k-Times Anonymous Authentication (Extended Abstract)

Isamu Teranishi, Jun Furukawa, Kazue Sako

Internet Systems Research Laboratories, NEC Corporation
1753 Shimonumabe, Nakahara-Ku, Kawasaki 211-8666, Japan.
teranisi@ah.jp.nec.com, j-furukawa@ay.jp.nec.com, k-sako@ab.jp.nec.com

Abstract. We propose an authentication scheme in which users can be authenticated anonymously so long as times that they are authenticated is within an allowable number. The proposed scheme has two features that allow 1) no one, not even an authority, identify users who have been authenticated within the allowable number, and that allow 2) anyone to trace, without help from the authority, dishonest users who have been authenticated beyond the allowable number by using the records of these authentications. Although identity escrow/group signature schemes allow users to be anonymously authenticated, the authorities in these schemes have the unnecessary ability to trace any user. Moreover, since it is only the authority who is able to trace users, one needs to make cumbersome inquiries to the authority to see how many times a user has been authenticated. Our scheme can be applied to e-voting, e-cash, electronic coupons, and trial browsing of content. In these applications, our scheme, unlike the previous one, conceals users' participation from protocols and guarantees that they will remain anonymous to everyone.

1 Introduction

1.1 Background

Many applications, such as e-voting [19, 21, 27, 29, 32], e-cash [1, 9, 12, 16, 30], electronic coupons [25, 26, 28], and trial browsing of content, often need to allow users to anonymously use these to protect privacy. At the same time, these applications need to restrict the number of times users can use them. These applications have three common requirements. The first is that they should provide honest users as much privacy as possible. The second is that they should be able to trace dishonest users easily. The third is that they should be able to restrict the number of times users can use applications.

However, if an application provider authenticates each user by receiving the user's signature when the user accesses it, a problem arises in that the provider is able to know who is using the application.

By following the authentication procedure of an identity escrow/group signature scheme [2, 4, 6, 7, 15, 24, 22], instead of an ordinary authentication scheme, users can be authenticated by the application provider without revealing their

ID to it. However, this method also does not fully satisfy all the requirements. First, an authority called the group manager can identify honest users. Second, providers needs to make cumbersome inquiries of the group manager to trace dishonest users. Third, there is no easy way for the provider to restrict the number of times users can use applications.

1.2 Properties of Proposed Scheme

We propose an authentication scheme called *k-times anonymous authentication* (k-TAA) that satisfies the three requirements mentioned in the previous section. An authority called the *group manager* first registers users in the proposed scheme. Each application provider(AP) then publishes the number of times a user is allowed to use their application. The registered users can be authenticated by various APs.

The proposed scheme satisfies the following properties:

1. No one, *not even the group manager*, is able to identify the authenticated user, if authenticated user is honest.
2. No one, *not even the group manager*, is able to decide whether two authentication procedures are performed by the same user or not, if the user(s) is/are honest.
3. Any user who was accurately detected as having accessed more than the allowed number of times can be correctly traced *using only the authentication log of the AP and public information*.
4. No colluders, *not even the group manager*, are able to be authenticated by an AP provider on behalf of an honest user.
5. Once a user has been registered by the group manager, the user does not need to access the group manager.
6. Each AP can independently determine the maximum number of times a registered user can anonymously access the AP.

We stress that the group manager of our scheme has less authority than one of an identity escrow/group signature scheme. He cannot trace honest users. His sole role is registering users.

The proposed scheme also has directly uses as a *k-times anonymous signature*.

We formalize security requirements of k-TAA, then prove that the proposed scheme is secure under strong RSA assumption and DDH assumption.

1.3 Comparison with Related Work

Using known schemes, one can construct a scheme that has similar properties to ours. However, these schemes have some problems.

Blind Signature Scheme Using the blind signature scheme [13], one can construct a scheme that has similar properties to ours. In each authentication, a user receives the group manager's blind signature and sends this signature to an AP. The AP accepts the authentication if the signature sent is valid. However, the scheme does not work well when there are multiple APs and their allowed number of access times is more than one.

Electronic Cash that can be spent k-times Using multi-show cash [9] (i.e., electronic cash that one can spend multiple times), we can construct another scheme that has similar properties to ours. The group manager plays the role of the bank. Before accessing an AP for the first time, a user asks the bank to give him digital cash that can be spent k times, where k is the number of the access times allowed by the AP. This cash plays the role of a ticket that allows users to access the AP, i.e., users send the digital cash to the AP when they are authenticated by the AP.

This scheme, however, has three drawbacks. First, the scheme is not efficient in the sense that users must access the group manager every time they access a new AP. Second, the group manager can learn which APs each user wants to be authenticated by. Third, one can determine whether two payment protocols have been performed by the same user or not by comparing the multi-show cash that was used in the protocols.

Electronic Coupon By using electronic coupons [25] as tickets, instead of electronic cash, one can construct another scheme, which also has similar properties to ours. This scheme, however, has the same problems that identity escrow/group signature schemes have. That is, the group manager can trace honest users, and an AP needs to make cumbersome inquiries of the group manager to trace dishonest users. The scheme also has a problem in that one can sometimes determine whether two authentication procedures have been performed by the same user or not[1].

List Signature and Direct Anonymous Attestation Independently proposed schemes [5] and [11] are similar to ours. However, these schemes are unmatched to our purpose. 1) These scheme cannot use two or more times signature. 2) A verifier of [5] cannot trace dishonest user without help of an authority. The scheme [11] has no way to trace dishonest user. 3) An authority of [5] can identify the authenticated user.

1.4 Applications

An example of an application of the k-TAA is trial browsing of content. Each provider wants to provide users with a service that allows them to browse content such as movies or music freely on trial. To protect user privacy, the providers allow users to use them anonymously. To prevent users from using the service too many times, the providers want to restrict the number of times that a user can access the service.

This privileged service is only provided to certain group members, say a member of the XXX community. The head of this community plays the role of the group manager, and registers users on behalf of providers in advance.

[1] Although the authors of [25] claim that no one can determine this, it does not. The reason is nearly same as that k-TAA scheme which an AP is able to know how many times users accesses to him. See 1) of 3.4.

The properties of the proposed scheme enables all honest users to browse content anonymously for an permitted number, but users who access beyond the allowed number of times are identified.

