K-Anonymous Multi-party Secret Handshakes

Shouhuai Xu[1] and Moti Yung[2]

[1] Department of Computer Science, University of Texas at San Antonio
shxu@cs.utsa.edu
[2] RSA Labs, EMC Corp. and Department of Computer Science, Columbia University
moti@cs.columbia.edu

Abstract. Anonymity-protection techniques are crucial for various commercial and financial transactions, where participants are worried about their privacy. On the other hand, authentication methods are also crucial for such interactions. Secret handshake is a relatively recent mechanism that facilitates privacy-preserving mutual authentication between communicating peers. In recent years, researchers have proposed a set of secret handshake schemes based on different assumptions about the credentials used: from one-time credentials to the more general PKI-like credentials. In this paper, we concentrate on k-anonymous secret handshake schemes based on PKI-like infrastructures. More specifically, we deal with the k-anonymous m-party ($m > 2$) secret handshake problem, which is significantly more involved than its two-party counterpart due to the following: When an honest user hand-shakes with $m - 1$ parties, it must be assured that these parties are distinct; otherwise, under the mask of anonymity a dishonest participant may clone itself in a single handshake session (i.e., assuming multiple personalities).

Key words: privacy-preserving authentication, anonymity, unlinkability, secret handshake, credential systems, reusable credentials, multiple personality prevention.

1 Introduction

"Privacy-preserving authentications" are a class of mechanisms that can achieve the traditional authentication goals, and at the same time protect the anonymity of an authenticator, a verifier, or both. Such tools can be useful in transactions where users require anonymity, yet at the same time, the authenticity of a party is needed (e.g., to assure payments, commitment to a contract, or membership in a proper authorized group). It is not a surprise then that in the last two decades a large and diverse set of settings and methods for privacy-preserving authentications have been investigated, among them: (1) anonymous e-coins [10] whereby a merchant can verify a buyer's capability in paying a certain amount of money; (2) group signatures [11] whereby anyone can verify a user's membership in an existing group where the user's privacy within the group is maintained; (3) ring signatures/authentication [12, 24, 22] whereby an authenticator can hide itself among an ad hoc group of users; (4) authenticated key exchange while protecting the anonymity of the participants [18, 21, 8, 1].

Secret handshakes are a new type of privacy-preserving authentications. They were introduced by Balfanz et al. [2] to fulfill the following functionality: *two* participating users authenticate each other in a way that no one reveals its own membership (or credential) unless the peer's legitimacy was already ensured of. As a prototypical setting pointed out in the past, the users can be, for example, CIA agents, and thus the all-or-nothing nature of the authentication result disclosure makes sense. They can also be members of a given commercial exchange or an industry sector, conducting preliminary negotiations regarding a deal, and must be authenticated to belong to the right group, or else (if the quorum is not completely from the group) no details should be learned by anyone (not even that there is indeed a quorum or a sub-quorum ready for the proposed deal in the legitimate group). There have been three different approaches to constructing secret handshake schemes.

- Secret handshakes based on one-time credentials. The pioneering two-party secret handshake scheme due to Balfanze et al. [2] and recent results due to Castelluccia et al. [9] and Jarecki et al. [19] adopted this approach. In this approach, the users are forced to use one-time credentials; otherwise, they suffer from the privacy degradation that all the sessions involving a same user (or credential) are trivially *linkable*. (The importance of ensuring unlinkability in anonymous transaction systems has been well recognized, e.g. [10, 11].) The adoption of one-time credentials could severely limit the usefulness of the resulting secret handshake schemes, since an attacker can easily deplete an honest user's credentials.
- Secret handshakes based on special credential schemes such as group signatures and group key management. This approach was taken by Tsudik and Xu [26] to construct a flexible framework that can seamlessly facilitate two-party and multi-party secret handshakes. This framework does not suffer from issues imposed by one-time credentials. They also presented some concrete constructions.
- Secret handshakes based on PKI-like key infrastructures. This approach was adopted by the k-anonymous *two-party* secret handshake scheme due to Xu and Yung [27]. Intuitively, k-anonymity means that a participant can only hide itself among a set of k users, rather than among a certain entire population. Thus, the fulfilled k-anonymity is *weaker* than the anonymity offered by the aforementioned schemes. However, this approach is valuable because it makes a *weaker* assumption about the underlying credential schemes, namely that it can be built on top of PKI-like infrastructures that may have been more widely deployed. This approach is interesting also because the underlying PKI-like infrastructures themselves do not have to offer any anonymity protection (i.e., this approach can protect anonymity based on non-anonymous infrastructures). We note that in real life, it is highly desirable to exploit a credential mechanism in more than one way (namely, with or without anonymity), which was indeed the motivation for ring signatures (compared to group signatures).

1.1 Our Contributions

We present a multi-party secret handshake method that builds upon the k-anonymous *two-party* secret handshake scheme [27]. Our resulting protocol does not impose any new restrictions on, or introduce any new assumptions into, the setting of [27]. This implies that the resulting multi-party secret handshake schemes achieve *unlinkability* with *reusable* credentials (i.e., unlinkability is not achieved by means of one-time credentials), and that the resulting anonymity still falls into the k-anonymity framework [25], where k is an adjustable parameter indicating the desired anonymity assurance.

We further notice that a straightforward extension to the two-party secret handshake protocol of [27] is subject to a newly emerged attack (i.e., this attack does not have a counterpart in the case of two-party secret handshakes). This attack has to do with a dishonest participant cloning itself in a single handshake session (i.e., assuming multiple personalities), without others realizing that the parties behind supposedly different parties are merely dittos of the same actual misbehaving party (we further discuss these attacks in Section 4.1). Consequently, the resulting k-anonymous m-party ($m > 2$) secret handshake protocol is significantly more involved. Specifically, we present a detailed k-anonymous m-party secret handshake scheme based on a standard PKI, and sketch another scheme based on a symmetric key pre-distribution scheme. Both schemes are efficient and provably secure (in the random oracle model).

Other Related Works: The anonymity fulfilled in this paper falls into the k-anonymity framework due to Sweeney [25]. This framework was originally motivated to protect privacy in the context of database systems so that released information limits what can be revealed about properties of the entities that are to be protected. This anonymity assurance has recently been utilized in some other applications. See also [27] for other, more loosely related, prior works.

Organization: In Section 2, we briefly review the utilized cryptographic tools. In Section 3 we present a system model and definition of k-anonymous m-party secret handshake schemes. In Section 4 we present a scheme based on public key cryptosystems, whereas in Section 5 we sketch a scheme based on symmetric key cryptosystems. We conclude the paper in Section 6.

2 Cryptographic Tools

A function $\epsilon : \mathbb{N} \rightarrow \mathbb{R}^+$ is negligible if for any c there exists κ_c such that $\forall \kappa > \kappa_c$ we have $\epsilon(\kappa) < 1/\kappa^c$.

