathtt{gpk}, \mathtt{sk}_b, \mathtt{cre}_b, \mathtt{tr}, m)$, where $(\mathtt{sk}_b, \mathtt{cre}_b)$ is the signing key for $\mathcal{P}_b$, and sends σ to $\mathcal{A}$.
4. Restricted Queries. After the challenge phase, $\mathcal{A}$ can make additional queries to $\mathcal{C}$, restricted as follows.
   (a) Join. $\mathcal{A}$ can make join queries as before.
   (b) Sign. $\mathcal{A}$ can make sign queries as before.
   (c) Corrupt. As before, but $\mathcal{A}$ cannot make corrupt queries at $\mathcal{P}_0$ and $\mathcal{P}_1$.
5. Output. Finally, $\mathcal{A}$ outputs a bit b'. The adversary wins if $b' = b$.

Definition 1. *Let $\mathcal{A}$ be the adversary. We use $\mathbf{Adv}[\mathcal{A}^{\mathrm{An}}_{OTAA}] = |\Pr[b = b'] - 1/2|$ to denote the advantage of $\mathcal{A}$ in breaking the user-controlled-anonymity game. The probability is taken over the coin tosses of $\mathcal{A}$, of the randomized setup, join, and sign algorithms, and over the choice of b. We say that an OTAA scheme is user-controlled-anonymity if for any probabilistic polynomial-time adversary, $\mathbf{Adv}[\mathcal{A}^{\mathrm{An}}_{OTAA}]$ is negligible.*

User-Controlled-Traceability. We say that an OTAA scheme satisfies the user-controlled-traceability property if no adversary can win the following traceability game. In the user-controlled-traceability game, the adversary's goal is to forge a valid non-traceable signature given that all private keys known to the adversary have been revoked or to create a traceable signature that cannot be opened properly. The traceability game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ is defined as follows.

1. Setup. $\mathcal{C}$ runs $(\mathtt{gpk}, \mathtt{isk}) \leftarrow \mathsf{Setup}(1^k)$ and sends $\mathtt{gpk}$ to $\mathcal{A}$. $\mathcal{C}$ sets empty revocation lists $\mathtt{sRL}$ and $\mathtt{tRL}$.
2. Queries. $\mathcal{A}$ can make the following queries to $\mathcal{C}$.
   (a) Join. $\mathcal{A}$ requests for creating a new platform $\mathcal{P}$ by choosing one of the following two types:
      i. $\mathcal{C}$ runs the join protocol as the issuer with $\mathcal{A}$ as the platform $\mathcal{P}$. In the end, $\mathcal{C}$ outputs $\mathtt{cre}$ while $\mathcal{A}$ outputs the signing key $(\mathtt{sk}, \mathtt{cre})$. $\mathcal{C}$ appends $\mathtt{tk}$ in $\mathtt{cre}$ to revocation list $\mathtt{tRL}$.
      ii. $\mathcal{C}$ runs the join protocol locally and generates a signing key $(\mathtt{sk}, \mathtt{cre})$.
   (b) Sign. $\mathcal{A}$ requests a signature on a message m with a traceability option $\mathtt{tr}$ for a platform $\mathcal{P}$. $\mathcal{C}$ finds $\mathcal{P}$'s signing key $(\mathtt{sk}, \mathtt{cre})$ and computes $\sigma \leftarrow \mathsf{Sign}(\mathtt{gpk}, \mathtt{sk}, \mathtt{cre}, \mathtt{tr}, m)$. $\mathcal{C}$ returns σ to $\mathcal{A}$.

(c) Corrupt. $\mathcal{A}$ requests the signing key of a platform $\mathcal{P}$. $\mathcal{C}$ responds with $(\mathtt{sk}, \mathtt{cre})$ of $\mathcal{P}$ to $\mathcal{A}$. $\mathcal{C}$ also appends $\mathtt{sk}$ to $\mathtt{sRL}$ and $\mathtt{tk}$ from $\mathtt{cre}$ to $\mathtt{tRL}$.