It can also be applied to voting, transferable cash, and coupons. In the one- or multiple-voting scheme constructed with the proposed scheme, a voter computes one- or k-times anonymous signatures on his ballot, and sends these anonymously to an election administrator. In this scheme, even authorities are unable to know whether a user has voted or not.

We can add transferability to the electronic cash scheme [9] with our scheme. To transfer cash to another entity, the owner of the cash computes a one-time anonymous signature on the electronic cash, and sends it with the signature to the receiver. Although a transferable electronic cash scheme has already been proposed in [16], our scheme has an advantage in that users does not need to access the bank each time they transfers cash to another entity.

One can construct an electronic coupon scheme by applying the k-times anonymous signature scheme directly. Our method has an advantage in that even an authority can not trace an honest user while anyone can trace a dishonest user.

2 Model

2.1 Entities

Three types of entities take part in the model, namely, the *group manager* (GM), *users*, and *application providers* (AP). The k-TAA scheme is comprised of the following five procedures: *setup*, *joining*, *bound announcement*, *authentication*, and *public tracing*.

In the setup, the GM generates a *group public key / group secret key* pair, and publishes the group public key. Joining is done between the GM and user who wants to join the group. After the procedure, the user obtains a *member public key / member secret key* pair. A user who has completed the joining procedure is called a *group member*.

In the bound announcement procedure, an AP announces the number of times each group member is allowed to access him. The AP v publishes his $\mathtt{ID}_v$, and the upper bound k_v.

An authentication procedure is performed between a user and an AP. The AP accepts the user if the user is a group member and has not accessed him more than the allowable times. The AP detects and rejects the user if he is not a group member, or if he is a group member but has accessed him more times than the announced bound allows. The AP records the data sent by the accepted or detected user in the *authentication log*.

Using only the public information and the authentication log, anyone can do public tracing. The procedure outputs some user ID i, "GM", or "NO-ONE", which respectively mean "the user i is authenticated by the AP more times than the announced bound", "the GM published the public information maliciously

", and "the public tracing procedure cannot find malicious entities". Note that we allow AP to delete some data from the log. Even if a member has been authenticated over the number of times, the tracing outputs `NO-ONE` if the AP deletes data about the member's authentication.

2.2 Requirements

A secure k-TAA must satisfy the following requirements:

- (**Correctness**): An honest group member will be accepted in authentication with an honest AP.
- (**Total Anonymity**): No one is able to identify the authenticated member, or to decide whether two accepted authentication procedures are performed by the same group member, if the authenticated user(s) has followed the authentication procedure within the permitted number of times per AP. These are satisfied even if other group members, the GM, and all APs collude with each other.
- (**Detectability**): Public tracing using an honest AP's authentication log does not output "`NO-ONE`", if a colluding subset of group members has been authenticated beyond the total number of times each colluding group member is able to be authenticated by the AP.
- (**Exculpability for users**) Public tracing does not output the ID of an honest user, even if other group members, the GM, and all APs collude with each other.
- (**Exculpability for the GM**) Public tracing does not output `GM` if the GM is honest. This is satisfied even if every group members and every APs collude with one another.

Note that these requirements implies the followings:

- (**Unforgeability**): Without the help of the GM or group members, no colluding group non-members can be authenticated as group members.
- (**Coalition Resistance**): A colluding subset of group members cannot generate a member public key/private key pair, which is not generated in the joining procedures.
- (**Traceability**): Any member who is detected of having accessed an AP predetermined bound can be traced from public information and the AP's authentication log.

As reasons the unforgeability and the coalition resistance properties are satisfied are almost the same as for the group signature case [7], we have not included an explanation. Traceability property is clearly satisfied.

3 Proposed Scheme

3.1 Notations and Terminologies

Let $\mathbb{N}$ and $\mathbb{Z}_n$ denote the ring of natural numbers and natural numbers from 0 to $n-1$, and $\mathrm{QR}(n)$ be the multiple group of quadratic residues of $\mathbb{Z}_n$. Let $\mathcal{H}_X$

denote a full domain hash function onto set X. Let $\mathtt{PROOF}(x \text{ s.t. } R(x))$ denote the proof of knowledge of x that satisfies the relation $R(x)$. We call prime p a *safe prime* if $(p-1)/2$ is also a prime number. We call n a *rigid integer* if natural number n can be factorized into two safe primes of equal length. Let G be a group with known order q, on which DDH problem is hard to solve. For simplification, we assume the bit length of q is equal to a security parameter κ.

3.2 Key Ideas

The proposed scheme is a modification of a group signature scheme. The GM is disabled from tracing an honest member, and anyone can identify who accessed over a number of times. Say an AP wishes to set the bound at k. Every time a member wants to be authenticated by the AP, he computes k intrinsic basis B_1, $\ldots$, B_k of AP which is called a *tag base*, then picks a tag base B_i which he has not used before. As long as the member uses different tag bases, he will not be identified. However, if he used the same tag base, anyone can identify who used the tag base twice.

3.3 Summary of Proposed Scheme

Let G be a group on which DDH problems are hard to solve. In the setup, the GM publishes a rigid integer n, elements $a, a_0 \in_U \mathrm{QR}(n)$, and an element b of G.

In joining, the a user and GM compute a member public key/secret key pair $((A,e),x)$ such that an equation $a^x a_0 = A^e$ is satisfied, x and e are elements of some previously determined intervals, and e is prime, and add b^x and his ID to public list, which is called *identification list*.

A tag base is a pair $(t, \check{t})$ of elements of the group G. They must be a hash values of some data, to prevent to be known the discrete logarithm of each others. In each authentication, an AP sends random number ℓ to a member, then the member sends back a *tag* $(\tau, \check{\tau}) = (t^x, (b^\ell \check{t})^x)$ with a validity proof. If the member does not have computed two tags using the same tag base, no one is able to trace that user, since DDH problem on G is hard to solve. However, if the member computes another tag $(\tau', \check{\tau}') = (t^x, (b^{\ell'} \check{t})^x)$ using the same tag base, AP can search these from his authentication log since these satisfy $\tau = \tau'$, and one can compute $(\check{\tau}/\check{\tau}')^{1/(\ell-\ell')} = ((b^\ell \check{t})^x/(b^{\ell'}\check{t})^x)^{1/(\ell-\ell')} = b^x$. Since identification list preserves user ID which corresponds b^x, one can identify the member.