Public Key Cryptosystems. A public key cryptosystem consists of three polynomial-time algorithms: $\mathsf{pKeyGen}$, pEnc, and pDec. The probabilistic key generation algorithm $\mathsf{pKeyGen}$ takes as input 1^κ and outputs a key pair (pk, sk). The probabilistic encryption algorithm pEnc takes as input a public key pk and a message m, and outputs a ciphertext c. The decryption algorithm pDec takes as input a ciphertext c and a private key sk, and returns a message m or $\perp$ (meaning that c is not valid). We will use public key cryptosystems that are secure against

adaptive chosen ciphertext attack (i.e., IND-CCA2) [23, 14]. Practical schemes are available in [5, 15, 13].

Pseudorandom Functions. A pseudorandom function (PRF) family $\{f_k\}$ parameterized by a secret value k of length κ has the following property [16]: An adversary $\mathcal{A}$ cannot distinguish f_k, where $k \in_R \{0,1\}^\kappa$, from a perfect random function (with the same domain and range) with a non-negligible (in κ) probability.

Symmetric Key Cryptosystems. A symmetric key cryptosystem consists of three polynomial-time algorithms: $\mathsf{sKeyGen}$, sEnc, and sDec. The probabilistic key generation algorithm $\mathsf{sKeyGen}$ takes as input 1^κ and outputs a key pair k. The encryption algorithm sEnc takes as input a key k and a message m, and outputs a ciphertext c. The decryption algorithm sDec takes as input a ciphertext c and a key sk, and returns a message m or $\perp$ (meaning that c is not valid). In parallel to the above public key cryptosystems, we will use symmetric key cryptosystems that are secure against adaptive chosen ciphertext attack or IND-CCA2 secure (we refer the reader to [20] for a thorough treatment on this subject).

Digital Signature Schemes. A digital signature scheme consists of three polynomial-time algorithms: $\mathsf{pKeyGen}$, Sign, and Ver. The probabilistic key generation algorithm $\mathsf{pKeyGen}$ takes as input 1^κ and outputs a key pair (pk, sk). The signing algorithm Sign takes as input a message m and a private key sk, and outputs a signature σ. The verification algorithm Ver takes as input a message m, a public key pk, and a candidate signature σ, returns "accept" if σ is a valid signature and "reject" otherwise. A signature schemes should be existentially unforgeable under an adaptive chosen-message attack [17]. Basically, this means that an adversary $\mathcal{A}$ cannot output a valid signature on a message that was not signed by the signing oracle with a non-negligible probability in κ.

3 Model and Definition

Let U be a set of all possible users. Suppose there is a set of l groups $\mathbf{G} = \{G_1, \cdots, G_l\}$, where each group $G \in \mathbf{G}$ is a set of users (i.e., $G \subset \mathsf{U}$) and managed by an authority CA (the term "group" in this paper always refers to a set of users unless explicitly stated otherwise). In other words, a CA is responsible for admitting users into a group and revoking their memberships when the need arises – just like a certificate authority in a standard PKI. All the participants are modeled as probabilistic polynomial-time algorithms. For simplicity, we assume that each user is a member of exactly one group.

We assume that the communication channels are under an adversary's control. However, the channels are (semi-)synchronous so as to facilitate guaranteed delivery. We assume that there is a broadcast channel between the handshaking participants. The polynomial-time adversary is allowed to corrupt some of the participants at the start of the protocol.

Definition: A multi-party secret handshake scheme SHS consists of the following algorithms:

SHS.CreateGroup This algorithm is executed by an authority, CA, to establish a group G. It takes as input appropriate security parameters, and outputs a cryptographic context specific to this group. In particular, the context may include a data structure called certificate/membership revocation list, CRL, which is originally empty. The cryptographic context is made public.

SHS.AdmitMember This algorithm is run by a CA to admit a user to become a member of the group that is under the CA's jurisdiction. The CA admits members according to a certain policy, which is orthogonal to the focus of this paper. For example, the CA may interact with the user to verify its real identity and its ownership of the private key corresponding to a claimed public key. After executing this algorithm, the group state information (e.g., the list of the members' certificates) is appropriately updated, and the member holds some secret(s) as well as a membership certificate. We will identify an anonymous user through its pseudonym $U \in \mathsf{U}$, which can be included in its certificate.

SHS.Handshake$(U_1, \ldots, U_m)$ This protocol is executed by a set of $m > 2$ distinct anonymous users, where $\{U_1, \ldots, U_m\} \subset \mathsf{U}$ are just a set of placeholders, and U_1 plays the role of the initiator. (It is even true that U_1 does not know any other U_i's pseudonym before a successful handshake, and vice versa.) The input to this protocol includes the anonymous users' secrets, and possibly some public information regarding the current state of the system. The output of this protocol, upon completion, ensures that, for all $i, j \in \{1, \ldots, m\}$ and $i \neq j$, U_i discovers $U_j \in G$ if and only if U_j discovers $U_i \in G$. We say that the protocol returns "1" if the handshake succeeds (i.e., $U_1, \ldots, U_m$ all belong to G), and "0" otherwise.

SHS.RemoveUser This algorithm is executed by an authority CA. It takes as input its current CRL and U's certificate/pseudonym. The output includes an updated CRL which includes the newly revoked certificate U, and perhaps the updated list of the members' certificates/pseudonyms.

Security Definition: Consider a probabilistic polynomial-time adversary $\mathcal{A}$ that may have access to the following oracles:

- $\mathcal{O}_{CG}(\cdot)$: This activates a new CA to create a new group via operation SHS.CreateGroup. The identity, CA, may be given by $\mathcal{A}$ as the input. We assume that a CA is not under $\mathcal{A}$'s control before the new group is established. However, the CA may be corrupt immediately after its establishment (i.e., before admitting any user into the group).
- $\mathcal{O}_{AM}(\cdot, \cdot)$: The input includes the identity of a CA and, optionally, the identity U of a user that is under $\mathcal{A}$'s control. By $\mathcal{O}_{AM}(CA, U)$, the CA may admit the *corrupt* user U by executing SHS.AdmitMember; by $\mathcal{O}_{AM}(CA)$, the CA executes SHS.AdmitMember to admit an *honest* user and assigning it with a unique pseudonym U.
- $\mathcal{O}_{HS}(\cdot, \ldots, \cdot)$: The oracle will activate SHS.Handshake between U_1, ..., and U_m that are given by $\mathcal{A}$. A corrupt user will execute whatever is determined by the adversary.

- $\mathcal{O}_{RU}(\cdot,\cdot)$: The input includes the identity of a CA and a pseudonym U. The oracle actives $\mathsf{SHS.RemoveUser}$ to insert U into the corresponding CRL, and the system state information is appropriately updated.
- $\mathcal{O}_{Corrupt}(\cdot,\cdot)$: The input includes the identity of a CA, and possibly a pseudonym U issued by CA. By $\mathcal{O}_{Corrupt}(CA,U)$, the oracle returns U's current internal state information (including all secrets) to $\mathcal{A}$; by $\mathcal{O}_{Corrupt}(CA)$, the oracle returns CA's current internal state information (including all secrets) to $\mathcal{A}$. Once the CA or U is corrupt, it will execute what $\mathcal{A}$ is pleased of, until such a corruption is detected by some outside mechanism (e.g., intrusion detection systems). When the corruption of a user U is detected, it is excluded from the group via $\mathsf{SHS.RemoveUser}$; when the corruption of an authority CA is detected, its group is simply excluded from the system.

