This paper was published in the *Proceedings of the International Conference on Applied Cryptography and Information Security (ACIS)*, 2006, pp. 327–337.

Traceable Signature: Better Efficiency and Beyond*

He Ge[1] and Stephen R. Tate[2]

[1] Dept. of Computer Science and Engineering, University of North Texas ge@unt.edu
[2] Dept. of Computer Science and Engineering, University of North Texas srt@cs.unt.edu

Abstract. In recent years one of the most active research areas in applied cryptography is the study of techniques for creating a group signature, a cryptographic primitive that can be used to implement anonymous authentication. Some variants of group signature, such as traceable signature, and authentication with variable anonymity in a trusted computing platform, have also been proposed. In this paper we propose a traceable signature scheme with variable anonymity. Our scheme supports two important properties for a practical anonymous authentication system, i.e., corrupted group member detection and fair tracing, which have unfortunately been neglected in most group signature schemes in the literature. We prove the new scheme is secure in the random oracle model, under the strong RSA assumption and the decisional Diffie-Hellman assumption.

Keywords: Group signature, Traceable Signature, Anonymous Authentication, Variable Anonymity, Cryptographic Protocol.

1 Introduction

In this paper, we present new techniques for performing anonymous authentication, in which authenticated users receive credentials from a designated group manager, and in later interactions a user can prove possession of such a credential in a privacy-preserving manner. Anonymous authentication has been one of the most active research areas in applied cryptography in recent years.

The most heavily studied type of anonymous authentication system is the "group signature scheme," which provides a well-defined set of services and security guarantees that we describe in more detail below.[3] However, several authors have identified various desirable properties not provided by the group signature definition, and have introduced variants of this basic scheme including work on "anonymous credential systems" [6], "traceable signatures" [14], and a system designed for trusted computing platforms called "direct anonymous attestation" [4]. Our contribution in this paper is to show how the well-known group signature scheme of Ateniese *et al*. [1], which we call the ACJT scheme, can be modified to a traceable signature so that it supports a particularly useful extension from the work on direct anonymous attestation that allows a prover and verifier to agree on a variable degree of signature linkability. Our modifications to the ACJT scheme replace operations with modified formulas that have the same computational complexity, so our system preserves the efficiency of the ACJT scheme while providing a unique set of features which is useful in many situations.

* This research is supported in part by NSF award 0208640.

[3] An extensive bibliography of group signature literature can be found at http://www.i2r.a-star.edu.sg/icsd/staff/guilin/bible/group-sign.htm

1.1 Background

Group signature is a privacy-preserving signature scheme introduced by Chaum and Heyst in 1991 [10]. In such a scheme, a group member can sign a message on behalf of the group without revealing his identity. Only the group manager or the specified open authority can open a signature and find its originator. Signatures made by the same user cannot be identified as from the same source, i.e, "linked". Recently, group signature has attracted considerable attention, and many schemes have been proposed in the literature (e.g., [8, 1, 6, 7, 5]). Creating an anonymous authentication scheme from a group signature is simple: the group is simply the set of authorized users, and authentication is performed by a group member placing a group signature on a challenge (nonce) sent by the service requiring authentication. From the properties of group signatures, all the service or an attacker can learn is that the signature was made by a valid group member (i.e., an authorized user).

However, group signature does not provide certain important features for a more hostile or realistic environment where group members could be malicious or compromised. In such settings, an efficient mechanism should be available to reveal all the malicious behaviors of corrupted members. In group signature, identification of signatures from corrupted members has to be done by opening all signatures. This is either inefficient (centralized operation by the group manager), or unfair (unnecessarily identifying all innocent group members' signatures). To overcome this shortcoming, Kiayias *et al.* proposed a variant of group signature, called traceable signature [14]. They define "traceability" as the ability to reveal all the signatures signed by a group member without requiring the open authority to open them. Tracing can be done by "trace agents" distributively and efficiently. They also introduced the concept of "self-traceability", or "claiming". That is, a group member himself can stand out, claiming a signature signed by himself without compromising his other signatures and secrets. The subtlety lies in that a group member should be able to do this without keeping all one time random values in his signatures. In group signature, a group member may also be able to claim his signatures, but he has to keep all his transaction transcripts including some random values, making "claiming" highly impractical and a security risk.