3. Response. Finally, $\mathcal{A}$ outputs a message m and a signature σ.

Assuming that $\mathcal{A}$ did not obtain σ by making a sign query on m.

1. If σ is a traceable signature, $\mathcal{A}$ wins if $\mathsf{Verify}(\mathtt{gpk}, \sigma, m, \mathtt{sRL}, \mathtt{tRL}) = \mathtt{valid}$.
2. If σ is a non-traceable signature, $\mathcal{A}$ wins if $\mathsf{Verify}(\mathtt{gpk}, \sigma, m, \mathtt{sRL}, \mathtt{tRL}) = \mathtt{valid}$ and $\mathcal{A}$ has never made any type (i) join query.

Note that if $\mathcal{A}$ has made a type (i) join query, it obtained a signing key $(\mathtt{sk}, \mathtt{cre})$ such that only the tracing key in $\mathtt{cre}$ is known to $\mathcal{C}$. Since $\mathcal{C}$ does not know $\mathtt{sk}$, $\mathcal{A}$ can produce a valid non-traceable signature using the signing key. Thus, if $\mathcal{A}$ outputs a non-traceable signature in the final response, $\mathcal{A}$ wins only if it has not made any type (i) join query.

Definition 2. *Let $\mathcal{A}$ denote an adversary that plays the traceability game above. We use $\mathbf{Adv}[\mathcal{A}^{\mathrm{Tr}}_{OTAA}] = \Pr[\mathcal{A} \text{ wins}]$ to denote the advantage that $\mathcal{A}$ breaks the traceability game. We say that an OTAA scheme is user-controlled-traceability if for any probabilistic polynomial-time adversary, $\mathbf{Adv}[\mathcal{A}^{\mathrm{Tr}}_{OTAA}]$ is negligible.*

4 Pairings and Complexity Assumptions

In this section, we first review the concept of pairings and then discuss some complexity assumptions related to our scheme.

Background on Bilinear Maps. Our OTAA scheme in this paper is based on asymmetric pairings. We follow the notation of Boneh, Boyen, and Shacham [7] to review some background on pairings. Let G_1 and G_2 to two multiplicative cyclic groups of prime order p. Let g_1 be a generator of G_1 and g_2 be a generator of G_2. We say $e : G_1 \times G_2 \rightarrow G_T$ is an admissible bilinear map, if it satisfies the following properties:

1. Bilinear. For all $u \in G_1, v \in G_2$, and for all $a, b \in \mathbb{Z}$, $e(u^a, v^b) = e(u, v)^{ab}$.
2. Non-degenerate. $e(g_1, g_2) \neq 1$ and is a generator of G_T.
3. Computable. There exists an efficient algorithm for computing $e(u, v)$ for any $u \in G_1, v \in G_2$.

We call the two groups (G_1, G_2) in the above a bilinear group pair. In the rest of this paper, we consider bilinear maps $e : G_1 \times G_2 \rightarrow G_T$ where G_1, G_2, and G_T are multiplicative groups of prime order p.

Strong Diffie-Hellman Assumption. The security of our DAA scheme is related to the hardness of the q-SDH problem introduced by Boneh and Boyen [6]. Let G_1 and G_2 be two cyclic groups of prime order p, respectively, generated by g_1 and g_2. The q-Strong Diffie-Hellman (q-SDH) problem in (G_1, G_2) is defined as follows: Given a (q+3)-tuple of elements $(g_1, g_1^{\gamma}, \ldots, g_1^{(\gamma^q)}, g_2, g_2^{\gamma})$ as input, output

a pair $(g_1^{1/(\gamma+x)}, x)$ where $x \in \mathbb{Z}_p^*$. An algorithm $\mathcal{A}$ has advantage ϵ in solving q-SDH problem in (G_1, G_2) if $\Pr[\mathcal{A}(g_1, g_1^{\gamma}, \ldots, g_1^{(\gamma^q)}, g_2, g_2^{\gamma}) = (g_1^{1/(\gamma+x)}, x)] \geq \epsilon$ where the probability is over the random choice of γ and the random bits of $\mathcal{A}$.