Consider the following security properties: `correctness`, `resistance to impersonation attacks`, and k-`anonymity` specified through k-`resistance to detection attacks`, k-`unlinkability`, and k-`indistinguishability to eavesdroppers`. Compared with the two-party case, major changes are made to the `resistance to impersonation attacks` property.

`Correctness`. If m distinct honest users $U_1,\ldots,U_m$ belong to the same group, then $\mathsf{SHS.Handshake}(U_1,\ldots,U_m)$ always returns "1;" otherwise, it returns "0."

`Resistance to impersonation attacks`. The adversary may corrupt as many members of the group G (needed and employed in the session), as long as there is at least one honest party that is not participating in a handshake session $\mathsf{SHS.Handshake}(X_1,\ldots,X_m)$ (namely, by the pigeon hole principle, the adversary has to play for a party not present, perhaps by using proper group credentials that are already used within the same session). Formally, let $\Xi(X_i)$ denote the (honest or corrupt) user corresponding to the role of X_i in a handshake session, where the adversary intends to impersonate an honest $\Xi(X_i)$. Consider the experiment specified below.

Experiment $\mathsf{RIA}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$:
  $(\mathsf{stateInfo}, CA, U, b, m) \leftarrow \mathcal{A}^{\mathcal{O}_{CG}(\cdot),\mathcal{O}_{AM}(\cdot,\cdot),\mathcal{O}_{AM}(\cdot),\mathcal{O}_{HS}(\cdot,\ldots,\cdot),\mathcal{O}_{RU}(\cdot,\cdot),\mathcal{O}_{Corrupt}(\cdot,\cdot)}(\cdot)$
    where $1 \le b \le m$ and U is a member of the group managed by CA
  Return "1" if the following hold and "0" otherwise:
    $1 \leftarrow \mathsf{SHS.Handshake}(X_1,\ldots,X_m)$ where the role of X_b is played by U
      (1) There is no $\mathcal{O}_{Corrupt}(CA)$ query
      (2) There is no $\mathcal{O}_{AM}(CA,U)$ query
      (3) There is no $\mathcal{O}_{RU}(CA,U)$ query
      (4) $|\{\Xi(X_1),\ldots,\Xi(X_{b-1}),\Xi(X_{b+1}),\ldots,\Xi(X_m)\}$
            $\cap\ \{U' : \exists\mathcal{O}_{AM}(\cdot,U') \vee \exists\mathcal{O}_{Corrupt}(U')\}| \le m-2$
      (5) $\exists X \in \{X_1,\ldots,X_{b-1},X_{b+1},\ldots,X_m\}$
          such that $\neg\,((\exists\mathcal{O}_{AM}(\cdot,\Xi(X)) \vee \exists\mathcal{O}_{Corrupt}(\Xi(X))) \wedge \neg\exists\mathcal{O}_{RU}(CA,\Xi(X)))$
          and $\Xi(x)$ does not participate in the session

The above (4) captures that there are at least two roles *corresponding to* two honest users (including U), whereas (5) captures that at least one role (corresponding to an honest user) *is being played* by the adversary. Let $\mathsf{AdvRIA}_{\mathsf{SHS},\mathcal{A}}(1^\kappa) = \Pr[\mathsf{RIA}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$ returns "1"$]$, which is the probability that $\mathsf{RIA}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$ returns "1", where probability is taken over all the tossed coins. A handshake scheme SHS is "`resistant to impersonation attacks`" if for $\forall\mathcal{A}$, $\mathsf{AdvRIA}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$ is

negligible in the security parameter of SHS. We notice that this definition can be smoothly degenerated to the two-party case of $m = 2$ to accommodate its counterpart definition in [27]. This is because the above restrictions (4) and (5) imply that the adversary intends to impersonate an honest user.

k-`resistance to detection attacks`. Suppose there are β groups such that none of their members or CAs is corrupt. Ideally, "`resistance to detection attacks`" means that no adversary $\mathcal{A}$ who does not belong to any of the β groups, can successfully guess the membership of an anonymous (and honest) handshaking peer U, who is a member of one of the β groups, with a non-negligible advantage over $1/\beta$. In this paper, we pursue a weaker, but practical and useful (see [27] for discussions on the usefulness of k-anonymity in the context of secret handshakes), notion we call "k-`resistance to detection attacks`", where $2 \leq k \leq \beta$ is a parameter indicating the desired anonymity assurance. Intuitively, it means that no adversary $\mathcal{A}$, who does not belong to any of the k groups (including the group U belongs to), can successfully guess the membership of U with a non-negligible advantage over $1/k$. Formally, consider the experiment specified below.

Experiment $\mathsf{RDA}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$:
  $(\mathsf{stateInfo}, b, m) \leftarrow \mathcal{A}^{\mathcal{O}_{CG}(\cdot),\mathcal{O}_{AM}(\cdot,\cdot),\mathcal{O}_{AM}(\cdot),\mathcal{O}_{HS}(\cdot,\ldots,\cdot),\mathcal{O}_{RU}(\cdot,\cdot),\mathcal{O}_{Corrupt}(\cdot,\cdot)}(\cdot)$
    where $1 \leq b \leq m$
  Execute $\mathsf{SHS.Handshake}(X_1, \ldots, X_{b-1}, U_b, X_{b+1}, \ldots, X_m)$
    where $X_1, \ldots, X_{b-1}, X_{b+1}, \ldots, X_m$ are placeholders for the other participants and $\mathcal{A}$ plays the role of at least one of them
  Let CA_i be the authority of group G_i, to which U_b belongs
  Return "1" if the following hold and "0" otherwise:
    (1) There is no $\mathcal{O}_{RU}(CA_i, U_b)$ query
    (2) There are at least k groups (including G_i) s.t. for each one managed by CA:
      (2.1) There is no $\mathcal{O}_{Corrupt}(CA)$ query
      (2.2) If there is an $\mathcal{O}_{AM}(CA, Y)$ query, then there is an $\mathcal{O}_{RU}(CA, Y)$ query
      (2.3) If there is an $\mathcal{O}_{Corrupt}(CA, Y)$ query, then there is an $\mathcal{O}_{RU}(CA, Y)$ query
    (3) $\mathcal{A}$ outputs i

Let $\mathsf{AdvRDA}_{\mathsf{SHS},\mathcal{A}}(1^\kappa) = |\Pr[\mathsf{RDA}_{\mathsf{SHS},\mathcal{A}}(1^\kappa) \text{ returns "1"}] - 1/k|$, which is the advantage that $\mathcal{A}$ successfully guesses the membership of an anonymous handshake peer X. The probability is taken over all the tossed coins. A scheme SHS is "k-`resistant to detection attacks`" if for $\forall\mathcal{A}$, $\mathsf{AdvRDA}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$ is negligible in the security parameter of SHS.

k-`unlinkability`. Suppose there are β groups such that none of their members or CAs is corrupt. Ideally, "`unlinkability`" means that no adversary $\mathcal{A}$ who does not belong to any of the β groups, can successfully associate two sessions involving a same honest user with a non-negligible advantage over $1/\beta$. (We remark that this corresponds to the worst case scenario that the honest user could have been identified by the adversary in one of the two sessions through whatever means; e.g., the adversary corrupted some users of the group the honest user belongs to when that specific session was conducted.) In this paper, we pursue a weaker, but still practical and useful, notion we call "k-`unlinkability`", where $2 \leq k \leq \beta$ is a parameter indicating the desired anonymity assurance. Intuitively, it means that no adversary $\mathcal{A}$ who does not belong to any of the k groups, can successfully associate two sessions, s_1 and s_2, involving a same honest user with

a non-negligible advantage over $1/k$. Formally, consider the experiment specified below.