The Trusted Computing Group [15] has recently proposed an architecture called the "trusted computing platform" to enhance computer security. A trusted computing platform is a computing device integrated with a cryptographic chip called the trusted platform module (TPM). The TPM is designed and manufactured so that all other remote parties can trust cryptographic computing results from this TPM. To protect the privacy of a TPM owner, an anonymous authentication technique, called Direct Anonymous Attestation (DAA), has been deployed in recent versions of the trusted computing platform. DAA can be seen as a group signature scheme without openability. DAA introduces the notion of "variable anonymity," which is conditionally linkable anonymous authentication: the same TPM will produce linkable signatures for a certain period of time. The period of time during which signatures can be linked can be determined by the parties involved and can vary from an infinitesimally short period (leading to completely unlinkable signatures) to an infinite period (leading to completely linkable signatures). Signatures made by the same user in different periods of time or to different servers cannot be linked. By setting the linkability period to a moderately short time period (a day to a week) a server can potentially detect if a key has been compromised and is being used by many different users, while still offering some amount of unlinkability.

1.2 Our results

In the previous section we briefly introduced some of the available techniques for anonymous authentication. Numerous constructions with different features have been proposed to accommodate different properties. This raised the question which we address in this paper: Can we devise a construction which combines the features from different authentication primitives? More specifically, can we have a traceable signature scheme which also supports variable anonymity? So far as we know, no such scheme has been proposed to work in this manner, probably because variable anonymity is a recently identified feature in anonymous authentication.

We consider the combination of traceability and variable anonymity to be particularly important for anonymous authentication. Variable anonymity is the only way key sharing violations can be detected, while traceability is the efficient and fair way to reveal all malicious behaviors. More specifically, while the standard group signature scheme can use the open authority to identify a user that performs malicious actions, consider what happens when one authorized user shares his authentication credential with a set of co-conspirators. For example, a large set of users could share a single subscription to some pay web site. Since all authentications are completely unlinkable in a group signature scheme, it would be impossible to determine whether 1000 requests coming in during a day are from 1000 different valid users or from 1000 people sharing a single valid credential. Introducing linkability for a limited time period is the only way to detect this, and if an unusually high number of requests using the same credential come in from different IP addresses during the same day, then this could be flagged as potentially malicious behavior. After that, the open authority can open the signatures to determine the real owner of this credential, and the tracing trapdoor associated with this credential is further revealed to trace agents by the group manager. Then the trace agents reveal all the behaviors associated with the trapdoor for further investigation. At the same time, a tracing trapdoor may be published on the revocation list for verifiers to identify future requests by this member. In our opinion, to build up a realistic anonymous authentication system, the combination of traceability and variable anonymity is a must.

In this paper, we present our construction for traceable signature that supports variable anonymity. Our construction is built up from the well-known ACJT group signature [1]. The traceable signature due to Kiayias *et al.*, which we refer to as the KTY scheme in this paper, is also built up from the ACJT scheme. However, our construction improves on the KTY scheme in three aspects. First, we adopt the same group membership certificate as in the ACJT scheme. The KTY scheme changes the group certificate in the ACJT scheme to integrate the tracing trapdoor. We show this change is unnecessary by identifying that tracing trapdoors in fact are already available in the ACJT scheme. Second, our tracing mechanism is more efficient than the KTY scheme. Our scheme uses a hash function to create generators while the KTY scheme uses expensive exponentiation computation. Finally, our scheme supports variable anonymity while the KTY scheme does not. Thus, our scheme is more efficient and flexible than the KTY scheme.

The rest of this paper is organized as follows. The next section introduces a concrete model for our signature scheme. Section 3 reviews some definitions, cryptographic assumptions, and building blocks of our proposed scheme. Section 4 presents the pro-

posed scheme. Security properties are considered in Section 5. Finally, we summarize and give conclusions in section 6.

2 The Model

This section introduces the model for traceable signature [14], which is a variant of the group signature model (e.g. [1]). Both of these two models include operations for Setup, Join, Sign, Verify, and Open. The traceable signature model has additional operations for traceability: Reveal, Trace, Claim (Self-trace) and Claim-Verify.