3.4 Concerns

To construct the scheme we propose, we need to consider the followings:
1) *If an AP is able to know, w, the member accesses to him, the total anonymity property is not satisfied.* Suppose the number of times, w_1, that member M_1 has accessed to an AP does not equals the number of times, w_2, that member M_2 has accessed to the same AP. If $w \neq w_1$ is satisfied, the AP can affirm that the

member is not M_1.
2) *If one can know the discrete logarithm of two tag bases, one can identify members* using the equation $\beta = (\check{\tau}_1{}^z/\check{\tau}_2)^{1/(l_1 z - l_2)}$. Here, β is a part of the public key of a member, $\check{\tau}_1$ and $\check{\tau}_2$ are second coordinate of tags computed by the member using tag bases t_1 and t_2 which satisfy $t_2 = t_1{}^z$. Similerly, *If $\tau_1{}^{x_1} = \tau_1{}^{x_2}$ are satisfied, a user who know x_1 can perform as a user who know x_2*
3) *An AP can add false data to the log.*
4) In joining, *a secret key x of a member must be selected randomly*, since "one more unforgeability" of member key pair is assured only if the condition is satisfied. (See Lemma 2).
5) In joining, *a user must add b^x to the identification list before he know* (A, e). If a user can know (A, e) before he adds b^x, he can stop the joining procedure, and get a member key pair $(x, (A, e))$ such that b^x is not written in the identification list. Therefore, he can be anonymously authenticated any number of times since b^x is needed to tracing procedure.
6) As a similar reason plain signature schemes needs CA, the proposed scheme needs some mechanizm to assure the correctness of the correspondence between each entity and his public key.
7)*If G is unknown order group, the number of exponentiations of public tracing is linear to the size of members.* In this case, since one cannot compute $(\check{\tau}/\check{\tau}')^{1/(\ell-\ell')}$, one must cumbersomely compute $\beta^{(\ell-\ell')}$ for each element β of the identification list and then check whether $\beta^{(\ell-\ell')} = \check{\tau}/\check{\tau}'$ is satisfied.

To avoid attacks of 1), ..., 5), we construct the proposed scheme which satisifes the following: 1) the validity proof conceals w, 2) tag bases are hash values of some data, 3) an authentication log contains validity proofs which members have computed, 4) x is randomized by the GM, and 5) b^x is added to the identification list before the GM computes $A = (a^x a_0)^{1/e} \bmod n$.

To avoid attacks of 6), we assume the GM's public key is distributed by some trust entity. Additionally, we assume some assumption about the identification list, to assuer the correspondence between each member and his public key. See 4.2 for more detailed discussion.

To avoid inefficient tracing descibed in 7), we set G as a known order group, especially $G \neq \mathrm{QR}(n)$.

3.5 Description of Proposed Scheme

PARAMETERS

The security parameters of our scheme are ν, ε, μ, and κ. Let λ and γ be parameters which are determined by the security parameters. (See Section 5 for a detailed description). We set Λ, Γ as sets of integers that were in $(0, 2^\lambda)$ and $(2^\gamma, 2^\gamma + 2^\lambda)$ respectively. Let $\{G_\kappa\}_{\kappa \in \mathbb{N}}$ be a set of cyclic groups with a known order. Let G be G_κ.

The parameters ν, ε, μ, and κ respectively control the difficulty of solving flexible RSA problem (the problem is also called strong RSA problem) on $\mathbb{Z}_n$, the tightness of the statistical zero-knowledge property, the soundness of the scheme, and the difficulty of solving DDH problem on G. We set, for example, $\nu = 1024$, $\varepsilon = \mu = \kappa = 160$, and set G as an elliptic curve group.

SETUP

1. The GM randomly chooses 2ν-bit rigid integer n. Then, it randomly chooses μ-bit string $R_{\mathtt{GM}}$, and computes $((a', a'_0), b) = \mathcal{H}_{\mathbb{Z}_n{}^2 \times G}(R_{\mathtt{GM}})$ and $(a, a_0) = (a'^2, a_0'^2) \bmod n \in \mathrm{QR}(n)^2$. The group secret key is (p_1, p_2) and the group public key is $(n, R_{\mathtt{GM}}, a, a_0, b)$.

JOINING

1. User U_i selects $x' \in_U \Lambda$, and sends its commitment C to the GM with a validity proof.
2. The GM verifies the proof, and sends $x'' \in_U \Lambda$ to U_i.
3. User U_i confirms that $x'' \in \Lambda$ is satisfied, computes $x = ((x' + x'') \bmod 2^\lambda)$ and $(\alpha, \beta) = (a^x \bmod n, b^x)$, and then adds new data (i, β) to the *identification list* `LIST`. Then, U_i sends (α, β) to the GM with a validity proof.
4. The GM verifies (i, β) is an element of the identification list, and the proof is valid. Then, the GM generates a prime $e \in_U \Gamma$, computes $A = (\alpha a_0)^{1/e} \bmod n$, and sends (A, e) to user U_i.
5. User U_i confirms that equation $a^x a_0 = A^e \bmod n$ is satisfied, e is a prime, and e is an element of Γ. The new member U_i's secret key is x, and his public key is (α, A, e, β).

BOUND ANNOUNCEMENT

1. AP V publishes $(\mathtt{ID}_{\mathrm{V}}, k_{\mathrm{V}})$. Here, $\mathtt{ID}_{\mathrm{V}}$ is his ID.

Let $(t_1, \check{t}_1) = \mathcal{H}_{G^2}(\mathtt{ID}_{\mathrm{V}}, k_{\mathrm{V}}, 1), \ldots, (t_{k_{\mathrm{V}}}, \check{t}_{k_{\mathrm{V}}}) = \mathcal{H}_{G^2}(\mathtt{ID}_{\mathrm{V}}, k_{\mathrm{V}}, k_{\mathrm{V}})$. We call $(t_w, \check{t}_w)$ the w-th *tag base* of the AP.