Experiment $\mathsf{Unlink}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$:
$(s_1, s_2) \leftarrow \mathcal{A}^{\mathcal{O}_{CG}(\cdot),\mathcal{O}_{AM}(\cdot,\cdot),\mathcal{O}_{AM}(\cdot),\mathcal{O}_{HS}(\cdot,\ldots,\cdot),\mathcal{O}_{RU}(\cdot,\cdot),\mathcal{O}_{Corrupt}(\cdot,\cdot)}(\cdot)$
Return "1" if the following hold and "0" otherwise:
(1) s_1 and s_2 involve a same user $U \in G_i$, where G_i is managed by CA_i
(2) There is no $\mathcal{O}_{RU}(CA_i, U)$ query
(3) There exist $z \in \{1, 2\}$ such that w.r.t. s_z the following holds:
There are at least k groups (including G_i) s.t. for each one managed by CA:
(3.1) There is no $\mathcal{O}_{Corrupt}(CA)$ query
(3.2) If there is an $\mathcal{O}_{AM}(CA, Y)$ query, then there is an $\mathcal{O}_{RU}(CA, Y)$ query
(3.3) If there is an $\mathcal{O}_{Corrupt}(CA, Y)$ query, then there is an $\mathcal{O}_{RU}(CA, Y)$ query

Let $\mathsf{AdvUnlink}_{\mathsf{SHS},\mathcal{A}}(1^\kappa) = |\Pr[\mathsf{Unlink}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$ returns "1"$] - 1/k|$, which is the advantage that $\mathcal{A}$ successfully associates two handshake sessions to a same honest user. The probability is taken over all the tossed coins. A scheme SHS is "k-`unlinkable`" if for $\forall\mathcal{A}$, $\mathsf{AdvUnlink}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$ is negligible in the security parameter of SHS.

k-`indistinguishability to eavesdroppers`. Consider an adversary $\mathcal{A}$ who corrupts some users, interacts with users, and observes a session of SHS.Handshake for incorrupt users $U_1, \ldots, U_m$. Suppose there are at least k groups (including the groups $U_1, \ldots, U_m$ belong to) such that none of their members or CAs is corrupt. Intuitively, this property means that $\mathcal{A}$ should not be able to learn from this handshake session anything that it did not already know. In order to capture this property, consider a simulated transcript of a handshake session, which is obtained by substituting all the strings derived from cryptographic secrets with random strings of appropriate lengths (i.e., no cryptographic secrets or memberships are involved). Yet, such a substitution cannot be detected by $\mathcal{A}$. Formally, consider the experiment specified below.

Experiment $\mathsf{INDeav}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$:
Let $R \notin \mathsf{U}$ be an algorithm/simulator having no access to any cryptographic secrets
$\mathsf{stateInfo} \leftarrow \mathcal{A}^{\mathcal{O}_{CG}(\cdot),\mathcal{O}_{AM}(\cdot,\cdot),\mathcal{O}_{AM}(\cdot),\mathcal{O}_{HS}(\cdot,\ldots,\cdot),\mathcal{O}_{RU}(\cdot,\cdot),\mathcal{O}_{Corrupt}(\cdot,\cdot)}(\cdot)$
Flip a random coin b
If $b = 0$ then give $\mathcal{A}$ a transcript $trans$ of $\mathsf{SHS.Handshake}(U_1, \ldots, U_m)$
else give $\mathcal{A}$ a simulated transcript $trans$ of $\mathsf{SHS.Handshake}(R, \ldots, R)$
$b' \leftarrow \mathcal{A}^{\mathcal{O}_{CG}(\cdot),\mathcal{O}_{AM}(\cdot,\cdot),\mathcal{O}_{AM}(\cdot),\mathcal{O}_{HS}(\cdot,\ldots,\cdot),\mathcal{O}_{RU}(\cdot,\cdot),\mathcal{O}_{Corrupt}(\cdot,\cdot)}(\mathsf{stateinfo}, trans)$
Suppose $U_i \in G_i$ and G_i is managed by CA_i for $1 \le i \le m$
Return "1" if the following hold and "0" otherwise:
(1) $b' = b$
(2) There are no $\mathcal{O}_{RU}(CA_i, U_i)$ queries for all $i \in \{1, \ldots, m\}$
(3) There are at least k groups (including those the $U_1, \ldots, U_m$ belong to)
such that for each group G managed by CA:
(3.1) There is no $\mathcal{O}_{Corrupt}(CA)$ query
(3.2) If there is an $\mathcal{O}_{AM}(CA, Y)$ query, then there is an $\mathcal{O}_{RU}(CA, Y)$ query
(3.3) If there is an $\mathcal{O}_{Corrupt}(CA, Y)$ query, then there is an $\mathcal{O}_{RU}(CA, Y)$ query

Let us define $\mathsf{AdvINDeav}_{\mathsf{SHS},\mathcal{A}}(1^\kappa) = |\Pr[\mathsf{INDeav}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$ returns "1"$|b = 0] -$ $\Pr[\mathsf{INDeav}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$ returns "1"$|b = 1]|$, which is the advantage that $\mathcal{A}$ successfully distinguishes a real transcript from a simulated one. The probability is taken over all the tossed coins. A scheme SHS is "k-`indistinguishable to eavesdroppers`" if for $\forall\mathcal{A}$, $\mathsf{AdvINDeav}_{\mathsf{SHS},\mathcal{A}}(1^\kappa)$ is negligible in the security parameter of SHS.

4 Public Key Cryptosystems based Construction

Notations. Recall that we denoted by $\mathbf{G} = \{G_1, \cdots, G_l\}$ a set of groups, where G_z $(1 \leq z \leq l)$ is a set of users whose public keys are certified by an authority CA_z using a secure signature scheme. Without loss of generality, we assume that both the groups G_1, $\cdots$, G_l and the users in a group $G \in \mathbf{G}$ are in an appropriate order (e.g., alphabetic), which means that partitions over the sets can be naturally defined. If X is a user, let $\mathsf{Group}(X)$ be a function that returns the identity of the group to which X belongs, and $Cert_X$ denote X's public key certificate. If $Cert$ is a certificate, let $\mathsf{CA}(Cert)$ be a function that returns the identity of the authority which issues it. For example, a user $X \in G_i$ owning a public key pk_X will be issued a certificate $Cert_X$ by authority CA_i. Moreover, $\mathsf{Group}(X)$ returns G_i and $\mathsf{CA}(Cert_X)$ returns CA_i.