Definition 3. We say that the (q, t, ϵ)-SDH assumption holds in (G_1, G_2) if no t-time algorithm has advantage at least ϵ in solving the q-SDH problem.

External Diffie-Hellman Assumption. Let G, generated by g, be a cyclic group of prime order p. The Decisional Diffie-Hellman (DDH) problem in G is defined as follows: Given a tuple of elements (g, g^a, g^b, g^c) as input, output 1 if $c = ab$ and 0 otherwise.

Definition 4. We say that the (t, ϵ)-DDH assumption holds in G if no t-time algorithm has advantage at least ϵ in solving the DDH problem in G.

Let (G_1, G_2) be a bilinear group pair. Our proposed OTAA scheme requires the DDH problem for G_1 to be hard. The DDH assumption on G_1 is often known as the External Diffie-Hellman (XDH) assumption.

5 The Proposed OTAA Scheme

In this section, we first present our construction of an OTAA scheme from bilinear maps. Our construction builds on top of the recent pairing-based EPID scheme [14] and Chen's DAA scheme [24], and Furukawa and Imai group signatures scheme [27].

5.1 Our OTAA Scheme

The OTAA scheme has the following algorithms Setup, Sign, Verify, and Open and one interactive protocol Join which are defined as follows.

Setup : The setup algorithm takes the following steps:

1. On input of 1^k, it chooses an asymmetric bilinear group pair (G_1, G_2) of prime order p and a pairing function $e : G_1 \times G_2 \rightarrow G_T$. Let g_1 and g_2 be the generators of G_1 and G_2, respectively.
2. It selects a collision resistant hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_p$.
3. It chooses $h_1, h_2 \leftarrow G_1$, $\gamma \leftarrow \mathbb{Z}_p^*$, and computes $w := g_2^{\gamma}$.
4. It computes $T_1 = e(g_1, g_2)$, $T_2 = e(h_1, g_2)$, $T_3 = e(h_2, g_2)$, $T_4 = e(h_1, w)$, and $T_5 = e(h_2, w)$.
5. It outputs the group public key and private key

$$(\mathtt{gpk}, \mathtt{isk}) := ((G_1, G_2, G_T, e, p, g_1, g_2, h_1, h_2, w, H, T_1, T_2, T_3, T_4, T_5), \gamma)$$

Note that T_1, T_2, T_3, T_4, and T_5 are optional in gpk, as they can be computed from g_1, g_2, h_1, h_2, w.

Join : The join protocol is performed by a platform $\mathcal{P}$ and the issuer $\mathcal{I}$. $\mathcal{P}$ takes $\mathtt{gpk}$ as input and $\mathcal{I}$ has $\mathtt{gpk}$, $\mathtt{isk}$, and $\mathtt{dbase}$ as input. The protocol has the following steps:

