Unique Group Signatures

Matthew Franklin and Haibin Zhang

Dept. of Computer Science, University of California, Davis, California 95616, USA
{franklin,hbzhang}@cs.ucdavis.edu

Abstract. We initiate the study of *unique group signature* such that signatures of the same message by the same user will always have a large common component (i.e., unique identifier). It enables an efficient detection algorithm, revealing the identities of illegal users, which is fundamentally different from previous primitives. We present a number of unique group signature schemes (without random oracles) under a variety of security models that extend the standard security models of ordinary group signatures. Our work is a beneficial step towards mitigating the well-known group signature paradox, and it also has many other interesting applications and efficiency implications.

Keywords: Anonymity, anonymous authentication, detection algorithm, group signature, unique signature, verifiable random function.

1 Introduction

Group signatures, introduced by Chaum and van Heyst [11], are very useful tools in applications where the signer's privacy should be protected and in case of abuse some authorities can identify the misbehaving user. However, a well-known group signature "paradox" is that it is difficult for the group manager to identify a "misbehaving" user since all of signatures are anonymous. The group manager obviously cannot afford to open *all* of group signatures signed, for this is inefficient, and more importantly, it would compromise the privacy of every signer. Typically, the group manager identifies possible misbehaving users by observing whether some surprising documents are signed, or a huge amount of documents are signed within a short period, or some other "rules" are broken. These empirical test methods only provide the group manager with rough estimation about what signatures are suspicious. Trying to open and reveal the identities of suspicious signatures has a risk of jeopardizing legal users, while the illegal users may still be well-hidden.

Let us consider the motivating example of group signature due to Chaum and van Heyst [11]: "A company has several computers, each connected to the local network. Each department of that company has its own printer (also connected to the network) and only persons of that department are allowed to use their department's printer. Before printing, therefore, the printer must be convinced that the user is working in that department. At the same time, the company

S. Foresti, M. Yung, and F. Martinelli (Eds.): ESORICS 2012, LNCS 7459, pp. 643–660, 2012.

wants privacy: the user's name may not be revealed. If, however, someone discovers at the end of the day that a printer has been used too often, the director must be able to discover who misused that printer, to send him a bill."

The above opening policy, in practice, is problematic: it is not fair to reveal all identities of the persons who use the printer that is "used too much", since the identities of legal users might as well be revealed. It does not even make sense to say what is "used too much", as a dedicated adversary might use the same printer every day such that the times of uses are always slightly below the daily threshold, while the others would not dare to use the printer.

In this case, the rule that this company would like to enforce is to limit the number of times within some period that group members can use the service. If anyone who accessed the service beyond the allowed quota then its identity should be revealed by the group authority. At the same time, it is equally desirable for this company to detect *other* malicious printing any time—for instance, one printing process that uses up all the paper—which is prohibitive. In other words, once a user signs a message more than a predetermined value then it shall be almost always (efficiently) detected, but the group manager can always open signatures any time in case of other misbehavior.

We define *unique group signature* as a first step towards mitigating this paradox. We may say that a group signature scheme is "unique" if it is computationally infeasible for a signer to produce two different group signatures of the same message, such that both will pass the verification procedure (by analogy with the well-studied notion of uniqueness for ordinary signature schemes). We adopt a less stringent but more general definition such that if a signer produces two different group signatures of the same message, then both signatures will always have a large common component (hereinafter *unique identifier*) which is otherwise highly unlikely to occur. Ideally, if one user indeed signs two different signatures on one message then there should be an (efficient) *detection algorithm* that can reveal the identity of this user. With carefully defined other security notions, this primitive (still called unique group signature) serves as a perfect solution of dealing with the above problem.

A closely related question was first asked by Damgård, Dupont, and Pedersen [12] in their paper on *unclonable group identification scheme*. An unclonable group identification scheme enables a user to authenticate to a server with complete anonymity provided that no other users try to use the first user's secret key to authenticate to the server within the same time period ("cloning attack"), while allowing the user's identity to be traced if they do misbehave in this way. They point out the inadequacy of existing group signature schemes for this purpose: "...This achieves anonymity but does not protect against cloning." Indeed, "This...is actually false for known schemes, since these are probabilistic and produce randomly varying signature even if the message is fixed." Our unique group signature can be deemed as important progress on this interesting open question, and it also has many applications beyond unclonable group identification.

Informally speaking, unique group signatures (suitably defined) are adequate for unclonable group identification. For example, the user might send

identification requests that include a signed message of the form "service_name || date" where || denotes concatenation. The server accepts if the signature is valid, and if it doesn't have the same large common component as another identification request received earlier in the day. For this application (and many others), we further need a *non-colliding* property for a unique group signature. A unique group signature is non-colliding if two different signers almost never produce the same unique identifier of the same message.

In another application, the user might send authentication requests that include a signed message of the form "service_name || date || j", where j is any integer between 1 and the (daily) authentication bound k. The server accepts if the signature is valid, and if it doesn't have the same large common component as another authentication request from earlier in the day. This yields a variant of *periodic k-times anonymous authentication scheme* [23,22,24,8]. Of course, many variations are possible by varying the space of messages to be signed.

Notice that for both of these applications, the server can choose whether or not to ask the group manager to reveal the identities of misbehaving users. For minor misbehavior (such as an attempt to authenticate to a service a few more times than the allowed bound, which might be due to innocent human or software or network error) the extra attempts could be detected and ignored. This lets the service provider reserve the relatively harsh penalty of anonymity revocation for more significant (sustained and persistent) misbehavior.

Also note that the deterministic and uniqueness property of our unique signature can lead to very fast processing of data. For example, a service provider carrying out a "first come, first kept" policy on a stream of ℓ requests would need only $\mathcal{O}(\ell \log \ell)$ operations (via appropriate tree structures), or $\mathcal{O}(\ell)$ expected operations (via hash tables). This is particularly useful when there are many users to be processed.

Though it can also deal with some applications that k-times anonymous authentication and more generalized e-token system [8] can, our primitive (even in this respect) is in essence a different one with distinct features and benefits (further discussion and comparison coming shortly).

Two models. This paper studies both the static group signature setting due to Bellare, Micciancio, and Warinschi (BMW) [4] and the dynamic group signature setting due to Bellare, Shi, and Zhang (BSZ) [5]. Intuitively, the static setting has a single authority (called the group manager), which the dynamic setting splits into two: an issuer for enrolling members, and an opener for tracing identities. One might feel that studying static setting is not quite necessary as one could focus on the more involved and generalized dynamic group signature setting. First, this does not make sense *syntactically*, since a dynamic group model is not simply an extension of a static group model. Static group signature models realistic scenarios that the group manager takes full control of the group user generation, and the secret signing key is distributed to each member, preferably, *without* interaction. (Otherwise, the members have to be supported by a trusted PKI, which usually is not the case in such a setting.) Instead, in the dynamic group setting, PKI support and interactive Join/Issue between the

group issuer and group users are both *inevitable*. Second, this does not make sense *technically*, as we shall see, asking for non-interaction raises a few subtle issues in the static setting, making constructing an efficient scheme equally difficult. Third, we believe that static group signature is still *conceptually* more simple and starting from such a non-trivial point will make our presentation much clearer. Last, *constructionally*, our results for static unique group signature are both general and more efficient, while for the dynamic group setting our results are only semi-modular and a little less efficient.

How to model unique group signature? We offer the "strongest" achievable definitions of security for both settings, but here we only highlight the case of dynamic model. On the one hand, the security requirements of dynamic unique group signatures are all simple and clear. Three of them (i.e., CCA-anonymity, traceability, and non-frameability) are based on previous security definitions of ordinary group signatures, while the uniqueness requirement is a quite natural and intuitive one. This is good, whether for understanding the definitions, or for designing the constructions. The uniqueness security notion formalizes the intuition that one signer can only sign one message once. Jumping ahead, we argue that defining uniqueness in the group signature setting raises subtle issues that must be carefully treated.