Let κ_0 and κ_1 be additional security parameters and q be a prime of length κ_0. Assume $f, g : \{0,1\}^{\kappa_1} \times \{0,1\}^* \rightarrow \{0,1\}^{\kappa_1}$ are pseudorandom functions, and $h : \{0,1\}^* \rightarrow \mathbb{Z}_q$ is an ideal hash function [4].

4.1 Basic Ideas

Basic idea underlying the k-anonymous two-party secret handshake scheme. The basic idea underlying the two-party scheme [27] is to let each party "draft" on-the-fly a certain number w of users to form a "crowd" while ensuring that the drafting algorithms leak no information about the drafting party. Specifically, the first participant drafts a crowd $U_{1,1}, \ldots, U_{1,w}$ such that it plays the role of $U_{1,i}$ for some $1 \leq i \leq w$, and the second participant drafts a crowd $U_{2,1}, \ldots, U_{2,w}$ such that it plays the role of $U_{2,j}$ for some $1 \leq j \leq w$. The drafting algorithm ensures that, for $1 \leq z \leq 2$, $U_{1,z}$ and $U_{2,z}$ belong to the the same group (i.e., they are managed by the same authority). If we let the first participant encrypt a random value v_1 using the public key of $U_{2,i}$, and the second participant encrypts a random value v_2 using the public key of $U_{1,j}$, then a successful handshake (i.e., $i = j$) allows the participants to reach a common secret $F(v_1, v_2)$ for some appropriate function F.

Why a straightforward extension to the two-party secret handshake scheme is not secure? As briefly reviewed above, a successful two-party secret handshake allows the participants to reach a common secret $F(v_1, v_2)$ by ensuring, for instance, that v_1 is encrypted **only** using the other involved public key certified by the same CA (i.e., the CA that certified the public key of the participant who selected v_1). In a straightforward extension to the two-party case, a successful multi-party secret handshake would allow the participants to reach a common secret $F(v_1, v_2, \ldots, v_m)$, where v_i is a random value selected by the ith participant. To facilitate this, for instance, v_1 should be encrypted using **all** the other involved public keys certified by the same CA (i.e., the CA that certified the public key of the participant who selected v_1). However, this would allow a dishonest user to cheat an honest user into believing that it is handshaking with $m - 1 > 1$ peers, while it is indeed handshaking with a single

dishonest user (who can clone its participation and thus exhibit multiple personalities). This is because using the adversarial participant's private key alone is enough to derive $F(v_1, v_2, \ldots, v_m)$. The attack is specific to, and emerged in, the multi-party scenario, because in the case of two-party secret handshakes, $m = 2$ and thus the above two cases (i.e., "the **only** other key" vs. "**all** the other keys") amount to the same thing. (We remark that the capability that "one can impersonate multiple parties belonging to the same group" may not be harmful in some specific application scenarios, but should be eliminated in general, and is an attack where a quorum is required.)

How should the newly emerged attack be dealt with? We adopt a two-stage strategy of constructing k-anonymous multi-party secret handshakes. In the first stage (corresponding to Phase I-III in the construction), a preliminary secret handshake is conducted to detect if the participants indeed belong to the same group, but there is no guarantee that the participants are distinct. This is quite similar to the aforementioned straightforward extension to the two-party case.

In the second stage (corresponding to Phase IV-VI in the construction), effort is taken to ensure that the participants in a successful preliminary handshake are indeed distinct. This is fulfilled by forcing that the private keys corresponding to all the other involved public keys belonging to the same group are utilized for decryption. For simplicity, let's consider the case of three-party secret handshakes, where each participant draws a "crowd" of size w. We can let the first participant encrypt $v_{1,2,1}, \ldots, v_{1,2,w}$ to the second participant using the respective public keys drawn by the second participant, and encrypt $v_{1,3,1}, \ldots, v_{1,3,w}$ to the third participant using the respective public keys drawn by the third participant. Then we can let the second participant "translate" (i.e., decrypt and then encrypt) $v_{1,2,1}, \ldots, v_{1,2,w}$ to the third participant, and let the third participant "translate" (i.e., decrypt and then encrypt) $v_{1,3,1}, \ldots, v_{1,3,w}$ to the second participant. If we let each participant act as the first participant (this is necessary; otherwise, attacks on anonymity are easy to conceive), then it is clear that a successful handshake does result in a common secret $F(v_{1,2,i}, v_{1,3,i}, v_{2,1,i}, v_{2,3,i}, v_{3,1,i}, v_{3,2,i})$, where i indicates the location of the group, to which the participants belong (i.e., in the case of a successful handshake), in the crowds. We notice that special care must also be taken to ensure, for instance, that by simply observing the traffic flow it is not possible for the adversary to tell a successful (preliminary) secret handshake from a failed one. This combination of properties explains why the resulting protocol is seemingly quite involved.

4.2 Subroutines

Two subroutines are specified below. Specifically, rSelect is for selecting w groups from $\mathbf{G} = \{G_1, \cdots, G_l\}$ where w is a parameter that will be determined later, mSelect is for selecting w members from the w groups, rSelectVer and mSelectVer are for verifying that the selections are appropriately done. For simplicity, we assume $w|l$. Proof of Lemma 1 can be found in the full version of the present

paper [28]. This lemma ensures that the adversary cannot tell who is the party that draws a "crowd" in the above rSelect and mSelect algorithms. Therefore, we will simply treat the two processes as being ideal.

The algorithm $\mathsf{rSelect}(\mathbf{G}, U_1, w, n_1, \ldots, n_m)$ has the following steps:

1. Partition $\mathbf{G}$ into $\mathbf{G}_0, \cdots, \mathbf{G}_{w-1}$ where $\mathbf{G}_z = \{G_{z_0}, \cdots, G_{z_{l/w-1}}\}$ for $0 \leq z \leq w-1$. Assume $U_1 \in G_{i_u}$ for some $0 \leq i \leq w-1$ and $0 \leq u \leq l/w-1$.
2. Set $\eta = h(n_1, \ldots, n_m, 0)$ and $x = h(n_1, \ldots, n_m, 1, i)$. Choose $r \in_R \{0, 1, \cdots, \lfloor (q-1)^2 w/l \rfloor\}$ and set $y_i = u + r \cdot l/w$ (in $\mathbb{Z}$). Solve $y_i = \eta \cdot x + \theta_1 \mod q$ to get θ_1.
3. For $z = 0$ to $w-1$ (except $z = i$), set $y_z = \eta \cdot h(n_1, \ldots, n_m, 1, z) + \theta_1 \mod q$, and $s_z = y_z \mod l/w$.
4. Output $(\mathbf{G}^* = \{G_{z_{s_z}}\}_{z=0}^{w-1}, \theta_1)$, where $G_{z_{s_z}}$ is the group selected from $\mathbf{G}_z$ and $s_i = u$.