1. $\mathcal{I}$ sends a nonce $n_I \in \{0,1\}^t$ as a challenge to $\mathcal{P}$.
2. $\mathcal{P}$ chooses at random $f \leftarrow \mathbb{Z}_p$ and computes $F := h_1^f$.
3. $\mathcal{P}$ sets $\mathtt{sk} := f$ and $\mathtt{id}_{\mathcal{P}} := F$ as its identity.
4. $\mathcal{P}$ chooses at random $r_f \leftarrow \mathbb{Z}_p$ and computes $R := h_1^{r_f}$.
5. $\mathcal{P}$ computes $c := H(\mathtt{gpk} \| F \| R \| n_I)$ and $s_f := r_f + c \cdot f \pmod p$.
6. $\mathcal{P}$ sets $\mathtt{comm} := (F, c, s_f)$ as a commitment of its $\mathtt{sk}$, and sends $\mathtt{comm}$ to $\mathcal{I}$.
7. $\mathcal{I}$ computes $\hat{R} := h_1^{s_f} \cdot F^{-c}$ and verifies $s_f \in \mathbb{Z}_p$ and $c = H(\mathtt{gpk} \| F \| \hat{R} \| n_I)$.
8. $\mathcal{I}$ chooses at random $x, y \leftarrow \mathbb{Z}_p$ and computes $A := (g_1 \cdot F \cdot h_2^y)^{1/(x+\gamma)}$.
9. $\mathcal{I}$ sets $\mathtt{tk} := y$ and $\mathtt{cre} := (A, x, y)$ and sends $\mathtt{cre}$ to $\mathcal{P}$.
10. $\mathcal{I}$ updates its tracing database $\mathtt{dbase}$ by appending $(\mathtt{id}_{\mathcal{P}}, \mathtt{tk}) := (F, y)$.
11. $\mathcal{P}$ verifies $e(A, w g_2^x) = e(g_1 h_1^f h_2^y, g_2)$ and outputs $(\mathtt{sk}, \mathtt{cre}) := (f, (A, x, y))$.

As in many DAA schemes [9,10,13,24], the above join protocol needs to be executed in a sequential manner, as $\mathtt{comm}$ is a proof of knowledge of the discrete logarithm of the value F. To support concurrent join, we could use verifiable encryption [21] of the f value or use the concurrent join technique described in [28] with some loss of efficiency.

Sign : On input of $\mathtt{gpk}$, $\mathtt{sk} = f$, $\mathtt{cre} = (A, x, y)$, a tracing option $\mathtt{tr} \in \{0,1\}$, a message $m \in \{0,1\}^*$, this signing algorithm takes the following steps:

1. If $\mathtt{tr} = 0$, the algorithm outputs a non-traceable signature on m as follows:
   (a) It chooses $B \leftarrow G_1$ and computes $K := B^f$.
   (b) It chooses $a \leftarrow \mathbb{Z}_p$, computes $b := y + ax \pmod p$ and $T := A \cdot h_2^a$.
   (c) It randomly picks

$$r_x \leftarrow \mathbb{Z}_p, \qquad r_f \leftarrow \mathbb{Z}_p, \qquad r_a \leftarrow \mathbb{Z}_p, \qquad r_b \leftarrow \mathbb{Z}_p.$$

   (d) It computes

$$\begin{aligned} R_1 &:= B^{r_f}, \\ R_2 &:= e(T, g_2)^{-r_x} \cdot e(h_1, g_2)^{r_f} \cdot e(h_2, g_2)^{r_b} \cdot e(h_2, w)^{r_a} \\ &= e(A, g_2)^{-r_x} \cdot e(h_2^a, g_2)^{-r_x} \cdot e(h_1, g_2)^{r_f} \cdot e(h_2, g_2)^{r_b} \cdot e(h_2, w)^{r_a} \\ &= e(A, g_2)^{-r_x} \cdot T_2^{r_f} \cdot T_3^{r_b - a r_x} \cdot T_5^{r_a}. \end{aligned}$$

   (e) It then computes

$$c := H(\mathtt{gpk} \| B \| K \| T \| R_1 \| R_2 \| m).$$

   (f) It computes in $\mathbb{Z}_p$

$$s_x := r_x + cx, \quad s := r_f + cf, \quad s_a := r_a + ca, \quad s_b := r_b + cb.$$

2. If $\mathtt{tr} = 1$, the algorithm outputs a traceable signature on m as follows:
   (a) It chooses $B \leftarrow G_1$ and computes $K := B^y$.
   (b) It chooses $a \leftarrow \mathbb{Z}_p$, computes $b := f + ax \pmod p$ and $T := A \cdot h_1^a$
   (c) It randomly picks

$$r_x \leftarrow \mathbb{Z}_p, \qquad r_y \leftarrow \mathbb{Z}_p, \qquad r_a \leftarrow \mathbb{Z}_p, \qquad r_b \leftarrow \mathbb{Z}_p.$$