AUTHENTICATION

1. Member M increases counter $C_{\mathtt{ID}_{\mathrm{V}}, k_{\mathrm{V}}}$. If value w of counter $C_{\mathtt{ID}_{\mathrm{V}}, k_{\mathrm{V}}}$ is greater than k_{V}, then M sends $\perp$ to V and stops.
2. AP V sends random integer $\ell \in_U [0, 2^{\mu+\varepsilon}] \cap \mathbb{N}$ to M.
3. Member M computes tag $(\tau, \check{\tau}) = (t_w^x, (b^\ell \check{t}_w)^x)$, using M's secret key x and the w-th tag base $(t_w, \check{t}_w)$, computes proof $(\tau, \check{\tau})$ is correctly computed, and sends $(\tau, \check{\tau})$ and the validity proof to V.
4. If the proof is valid and if τ is different from all search tags in his authentication log, V adds tuple $(\tau, \check{\tau}, \ell)$ and the proof to the authentication log `LOG` of V, and outputs `accept`.

PUBLIC TRACING

1. From LOG, one finds two data $(\tau, \check{\tau}, \ell, \mathtt{PROOF})$ and $(\tau', \check{\tau}', \ell', \mathtt{PROOF}')$ that satisfy $\tau = \tau'$ and $\ell \neq \ell'$, and that PROOF and PROOF′ are valid. If one cannot find such data, then one outputs NO-ONE.
2. One computes $\beta' = (\check{\tau}/\check{\tau}')^{1/(\ell-\ell')} = ((b^{\ell}\check{t})^x/(b^{\ell'}\check{t})^x)^{1/(\ell-\ell')} = b^x$, and searches pair (i, β) that satisfies $\beta = \beta'$ from the identification list. Then, one outputs a member's ID i. If there is no such (i, β), then one affirms that the GM has deleted some data from the identification list, and outputs GM.

3.6 Details

- setup.
  - The GM must additionally publish 1) $(g, h) \in_U \mathrm{QR}(n)^2$, which shall be used by users to compute commitment C in joining, 2) a zero-knowledge proof that n is a rigid integer, and 3) a zero-knowledge proof that (g, h) is an element of $\mathrm{QR}(n)$. The GM provides the proof 2) using the technique of [10], and provide the proof 3) by proving knowledge of $(\tilde{g}', \tilde{h}') \in {\mathbb{Z}_n}^2$, which satisfies $(\tilde{g}'^2, \tilde{h}'^2) = (g, h) \bmod n$.
- joining.
  - At step 1, the user must compute $((a', a_0'), b) = \mathcal{H}_{{\mathbb{Z}_n}^2 \times G}(R_{\mathtt{GM}})$, and verify equation $(a, a_0) = (a'^2, {a_0'}^2) \bmod n$, and the proofs.
  - Commitment C is $g^{x'}h^{s'} \bmod n$. Here s' is a $(2\nu + \varepsilon)$-bit random natural number.
  - The formal description of validity proofs of step 1 and 3 are, respectively, $\mathtt{PROOF}_1 = \mathtt{PROOF}((\tilde{x}', \tilde{s}')$ s.t. $\tilde{x}' \in \Lambda \wedge C = g^{\tilde{x}'}h^{\tilde{s}'} \bmod n)$ and $\mathtt{PROOF}_2 = \mathtt{PROOF}((\tilde{x}, \tilde{\theta}, \tilde{s}')$, which satisfies the (a), ..., (d) below.), where (a) $\tilde{x} \in \Lambda$, (b) $a^{\tilde{x}} = \alpha \bmod n$, (c) $Cg^{x''} = g^{\tilde{x}}(g^{2^{\lambda}})^{\tilde{\theta}}h^{\tilde{s}'} \bmod n$, and (d) $b^{\tilde{x}} = \beta$. These proofs must be statistically zero knowledge on security parameter ε. We have omitted a detailed description of proofs. See [8] for the proof that committed number lies in the interval.
- authentication.
  - At step 4, if the proof is invalid, V outputs `reject` and stops. If τ is already written in the identification list, V adds tuple $(\tau, \check{\tau}, \ell)$ and the proof to the LOG of the AP, outputs (`detect`, LOG) and stops.
  - The proof of step 3 is rather more complex. Its details are described in the full version of this paper.

3.7 Efficiency

The proposed scheme satisfies the followings:

- (**Compactness**) The GM is able to add new members to the group without modifying any keys which was previously generated. In particular, the size of the member's key pair does not depend on the group size.
- Once a user has been registered by the GM, the user does not need to access the GM.

– Each AP is able to solely determine the bound of himself.
– The computational cost of authentication is $\mathcal{O}(k_{\mathrm{V}})$. However, if G is taken as an elliptic curve group, the factor which depend on k_{V} is small, since the exponentiation on G is faster than that on $\mathbb{Z}_n$.
– The number of exponentiations of public tracing is independent of the size of an authentication log and the identification list.

3.8 Variants of Proposed Scheme

1) Although the proposed scheme merely restricts the number of authentications, one can construct, using the "and/or"-proof technique, a scheme such as "a trace procedure identifies a user if and only if the user is authenticated either 1) k_1 times from AP V_1 or 2) k_2 times from AP V_2 and k_3 times from AP V_3".
2) By changing a data in LIST from (i, β) to $(\mathcal{H}(\beta), \mathcal{E}_\beta(i))$, one can construct a k-TAA scheme in which no one, except a member himself and the GM, can detect who is a member of the group. Here, $\mathcal{H}$ is a hash function and $\mathcal{E}$ is a symmetric encryption scheme. To trace dishonest user, one computes β as in the proposed scheme, and then computes $\mathcal{H}(\beta)$, searches (h, e) from LIST that satisfies $h = \mathcal{H}(\beta)$, and decrypts e.

4 Formal Security Requirements

4.1 Notations

We describe the five procedures for a k-TAA scheme as SETUP, JOIN $= (\mathcal{U}_{\texttt{JOIN-GM}}, \mathcal{U}_{\texttt{JOIN-U}})$, BOUND-ANNOUNCEMENT, (abbrev. BD-ANN), AUTH $= (\mathcal{U}_{\texttt{AUTH-U}}, \mathcal{U}_{\texttt{AUTH-AP}})$, and TRACE. The procedures $\mathcal{U}_{\texttt{JOIN-GM}}$ and $\mathcal{U}_{\texttt{JOIN-U}}$ (resp. $\mathcal{U}_{\texttt{AUTH-AP}}$ and $\mathcal{U}_{\texttt{AUTH-U}}$) are what the GM and user (resp. AP and user) follow in joining (resp. authentication). Let $(\mathtt{gpk}, \mathtt{gsk})$ and $(\mathtt{mpk}, \mathtt{msk})$ denote the public key/secret key pair of group manager and member respectively.