The algorithm $\mathsf{rSelectVer}(\mathbf{G}, \mathbf{G}^*, w, n_1, \ldots, n_m, \theta_1)$ has the following steps:

1. Partition $\mathbf{G}$ into $\mathbf{G}_0, \cdots, \mathbf{G}_{w-1}$ where $\mathbf{G}_z = \{G_{z_0}, \cdots, G_{z_{l/w-1}}\}$ for $0 \leq z \leq w-1$.
2. Parse $\mathbf{G}^*$ as $\{G_{z_{s_z}}\}_{z=0}^{w-1}$, and set $\eta = h(n_1, \ldots, n_m, 0)$.
3. Accept if for $z = 0$ to $w-1$, it holds that $s_z = y_z \mod l/w$ where $y_z = \eta \cdot h(n_1, \ldots, n_m, 1, z) + \theta_1 \mod q$; reject otherwise.

The algorithm $\mathsf{mSelect}(\mathbf{G}^*, X, w, n_1, \ldots, n_m)$ has the following steps:

1. Parse $\mathbf{G}^*$ as $\{G_{z_{s_z}}\}_{z=0}^{w-1}$. Assume X is the λ-th member of group $G_{a_{s_a}}$ for some $0 \leq a \leq w-1$ and $0 \leq \lambda \leq |G_{a_{s_a}}| - 1$.
2. Set $\eta = h(n_1, \ldots, n_m, 2)$ and $x = h(n_1, \ldots, n_m, 3, a, s_a)$. Choose $r \in_R \{0, 1, \cdots, \lfloor (q-1)^2/|G_{a_{s_a}}| \rfloor\}$, and set $y_a = \lambda + r \cdot |G_{a_{s_a}}|$ (in $\mathbb{Z}$). Solve $y_a = \eta \cdot x + \theta_2 \mod q$ to get θ_2.
3. For $z = 0$ to $w-1$ (except $z = a$), set $\lambda_z = y_z \mod |G_{z_{s_z}}|$ where $y_z = \eta \cdot h(n_1, \ldots, n_m, 3, z, s_z) + \theta_2 \mod q$.
4. Output $(\mathbf{X} = \{X_{z_{s_z, \lambda_z}}\}_{z=0}^{w-1}, \theta_2)$, where $\lambda_a = \lambda$ and $X_{z_{s_z, \lambda_z}}$ is the λ_z-th member of group $G_{z_{s_z}}$.

The algorithm $\mathsf{mSelectVer}(\mathbf{G}^*, \mathbf{X}, w, n_1, \ldots, n_m, \theta_2)$ has the following steps:

1. Parse $\mathbf{G}^*$ as $\{G_{z_{s_z}}\}_{z=0}^{w-1}$, and parse $\mathbf{X}$ as $\{X_{z_{s_z, \lambda_z}}\}_{z=0}^{w-1}$.
2. Accept if for $z = 0$ to $w-1$, it holds that $\lambda_z = y_z \mod |G_{z_{s_z}}|$ where $y_z = \eta \cdot h(n_1, \ldots, n_m, 3, z, s_z) + \theta_2 \mod q$; reject otherwise.

Lemma 1. *Let q be a positive number super-polynomial in security parameter κ_0, and p_1 be uniformly distributed over $\mathbb{Z}_q$, and p_2, p_3 be some positive numbers bounded by some polynomials in κ_0. Let further (a, x, b) be uniformly distributed over $(\mathbb{Z}_q)^3$, and r be uniformly distributed over $\{0, \ldots, \lfloor (q-1)^2 p_2/p_3 \rfloor\}$. Then the distribution of $y = ax + b$ in $\mathbb{Z}$ and the distribution of $y' = p_1 + r \cdot p_3/p_2$ in $\mathbb{Z}$ are statistically close to each other. Moreover, the distribution of $y \mod q$ and the distribution of $y' \mod q$ are also statistically close to each other.*

4.3 The Construction

Since all the algorithms other than $\mathsf{SHS.Handshake}(U_1, \ldots, U_m)$ are merely as in a standard and well-known public key infrastructure (i.e., certifying, signing, etc.), we only specify $\mathsf{SHS.Handshake}(U_1, \ldots, U_m)$ which enables $U_1, \ldots, U_m$ to figure out if they belong to the same group.

1. Let U_i, $1 \leq i \leq m$, select and broadcast $n_i \in_R \{0,1\}^{\kappa_2}$. This is for the peers to exchange nonces, and typically could have been done before the handshake protocol is activated.