   (d) It computes

$$\begin{aligned} R_1 &:= B^{r_y}, \\ R_2 &:= e(T, g_2)^{-r_x} \cdot e(h_2, g_2)^{r_y} \cdot e(h_1, g_2)^{r_b} \cdot e(h_1, w)^{r_a} \\ &= e(A, g_2)^{-r_x} \cdot e(h_1^a, g_2)^{-r_x} \cdot e(h_2, g_2)^{r_y} \cdot e(h_1, g_2)^{r_b} \cdot e(h_1, w)^{r_a} \\ &= e(A, g_2)^{-r_x} \cdot T_3^{r_y} \cdot T_2^{r_b - a r_x} \cdot T_4^{r_a}. \end{aligned}$$

   (e) It then computes

$$c := H(\mathtt{gpk} \| B \| K \| T \| R_1 \| R_2 \| m).$$

   (f) It computes in $\mathbb{Z}_p$

$$s_x := r_x + cx, \quad s := r_y + cy, \quad s_a := r_a + ca, \quad s_b := r_b + cb.$$

3. It outputs $\sigma := (\mathtt{tr}, B, K, T, c, s_x, s, s_a, s_b)$.

Note that $e(A, g_2)$ in the computation of R_2 can be pre-computed and re-used. Also observe that the traceable and non-traceable signatures have the same size and format. The signature generations for both signature types have the same complexity as well.

Verify : On input of $\mathtt{gpk}$, a message m, a signature $\sigma = (\mathtt{tr}, B, K, T, c, s_x, s, s_a, s_b)$, a list of revoked secret keys $\mathtt{sRL}$, and a list of revoked tracing keys $\mathtt{tRL}$, the verifying algorithm has the following steps:

1. It verifies that $B, K, T \in G_1$ and $s_x, s, s_a, s_b \in \mathbb{Z}_p$.
2. It computes $R_1 := B^s \cdot K^{-c}$.
3. If $\mathtt{tr} = 0$, it computes

$$\begin{aligned} \hat{R}_2 &:= e(T, g_2)^{-s_x} \cdot e(h_1, g_2)^{s} \cdot e(h_2, g_2)^{s_b} \cdot e(h_2, w)^{s_a} \cdot (e(g_1, g_2)/e(T, w))^{c} \\ &:= e(T, g_2^{-s_x} \cdot w^{-c}) \cdot T_1^{c} \cdot T_2^{s} \cdot T_3^{s_b} \cdot T_5^{s_a}, \end{aligned}$$

   otherwise, it computes

$$\begin{aligned} \hat{R}_2 &:= e(T, g_2)^{-s_x} \cdot e(h_2, g_2)^{s} \cdot e(h_1, g_2)^{s_b} \cdot e(h_1, w)^{s_a} \cdot (e(g_1, g_2)/e(T, w))^{c} \\ &:= e(T, g_2^{-s_x} \cdot w^{-c}) \cdot T_1^{c} \cdot T_3^{s} \cdot T_2^{s_b} \cdot T_4^{s_a}. \end{aligned}$$

4. It verifies that

$$c \stackrel{?}{=} H(\mathtt{gpk} \| B \| K \| T \| \hat{R}_1 \| \hat{R}_2 \| m).$$

5. If any of the above steps fails, it quits and outputs $\mathtt{invalid}$.
6. If $\mathtt{tr} = 0$, for each $f' \in \mathtt{sRL}$, if $K = B^{f'}$, it quits and outputs $\mathtt{revoked}$,

7. If $\mathtt{tr} = 1$, for each $y' \in \mathtt{tRL}$, if $K = B^{y'}$, it quits and outputs `revoked`.
8. If none of the above steps fails, it outputs `valid`.