4.2 List Oracle Model

We must assume the existence of an infrastructure which enables to assure the correct correspondence between each member and his public key to formalize the security requirements. If we do not assume such thing, no scheme satisfies the exculpability properties for users as in a group signature case[7]. One of a such infrastructure is a PKI, but a formalization on the PKI model is rather complicated, since it must include description of the signing oracle, what an adversary can do in a PKI key setup, etc. To simplify, we introduce new model *list oracle model*. In the model, it is assumed the existence of a *list oracle* $\mathcal{O}_{\texttt{LIST}}$, which manages the identification list[2] LIST. The oracle $\mathcal{O}_{\texttt{LIST}}$ allows anyone to view any data of LIST. However, it allows entities to write data $(i, \mathtt{mpk}_i)$ to LIST

[2] Although the LIST of the proposed scheme stores a parts of public keys, β, we deal with the case LIST stores the whole public key, to simplify.

only if the entity is user i or i's colluder and to delete data of $\mathtt{LIST}$ only if the entity is the GM or GM's colluder. We need to stress that even the GM cannot write data $(i, \mathtt{mpk}_i)$ without colluding with the user i, and even user i cannot delete data $(i, \mathtt{mpk}_i)$ without colluding with the GM. A more formal definition is described in Figure 1, where X is a set of entities which collude with an entity who accesses to $\mathcal{O}_{\mathtt{LIST}}$

Note that *a scheme on the list oracle model can be easily transformed into a scheme on the PKI model*, by changing $(i, \mathtt{mpk}_i)$ to $(\mathtt{mpk}_i, \sigma_i(\mathtt{mpk}_i))$ and $\mathtt{LIST}$ to $(\mathtt{LIST}, \sigma_{\mathtt{GM}}(\mathtt{LIST}))$. Here, $\sigma_i(\cdot)$ is a signature of an entity i. The authority of the GM in the list oracle model to delete data from $\mathtt{LIST}$ corresponds to the authority of the GM in the PKI model to publish $(\mathtt{LIST}', \sigma_{\mathtt{GM}}(\mathtt{LIST}'))$ in spite of $(\mathtt{LIST}, \sigma_{\mathtt{GM}}(\mathtt{LIST}))$. Here $\mathtt{LIST}' = \mathtt{LIST} \setminus \{(\mathtt{mpk}_i, \sigma_i(\mathtt{mpk}_i))\}$.

4.3 Experiments

An adversary is allowed the following in experiments on security properties:

- If an adversary colludes with the GM, the adversary can maliciously execute $\mathtt{SETUP}$ and $\mathcal{U}_{\mathtt{JOIN\text{-}GM}}(\mathtt{gpk}, \mathtt{gsk})$.
- If an adversary colludes with a user i, the adversary can maliciously execute $\mathcal{U}_{\mathtt{JOIN\text{-}U}}(\mathtt{gpk}, i)$ and $\mathcal{U}_{\mathtt{AUTH\text{-}U}}(\mathtt{gpk}, \mathtt{msk})$ where $\mathtt{msk}$ is a secret key of i.
- If an adversary colludes with an AP, the adversary can choose the public information $(\mathtt{ID}, k)$ of the AP and maliciously execute $\mathcal{U}_{\mathtt{AUTH\text{-}AP}}(\mathtt{gpk}, (\mathtt{ID}, k))$. Moreover, the adversary can use different AP information $(\mathtt{ID}, k)$ for each authentication.
- An adversary is only allowed to execute many joining and authentication procedures sequentially.

Total Anonymity An adversary is allowed to collude with the GM, all APs, and all users except target users i_1 and i_2. It is also allowed to authenticate the oracle $\mathcal{O}_{\mathtt{QUERY}}(b, \mathtt{gpk}, (i_1, i_2), (\mathtt{ID}, k), (d, \cdot))$ once only for $d = 0, 1$. If it sends (d, M) to $\mathcal{O}_{\mathtt{QUERY}}$, oracle $\mathcal{O}_{\mathtt{QUERY}}$ regards M as data sent by a member and executes $\mathcal{U}_{\mathtt{AUTH\text{-}U}}$ using the key pair of user $i_{b \oplus d+1}$ and the APs public information $(\mathtt{ID}, k)$. Recall that k-TAA schemes provide anonymity only if a member has been authenticated less than the allowed number of times. Therefore, the adversary must authenticate user i_1 or i_2 using $(\mathtt{ID}, k)$ within k times. If the adversary keeps to the rule and outputs b, the adversary wins. See Figure 1 for the formal definition of $\mathcal{O}_{\mathtt{QUERY}}$. Here, $S_{\mathtt{QUERY}}$ is a set, using which $\mathcal{O}_{\mathtt{QUERY}}$ memorize the session IDs.

Contrary to [7, 22], the secret key of the target users is not input to an adversary. If the secret keys is input to an adversary, the adversary is able to determine b as follows: it colludes with AP publishing $(\mathtt{ID}, k)$, authenticated k times from the AP using i_1's secret key, and obtains the log $\mathtt{LOG}$ for the authentications. Then, it communicates with $\mathcal{O}_{\mathtt{QUERY}}(b, \mathtt{gpk}, (i_1, i_2), (\mathtt{ID}, k), (0, \cdot))$ and obtains the log L of the authentications. Secret b equals to 0 if and only if $\mathtt{TRACE}^{\mathcal{O}_{\mathtt{LIST}}(\emptyset, \cdot)}(\mathtt{gpk}, \mathtt{LOG} \cup \{L\}) = i_1$ is satisfied.