On the other hand, they are in fact very carefully defined *on the whole*. Recall that our goal is to present a group signature system where each group member can only sign any message once, equipped with an (efficient) detection algorithm such that the identities of ones who disobey such a rule can be revealed and should otherwise be never leaked. All definitions of security are designed to this end. A few seemingly reasonable variants of definitions turn out to be inadequate.

The detection algorithm of our dynamic CCA-anonymous unique group signature is as simple as one could imagine: if the detection authority (i.e., the opener) ever found two different valid group signatures on the same message with the same unique identifier, then it runs the opening algorithm Open to extract their identities i and j (possibly i equals j), and adds them (it) to the misbehaving user set. However, all of these on detection algorithm have to be formally defined, otherwise it leaves one without any notion for what it means to have a *good* detection algorithm. Also note that our defined security properties do *not* even involve any properties of detection algorithms. Instead, we show that once the group system satisfies the four basic security requirements, it gives rise to a good (*complete* and *sound*) detection algorithm.

Constructions. In this paper, we present both the general constructions and efficient instantiations for both static and dynamic group models without relying on random oracles. In the static setting, our general scheme follows the BMW two-level signature construction but uses a *verifiable random function* (VRF) [21] as the second-level signature. We also give a simpler construction for a unique group signature that is secure in a relaxed yet reasonable model. They *together* lead to our final efficient instantiation using Groth-Sahai proof system [19]. All of our constructions (either general or specific) are constant-size, and the instantiation is as efficient as the-state-of-the-art. Our construction for

the unique group signatures in the dynamic setting is semi-modular, and can be instantiated efficiently. The construction can even admit efficient *concurrent-join* which allows many entities concurrently engage in the Join/Issue protocol with the issuer. In building the schemes, we identify new and useful techniques that we believe can be used in other privacy-preserving primitives. We highlight two of them. The first one is a PRF with NIZK proof that can degenerate into a unique signature. In many signature-related primitives, one not only need prove a deterministic function in a zero-knowledge sense but also prove knowledge of input to the function. There are many existing techniques, but ours gives the constructions that can be more efficient and rely on weaker assumptions. The other technique is what we call "*double-chaining certification*", which is used to achieve our unique group signature in the dynamic setting. In essence, this allows us to separate the unique identifier generation process from tracing process, thereby resulting in efficient and intuitive constructions.

Applications and Comparison between other primitives. Our primitive is designed to mitigate the group signature paradox and also motivated by other privacy-preserving constructions, such as k-times anonymous authentication, unclonable group identification protocol, and more generalized e-token systems (periodic k-times anonymous authentication) [8]. The latter primitives are closely related to group signatures, but do *not* have an opening authority that can *always* de-anonymize signed messages.

On the other hand, our primitive can be as well used in applications where (periodic) k-times anonymous authentication is needed as illustrated earlier. Indeed, one can simply use a range proof to extend unique group signature to handle cases for $k > 1$, or one can easily achieve constant-size scheme by registering k public keys for one user at a time. (Note one of our instantiations supports efficient concurrent-join.) However, our primitive, in this respect, has distinct features.

First, the detection algorithms for other primitives are made *public*, meaning that if the a user signs more than the authentication bound k then its identity can be publicly known. This can be both *good* and *bad*: if an honest user accidentally signs slightly more than what is required because of hardware breakdown or clock desynchronization, then the public identity disclosure might not be the most reasonable choice. In fact, we are not aware of any implementations with such stringent mechanisms in *real* applications. Our unique group signature in the dynamic group setting supports in essence a different identity disclosure strategy where the detection authority (other than the group provider) is responsible to detect and reveal disobeyers by the detection algorithm Det. Anyone including the group provider and group members can find publicly misbehaving signatures and report to the detection authority. In our setting, this algorithm is even coupled with a detection proving algorithm DetProve that ensures the detection authority to behave correctly with a proof that the revealed identities are ones of the disobeyers. The opener reserves the right to open persistent misbehaving users to the public, or contact and warn them privately, or send the identity and the corresponding proof into court as it sees fit. As far as we are concerned,

two flavors of revelation are both interesting and should be used depending on specific applications.

Second, it was argued in [23], for their applications only, of course, that it is preferable that the users (who honestly follow the protocol specification) should enjoy anonymity even from the group provider. For the traditional group signature schemes, this requirement is not satisfied. But in the dynamic group model, the group provider might be a distinct entity from the opener who acts as the detection authority. Indeed, the reliance on some other party is inevitable if we do not want to enforce public identity discovery.

Third, in the context of k-times anonymous authentication, to the best of our knowledge, all previous constructions (e.g., [8,22,24,23]) uses an idea originally from e-cash system. The detection algorithm of our primitive is fundamentally different from those. It turns out, perhaps somewhat counter-intuitive, that modeling and achieving "right" detection without using public discovery is actually more challenging.

Last, as mentioned earlier, our primitives can be used in a more efficient way such that no detection algorithm is involved. Namely, the deterministic and unique property of our unique signature lead to very fast processing of data. We are not aware of other primitives admitting such efficient detection.

2 Preliminaries

NOTATIONS. If x is a string then $|x|$ denotes its length. The empty string is denoted ε. If S is a set then $|S|$ denotes its size and $s \xleftarrow{\$} S$ denotes the operation of selecting an element s of S uniformly at random. $\emptyset$ denotes the empty set, while $\boldsymbol{\emptyset}$ denotes a vector of empty sets. If n is an integer $[n]$ denotes the set $\{1, 2, \cdots, n\}$. If $\mathcal{A}$ is a randomized algorithm then we write $z \xleftarrow{\$} \mathcal{A}(x, y, \cdots)$ to indicate the operation that runs $\mathcal{A}$ on inputs $x, y, \cdots$ and a uniformly selected r from an appropriately required domain and outputs z. A function $\epsilon(\lambda)\colon \mathbb{N} \to \mathbb{R}$ is *negligible* if, for any positive number d, there exists some constant $\lambda_0 \in \mathbb{N}$ such that $\epsilon(\lambda) < (1/\lambda)^d$ for any $\lambda > \lambda_0$. For definitions of primitives and cryptographic assumptions, please refer the full version [15, Section 2].

3 Unique Group Signature Models

In this section we present models of unique group signatures in the static setting (following BMW [4]) and in the dynamic setting (following BSZ [5]).

3.1 Static Setting Model

Following [4], a *static group signature scheme* $\mathcal{SGS}$ consists of four algorithms $(\mathsf{GK}, \mathsf{GS}, \mathsf{GV}, \mathsf{Open})$. There is only one group authority which we call the *group manager*. The *group key generation* algorithm GK takes as input the security parameter λ to form a fixed-size group with n members where n may be related

to λ, returning a tuple $(gpk, gmsk, \boldsymbol{gsk})$, where gpk is the *group public key*, $gmsk$ is the *group manager secret key*, and $\boldsymbol{gsk}$ is an n-vector of *secret signing keys* with $gsk[i]$ for each user i. The secret keys are usually distributed to members without interaction. The *group signing* algorithm GS takes as input $gsk[i]$ and a message m to return a signature σ under $gsk[i]$. The *group verification* algorithm GV takes as input the group public key gpk, a message m, and a signature σ for m to return a single bit b. We say that σ is a *valid* signature of m if $\mathsf{GV}(gpk, m, \sigma) = 1$. The *opening* algorithm Open takes the group public key gpk, group manager secret key $gmsk$, a message m, and a signature σ to return an identity i or $\perp$ (indicating failure). Basic *correctness* property is required: for all security parameter λ and integer n, all $(gpk, gmsk, \boldsymbol{gsk}) \xleftarrow{\$} \mathsf{GK}(1^\lambda)$, all $i \in [n]$, and all message $m \in \{0,1\}^*$, it holds that $\mathsf{GV}(gpk, m, \mathsf{GS}(gsk[i], m)) = 1$ and $\mathsf{Open}(gpk, gmsk, m, \mathsf{GS}(gsk[i], m)) = i$.