Open : On input of gpk, a message m, a signature $\sigma = (\mathtt{tr}, B, K, T, c, s_x, s, s_a, s_b)$, a tracing database `dbase`, the open algorithm has the following steps:

1. If $\mathsf{Verify}(\mathsf{gpk}, m, \sigma, \emptyset, \emptyset) = \mathtt{invalid}$, it quits and outputs `invalid`.
2. If $\mathtt{tr} = 0$, it quites and outputs `untraceable`.
3. For each entry $(\mathtt{id}, \mathtt{tk})$ in `dbase`, it computes $K' = B^{\mathtt{tk}}$. If $K = K'$, it quits and outputs $(\mathtt{id}, \mathtt{tk})$.
4. If none of the entry in `dbase` matches, it outputs `untraceable`.

The above OTAA scheme is secure under the OTAA definition in Section 3. It is correct, user-controlled-anonymous under the XDH assumption, and user-controlled-traceable under the q-SDH assumption. Due to the space limit, the security proof will be given in the full version of this paper.

5.2 Efficiency of Our Scheme

The signing key of the above scheme comprises three elements in $\mathbb{Z}_p$ and one element in G_1. The signature of the above scheme takes one boolean variable, three elements in G_1, and five elements in $\mathbb{Z}_p$. Let p be a 256-bit prime number. Using 256-bit Barreto-Naehrig curves [4], security is approximately 128-bit and is about the same as a standard 3072-bit RSA signature. Each element in G_1 is 257-bit. Thus the signing key is only 1025-bit and the signature is 2052-bit in the above scheme. Using 170-bit MNT curves with approximate 80-bit security, the signing key is 681-bit and the signature is 1364-bit.

The signature generation requires three exponentiations in G_1 and one multi-exponentiation in G_T. The signature verification algorithm requires one multi-exponentiation in G_1, G_2, and G_T, respectively, one pairing operation, and n exponentiations in G_1, where n is the size of the revocation list. The open algorithm includes one signature verification and m exponentiations in G_1, where m is the total number of platforms issued by the issuer.

6 Comparisons with Group Signature and DAA Schemes

As we mentioned earlier, our OTAA scheme becomes a group signature scheme if all the signatures are traceable. Note that the group signature scheme derived from our OTAA scheme does not have the non-frameability property, as the issuer has the tracing keys for all the platforms and can frame any platforms.

We now show that our OTAA scheme is more efficient than the existing group signature schemes [7,20,8,27] in the following table. The BS group signature scheme [8] has a slightly smaller signature size than our OTAA scheme, but is less inefficient. In particular, the revocation check in [8] requires two pairings per revocation item and is much inefficient than our scheme. The FI group signature scheme [27] is more efficient than BBS scheme [7] and CL scheme [20]. The FI scheme is still inefficient than our OTAA scheme. In addition, the FI scheme

Table 1. A comparison between our OTAA scheme and pairing-based group signature schemes and DAA schemes with 80-bit security level, where EXP denotes a exponentiation or a scaler multiplication operation and P denotes a pairing operation.

	sign	verify	signature size
Our OTAA scheme	7 EXP	8 EXP + 1 P	1364-bit
BS Group Signatures [8]	8 EXP + 2 P	9 EXP + 3 P	1192-bit
BBS Group Signatures [7]	14 EXP	17 EXP + 1 P	2057-bit
CL Group Signatures [20]	16 EXP	13 EXP + 5 P	5296-bit
FI Group Signatures [27]	11 EXP	12 EXP + 1 P	1711-bit
BCL DAA scheme [11]	6 EXP + 3 P	5 EXP + 5 P	3063-bit
CMS DAA scheme [26]	5 EXP + 1 P	4 EXP + 5 P	1355-bit
Chen DAA scheme [24]	7 EXP	8 EXP + 1 P	1363-bit
BL DAA scheme [15]	7 EXP + 1 P	8 EXP + 1 P	1363-bit

itself does not support any revocation mechanisms. Revocation could be added to the FI scheme, but with an additional cost.