$\mathbf{Exp}^{\text{anon-}((i_1,i_2),(\mathtt{ID},k),b)}_{\mathcal{A},\mathcal{H}}(\omega)$

$(\mathtt{gpk}, St) \leftarrow \mathcal{A}(1^{\omega})$

$b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathtt{LIST}}(\{i_1,i_2\}^c,\cdot),\mathcal{O}_{\mathtt{JOIN\text{-}U}}(\mathtt{gpk},\cdot),\mathcal{O}_{\mathtt{AUTH\text{-}U}}(\mathtt{gpk},\cdot),\mathcal{O}_{\mathtt{QUERY}}(b,\mathtt{gpk},(i_1,i_2),(\mathtt{ID},k),(\cdot,\cdot))}(St)$

If ($\mathcal{O}_{\mathtt{QUERY}}$ has output $\mathtt{OVER}$) Return $\perp$.

Return b'.

***$\mathcal{O}_{\mathtt{LIST}}(X, M)$ ***

Parse M as $(\mathtt{command}, i, \mathtt{mpk})$.

If($\mathtt{command} = \mathtt{view}$ and $\mathtt{mpk} =$ "-")

  If($^{\exists}\mathtt{mpk}'$ s.t. $(i, \mathtt{mpk}') \in \mathtt{LIST}$)

    Return $\mathtt{mpk}'$.

If($\mathtt{command} = \mathtt{add}$)

  If($i \in X$ and $^{\nexists}\mathtt{mpk}'$ s.t. $(i, \mathtt{mpk}) \in \mathtt{LIST}$)

    $\mathtt{LIST} \leftarrow \mathtt{LIST} \cup \{(i, \mathtt{mpk})\}$.

Else if($\mathtt{command} = \mathtt{delete}$)

  If ($\mathtt{GM} \in X$) $\mathtt{LIST} \leftarrow \mathtt{LIST} \setminus \{(i, \mathtt{mpk})\}$.

Return $\perp$.

$\mathcal{O}_{\mathtt{QUERY}}(b, \mathtt{gpk}, (i_1, i_2), (\mathtt{ID}, k)(d, M))$

If($d \notin \{0, 1\}$) Return $\perp$.

If($^{\nexists}\mathtt{sid}$ s.t. $(d, \mathtt{sid}) \in S_{\mathtt{QUERY}}$)

  Choose new session ID $\mathtt{sid}$

    which has been ever used.

  $S_{\mathtt{QUERY}} \leftarrow S_{\mathtt{QUERY}} \cup \{(d, \mathtt{sid})\}$.

Return

  $\mathcal{O}_{\mathtt{AUTH\text{-}U}}(\mathtt{gpk}, (\mathtt{sid}, i_{1+(b \oplus d)}, (\mathtt{ID}, k)||M))$.

$\mathbf{Exp}^{\text{decis}}_{\mathcal{A},\mathcal{H}}(\omega)$

$(\mathtt{gpk}, \mathtt{gsk}) \leftarrow \mathtt{SETUP}(1^{\omega})$

$\mathcal{A}^{\mathcal{O}_{\mathtt{LIST}}(\{\mathtt{GM}\}^c,\cdot),\mathcal{O}_{\mathtt{JOIN\text{-}GM}}(\mathtt{gpk},\mathtt{gsk},\cdot),\mathcal{O}_{\mathtt{AUTH\text{-}AP}}(\mathtt{gpk},\cdot,\cdot)}(1^{\omega})$.

If ($^{\exists}(\mathtt{ID}, k) \in S_{\mathtt{AUTH\text{-}AP}}$ s.t. $\#\mathtt{LOG}_{\mathtt{ID},k} > k \cdot \#\mathtt{LIST}$)

  Return $\mathtt{TRACE}^{\mathcal{O}_{\mathtt{LIST}}(\emptyset,\cdot)}(\mathtt{gpk}, \mathtt{LOG}_{\mathtt{ID},k})$.

Return $\perp$.

$\mathbf{Exp}^{\text{excul-}i_1}_{\mathcal{A},\mathcal{H}}(\omega)$

$(\mathtt{gpk}, St) \leftarrow \mathcal{A}(1^{\omega})$.

$\mathtt{LOG} \leftarrow \mathcal{A}^{\mathcal{O}_{\mathtt{LIST}}(\{i_1\}^c,\cdot),\mathcal{O}_{\mathtt{JOIN\text{-}U}}(\mathtt{gpk},\cdot),\mathcal{O}_{\mathtt{AUTH\text{-}U}}(\mathtt{gpk},\cdot)}(St)$.

Return $\mathtt{TRACE}^{\mathcal{O}_{\mathtt{LIST}}(\emptyset,\cdot)}(\mathtt{gpk}, \mathtt{LOG})$.

$\mathbf{Exp}^{\text{excul-GM}}_{\mathcal{A},\mathcal{H}}(\omega)$

$(\mathtt{gpk}, \mathtt{gsk}) \leftarrow \mathtt{SETUP}(1^{\omega})$

$\mathtt{LOG} \leftarrow \mathcal{A}^{\mathcal{O}_{\mathtt{LIST}}(\{\mathtt{GM}\}^c,\cdot),\mathcal{O}_{\mathtt{JOIN\text{-}GM}}(\mathtt{gpk},\mathtt{gsk},\cdot)}(\omega)$.

Return $\mathtt{TRACE}^{\mathcal{O}_{\mathtt{LIST}}(\emptyset,\cdot)}(\mathtt{gpk}, \mathtt{LOG})$.

Comments:

1. To simplify, we abbreviate the hash oracle $\mathcal{O}_{\mathcal{H}}$.

2. $\mathcal{O}_{\mathtt{AUTH\text{-}U}}(\mathtt{gpk}, (\cdot, i, \cdot))$ outputs $\mathtt{OVER}$ if $\mathcal{A}$ authenticate user i more than allowed number of times.

Fig. 1. The oracles and the experiments.

Detectability An adversary is allowed to collude with all of group members. If the adversary succeeds in being accepted by some AP in more than kn authentications, the adversary wins. Here, k is the number of times the AP allows access for each user, and n is the number of users who collude with the adversary.

Exculpability (for users, and for GM) An adversary is allowed to collude with all entities except the target entity. If the adversary succeeds in computing the log with which the public tracing procedure outputs the ID of the target entity, the adversary wins.