2. Phase-I protocol for U_1:
   (a) Run algorithm $\mathsf{rSelect}(\mathbf{G}, U_1, w, n_1, \ldots, n_m)$ to obtain $(\mathbf{G}^*, \theta_1)$, where $U_1 \in G_{i_{s_i}}$.
   (b) Run algorithm $\mathsf{mSelect}(\mathbf{G}^*, U_1, w, n_1, \ldots, n_m)$ to obtain $(\mathbf{X}_1, \theta_{1,2})$. In order to clarify presentation, parse $\mathbf{X}_1$ also as $\{U_{1,z,z_{s_z,\lambda_{1,z}}}\}_{z=0}^{w-1}$, where $U_{1,z,z_{s_z,\lambda_{1,z}}} \in G_{z_{s_z}}$ and $0 \le \lambda_{1,z} \le |G_{z_{s_z}}| - 1$.
   (c) Broadcast $(\mathbf{G}^*, \mathbf{X}_1, \theta_1, \theta_{1,2}, \Delta_1 = \{Cert_{U_{1,z,z_{s_z,\lambda_{1,z}}}}\}_{z=0}^{w-1})$.
3. Phase-I protocol for U_i $(2 \le i \le m)$: Upon receiving $\{(\mathbf{G}^*, \mathbf{X}_\tau, \theta_1, \theta_{\tau,2}, \Delta_\tau = \{Cert_{U_{\tau,z,z_{s_z,\lambda_{\tau,z}}}}\}_{z=0}^{w-1})\}_{\tau=1}^{i-1}$, check for every $1 \le \tau \le i-1$ if the selected groups (consistently indicated by $\mathbf{G}^*$, $\mathbf{X}_\tau$, and the certificates) are distinct, if $\mathsf{CA}(Cert_{U_i}) = \mathsf{CA}(Cert_{U_{\tau,j,j_{s_j},\lambda_{\tau,j}}})$ for some $0 \le j \le w-1$, if the groups are selected via $\mathsf{rSelectVer}(\mathbf{G}, \mathbf{G}^*, w, n_1, \ldots, n_m, \theta_1)$, if $Cert_{U_i} \notin \Delta_\tau$, and if the members are selected via $\mathsf{mSelectVer}(\mathbf{G}^*, \mathbf{X}_\tau, w, n_1, \ldots, n_m, \theta_{\tau,2})$. If any condition is not satisfied, quit; otherwise, execute as follows:
   (a) Set $\mathbf{G}' = \{G'_{z_{s_z}}\}_{z=0}^{w-1}$, where $G'_{z_{s_z}} = G_{z_{s_z}} - \{U_{1,z,z_{s_z,\lambda_{1,z}}}, \ldots, U_{i-1,z,z_{s_z,\lambda_{i-1,z}}}\}$.
   (b) Run algorithm $\mathsf{mSelect}(\mathbf{G}', U_i, w, n_1, \ldots, n_m)$, which returns $(\mathbf{X}_i, \theta_{i,2})$. In order to clarify presentation, parse $\mathbf{X}_i$ also as $\{U_{i,z,z_{s_z,\lambda_{i,z}}}\}_{z=0}^{w-1}$, where $U_{i,z,z_{s_z,\lambda_{i,z}}} \in G'_{z_{s_z}}$ and $0 \le \lambda_{i,z} \le |G'_{z_{s_z}}| - 1$.
   (c) Broadcast $(\mathbf{G}^*, \mathbf{X}_i, \theta_1, \theta_{i,2}, \Delta_i = \{Cert_{U_{i,z,z_{s_z,\lambda_{i,z}}}}\}_{z=0}^{w-1})$.
4. Phase-II protocol for U_i $(1 \le i \le m)$: At this point, U_i has received $\{(\mathbf{G}^*, \mathbf{X}_\tau, \theta_1, \theta_{\tau,2}, \Delta_\tau = \{Cert_{U_{\tau,z,z_{s_z,\lambda_{\tau,z}}}}\}_{z=0}^{w-1})\}_{1 \le \tau \le m}$. For $\tau = i+1$ to m, U_i checks if $\mathsf{CA}(Cert_{U_{\tau,z,z_{s_z,\lambda_{\tau,z}}}}) = \mathsf{CA}(Cert_{U_{i,z,z_{s_z,\lambda_{i,z}}}})$ for $0 \le z \le w-1$, and if the members are selected by running $\mathsf{mSelectVer}(\mathbf{G}^*, \mathbf{X}_\tau, w, n_1, \ldots, n_m, \theta_{\tau,2})$. If not, quit; otherwise, execute as follows:
   (a) To simplify the presentation, rename $U_{\tau,z,z_{s_z,\lambda_{\tau,z}}}$ to $U_{\tau,z}$ because once $1 \le \tau \le m$ and $0 \le z \le w-1$ are determined, $z_{s_z,\lambda_{\tau,z}}$ is uniquely defined. Therefore, the drawn $m \times w$ users can be denoted by a "matrix" $U_{1,0}, \ldots, U_{1,w-1}, \ldots, U_{m,0}, \ldots, U_{m,w-1}$, where $U_{i,0}, \ldots, U_{i,w-1}$ is the "crowd" drawn by U_i. We will call each column, $U_{1,z}, \ldots, U_{m,z}$, a "handshaking m-tuple of users" where $0 \le z \le w-1$.
   (b) For $z = 0$ to $w-1$,
      i. Choose $\delta_{i,z} \in_R \{0,1\}^{\kappa_1}$.
      ii. For every $\tau \in \{1, \ldots, m\} \setminus \{i\}$, encrypt $\delta_{i,z}$ using $pk_{U_{\tau,z}}$ (certified via $Cert_{U_{\tau,z}}$) to obtain $\alpha_{i,\tau,z} = \mathsf{pEnc}(pk_{U_{\tau,z}}, \delta_{i,z})$.
   (c) Broadcast $\Gamma_i = \{\alpha_{U_{i,1,z}}, \ldots, \alpha_{U_{i,i-1,z}}, \alpha_{U_{i,i+1,z}}, \ldots, \alpha_{U_{i,m,z}}\}_{z=0}^{w-1}$.
5. Phase-III protocol for U_i $(1 \le i \le m)$: At this point, U_i (i.e., U_{i,j_i} for some $0 \le j_i \le w-1$) has received $(m-1)$ ciphertexts $(\alpha_{U_{1,i,j_i}}, \ldots, \alpha_{U_{i-1,i,j_i}}, \ldots, \alpha_{U_{i+1,i,j_i}}, \ldots, \alpha_{U_{m,i,j_i}})$. Then it decrypts them to obtain the corresponding plaintexts $\delta_{1,j_i}, \ldots, \delta_{i-1,j_i}, \delta_{i+1,j_i}, \ldots, \delta_{m,j_i}$.
   (a) Compute $\pi_{i,j_i} = \oplus_{\tau=1}^{m} \delta_{\tau,j_i}$. (Note that δ_{i,j_i} is selected by and known to U_i.) For every $z \in \{0, \ldots, w-1\} \setminus \{j_i\}$, set $\pi_{i,z}$ to be an element uniformly selected from $\{0,1\}^{\kappa_1}$.

(b) For $z = 0$ to $w-1$, compute $\sigma_{i,z} = f_{g_{\pi_{i,z}}(1)}(i, z, \Gamma_1, \ldots, \Gamma_m)$. Broadcast $(\sigma_{i,0}, \ldots, \sigma_{i,w-1})$.

(c) For every $\tau \in \{1, \ldots, m\} \setminus \{i\}$: If $\sigma_{\tau,j_i} = f_{g_{\pi_{i,j_i}}(1)}(\tau, j_i, \Gamma_1, \ldots, \Gamma_m)$, this is a successful preliminary handshake; otherwise, it is not.

6. Phase-IV protocol for U_i $(1 \le i \le m)$:

Case I: The preliminary handshake is successful. For every $\tau \in \{1, \ldots, m\} \setminus \{i\}$, choose $\delta'_{i,\tau} \in_R \{0,1\}^{\kappa_1}$ and encrypt it using $pk_{U_{\tau,j_i}}$ (certified via $Cert_{U_{\tau,j_i}}$) to obtain $\alpha'_{i,\tau} = \mathsf{sEnc}(g_{\pi_{i,j_i}}(2), \mathsf{pEnc}(pk_{U_{\tau,j_i}}, \delta'_{i,\tau}))$. Finally, broadcast $\Gamma'_i = (\alpha'_{i,1}, \ldots, \alpha'_{i,i-1}, \alpha'_{i,i+1}, \ldots, \alpha'_{i,m})$.

Case II: The preliminary handshake is not successful. Broadcast $\Gamma'_i = (\alpha'_{i,1}, \ldots, \alpha'_{i,i-1}, \alpha'_{i,i+1}, \ldots, \alpha'_{i,m})$, where $\alpha'_{i,\tau}$ is uniformly selected from the ciphertext space corresponding to sEnc and $\mathsf{pEnc}(pk_{U_{i,\tau}}, \cdot)$.

7. Phase-V protocol for U_i $(1 \le i \le m)$: At this point, U_i (i.e., U_{i,j_i}) has received $(\alpha'_{1,i}, \ldots, \alpha'_{i-1,i}, \alpha'_{i+1,i}, \ldots, \alpha'_{m,i})$.

Case I: The preliminary handshake is successful. Decrypt them to obtain the corresponding plaintexts $\delta'_{1,i}, \ldots, \delta'_{i-1,i}, \delta'_{i+1,i}, \ldots, \delta'_{m,i}$.