Definition 1. *A traceable signature is a digital signature scheme with four types of participants: Group Manager, Group Members, Open Authorities, and Trace Agents. It consists of the following procedures:*

- **Setup***: For a given security parameter σ, the group manager produces system-wide public parameters and a group manager master key for group membership certificate generation.*
- **Join***: An interactive protocol between a user and the group manager. The user obtains a group membership certificate to become a group member. The public certificate and the user's identity information are stored by the group manager in a database for future use.*
- **Sign***: Using its group membership certificate and private key, a group member creates a group signature for a message.*
- **Verify***: A signature is verified to make sure it originates from a legitimate group member without the knowledge of which particular one.*
- **Open***: Given a valid signature, an open authority discloses the underlying group membership certificate.*
- **Reveal***: The group manager outputs the tracing trapdoor associated with a group membership certificate.*
- **Trace***: Trace agents check whether a signature is associated with a tracing trapdoor.*
- **Claim (Self-trace)***: A group member creates a proof that he created a particular signature.*
- **Claim_Verify***: A party verifies the correctness of the claiming transcript.*

Similar to group signatures, a traceable signature scheme should satisfy the following properties:

- **Correctness**: Any valid signature can be correctly verified by the Verify protocol and a valid claiming proof can be correctly verified.
- **Forgery-Resistance**: A valid group membership certificate can only be created by a user and the group manger through Join protocol.
- **Anonymity**: It is infeasible to identify the real signer of a signature except by the open authority or if the signature has been claimed.
- **Unlinkability**: It is infeasible to link two different signatures of the same group member.
- **Non-framing**: No one (including the group manager) can sign a message in such a way that it appears to come from another user if it is opened.
- **Traceability**: Given a tracing trapdoor, trace agents can reveal all signatures associated with the trapdoor. A group member can claim (self-trace) his signatures.

3 Definitions and Preliminaries

This section reviews some definitions, widely accepted complexity assumptions that we will use in this paper, and building blocks for our construction.

Definition 2 (Special RSA Modulus). *An RSA modulus $n = pq$ is called special if $p = 2p' + 1$ and $q = 2q' + 1$ where p' and q' also are prime numbers.*

Definition 3 (Quadratic Residue Group QR_n). *Let Z_n^* be the multiplicative group modulo n, which contains all positive integers less than n and relatively prime to n. An element $x \in Z_n^*$ is called a* quadratic residue *if there exists an $a \in Z_n^*$ such that $a^2 \equiv x \pmod{n}$. The set of all quadratic residues of Z_n^* forms a cyclic subgroup of Z_n^*, which we denote by QR_n. If n is the product of two distinct primes, then $|QR_n| = \frac{1}{4}|Z_n^*|$.*

The security of our techniques relies on the following security assumptions which are widely accepted in the cryptography literature (see, for example, [2, 13, 8, 1]).

Assumption 1 (Strong RSA Assumption) *Let n be an RSA modulus. The* Flexible RSA Problem *is the problem of taking a random element $u \in Z_n^*$ and finding a pair (v, e) such that $e > 1$ and $v^e = u \pmod{n}$. The* Strong RSA Assumption *says that no probabilistic polynomial time algorithm can solve the flexible RSA problem with non-negligible probability.*

Assumption 2 (Decisional Diffie-Hellman Assumption for QR_n) *Let n be a special RSA modulus, and let g be a generator of QR_n. For the two distributions (g, g^x, g^y, g^{xy}), (g, g^x, g^y, g^z), $x, y, z \in_R Z_n$, there is no probabilistic polynomial-time algorithm that distinguishes them with non-negligible probability.*

The building blocks of our technique are *statistical honest-verifier zero knowledge proofs of knowledge* related to discrete logarithms over QR_n [9, 8]. They may include protocols for problems such as the knowledge of the discrete logarithm, the knowledge of equality of two discrete logarithms, the knowledge of the discrete logarithm that lies in certain interval, etc. We introduce one of them here. Readers may refer to the original papers for more details.