Figure 1 denotes the experiments, formally. $\mathcal{O}_{\mathtt{JOIN\text{-}GM}}$, $\mathcal{O}_{\mathtt{JOIN\text{-}U}}$, $\mathcal{O}_{\mathtt{AUTH\text{-}U}}$, and $\mathcal{O}_{\mathtt{AUTH\text{-}AP}}$ are the oracles that manage and execute multiple sessions of $\mathcal{U}_{\mathtt{JOIN\text{-}GM}}$, $\mathcal{U}_{\mathtt{JOIN\text{-}U}}$, $\mathcal{U}_{\mathtt{AUTH\text{-}U}}$, and $\mathcal{U}_{\mathtt{AUTH\text{-}AP}}$ respectively, and ω is the security parameter. The set $S_{\mathtt{AUTH\text{-}AP}}$ contains all AP's information that was used by $\mathcal{O}_{\mathtt{AUTH\text{-}AP}}$, and $\mathtt{LOG}_{\mathtt{ID},k}$ is the log of authentications engaged by $\mathcal{O}_{\mathtt{AUTH\text{-}AP}}$ using AP's information $(\mathtt{ID}, k)$. See the full version of this paper for a formal definition of the oracles.

4.4 Definition

Definition 1. *Let ω be a security parameter, $\mathcal{A}$ be an adversary, b be an element of $\{0,1\}$, i_1 and i_2 be natural numbers, and $(\mathtt{ID}, k)$ is a some AP's public information.*

If $\mathbf{Adv}_{\mathcal{A}}^{anon\text{-}((i_1,i_2),(\mathtt{ID},k))}(\omega) = |\Pr(\mathbf{Exp}_{\mathcal{A},\mathcal{H}}^{anon\text{-}(0,(i_1,i_2),(\mathtt{ID},k))}(\omega) = 1)$ $- \Pr(\mathbf{Exp}_{\mathcal{A},\mathcal{H}}^{anon\text{-}(1,(i_1,i_2),(\mathtt{ID},k))}(\omega) = 1)|$ *is negligible for security parameter ω for all $(\mathcal{A}, i_1, i_2, (\mathtt{ID}, k))$, we say a k-TAA scheme satisfies* total anonymity*.*

If $\mathbf{Adv}_{\mathcal{A}}^{decis}(\omega) = \Pr(\mathbf{Exp}_{\mathcal{A},\mathcal{H}}^{decis}(\omega) = \mathtt{NO\text{-}ONE})$ *is negligible for security parameter ω for all $\mathcal{A}$, we say a k-TAA scheme satisfies* detectability*.*

If $\mathbf{Adv}_{\mathcal{A}}^{excul\text{-}i_1}(\omega) = \Pr(\mathbf{Exp}_{\mathcal{A},\mathcal{H}}^{excul-i_1}(\omega) = i_1)$ *is negligible for security parameter ω for all $(\mathcal{A}, i_1)$, we say a k-TAA scheme satisfies* exculpability for users.

If $\mathbf{Adv}_{\mathcal{A}}^{excul\text{-}\mathtt{GM}}(\omega) = \Pr(\mathbf{Exp}_{\mathcal{A},\mathcal{H}}^{excul-\mathtt{GM}}(\omega) = \mathtt{GM}))$ *is negligible for security parameter ω for all $\mathcal{A}$, we say a k-TAA scheme satisfies* exculpability for users*.*

5 Security of Proposed Scheme

To prove the security of the proposed scheme, we use two key lemmata. First, since each member generates element a^x of $\mathrm{QR}(n)$ and element b^x of G using the same x, we must be particularly concerned about secrecy. We will prove the difficulty of a variant in the DDH problem, where two components of a DH-tuple are elements of $\mathrm{QR}(n)$ and the other two components are elements of G:

Lemma 1. (Separation Lemma) *Let a be an element of $\mathrm{QR}(n)$, and b be an element of G. Then, the following two distributions are statistically indistinguishable: 1) the distribution of $(a^x \bmod n, b^x) \in \mathrm{QR}(n) \times G$, where x is randomly chosen from Λ, and 2) the distribution of $(\alpha, \beta) \in \mathrm{QR}(n) \times G$, where α and β are randomly chosen from $\mathrm{QR}(n)$ and G respectively.*

Since $|\Lambda|$ is ε times greater than $|\mathrm{QR}(n) \times G|$, the variation distance between the two distributions is less than $1/2^{\varepsilon}$, and therefore, Lemma 1 holds. Note that, if we injudiciously choose a narrow Λ, the security of the proposed scheme will rely on a non-standard assumption that those two distributions will still be computationally indistinguishable.

Detectability and GM's exculpability of the proposed scheme depends on "one more unforgeability" of a $((A, e), x)$:

Lemma 2. *If a) a member's secret key is randomly generated in each joining procedure, and if b) for all $x \in \Lambda$ and $e \in \Gamma$, $x < e$ is satisfied, then no adversary can generate a (x, A, e) which satisfies $a^x a_0 = A^e$, $x \in \Lambda$, and $e \in \Gamma$, and which has not been made in the joinings.*

The proof for Lemma 2 is almost same as the proof for Theorem 1 of [2].

The proposed scheme satisfies the conditions for Lemma 2. Conditions a) and b), respectively, follow the method of choosing x in the joining procedure, and the choice of (λ, γ).

Using these lemmata, we can prove the security of the proposed scheme. See the full version of this paper for the detailed proof.