Our OTAA scheme is efficient than many pairing-based DAA schemes [11,26,15]. It has almost the same complexity as Chen's DAA scheme [24]. Note that in DAA schemes, the sign algorithm is split between a TPM and a host. In the comparison below, we sum up all the computations of the TPM and host.

7 Conclusion and Future Work

In this paper, we have presented a new cryptographic primitive called Optionally Traceable Anonymous Attestation (OTAA). OTAA can be used in trusted computing, content protection, e-commerce, identity cards, and beyond. OTAA is more capable at revocation than DAA, with relatively minimal compromise of privacy to the issuer, e.g., when a platform creates traceable signatures. In most of the transactions, the platform can use non-traceable signatures to safeguard his privacy. We provided an efficient construction of OTAA from bilinear map. The security model and OTAA scheme developed in this paper is a merely first step. There is room for improvement in the following areas.

1. In many group signature schemes such as in [1,27], there is a feature called non-frameability, in which the issuer cannot frame a honest signer for creating a group signature. OTAA does not have this feature. In fact, our OTAA construction is frameable, as the issuer has the tracing keys for all the platforms and can forge a signature using any platform's tracing key. It may be interesting to formally define non-frameability in OTAA and give a corresponding construction.
2. There are features in DAA that can be added to OTAA, such as (1) the signer and verifier can negotiate whether the signature is linkable, and (2) the signer computation can be split between a weak TPM as the main signer and a host. Adding those features to the OTAA scheme is not difficult, but it is a challenge to build a simple security model that captures all the features.

Acknowledgement

We thank Liqun Chen for her helpful discussions and feedback. We also thank the anonymous reviewers for their useful comments.

References

1. Ateniese, G., Camenisch, J., Joye, M., Tsudik, G.: A practical and provably secure coalition-resistant group signature scheme. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 255–270. Springer, Heidelberg (2000)
2. Ateniese, G., Song, D.X., Tsudik, G.: Quasi-efficient revocation in group signatures. In: Blaze, M. (ed.) FC 2002. LNCS, vol. 2357, pp. 183–197. Springer, Heidelberg (2003)
3. Backes, M., Maffei, M., Unruh, D.: Zero-knowledge in the applied pi-calculus and automated verification of the direct anonymous attestation protocol. In: Proceedings of IEEE Symposium on Security and Privacy, pp. 202–215. IEEE Computer Society, Los Alamitos (2008)
4. Barreto, P.S.L.M., Naehrig, M.: Pairing-friendly elliptic curves of prime order. In: Preneel, B., Tavares, S. (eds.) SAC 2005. LNCS, vol. 3897, pp. 319–331. Springer, Heidelberg (2006)
5. Bellare, M., Shi, H., Zhang, C.: Foundations of group signatures: The case of dynamic groups. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 136–153. Springer, Heidelberg (2005)
6. 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)
7. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
8. Boneh, D., Shacham, H.: Group signatures with verifier-local revocation. In: Proceedings of 11th ACM Conference on Computer and Communications Security, October 2004, pp. 168–177 (2004)
9. Brickell, E., Camenisch, J., Chen, L.: Direct anonymous attestation. In: Proceedings of the 11th ACM Conference on Computer and Communications Security, pp. 132–145. ACM Press, New York (2004)
10. Brickell, E., Chen, L., Li, J.: A new direct anonymous attestation scheme from bilinear maps. In: Lipp, P., Sadeghi, A.-R., Koch, K.-M. (eds.) Trust 2008. LNCS, vol. 4968, pp. 166–178. Springer, Heidelberg (2008)
11. Brickell, E., Chen, L., Li, J.: Simplified security notions of direct anonymous attestation and a concrete scheme from pairings. International Journal of Information Security 8(5), 315–330 (2009)
12. Brickell, E., Li, J.: Enhanced Privacy ID: A direct anonymous attestation scheme with enhanced revocation capabilities. In: Proceedings of the 6th ACM Workshop on Privacy in the Electronic Society, October 2007, pp. 21–30. ACM Press, New York (2007)
13. Brickell, E., Li, J.: Enhanced Privacy ID: A remote anonymous attestation scheme for hardware devices. Intel. Technology Journal: Advances in Internet Security 13(2) (2009)
14. Brickell, E., Li, J.: Enhanced Privacy ID from bilinear pairing. Cryptology ePrint Archive, Report 2009/095 (2009), `http://eprint.iacr.org/`