For our purposes, we consider *static unique group signatures* where the signatures should have the form of $(m, \sigma) = (m, \tau, \psi)$ where τ is the *unique identifier* for the message m and some group member i, and ψ is the rest of the signature. (One can view the unique identifier as a special *tag*.) We define for static unique group signature three security requirements: uniqueness, anonymity, and traceability. The uniqueness requirement formalizes the intuition that one user can only sign one message once, while the last two requirements are adapted from ones for the regular static group signatures with the restraints of being unique.

Uniqueness. Unlike defining uniqueness for a stand-alone signature (i.e., unique signature), it is "tricky" to do so in the context of group signature that involves multiple users. Intuitively, any single group member should not generate more than one valid signatures for any message m. However, it is not quite adequate, for, an adversary may (adaptively) corrupt multiple group members to gain an additional advantage. (In the full version [15, Appendix B], we give a separation result, showing that there exist schemes satisfying a weakened uniqueness definition where the adversary can only corrupt one user but not the standard uniqueness that we define shortly.) We thus give adversary access to a *user secret oracle*, $\mathsf{USK}(\cdot)$, which, when queried with an identity $i \in [n]$, answers with the secret signing key $gsk[i]$ for user i. In the static group signature setting, once the secret key of a user is revealed then it is said to be *corrupted*. Let $\mathtt{CU}$ denote a set of corrupted users. Since the group has a fixed-size n, a set of uncorrupted (i.e., honest) users is $[n]/\mathtt{CU}$. The adversary is also given access to a *user signing oracle*, $\mathsf{GS}(\cdot,\cdot)$, which when queried with an identity i of a user and a message m, returns $\mathsf{GS}(gsk[i], m)$. Note that we do not require that adversary only ask uncorrupted users for this oracle. Let $\mathtt{GS}$ denote a set of message-signature pairs queried via the $\mathsf{GS}(\cdot,\cdot)$ oracle. We write $\mathtt{GS}_m$ to denote a set of users with which adversary calls $\mathsf{GS}(\cdot, m)$. We write $\mathtt{GS}_{\mathbf{M}}$ where $\mathbf{M}$ is a set of the messages queried to denote a vector of sets with $\mathtt{GS}_m$ for each $m \in \mathbf{M}$. For maximal security, we also provide adversary with the secret of the group manager $gmsk$. Formally, given a static signature scheme $\mathcal{SGS}$ of a fixed-size n, we associate to an adversary $\mathcal{A}$ the following experiment:

Experiment $\mathbf{Exp}^{\text{unique}}_{\mathcal{SGS},n}(\mathcal{A})$
$(gpk, gmsk, \boldsymbol{gsk}) \xleftarrow{\$} \mathcal{SGS}.\mathsf{Gen}(1^\lambda)$; $\mathtt{CU} \leftarrow \emptyset$; $\mathtt{GS} \leftarrow \emptyset$
$(m, \sigma_1, \cdots, \sigma_{|\mathtt{CU}|+1}) \xleftarrow{\$} \mathcal{A}^{\mathsf{USK}(\cdot),\mathsf{GS}(\cdot,\cdot)}(gpk, gmsk)$
for $i \leftarrow 1$ **to** $|\mathtt{CU}| + 1$ **do**
  if $\mathsf{GV}(gpk, m, \sigma_i) = 0$ **or** $(m, \sigma_i) \in \mathtt{GS}$ **then return** 0
for $i, j \leftarrow 1$ **to** $|\mathtt{CU}| + 1$ **do**
  if $i \neq j$ **and** $\tau_i = \tau_j$ **then return** 0
return 1

where, above, each σ_i is of the form (τ_i, ψ_i). We define the advantage of $\mathcal{A}$ in the above experiment as

$$\mathbf{Adv}^{\text{unique}}_{\mathcal{SGS},n}(\mathcal{A}) = \Pr[\mathbf{Exp}^{\text{unique}}_{\mathcal{SGS},n}(\mathcal{A}) = 1].$$

In the above experiment, adversary is expected to output *exactly* $|\mathtt{CU}|+1$ *new* and *valid* signatures which have *distinct* unique identifiers w.r.t. the *same* message.

A CAVEAT. We first emphasize that the above notion is the one that we shall use in this paper. However, we do point out some "inadequacies" by considering the following scenario: it is entirely possible that some of keys correspond to one same unique identifier (i.e., they "collide"), while some other keys might generate more unique identifiers than *required*. To put it differently, it might be the case that a set of users of size k who do not collude ought to create $k-1$ unique identifiers as two of them collide, but when they collude they can create k unique identifiers. This does not contract our uniqueness security, but such a collusion clearly makes them sign messages beyond their own.

NON-COLLIDING PROPERTY. In light of this (and as required by some applications mentioned in the introduction), we impose a restriction on our static unique group signature. We say that a group signature is *non-colliding* if any of two different (honest) signers (who follow the scheme specification) almost never produce the same *unique identifier* of the same message. More formally, for all security parameter λ and integer n, all $(gpk, gmsk, \boldsymbol{gsk}) \xleftarrow{\$} \mathsf{GK}(1^\lambda)$, all $i, j \in [n]$ and $i \neq j$, and all message $m \in \{0,1\}^*$, it holds that

$$\Pr[(\tau_i, \psi_i) \xleftarrow{\$} \mathsf{GS}(gsk[i], m); (\tau_j, \psi_j) \xleftarrow{\$} \mathsf{GS}(gsk[j], m) : \ \tau_i = \tau_j] \leq \epsilon(\lambda).$$

Above, the probability is taken over the coins of the group key generation algorithm and group signing algorithm.

The above requirement can resolve the "issue" above. Indeed, if the above-mentioned circumstance happens then an adversary who corrupted a set of group members can always "honestly" generate signatures *again* and pick "enough" signatures with different unique identifiers to attack the uniqueness property. It also makes our primitive justifiable in a few applications—only via this property one can safely achieve the functionality of restricted anonymous authentication (as mentioned in the introduction). Jumping ahead, we claim that the non-colliding property is needed as well in justifying the security of the detection

algorithm of unique group signature. We refer to the full version [15] for further discussion on definitional choices and issues on uniqueness.