Protocol 1 *Let n be a special RSA modulus, QR_n be the quadratic residue group modulo n, and g be a generator of QR_n. ϵ, l, l_c are security parameters that are all greater than 1. X is a constant number. A prover Alice knows x, the discrete logarithm of T_1, and $x \in [X - 2^l, X + 2^l]$. Alice demonstrates her knowledge of $x \in [X - 2^{\epsilon(l+l_c)}, X + 2^{\epsilon(l+l_c)}]$ as follows.*

1. *Alice picks a random $t \in \pm\{0,1\}^{\alpha(l+l_c)}$ and computes $T_2 = g^t \pmod{n}$. Alice sends (T_1, T_2) to a verifier Bob.*
2. *Bob picks a random $c \in \{0,1\}^{l_c}$ and sends it to Alice.*
3. *Alice computes $w = t - c(x - X)$, and $w \in \pm\{0,1\}^{\alpha(l+l_c)+1}$. Alice sends w to Bob.*
4. *Bob checks $w \in \pm\{0,1\}^{\alpha(l+l_c)+1}$ and*

$$g^{w-cX} T_1^c =? T_2 \pmod{n}.$$

If the equation holds, Alice proves knowledge of the discrete logarithm of T_1 lies in the range $[X - 2^{\epsilon(l+l_c)}, X + 2^{\epsilon(l+l_c)}]$.

Remark 1. It should be emphasized that while Alice knows a secret x in $[X-2^l, X+2^l]$, the protocol only guarantees that x lies in the extended range $[X-2^{\epsilon(l+l_c)}, X+2^{\epsilon(l+l_c)}]$.

Remark 2. Using the Fiat-Shamir heuristic [12], the protocol can be turned into a non-interactive "signature of knowledge" scheme, which is secure in the random oracle model [3]. We will introduce the proposed scheme in the manner of "signature of knowledge" in next section.

4 Traceable Signature

Our construction is built upon the ACJT group signature scheme. We adopt the same system parameters, group certificates, and Join protocol. The Sign and Verify protocols have been changed to support traceability and variable anonymity. In the following presentation, we use the same notation as in the original paper to make it easier for readers to see how we convert the ACJT scheme into a traceable signature scheme.

4.1 The System Parameters

The following system parameters are set up when the system is initialized and the group manager key is generated.

- A special RSA modulus $n = pq$, $p = 2p' + 1$, $q = 2q' + 1$, with p, p', q, q' all prime
- Random elements $a, a_0, g \in QR_n$ of order $p'q'$, i.e., these numbers are generators of QR_n
- Security parameters used in protocols: $\epsilon > 1, k, l_p$
- Length parameters $\lambda_1, \lambda_2, \gamma_1, \gamma_2$. $\lambda_1 > \epsilon(\lambda_2 + k) + 2$, $\lambda_2 > 4l_p$, $\gamma_1 > \epsilon(\gamma_2 + k) + 2$, and $\gamma_2 > \lambda_1 + 2$
- Integer ranges $\Lambda =]2^{\lambda_1} - 2^{\lambda_2}, 2^{\lambda_1} + 2^{\lambda_2}[$ and $\Gamma =]2^{\gamma_1} - 2^{\gamma_2}, 2^{\gamma_1} + 2^{\gamma_2}[$
- Three strong collision-resistant hash functions: $\mathcal{H}_1, \mathcal{H}_2 : \{0,1\}^* \rightarrow Z_n^*$, and $\mathcal{H}_3 : \{0,1\}^* \rightarrow \{0,1\}^k$
- A message to be signed: $m \in \{0,1\}^*$
- The public parameters are (n, a, a_0, g).
- The secret parameters for the group manager are (p', q').

The open authority creates his ElGamal public keypair [11], i.e., private key x and public key y such that $y = g^x \pmod n$.

4.2 Variable Anonymity Parameter

To achieve variable anonymity, each signature will belong to a "linkability class" that is identified using a "linkability class identifier," or LCID. All signatures made by the same group member with the same LCID are linkable, and in an interactive authentication protocol the LCID can be negotiated and determined by the two parties. For example, to link authentications to a single server over a single day, the LCID could simply be the server name concatenated with the date. If the same LCID is always used with a particular server (e.g., the server name), then the result is a pseudo-anonymity system. If complete anonymity is desired, the signer can simply pick a random LCID (which is possible if the server isn't concerned with linkability and allows arbitrary LCIDs).

4.3 Join Protocol

The same Join protocol is adopted as in the original scheme. A group membership certificate is in the form of $A_i = (a^{x_i} a_0)^{1/e_i} \pmod n$ where $x_i \in \Lambda$ is the secret of the group member, and $e_i \in_R \Gamma$ is a random prime number that is known to both the group member and group manager.[4]

In our scheme, e_i is treated as tracing trapdoor, and kept secret by the group member and group manager. When an open authority reveals A_i for a signature, the group manager sends the corresponding e_i to the trace agents in order to trace all signatures associated with e_i.

x_i is treated as self-tracing trapdoor, which is used by a group member to claim his signatures. Since x_i is the secret of group member, only group member himself have the ability to claim his signatures.