Theorem 1. *Let* λ *be* $2\nu+\kappa+\varepsilon$*, and* γ *be* $\lambda+\mu+\varepsilon+8$*. Then, the proposed scheme on list oracle model satisfies the security requirements of Definition 1 under the strong RSA assumption, the DDH assumption on* $\{G_\kappa\}$*, and the random oracle assumption.*

References

1. How to Date Blind Signatures, M. Abe, and E. Fujisaki, In *ASIACRYPT 1996*, *LNCS* 1163, pp. 244-251, Springer-Verlag, 1996.
2. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A Practical and Provably Secure Coalition-Resistant Group Signature Scheme. In *CRYPTO 2000*, *LNCS* 1880, pp. 255-270, Springer-Verlag, 2000.
3. Giuseppe Ateniese and Breno de Medeiros. Efficient Group Signatures without Trapdoors. In *ASIACRYPT 2003*, *LNCS* 2094, pp. 246-268, Springer-Verlag, 2003.
4. G. Ateniese and G. Tsudik. Some Open Issues and New Directions in Group Signatures. In *Financial Cryptography '99*, *LNCS* 1648,pp. 196-211, Springer-Verlag, 1999.
5. Direct Anonymous Attestation. Ernie Brickell, Jan Camenisch, and Liqun Chen. ZISC Information Security Colloquium SS 2004, June 2004 http://www.hpl.hp.com/techreports/2004/HPL-2004-93.pdf
6. Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi, Foundations of Group Signatures: Formal Definitions, Simplified Requirements, and a Construction Based on General Assumptions In *EUROCRYPT 2003*, *LNCS* 2656, pp. 614-629, Springer-Verlag, 2003.
7. Mihir Bellare, Haixia Shi, and Chong Zhang, Foundations of Group Signatures: The Case of Dynamic Groups. http://eprint.iacr.org/2004/077.ps
8. Fabrice Boudot. Efficient Proofs that a Committed Number Lies in an Interval. In *EUROCRYPT 2000*, *LNCS* 1807, pp. 255-270, Springer-Verlag, 2000.
9. Stefan Brands. An Efficient Off-line Electronic Cash System Based On The Representation Problem. Technical Report CS-R9323, Centrum voor Wiskunde en Informatica,
10. Jan Camenisch and Markus Michels. Proving in Zero-Knowledge that a Number is the Product of Two Safe Primes. In *EUROCRYPT'99*, *LNCS* 1592, pp. 107-122, Springer-Verlag, 1999.
11. Sébastien Canard and Jacques Traoré List Signature Schemes and Application to Electronic Voting. In *International Workshop on Coding and Cryptography* 2003. pp.24-28, March 2003.
12. A. Chan, Y. Frankel, and Y.Tsiounis, Easy Come - Easy Go Divisible Cash. *EUROCRYPT '98*, *LNCS* 1403, pp. 614-629, Springer-Verlag, 1998.
13. D. Chaum. Blind signature system, In *CRYPTO'83*, pp. 153-153, Plenum Press, 1984
14. D. Chaum. *Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms*, Communications of the ACM, vol. 24, No. 2, pp. 84-88, (1981).
15. D. Chaum and E. van Heijst. Group signatures. In *EUROCRYPT '91*, vol. *LNCS* 547, pp. 257-265, Springer-Verlag, 1991.

16. D. Chaum, T. Pedersen, Transferred Cash Grows in Size, In *EUROCRYPT'92*, *LNCS* 658, pp. 390-407, Springer-Verlag, 1993.
17. R.Cramer, I.Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO'94*, *LNCS* 2139, pp. 174-187, Springer-Verlag, 1994.
18. Yevgeniy Dodis, Aggelos Kiayias, Antonio Nicolosi, Victor Shoup. Anonymous Identification in Ad Hoc Groups. In *EUROCRYPT 2004*, *LNCS* 3027, pp. 609-626, Springer-Verlag, 2004.
19. I. Damgård and M. Jurik. A Generalization, a Simplification and Some Applications of Paillier's Probabilistic Public-key system. In *Proceedings. of Public Key Cryptography 2001. LNCS* 1992, pp. 119-136, Springer-Verlag, 2001.
20. A. Fiat and A. Shamir. How to prove yourself: practical solution to identification and signature problems. In *CRYPTO'86*, *LNCS* 263, pp. 186-194, Springer-Verlag, 1987.
21. Jun Furukawa, and Kazue Sako. An Efficient Scheme for Proving a Shuffle. In *CRYPTO 2001*, *LNCS* 2139, pp. 368-387, Springer-Verlag, 2001.
22. Aggelos Kiayias, and Moti Yung. Group Signatures: Provable Secure, Efficient Constructions and Anonymity from Trapdoor Holders. http://eprint.iacr.org/2004/076.ps
23. Aggelos Kiayias, Yiannis Tsiounis, and Moti Yung. Traceable Signatures. In *EUROCRYPT 2004*, *LNCS* 3027, pp. 571-589, Springer-Verlag, 2004.
24. Joe Kilian, and Erez Petrank. Identity Escrow. In *CRYPTO'98*, *LNCS* 1462, pp. 169-185, Springer-Verlag, 1998.
25. Toru Nakanishi, Nobuaki Haruna, and Yuji Sugiyama. Unlinkable Electronic Coupon Protocol with Anonymity Control, In *ISW'99*, *LNCS* 1729, pp. 37-46, Springer-Verlag, 1999.
26. Toru Nakanishi, Nobuaki Haruna, and Yuji Sugiyama. Electronic Coupon Ticket Protocol with Unlinkable Transcripts of Payments. In *Proceedings of the 1999 Symposium on Cryptography and Information Security*, pp. 359-363, 1999. (Japanese).
27. C.A. Neff, A Verifiable Secret Shuffle and its Application to E-Voting, ACMCCS 01 pp. 116-125 2001.
28. Tatsuaki Okamoto and Kazuo Ohta. One-Time Zero-Knowledge Authentications and Their Applications to Untraceable Electronic Cash. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol E81-A, No. 1, pp. 2-10, 1998.
29. M. Ookubo, F. Miura, M. Abe A. Fujioka, and T. Okamoto. An improvement of a practical secret voting scheme. In *ISW'99*, *LNCS* 1729, pp. 37-46, Springer-Verlag, 1999.
30. Chris Pavlovski, Colin Boyd, and Ernest Foo. Detachable Electronic Coins. In *Information and Communication Security, Second International Conference, ICICS'99*, *LNCS* 1726, pp. 54-70, Springer-Verlag, 1999.
31. K. Sako. Restricted Anonymous Participation. In *Proceedings of the 2000 Symposium on Cryptography and Information Security*, B12, January 2000. (Japanese).
32. K. Sako and J. Kilian. Secure Voting using Partially Compatible Homomorphisms. In *CRYPTO '94*, *LNCS* 839, pp. 411–424, Springer-Verlag, 1994.
33. Isamu Teranishi and J. Furukawa. Tag Signature. (Preliminary version of this paper). In *Proceedings of the 2003 Symposium on Cryptography and Information Security*, 6C-2, January 2003.