Anonymity. Due to the uniqueness property, we cannot achieve the strongest anonymity definition of security as defined in BMW [4]. (The group signature signed by each member i is a partly deterministic function of the gpk, $gsk[i]$, and the message m. If the adversary is given all of the secret keys $\boldsymbol{gsk}$ then it can attack the full-anonymity game simply by re-computing.) Thus a slightly weaker yet still very strong anonymity security notion is used: the adversary can adaptively corrupt the users of the group; for uncorrupted users, adversary is given a signing oracle; in the challenge stage, adversary is not allowed to submit challenge queries with identities of corrupted users, and not allowed to submit challenge queries with at least one of the identities and the message being the same as ones queried before. We write $\mathsf{Open}(\cdot,\cdot)$ to denote the *opening oracle*, which when queried with a message m and a candidate signature σ, answers with $\mathsf{Open}(gpk, gmsk, m, \sigma)$. Specifically, given a static group signature scheme $\mathcal{SGS}$ of a fixed-size n, we associate to an adversary $\mathcal{A}$ the following experiment:

Experiment $\mathbf{Exp}^{\mathrm{anon}}_{\mathcal{SGS},n}(\mathcal{A})$
$(gpk, gmsk, \boldsymbol{gsk}) \xleftarrow{\$} \mathcal{SGS}.\mathsf{Gen}(1^\lambda)$; $\mathtt{CU} \leftarrow \emptyset$; $\mathtt{GS}_{\mathbf{M}} \leftarrow \boldsymbol{\emptyset}$
$(i_0, i_1, m, \mathsf{s}) \xleftarrow{\$} \mathcal{A}^{\mathsf{USK}(\cdot),\mathsf{GS}(\cdot,\cdot),\mathsf{Open}(\cdot,\cdot)}(\mathsf{find}, gpk)$
$b \xleftarrow{\$} \{0,1\}$; $\sigma \xleftarrow{\$} \mathsf{GS}(gsk[i_b], m)$
$b' \xleftarrow{\$} \mathcal{A}^{\mathsf{USK}(\cdot),\mathsf{GS}(\cdot,\cdot),\mathsf{Open}(\cdot,\cdot)}(\mathsf{guess}, \sigma, \mathsf{s})$
if $b' \neq b$ **then return** 0
return 1

where it is mandated that for each $d \in \{0,1\}$ we have $i_d \notin \mathtt{CU}$ and $i_d \notin \mathtt{GS}_m$, and in the guess phase the adversary $\mathcal{A}$ did not query $\mathsf{Open}(\cdot,\cdot)$ with m and σ. We define the advantage of $\mathcal{A}$ in the above experiment as

$$\mathbf{Adv}^{\mathrm{anon}}_{\mathcal{SGS},n}(\mathcal{A}) = \Pr[\mathbf{Exp}^{\mathrm{anon}}_{\mathcal{SGS},n}(\mathcal{A}) = 1] - 1/2.$$

We use the term "CPA-anonymity" to denote the following weakening of the security definition for anonymity [7]: The adversary is never given access to the opening oracle.

Traceability. The traceability security definition is the same as one in BMW [4]. We recall it by considering the experiment that associated to an adversary $\mathcal{A}$:

Experiment $\mathbf{Exp}^{\mathrm{trace}}_{\mathcal{SGS},n}(\mathcal{A})$
$(gpk, gmsk, \boldsymbol{gsk}) \xleftarrow{\$} \mathcal{SGS}.\mathsf{Gen}(1^\lambda)$; $\mathtt{CU} \leftarrow \emptyset$; $\mathtt{GS}_{\mathbf{M}} \leftarrow \boldsymbol{\emptyset}$
$(m, \sigma) \xleftarrow{\$} \mathcal{A}^{\mathsf{USK}(\cdot),\mathsf{GS}(\cdot,\cdot)}(gpk, gmsk)$
if $\mathsf{GV}(gpk, m, \sigma) = 0$ **then return** 0
if $\mathsf{Open}(m, \sigma) = \perp$ **then return** 1
if $\mathsf{Open}(m, \sigma) = i$ **and** $i \notin \mathtt{CU}$ **and** $i \notin \mathtt{GS}_m$ **then return** 1
return 0

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

$$\mathbf{Adv}^{\mathrm{trace}}_{\mathcal{SGS},n}(\mathcal{A}) = \Pr[\mathbf{Exp}^{\mathrm{trace}}_{\mathcal{SGS},n}(\mathcal{A}) = 1].$$

3.2 Dynamic Setting Model

In the dynamic group setting, there are two more features: it allows one to add members to the group; the authority is separated into the *opener* and the *issuer*. An issuer is responsible to enroll members, while an opener traces the identities of signatures signed by the users enrolled. A *dynamic group signature scheme* $\mathcal{DGS}$ consists of six algorithms (GK, $\mathsf{Join}/\mathsf{Issue}$, GS, GV, Open, Judge). We consider *dynamic unique group signatures* having the form of (m, τ, ψ) where τ is the *unique identifier*. A secure unique group signature in the dynamic setting should satisfy correctness and non-colliding property and four security notions: uniqueness, anonymity, traceability and non-frameability. Overall, the definitions in dynamic setting are more *involved* and refer the full version [15] for details.

3.3 Detection Algorithms

We show how our security definitions in *both* settings imply efficient *detection algorithms* that can find who do not follow the algorithm specification *and* disobey the rule that one group member can only sign any message once. Here we only focus on the more involved dynamic setting, and one can easily get similar (but weak) results for the static group setting.

The detection algorithm Det takes as input two different group signatures σ_1 and σ_2 for the same message m and outputs $\perp$ or $\mathcal{I}$ or (b, i, j, θ) for $b \in \{0, 1\}$. The algorithm returns $\perp$ if at least for one of σ_1 and σ_2 it holds that $\mathsf{GV}(gpk, m, \sigma_t) = 0$ ($t \in \{0, 1\}$). If $b = \mathcal{I}$ then the detection algorithm is claiming that at least one of the two signatures was not generated by the group members registered in the ***reg***. (Note that group *issuer* can always generate group signatures on his own by adding dummy users.) In this case, it might have an additional output μ that is a proof that at least one of the signatures was generated by the group issuer. If $b = 0$ then it is claiming that two signatures were generated by two different signers—a rule that the system would like to enforce. In this case, it does not need a proof of the claim. (But one could ask a proof if desired.) In case $b = 1$, it is claiming that two signatures were generated by rule disobeyers i and j, where $i, j \geq 1$, i could be equal to j, and θ is a proof of this claim that is verified by the $\mathsf{DetProve}$ algorithm.

The *detection proving* algorithm $\mathsf{DetProve}$ takes as input the group public key gpk, two valid signatures σ_1 and σ_2 of m, and a vector (b, i, j, θ) output from Det (m, σ_1, σ_2) where $b = 1$, $i, j \geq 1$, and θ is a non-empty string to output a single bit d indicating whether θ is a correct proof that both of i and j disobey the rule.

The detection algorithm should satisfy *completeness* and *soundness* properties described below.

Completeness. The set $\mathtt{LU}$ of legal users (who follow the rule that one signer can only sign one message once) will almost never be wrongly detected by the detection algorithm.

Soundness. If $\mathsf{Det}(m, \sigma_1, \sigma_2) = (1, i, j, \theta)$ and $\mathsf{DetProve}(gpk, m, \sigma_1, \sigma_2, \mathsf{Det}(m, \sigma_1, \sigma_2)) = 1$ then *both* i and j are illegal users (who did not follow the specification of the protocol or the rule).

Alg $\mathsf{Det}(m, \sigma_1 = (\tau_1, \psi_1), \sigma_2 = (\tau_2, \psi_2))$
  if $\mathsf{GV}(m, \sigma_1) = 0$ **or** $\mathsf{GV}(m, \sigma_1) = 0$ **then**
    return $\perp$
  $(i, \omega_i) \leftarrow \mathsf{Open}(m, \sigma_1)$
  $(j, \omega_j) \leftarrow \mathsf{Open}(m, \sigma_2)$
  if $i = 0$ **or** $j = 0$ **then**
    return $(\mathcal{I}, \mu)$
  if $\tau_1 = \tau_2$ **then**
    return $(1, i, j, (\omega_i, \omega_j))$
  return $(0, \varepsilon)$

Alg $\mathsf{DetProve}(m, \sigma_1, \sigma_2)$
  if $\mathsf{Det}(m, \sigma_1, \sigma_2) \neq (1, i, j, (\omega_i, \omega_j))$ **then**
    return 0
  if $\mathsf{Judge}(gpk, (i, \omega_i), m, \sigma_1, \boldsymbol{reg}) = 1$ **and**
    $\mathsf{Judge}(gpk, (j, \omega_j), m, \sigma_2, \boldsymbol{reg}) = 1$ **then**
    return 1
  return 0

Fig. 1. Det and DetProve algorithms

Our dynamic CCA-anonymous unique group signature immediately has an *efficient* complete and sound detection algorithm Det coupled with a detection proving algorithm DetProve, as illustrated in Figure 1. We justify the detection algorithm by providing the following theorem (with proof in the full version [15, Appendix C.1]). We also refer to [15] for further discussion and applications.