4.4 Sign Protocol

In order to sign a message m, a group member does the following:

- Derive two generators i and j of QR_n by hashing the LCID of this signature.

  $$i = (\mathcal{H}_1(LCID))^2 \pmod n,\ j = (\mathcal{H}_2(LCID))^2 \pmod n.$$

  In the random oracle model, with the hash functions modeled by random oracles, each distinct LCID results in i and j being random generators of QR_n with overwhelming probability.
- Generate a random value $w \in_R \{0,1\}^{2l_p}$ and compute:

  $$T_1 = A_i y^w \pmod n,\ T_2 = g^w \pmod n,\ T_3 = i^{e_i} \pmod n,\ T_4 = j^{x_i} \pmod n$$

- Randomly (uniformly) choose $r_1 \in_R \pm\{0,1\}^{\epsilon(\gamma_2+k)}, r_2 \in_R \pm\{0,1\}^{\epsilon(\lambda_2+k)}$, and $r_3 \in_R \pm\{0,1\}^{\epsilon(\lambda_1+2l_p+k+1)}$, and compute
  - $d_1 = T_1^{r_1}/(a^{r_2} y^{r_3}) \pmod n$, $d_2 = T_2^{r_1}/g^{r_3} \pmod n$, $d_3 = i^{r_1} \pmod n$, $d_4 = j^{r_2} \pmod n$.
  - $c = \mathcal{H}_3(g||i||j||y||a_0||a||T_1||T_2||T_3||d_1||d_2||d_3||d_4||m)$;
  - $s_1 = r_1 - c(e_i - 2^{\gamma_1}), s_2 = r_2 - c(x_i - 2^{\lambda_1}), s_3 = r_3 - c e_i w$ (all in Z_n).
- Output the signature tuple $(LCID, c, s_1, s_2, s_3, T_1, T_2, T_3, T_4)$.

4.5 Verify Protocol

To verify a signature $(LCID, c, s_1, s_2, s_3, T_1, T_2, T_3, T_4)$, a verifier does the following.

- Compute the same generators i and j, and then

  $$c' = \mathcal{H}_3(g||i||j||a_0||a||T_1||T_2||T_3||T_4||a_0^c T_1^{s_1 - c2^{\gamma_1}}/(a^{s_2 - c2^{\lambda_1}} y^{s_3})||$$

  $$T_2^{s_1 - c2^{\gamma_1}}/g^{s_3}||i^{s_1 - c2^{\gamma_1}} T_3^c||j^{s_2 - c2^{\lambda_1}} T_4^c||m)$$

- Accept the signature if and only if $c = c'$ and $s_1 \in \pm\{0,1\}^{\epsilon(\gamma_2+k)+1}$, $s_2 \in \pm\{0,1\}^{\epsilon(\lambda_2+k)+1}$, $s_3 \in \pm\{0,1\}^{\epsilon(\lambda_1+2lp+k+1)+1}$.

[4] Kiayias *et al.* have showed the range of x_i, e_i can be much smaller without compromising the scheme's security [14]. For simplicity, we still follow the definition in ACJT scheme.

4.6 Open and Reveal Protocol

For a valid signature, the open authority opens a signature to find its originator by ElGamal decryption:

$$A_i = T_1/T_2^x \pmod{n}.$$

For the non-framing property, the open authority must also issue a proof that it correctly revealed the group member, which can be done identically to the method used by the ACJT group signatures.

The opened certificate A_i is submitted to the group manager, and the group manager reveals the corresponding tracing trapdoor e_i to the trace agents.

4.7 Trace Protocol

To trace a group member, trace agents use e_i to reveal all the signatures by a group member by checking whether

$$i^{e_i} =? T_3 \pmod{n}.$$

To claim a signature, a group member proves its knowledge of discrete logarithm of T_4 with base j through Protocol 1.

5 Security Properties

Our scheme uses the same certificate as in the ACJT group signature. We have changed their Sign and Verify protocols. The security properties, such as, forgery-resistance, anonymity, non-framing, are unaffected by these changes. In this section, we only discuss the security properties affected by our change. Readers may refer to the original paper for other security arguments — the following theorem is representative, and further discussion is available in the full version of this paper.