15. Brickell, E., Li, J.: A pairing-based DAA scheme further reducing TPM resources. In: Acquisti, A., Smith, S.W., Sadeghi, A.-R. (eds.) TRUST 2010. LNCS, vol. 6101, pp. 181–195. Springer, Heidelberg (2010)
16. Brickell, E.F., Gordon, D.M., McCurley, K.S., Wilson, D.B.: Fast exponentiation with precomputation. In: Rueppel, R.A. (ed.) EUROCRYPT 1992. LNCS, vol. 658, pp. 200–207. Springer, Heidelberg (1993)
17. Camenisch, J., Groth, J.: Group signatures: Better efficiency and new theoretical aspects. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 120–133. Springer, Heidelberg (2005)
18. Camenisch, J., Kohlweiss, M., Soriente, C.: An accumulator based on bilinear maps and efficient revocation for anonymous credentials. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 481–500. Springer, Heidelberg (2009)
19. Camenisch, J., Lysyanskaya, A.: Dynamic accumulators and application to efficient revocation of anonymous credentials. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 61–76. Springer, Heidelberg (2002)
20. 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)
21. Camenisch, J., Shoup, V.: Practical verifiable encryption and decryption of discrete logarithms. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 126–144. Springer, Heidelberg (2003)
22. Camenisch, J., Stadler, M.: Efficient group signature schemes for large groups. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 410–424. Springer, Heidelberg (1997)
23. Chaum, D., van Heyst, E.: Group signatures. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 257–265. Springer, Heidelberg (1991)
24. Chen, L.: A DAA scheme requiring less TPM resources. In: Proceedings of the 5th China International Conference on Information Security and Cryptology, LNCS. Springer, Heidelberg (2009)
25. Chen, L., Morrissey, P., Smart, N.P.: Pairings in trusted computing. In: Galbraith, S.D., Paterson, K.G. (eds.) Pairing 2008. LNCS, vol. 5209, pp. 1–17. Springer, Heidelberg (2008)
26. Chen, L., Morrissey, P., Smart, N.P.: DAA: Fixing the pairing based protocols. Cryptology ePrint Archive, Report 2009/198 (2009), `http://eprint.iacr.org/`
27. Furukawa, J., Imai, H.: An efficient group signature scheme from bilinear maps. IEICE Transactions 89-A(5), 1328–1338 (2006)
28. Kiayias, A., Yung, M.: Group signatures with efficient concurrent join. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 198–214. Springer, Heidelberg (2005)
29. Leung, A., Mitchell, C.J.: Ninja: Non identity based, privacy preserving authentication for ubiquitous environments. In: Krumm, J., Abowd, G.D., Seneviratne, A., Strang, T. (eds.) UbiComp 2007. LNCS, vol. 4717, pp. 73–90. Springer, Heidelberg (2007)
30. Trusted Computing Group. TCG TPM specification 1.2 (2003), `http://www.trustedcomputinggroup.org`
31. Trusted Computing Group website, `http://www.trustedcomputinggroup.org`
32. Tsang, P.P., Au, M.H., Kapadia, A., Smith, S.W.: Blacklistable anonymous credentials: Blocking misbehaving users without TTPs. In: ACM Conference on Computer and Communications Security, pp. 72–81. ACM Press, New York (2007)