Theorem 1. *Given a dynamic unique group signature $\mathcal{DGS}$, if it is correct and non-colliding, and satisfies CCA-anonymity, uniqueness, traceability, and non-frameability requirements, then the Det algorithm given in Figure 1 is complete and sound.* ∎

4 Unique Group Signature Construction – Static Setting

In this section, we first present general constructions for CCA-anonymous unique group signature and for its meaningful relaxations in the static setting. They *together* motivate efficient instantiations by using Groth-Sahai proof system.

A GENERAL CCA-ANONYMOUS UNIQUE GROUP SIGNATURE. Our construction basically follows the general two-level signature constructions of [4]. The difference is that we replace the second-level signature with a verifiable random function, where its public key is signed by the certification key of group manager. We give our general construction using a first-level signature scheme that provides security against random message attacks.[1] Define a verifiable random function $\mathcal{VRF} = (\mathsf{Gen}, \mathsf{Eva}, \mathsf{Prove}, \mathsf{Ver})$ with input domain $\mathcal{X}$ and output range $\mathcal{Y}$. Let $\mathcal{DS} = (\mathsf{Gen}, \mathsf{Sig}, \mathsf{Vrf})$ be a signature scheme. Let $\mathcal{E} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be a public key encryption scheme. Let (P_1, V_1) be a NIZK proof system for a language $\mathcal{L}_1 := \{(m, vk, ek, \tau, C) | \exists (r, vk', \nu', \mathsf{cert})[\mathsf{Vrf}(vk, vk', \mathsf{cert}) = 1, \mathsf{Ver}(vk', m, \tau, \nu') = 1, \text{and } C = \mathsf{Enc}(ek, r, (vk', \nu', \mathsf{cert}))]\}$ where we write $\mathsf{Enc}(ek, r, M)$ for the encryption of a message M under the public key ek using the randomness r. We define a group signature scheme $\mathcal{SGS}_1$ in Figure 2. We have the following theorem:

[1] Informally, a signature is unforgeable against random message attack [14] if it cannot forge a signature on a new message having access to a special oracle that returns signatures on randomly chosen messages.

Alg $\mathsf{GK}(1^\lambda)$
$R \xleftarrow{\$} \{0,1\}^{p(\lambda)}$
$(vk, sk) \xleftarrow{\$} \mathcal{DS}.\mathsf{Gen}(1^\lambda)$
$(ek, dk) \xleftarrow{\$} \mathcal{E}.\mathsf{Gen}(1^\lambda)$
$gpk \leftarrow (R, ek, vk)$
for $i \leftarrow 1$ **to** n **do**
$(sk_i, vk_i) \xleftarrow{\$} \mathcal{VRF}.\mathsf{Gen}(1^\lambda)$
$\mathsf{cert}_i \xleftarrow{\$} \mathsf{Sig}(sk, vk_i)$
$gsk[i] \leftarrow (sk_i, vk_i, \mathsf{cert}_i, gpk)$
$reg[i] \leftarrow vk_i$
$gmsk \leftarrow (dk, \boldsymbol{reg})$
return $(gpk, gmsk, \boldsymbol{gsk})$

Alg $\mathsf{GS}(gsk[i], m)$
$\tau \leftarrow \mathsf{Eva}(sk_i, m)$; $\nu \xleftarrow{\$} \mathsf{Prove}(sk_i, m)$
$C \leftarrow \mathsf{Enc}(ek, r, (vk_i, \nu, \mathsf{cert}_i))$
$\pi \xleftarrow{\$} P_1(R, (m, vk, ek, \tau, C), (r, vk_i, \nu, \mathsf{cert}_i))$
$\sigma \leftarrow (\tau, C, \pi)$
return (m, σ)

Alg $\mathsf{GV}(gpk, m, \sigma)$
return $V_1(R, (m, vk, ek, \tau, C), \pi)$

Alg $\mathsf{Open}(gpk, gmsk, m, \sigma)$
if $V_1(R, (m, vk, ek, \tau, C, \pi)) = 0$ **return** $\perp$
$(vk', \nu', \mathsf{cert}) \leftarrow \mathsf{Dec}(dk, C)$
if $vk' = reg[i]$ **then return** i

Fig. 2. Static unique group signature (a general construction). We write $\boldsymbol{reg}$ to denote $reg[1] \cdots reg[n]$. R is the common reference string for the underlying NIZK proof system (P_1, V_1). $\mathcal{SGS}_1$ is a CCA-anonymous unique group signature, if $\mathcal{DS}$ is unforgeable under random message attacks, $\mathcal{E}$ is CCA-secure, and $\mathcal{VRF}$ is a verifiable random function, and (P_1, V_1) is a simulation-sound NIZK proof system. $\mathcal{SGS}_1$ is CPA-anonymous, if $\mathcal{E}$ is semantically secure and (P_1, V_1) is a regular NIZK proof system.

Theorem 2. *If $\mathcal{VRF}$ is a verifiable random function, $\mathcal{DS}$ is a secure signature against random message attack, scheme, and the underlying NIZK proof system (P_1, V_1) is sound, zero-knowledge, and one-time simulation-sound then the construction $\mathcal{SGS}_1$ in Figure 2 is a secure CCA-anonymous unique group signature in the static setting.* ∎

Relaxations and Separations. The above construction is general but does not seem to immediately give rise to efficient instantiations. This is due, first, to the fact current *simulation-sound* NIZK proof systems are not efficient enough. This is further due to the fact that the VRF proof ν may be incompatible with the efficient proof systems. In light of this, we consider two meaningful relaxations of CCA-anonymous unique group signature. The first natural relaxation is to consider CPA-anonymous unique group signature where the anonymity adversary is never given the opening oracle. This immediately helps avoid using simulation-sound property of NIZK proof system and chosen ciphertext security for the underlying encryption scheme. Namely, we have a group signature the same as illustrated in Figure 2 except that we only use a regular NIZK proof system and a semantic-secure encryption.

Theorem 3. *If $\mathcal{VRF}$ is a verifiable random function, $\mathcal{DS}$ is a secure signature against random message attack, $\mathcal{E}$ is a CPA-secure encryption scheme, and the underlying NIZK proof system (P_1, V_1) is sound and zero-knowledge, then the construction $\mathcal{SGS}_1$ in Figure 2 is a secure CPA-anonymous unique group signature in the static setting.* ∎

The other meaningful relaxation is that we no longer give the uniqueness and traceability adversaries the group manager secret key $gmsk$. This relaxation makes sense as an external adversary usually does not obtain the opening key

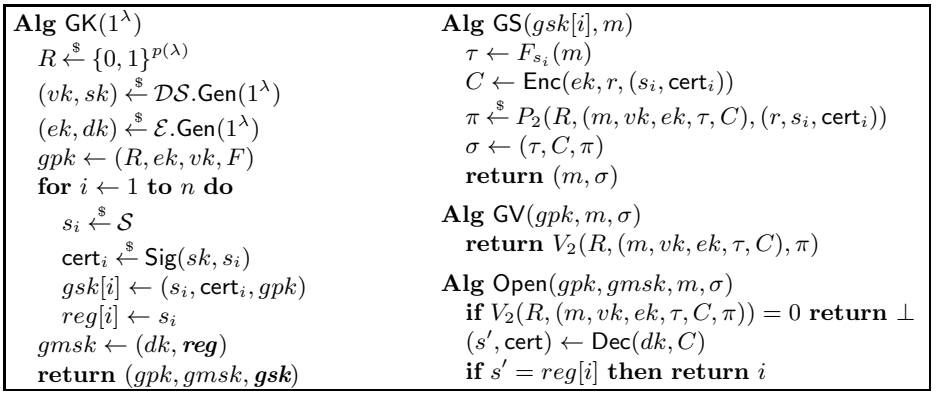

Fig. 3. Static unique group signature $\mathcal{SGS}_2$, with *relaxed* uniqueness and traceability notions, where the adversaries are not given the group manager secret key $gmsk$

of group manager unless it corrupts the group manager which looks less likely. We find that if we restrict the adversary in such a way then we can simply use PRF instead of VRF such that the second problem can be solved.