Theorem 1 (Coalition-resistance). *Under the strong RSA assumption, a group certificate* $[A_i = (a^{x_i}a_0)^{1/e_i} \pmod{n}, e_i]$ *with* $x \in \Lambda$ *and* $e_i \in \Gamma$ *can be generated only by the group manager provided that the number* K *of certificates the group manager issues is polynomially bounded.*

Now, we address the security of Sign and Verify protocol, which is described as the following theorem.

Theorem 2. *Under the strong RSA assumption, and the decisional Diffie-Hellman assumption, the interactive protocol underlying the group signature scheme is a statistical zero-knowledge (honest-verifier) proof of knowledge of a membership certificate and a corresponding membership secret key.*

Proof. The proof for correctness is straightforward. A proof for the zero-knowledge property (simulator) following the same method in the KTY scheme (*Lemma 20*) appears in the full version of this paper. We only address the existence of a knowledge extractor, which is able to recover the group certificate when it has found two accepting tuples under the same commitment and different challenges from a verifier. Let $(T_1, T_2, T_3, d_1, d_2, d_3, c, s_1, s_2, s_3)$ and $(T_1, T_2, T_3, d_1, d_2, d_3, c', s_1', s_2', s_3')$ be such tuples.

Since $d_4 \equiv j^{s_2 - c2^{\lambda_1}} T_4^c \equiv j'^{s_2 - c'2^{\lambda_1}} T_4^{c'} \pmod{n}$, we have

$$j^{(s_2' - s_2) + (c-c')2^{\lambda_1}} \equiv T_4^{c-c'} \pmod{n}.$$

Under the strong RSA assumption, $c - c'$ has to divide $(s_2' - s_2) + (c - c')2^{\lambda_1}$. Therefore we have $\tau_1 = (s_2' - s_2)/(c - c') + 2^{\lambda_1}$.

Since $d_3 \equiv i^{s_1 - c2^{\gamma_1}} T_3^c \equiv i^{s_1' - c'2^{\gamma_1}} T_3^{c'} \pmod{n}$, we have

$$i^{(s_1' - s_1) + (c-c')2^{\gamma_1}} \equiv T_3^{c-c'} \pmod{n}.$$

Likewise, under the strong RSA assumption, $c - c'$ has to divide $(s_1' - s_1)$. We obtain $\tau_2 = (s_1' - s_1)/(c - c') + 2^{\gamma_1}$.

Since $d_2 \equiv T_2^{s_1 - c2^{\gamma_1}} / g^{s_3} \equiv T_2^{s_1' - c'2^{\gamma_1}} / g^{s_3'} \pmod{n}$, we have

$$T_2^{(s_1' - s_1) + (c-c')2^{\gamma_1}} \equiv g^{s_3' - s_3} \pmod{n}.$$

Similarly, we have $\tau_3 = (s_3' - s_3)/((s_1' - s_1) + (c - c')2^{\gamma_1})$.

Since $d_1 \equiv a_0^c T_1^{s_1 - c2^{\gamma_1}} / (a^{s_2 - c2^{\lambda_1}} y^{s_3}) \equiv a_0^{c'} T_1^{s_1 - c'2^{\gamma_1}} / (a^{s_2 - c'2^{\lambda_1}} y^{s_3}) \pmod{n}$, We have

$$a^{s_2' - s_2 + (c-c')2^{\lambda_1}} a_0^{c-c'} \equiv T_1^{s_1' - s_1 + (c-c')2^{\gamma_1}} / y^{s_3' - s_3} \pmod{n}.$$

We further obtain

$$a^{(s_2' - s_2)/(c-c') + 2^{\lambda}} a_0 \equiv (T_1 / y^{(s_3' - s_3)/((s_1' - s_1) + (c-c')2^{\gamma_1})})^{(s_1' - s_1)/(c-c') + 2^{\gamma_1}} \pmod{n}.$$

Finally, let $A_i = T_1 / y^{\tau_3} \pmod{n}$, and then we obtain a valid certificate (A_i, τ_2, τ_1) such that $A_i^{\tau_2} = a^{\tau_1} a_0 \pmod{n}$, and τ_1, τ_2 lie in the valid range due to the length restriction on s_1, s_2, s_3 and c. Therefore we have demonstrated the existence of a knowledge extractor that can fully recover a valid group certificate. □

Unlinkability follows the same argument in the ACJT group signature for T_1, T_2. Since we define a new T_3, T_4 in our traceable signature, we need to show this change still keeps the unlinkability property (for different generators i and i'). Similar to the case in the ACJT group signature, the problem of linking two tuples (i, T_3), (i', T_3'), is equivalent to deciding the equality of the discrete logarithms of T_3, T_3' with base i, i' respectively. This is assumed to be infeasible under the decisional Diffie-Hellman assumption over QR_n. (j, T_4), (j', T_4') also follows the same argument. Therefore, we have the following result.

Theorem 3 (Unlinkability). *Under the decisional Diffie-Hellman assumption over QR_n and with $\mathcal{H}_1$ and $\mathcal{H}_2$ as random oracles, there exists no probabilistic polynomial-time algorithm that can make the linkability decision for any two arbitrary tuples $(i, T_3), (i', T_3')$, or $(j, T_4), (j', T_4')$ with non-negligible probability.*

6 Conclusion

We have presented a traceable signature scheme which is an enhancement of the ACJT group signature scheme [1] that supports variable anonymity. Our scheme is a more general solution to anonymous authentication, due to its support of traceability and variable anonymity. Traceability provides an efficient and fair mechanism to reveal and revoke corrupted group members, which is very important to a large, realistic anonymous authentication system. Variable anonymity can be adjusted to provide a wide range of linkability properties, from completely unlinkable signatures, to signatures linkable within a fixed time period, to completely linkable signatures (giving what is essentially a fixed pseudonym system). In practice, the amount of linkability would be determined by a risk analysis of the application, balancing the goal of protecting a user's privacy against a provider's goal of detecting inappropriate uses of keys. As our scheme supports the full range of linkability options, it provides the best available flexibility to users as well as providers. Finally, we have proved that our new signature scheme is secure under the strong RSA assumption and the Decisional Diffie-Hellman assumption over QR_n.

References

1. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *Advances in Cryptology — Crypto*, pages 255–270, 2000.
2. N. Baric and B. Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In *Advances in Cryptology — Eurocrypto*, pages 480–494, 1997.
3. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient procotols. In *ACM Conference on Computer and Communication Security*, pages 62–73, 1993.
4. E. Brickell, J. Camenisch, and L. Chen. Direct anonymous attestation. In *ACM Conference on Computer and Communications Security*, pages 132–145, 2004.
5. J. Camenisch and J. Groth. Group signatures: Better efficiency and new theoretical aspects. In *Security in Communication Networks (SCN 2004), LNCS 3352*, pages 120–133, 2005.
6. J. Camenisch and A. Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *Advances in Cryptology — Crypto'02, LNCS 2442*, pages 61–76, 2002.
7. J. Camenisch and A. Lysyanskaya. A signature scheme with efficient protocols. In *SCN'02, LNCS 2576*, pages 268–289, 2002.
8. J. Camenisch and M. Stadler. A group signature scheme with improved efficiency. In *Advances in Cryptology — ASIACRYPT'98, LNCS 1514*, pages 160–174, 1998.
9. A. Chan, Y. Frankel, and Y. Tsiounis. Easy come - easy go divisible cash. In *K. Yyberg, editor, Advances in Cryptology – Eurocrypt'98, LNCS 1403*, pages 561 – 574. Sringer-Verlag, 1998.
10. D. Chaum and E. van Heyst. Group signature. In *Advances in Cryptology — Eurocrypt*, pages 390–407, 1992.
11. T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Advances in Cryptology — Crypto*, pages 10–18, 1984.
12. A. Fiat and A. Shamir. How to prove yourself: practical solutions to identification and signature problems. In *Advances in Cryptology — CRYPTO'86, LNCS 263*, pages 186–194. Springer-Verlag, 1987.
13. E. Fujisaki and T. Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *Advances in Cryptology — Crypto*, pages 16–30, 1997.
14. A. Kiayias, Y. Tsiounis, and M. Yung. Traceable signatures. In *Advances in Cryptology— Eurocypt, LNCS 3027*, pages 571–589. Springer-Verlag, 2004.
15. TCG. http://www.trustedcomputinggroup.org.