Define a PRF family $F\colon \mathcal{S} \times \mathcal{X} \to \mathcal{Y}$ where $\mathcal{S}$ is the key space, $\mathcal{X}$ is the message space, and $\mathcal{Y}$ is the range. We write $F_s(\cdot)$ to denote a PRF for every $s \in \mathcal{S}$. Let $\mathcal{DS}$ and $\mathcal{E}$ be a digital signature and a public key encryption scheme respectively. Let (P_2, V_2) be a NIZK proof system for a language $\mathcal{L}_2 := \{(m, vk, ek, \tau, C) | \exists (r, s, \mathsf{cert})[\tau = F_s(m), \mathsf{Vrf}(vk, s, \mathsf{cert}) = 1$, and $C = \mathsf{Enc}(ek, r, (s, \mathsf{cert}))]\}$. We define a unique group signature scheme $\mathcal{SGS}_2$ as illustrated in Figure 3. The following theorem establishes its security.

Theorem 4. *If F is a PRF, $\mathcal{DS}$ is a secure signature against random message attack, $\mathcal{E}$ is a CCA2 secure encryption scheme, and the underlying NIZK proof system (P_2, V_2) is sound,zero-knowledge, and one-time simulation-sound then the construction $\mathcal{SGS}_2$ given in Figure 3 is a CCA-anonymous unique group signature with relaxed uniqueness and traceability where the adversaries are not given $gmsk$.* ∎

One can verify that $\mathcal{SGS}_1$ (i.e., the CPA-anonymous construction) may be not CCA-anonymous, and $\mathcal{SGS}_2$ may be not secure in the sense of standard uniqueness and traceability. Thus, they give natural separations results for these definitions of security. See [15, Appendix B] for proofs and discussion.

Efficient Instantiations. The above concerns do not rule out *ad hoc* constructions in the strongest model. It turns out that we can provide efficient constructions using the Groth-Sahai proof system. The encryption scheme can be replaced with a Groth-Sahai extractable commitment scheme. Given a bilinear group $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, h)$, a commitment to $x \in \mathbb{G}$ (either $\mathbb{G}_1$ or $\mathbb{G}_2$) with randomness r_x is denoted $\mathsf{Com}(x, r_x)$, and an extraction algorithm Extr takes as input the extraction key xk and a commitment C to return a group element.

The key component is a PRF that supports efficient NIZK proof that can *degenerate* into a unique signature scheme where they share the *same* tag. In general, the former helps achieve the anonymity security, where the tag has to

Alg $\mathsf{GK}(1^\lambda)$
$(\mathsf{crs}, xk) \xleftarrow{\$} \mathsf{Groth\text{-}Sahai.Gen}(1^\lambda)$
$(vk, sk) \xleftarrow{\$} \mathcal{DS}.\mathsf{Gen}(1^\lambda)$
$gpk \leftarrow (\mathsf{crs}, vk)$
for $i \leftarrow 1$ **to** n **do**
$s_i \xleftarrow{\$} \mathbb{Z}_q$
$\mathsf{cert}_i \xleftarrow{\$} \mathsf{Sig}(sk, h^{s_i})$
$gsk[i] \leftarrow (s_i, \mathsf{cert}_i, gpk)$
$reg[i] \leftarrow h^{s_i}$
$gmsk \leftarrow (xk, \boldsymbol{reg})$
return $(gpk, gmsk, \boldsymbol{gsk})$

Alg $\mathsf{GS}(gsk[i], m)$
$\tau \leftarrow g^{1/(s_i+m)}$
$C_s \xleftarrow{\$} \mathsf{Com}(h^{s_i})$
$\theta \xleftarrow{\$} \mathsf{Sig}(sk, h^{s_i})$
return $(m, \tau, C_s, C_\theta, \pi_1, \pi_2)$

Alg $\mathsf{GV}(gpk, m, \sigma)$
return $V_3((m, \tau, C_s), \pi_1) \wedge V_4(C_s, C_\theta, vk), \pi_2)$

Alg $\mathsf{Open}(gpk, gmsk, m, \sigma)$
if $\mathsf{GV}(gpk, m, \sigma) = 0$ **return** $\perp$
$S' \leftarrow \mathsf{Extr}(xk, C_s)$
if $S' = reg[i]$ **then return** i

Fig. 4. Efficient CPA-anonymous unique group signatures. Let V_3 and V_4 be the corresponding verification algorithms for the languages $\mathcal{L}_3$ and $\mathcal{L}_4$. The common reference string crs contains the bilinear map parameter $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, h)$ besides the Groth-Sahai proof parameter.

be random, while the latter is used to prove the uniqueness and traceability security, where the tag only needs to be unique and unpredictable.

Specifically, we make use of a *variant* of the PRF with NIZK proof proposed by Belenkiy et al. [3]. We define a language $\mathcal{L}_3 := \{(m, \tau, C_s) | \exists (s, r_s)[\tau = F_s(m)$ and $C_s = \mathsf{Com}(h^s, r_s)]\}$, where $F_s(\cdot) := g^{1/(s+\cdot)}$. The corresponding NIZK proof π_1 is of the form $(C_\tau, \pi_\tau, C'_s, \pi_s, \pi')$, where C_τ is a commitment to τ and π_τ is a NIZK proof for that C_τ is a commitment to τ, C'_s is a commitment to h^s, π_s is a NIZK proof that C_s and C'_s are commitments to the same value, and π' is a witness-indistinguishable proof that C_τ is a commitment to $\bar{\tau}$, C'_s is a commitment to S such that $e(\bar{\tau}, Sh^m) = e(g, h)$. The above proof system is a NIZK proof system for $\mathcal{L}_3$ if DDHI assumption [9,3] holds and Groth-Sahai proof system is secure. As shown above, if we directly let group manager sign each secret key $s \in \mathbb{Z}_q$ (and add each s to $\boldsymbol{reg}$ which is part of $gmsk$) and run a corresponding NIZKPoK then we can get a CPA-anonymous unique group signature yet with relaxed uniqueness and traceability security. Still, this appears hard to find an efficient instantiation in the framework of Groth-Sahai proof system, since the secret s is a scalar rather a group element.

Note that we cannot as well expose the value h^s in the above PRF with NIZK proof system, because neither the above system would be zero-knowledge nor we are able to prove its security based on DDHI assumption. We can, however, degenerate the above PRF with NIZK proof to get a unique signature scheme, where one can view h^s as the public key and $g^{1/(s+m)}$ as the signature of m.Then, the manager can sign each h^s instead of s, and add h^s to $\boldsymbol{reg}$. Fortunately, we can show that uniqueness property and standard unforgeability security (rather than pseudorandomness) suffice to give the security of uniqueness and traceability. This prevents us from using rather strong assumptions such as SDDHI assumption [8] in bilinear groups. In fact, one can prove security of the unforgeability under DHI assumption [13] (with less tight reduction) or SDHI assumption that we formalize where the adversary is only asked to output a new message-signature pair (see the full version [15, Section 2.2]).

It remains to be shown how to choose the first-level signature. Recall that Groth-Sahai commitment, given the extraction trapdoor, can only extract group elements. The first solution is to use the F-unforgeable signature by Belenkiy et al. [2]. They proposed two F-unforgeable signature schemes, one of which has a simple structure, yet using an interactive assumption (i.e., interactive Hidden SDH assumption). We can build our scheme on this signature, while the security can be proven using a weaker and more natural non-interactive q-type assumption. The other is to employ a structure-preserving signature [1] that is only needed secure in the weak random message attack (e.g., one from [17]) to sign h^s directly. To be as general as possible, we let $\mathcal{DS} = (\mathsf{Gen}, \mathsf{Sig}, \mathsf{Vrf})$ be the first-level signature that can sign at least one group element and π_2 is a corresponding Groth-Sahai NIZK proof for the language $\mathcal{L}_4 := \{(C_s, C_\theta, vk) | \exists (S, r_s, \theta, r_\theta)[C_s = \mathsf{Com}(S, r_s)$, and $C_\theta = \mathsf{Com}(\theta, r_\theta)$, and $\mathsf{Vrf}(vk, S, \theta) = 1\}$.The construction is illustrated in Figure 4 and we have the following theorem.

Theorem 5. *The construction in Figure 4 is a CPA-anonymous unique group signature if DDHI and DHI (or SDHI) assumptions hold and Groth-Sahai proof system is secure, and the $\mathcal{DS}$ is structure-preserving and unforgeable under random message attack (or F-unforgeable under random message attack).* ∎

5 Unique Group Signature – Dynamic Setting

Similar to the construction of Section 4, the starting point for a CPA-anonymous unique group signature scheme in the dynamic setting is a two-level certification protocol (with the first-level signature $\mathcal{DS}_1$ and second-level signature $\mathcal{DS}_2$): the issuer signs the verification key of users, and the users can then sign their own messages. This process should be achieved in a zero-knowledge sense.

To make the signature *unique*, one can consider using a PRF F instead of a signature scheme at the second level. Moreover, an interactive protocol is used to get a signature of the secret PRF key s_i of user i under vk, without letting the issuer know the secret. To sign a message m, it computes $\tau := F_{s_i}(m)$, which we would like to use as the unique identifier. It then gets a NIZK proof of knowledge π that there exists a certification chain (s_i, cert_i) such that $\tau = F_{s_i}(m)$ and $\mathcal{DS}_1.\mathsf{Vrf}(vk, s_i, \mathsf{cert}_i) = 1$. The group signature is now (m, τ, π).

It is important that the issuer should not learn the PRF keys that it signs, or the issuer may now attack the CPA-anonymity by simply checking which of the PRF keys could have produced a given unique identifier. In general, we can resort to two-party secure computation. More efficiently, in order for the user i to get cert_i without letting the issuer know s_i (or g^{s_i}, for our construction), they can run a "signing on a committed value" protocol to get a certification of the secret, and user later makes a proof of knowledge of the signature. (They are known as "CL-signatures" [10], and signatures with non-interactive proofs of knowledge are termed as P-signatures [2]). However, this above process does not make the tracing and judging algorithm available. To solve this, we introduce a second chaining of two-level certification; namely, two new signature schemes $\mathcal{DS}'_1$ and $\mathcal{DS}'_2$ are selected. This time, we use Groth-Sahai commitments such

Alg $\mathsf{GK}(1^\lambda)$
$(vk, sk) \xleftarrow{\$} \mathcal{DS}_1.\mathsf{Gen}(1^\lambda)$
$(vk', sk') \xleftarrow{\$} \mathcal{DS}'_1\mathsf{Gen}(1^\lambda)$
$(\mathsf{crs}, xk), (\mathsf{crs}', xk') \xleftarrow{\$} \mathsf{Groth\text{-}Sahai.Gen}(1^\lambda)$
$(X_1, X_2, Y_1, Y_2) \xleftarrow{\$} \mathcal{TE}.\mathsf{Gen}(\mathsf{crs}, 1^\lambda)$
$ek \leftarrow (X_1, X_2, Y_1, Y_2)$
$gpk \leftarrow (\mathsf{crs}, \mathsf{crs}', vk, vk', ek, F)$
$ik \leftarrow (sk, sk')$; $ok \leftarrow xk$
return (gpk, ik, ok)

Alg $\mathsf{Join/Issue}$ $(user\ i,\ issuer)$
$(user\ i : gpk, s_i, vk'_i sk'_i)$
$\leftrightarrows (issuer : gpk, ik)$
$gsk[i] \leftarrow (gpk, s_i, \mathsf{cert}_i, sk'_i, vk'_i, \mathsf{cert}'_i)$
$reg[i] \leftarrow vk'_i$

Alg $\mathsf{GS}(gsk[i], m)$
$(vk_o, sk_o) \xleftarrow{\$} \mathcal{OT}.\mathsf{Gen}(1^\lambda)$
$\tau \leftarrow F_{s_i}(m)$; $\phi \xleftarrow{\$} \mathcal{DS}'_2.\mathsf{Sig}(sk'_i, vk_o)$
$\pi'_1 \xleftarrow{\$} P'_1(\mathsf{crs}', (gpk, m, \tau), (s_i, \mathsf{cert}_i))$
$\pi'_2 \xleftarrow{\$} P'_2(\mathsf{crs}, (gpk, vk_o), (sk'_i, vk'_i, \phi, \mathsf{cert}'_i))$
$C \xleftarrow{\$} \mathcal{TE}.\mathsf{Enc}(ek, vk_o, \phi)$
$\pi'_3 \xleftarrow{\$} P'_3(\mathsf{crs}, (gpk, C, \pi'_2))$
$\phi_o \xleftarrow{\$} \mathcal{OT}.\mathsf{Sig}(sk_o, (vk_o, m, C, \pi'_1, \pi'_2, \pi'_3))$
$\sigma \leftarrow (vk_o, \tau, C, \pi'_1, \pi'_2, \pi'_3, \phi_o)$
return (m, σ)

Alg $\mathsf{GV}(gpk, m, \sigma)$
if $\mathcal{OT}.\mathsf{Vrf}(vk_o, (vk_o, m, C, \pi'_1, \pi'_2, \pi'_3), \phi_o) = 1$
and $V'_1(\mathsf{crs}', (gpk, m, \tau), \pi'_1) = 1$
and $V'_2(\mathsf{crs}, (gpk, vk_o), \pi'_2) = 1$
and $V'_3(\mathsf{crs}, (gpk, C, \pi'_2), \pi'_3) = 1$ **then**
return 1

Alg $\mathsf{Open}(ok, gpk, (m, \sigma))$
$(vk^*, \sigma^*, \mathsf{cert}^*) \leftarrow \mathsf{Extr}(xk, \pi'_2)$
$\omega \leftarrow (vk^*, \sigma^*, \mathsf{cert}^*)$
if $vk^* = reg[i]$ **then return** (i, ω)
return $(0, \omega)$

Alg $\mathsf{Judge}(gpk, (m, \sigma), (i, \omega))$
if $\mathsf{GV}(gpk, m, \sigma) = 1$
and $vk^* = reg[i]$
and $\mathcal{DS}'_1.\mathsf{Vrf}(vk', vk^*, \mathsf{cert}^*) = 1$
and $\mathcal{DS}'_2.\mathsf{Vrf}(vk^*, vk_o, \sigma^*) = 1$ **then**
return 1

Fig. 5. CCA-anonymous unique group signature—Dynamic Setting

that the witnesses can be extracted using the trapdoor given to the opener. Moreover, we can also use this chain to combine the technique of Groth [18] to achieve CCA anonymity. We call this technique "double-chaining certification".

Our algorithm. The CCA-anonymous unique group signature is illustrated in Figure 5. We define a PRF family $F\colon \mathcal{S} \times \mathcal{X} \to \mathcal{Y}$ with key space $\mathcal{S}$. Let $\mathcal{DS}_1$, $\mathcal{DS}'_1$, and $\mathcal{DS}'_2$ be three signature schemes, all of which are secure under adaptive chosen message attacks. Issuer runs $(vk, sk) \xleftarrow{\$} \mathcal{DS}_1.\mathsf{Gen}(1^\lambda)$ and $(vk', sk') \xleftarrow{\$} \mathcal{DS}'_1.\mathsf{Gen}(1^\lambda)$, where (vk, sk) is used to certify PRF keys, and (vk', sk') is used for double-chaining certification. Correspondingly, we use two Groth-Sahai proof systems with the same security parameter but with independently generated common reference strings (crs, xk) and (crs', xk')—the former for the double-chaining certification and the latter for certifying the PRF protocol and proving the knowledge of the corresponding signature. Let $\mathcal{OT}$ be a strong one-time signature scheme secure against weak chosen message attacks. Let $\mathcal{TE}$ be Kiltz's selective-tag weakly CCA-secure encryption scheme [20], with the public key compatible with Groth-Sahai proof system setup. (The secret keys of Kiltz's encryption and xk' can be safely discarded.) We write $\mathsf{Enc}(ek, t, M)$ for the

encryption of a message M under the public key ek and a tag t. User i and the issuer run an interactive $\mathsf{Join/Issue}$ protocol. This includes two steps. First, user i randomly picks its PRF key s_i; the user and issuer run a protocol on signing on committed value s_i, and finally the user gets a signature cert_i on s_i such that $\mathcal{DS}_1.\mathsf{Vrf}(vk, s_i, \mathsf{cert}_i) = 1$. Second, user i runs $(vk'_i, sk'_i) \xleftarrow{\$} \mathcal{DS}'_2.\mathsf{Gen}(1^\lambda)$, sends vk'_i to the issuers, and obtains a cert'_i such that $\mathcal{DS}'_1.\mathsf{Vrf}(vk', vk'_i, \mathsf{cert}'_i) = 1$. After the $\mathsf{Join/Issue}$ procedure, user will get its secret key $(s_i, \mathsf{cert}_i, sk'_i, vk'_i, \mathsf{cert}'_i)$, while the issuer puts vk'_i to $reg[i]$. We now specify the three NIZK proof systems in a general NIZK framework. (P'_1, V'_1) is a NIZK proof system for a language $\mathcal{L}'_1 := \{(gpk, m, \tau) | \exists (s, \mathsf{cert})[\tau = F_s(m)$ and $\mathcal{DS}_1.\mathsf{Vrf}(vk, s, \mathsf{cert}) = 1]$. (P'_2, V'_2) is a NIZK proof system for a language $\mathcal{L}'_2 := \{(gpk, vk_o) | \exists (\overline{vk'}, \phi', \mathsf{cert}')[\mathcal{DS}'_1.\mathsf{Vrf}(vk', \overline{vk'}, \mathsf{cert}') = 1$ and $\mathcal{DS}'_2.\mathsf{Vrf}(\overline{vk'}, vk_o, \phi') = 1]$. (P'_3, V'_3) is a NIZK proof system that the plaintext of C and second-level signature in π'_3 are the same (see [18]).

All of the primitives used in the above construction can be efficiently instantiated using Groth-Sahai proofs. In particular, the first chaining (including the signing on committed value protocol and $\mathcal{L}'_1$) can be achieved by combining the PRF with NIZK proof [3] and the P-signatures [2] (that relies on F-unforgeability). Clearly, we can use the technique in Section 4 (PRF with NIZK that can degenerate into unique signature) to improve the security as well as achieve extractability. $\mathcal{L}'_2$ can be instantiated using any structure-preserving signature combining any signature whose public keys are group elements. (Please refer the full version [15] for more efficient instantiation with concurrent-join.)

Theorem 6. *The construction illustrated in Figure 5 is a secure unique group signature (CCA-anonymous, dynamic setting).* ∎

Acknowledgments. The authors would like to thank Sherman Chow and anonymous reviewers for their helpful and insightful comments.

References

1. Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K., Ohkubo, M.: Structure-Preserving Signatures and Commitments to Group Elements. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 209–236. Springer, Heidelberg (2010)
2. Belenkiy, M., Chase, M., Kohlweiss, M., Lysyanskaya, A.: P-signatures and Noninteractive Anonymous Credentials. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 356–374. Springer, Heidelberg (2008)
3. Belenkiy, M., Chase, M., Kohlweiss, M., Lysyanskaya, A.: Compact E-Cash and Simulatable VRFs Revisited. In: Shacham, H., Waters, B. (eds.) Pairing 2009. LNCS, vol. 5671, pp. 114–131. Springer, Heidelberg (2009)
4. Bellare, M., Micciancio, D., Warinschi, B.: Foundations of Group Signatures: Formal Definitions, Simplified Requirements, and a Construction Based on General Assumptions. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 614–629. Springer, Heidelberg (2003)
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. Camenisch, J., Hohenberger, S., Kohlweiss, M., Lysyanskaya, A., Meyerovich, M.: How to win the clone wars: efficient periodic n-times anonymous authentication. In: ACM CCS 2006, pp. 201–210. ACM (2006)
9. Camenisch, J.L., Hohenberger, S., Lysyanskaya, A.: Compact E-cash. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 302–321. Springer, Heidelberg (2005)
10. Camenisch, J.L., Lysyanskaya, A.: An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 93–118. Springer, Heidelberg (2001)
11. Chaum, D., van Heyst, E.: Group Signatures. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 257–265. Springer, Heidelberg (1991)
12. Damgård, I.B., Dupont, K., Pedersen, M.Ø.: Unclonable Group Identification. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 555–572. Springer, Heidelberg (2006)
13. Dodis, Y., Yampolskiy, A.: A Verifiable Random Function with Short Proofs and Keys. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 416–431. Springer, Heidelberg (2005)
14. Even, S., Goldreich, O., Micali, S.: On-Line/Off-Line Digital Signatures. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 263–275. Springer, Heidelberg (1990)
15. Franklin, M., Zhang, H.: Unique group signatures. Full version. Cryptology ePrint Archive: Report 2012/204, `http://eprint.iacr.org`
16. Fuchsbauer, G.: Automorphic signatures in bilinear groups and an application to round-optimal blind signatures. Cryptology ePrint Archive: Report 2009/320
17. Green, M., Hohenberger, S.: Universally Composable Adaptive Oblivious Transfer. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 179–197. Springer, Heidelberg (2008)
18. Groth, J.: Fully Anonymous Group Signatures Without Random Oracles. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 164–180. Springer, Heidelberg (2007)
19. Groth, J., Sahai, A.: Efficient Non-interactive Proof Systems for Bilinear Groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008)
20. Kiltz, E.: Chosen-Ciphertext Security from Tag-Based Encryption. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 581–600. Springer, Heidelberg (2006)
21. Micali, S., Rabin, M., Vadhan, S.: Verifiable random functions, pp. 120–130. IEEE Computer Society (1999)
22. Nguyen, L., Safavi-Naini, R.: Dynamic k-Times Anonymous Authentication. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 318–333. Springer, Heidelberg (2005)
23. Teranishi, I., Furukawa, J., Sako, K.: k-Times Anonymous Authentication (Extended Abstract). In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 308–322. Springer, Heidelberg (2004)
24. Teranishi, I., Sako, K.: k-Times Anonymous Authentication with a Constant Proving Cost. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 525–542. Springer, Heidelberg (2006)