(a) For every $\tau \in \{1, \ldots, m\} \setminus \{i\}$

i. For every $b \in \{1, \ldots, m\} \setminus \{i, \tau\}$, encrypt to obtain $\alpha^*_{\tau,i,b} = \mathsf{sEnc}(g_{\pi_{i,j_i}}(2), \mathsf{pEnc}(pk_{U_{b,j_i}}, \delta'_{\tau,i}))$.

(b) Broadcast $\{\{\alpha^*_{\tau,i,b}\}_{b \in \{1,\ldots,m\} \setminus \{i,\tau\}}\}_{\tau \in \{1,\ldots,m\} \setminus \{i\}}$.

Case II: The preliminary handshake is not successful. Then broadcast $\{\{\alpha^*_{\tau,i,b}\}_{b \in \{1,\ldots,m\} \setminus \{i,\tau\}}\}_{\tau \in \{1,\ldots,m\} \setminus \{i\}}$ where $\alpha^*_{\tau,i,b}$ is uniformly selected from the ciphertext space w.r.t. sEnc and $\mathsf{pEnc}(pk_{U_{b,j_i}}, \cdot)$.

8. Phase-VI protocol for U_i $(1 \le i \le m)$: At this point U_i (i.e., U_{i,j_i}) has received $\{\{\alpha^*_{\tau,i,b}\}_{b \in \{1,\ldots,m\} \setminus \{i,\tau\}}\}_{\tau \in \{1,\ldots,m\} \setminus \{i\}}$.

Case I: The preliminary handshake is successful.

(a) Decrypt them to obtain $\Theta' = \{\{\delta'_{\tau,b}\}_{b \in \{1,\ldots,m\} \setminus \{i,\tau\}}\}_{\tau \in \{1,\ldots,m\} \setminus \{i\}}$.

(b) Set $\pi'_i = (\oplus_{\delta' \in \Theta'} \delta') \oplus (\oplus_{\tau \in \{1,\ldots,m\} \setminus \{i\}} \delta'_{i,\tau}) \oplus (\oplus_{\tau \in \{1,\ldots,m\} \setminus \{i\}} \delta'_{\tau,i})$.

(c) Broadcast (i, σ'_i), where $\sigma'_i = f_{\pi'_i}(i, j_i, \Gamma'_1, \ldots, \Gamma'_m)$.

(d) For every $\tau \in \{1, \ldots, m\} \setminus \{i\}$: If $\sigma'_\tau = f_{\pi'_i}(\tau, j_i, \Gamma'_1, \ldots, \Gamma'_m)$, this is a successful handshake with m distinct parties; otherwise, it is not with m distinct parties.

Case II: The preliminary handshake is not successful. Broadcast (i, σ'_i) where σ'_i is randomly chosen from the range of $\{f_k\}$, and then quit.

Security: Proof of the following theorem is left to the full paper [28].

Theorem 1. *Assume that the public key cryptosystems, the signature schemes, and the pseudorandom functions are secure as specified in Section 2. Then the above scheme is a secure k-anonymous secret handshake scheme in the random oracle model.*

Efficiency Analysis and Improvement: The handshake protocol itself has 6 round-trips excluding the exchange of the plaintext nonces $n_1, \ldots, n_m$, which could have been done before the handshake protocol is executed (e.g., when parties exchange their cipher suits). We only consider the complexity involving

public key cryptography (which dominates the overall cost). Each party needs to verify $O(mw)$ signatures of the public key certificates (which only corresponds to a naive implementation), execute $O(m^2)$ encryptions if the ciphertext spaces can be more efficiently sampled than conducting encryption as is the case with the known notion of dense encryption functions, (otherwise $O(mw)$ encryptions suffice), and $O(m^2)$ decryptions. Each party needs to send $O(mw)$ ciphertexts.

Note that the performance can be improved [28].

5 Symmetric Key Cryptosystems based Construction

As in the case of two-party case of [27], we notice that the basic idea underlying the above multi-party secret handshake scheme based on public key cryptosystems also applies to the setting of symmetric key cryptosystems. In this section we sketch such a scheme. This scheme is based on the key pre-distribution scheme of [7, 6] which we now briefly review: A trusted third party chooses a bivariate symmetric polynomial $poly(x, y)$ of degree t over $\mathbb{F}_q$, where t is the tolerated number of corrupt users. A user with a unique identity i holds $poly(i, y)$. Then two users i and j can derive a common secret $poly(i, j) = poly(j, i)$.

The k-anonymous multi-party secret handshake scheme based on symmetric key cryptosystems is analogous to the one based on public key cryptosystems (i.e., each group is managed by a trusted third party or TTP just as in the key pre-distribution case, the users utilize the rSelect and mSelect to form crowds, no TTP is involved in a secret handshake, etc.), except that a TTP may also adopt a CRL to publish the identities that have been revoked.

6 Conclusion

We presented two multi-party secret handshake schemes: one detailed construction based on public key cryptosystems and one sketched construction based on symmetric key cryptosystems. The added requirements compared to the two party case ([27]) are non-trivial due to new attacks in this setting.

Acknowledgement: This first author was supported in part by ARO, NSF, and UTSA.

References

1. W. Aiello, S. Bellovin, M. Blaze, J. Ioannidis, O. Reingold, R. Canetti, and A. Keromytis. Efficient, dos-resistant, secure key exchange for internet protocols. In *Proc. of ACM CCS'02*, pp 48–58.
2. D. Balfanz, G. Durfee, N. Shankar, D. Smetters, J. Staddon, and H. Wong. Secret handshakes from pairing-based key agreements. In *Proc. of 2003 IEEE Symposium on Security and Privacy.*
3. M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In *Proc. of ASIACRYPT01*, pp 566–582.

4. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proc. ACM CCS'93*, pp 62–73.
5. M. Bellare and P. Rogaway. Optimal asymmetric encryption. In *Proc. of EUROCRYPT'94*, pp 92–111.
6. R. Blom. An optimal class of symmetric key generation systems. In *Proc. of EUROCRYPT'84*, pp 335–338.
7. C. Blundo, A. DeSantis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung. Perfectly-secure key distribution for dynamic conferences. In *Proc. of CRYPTO'92*, pp 471–486.
8. C. Boyd, W. Mao, and K. Paterson. Deniable authenticated key establishment for internet protocols.
9. C. Castelluccia, S. Jarecki, and G. Tsudik. Secret handshakes from ca-oblivious encryption. In *Proc. of ASIACRYPT'04*,pp 293–307.
10. D. Chaum. Blind signatures for untraceable payments. In *Proc. of CRYPTO'82*, pp 199–203.
11. D. Chaum and E. Van Heyst. Group signatures. In *Proc. of Eurocrypt'91*, pp 257–265.
12. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Proc. of CRYPTO'p4*, pp 174–187.
13. R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *Proc. of CRYPTO'98*, pp 13–25.
14. D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography. In *Proc. of ACM STOC'91*, pp 542–552.
15. E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern. Rsa-oaep is secure under the rsa assumption. In *Proc. of CRYPTO'01*, pp 260–274.
16. O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, 1986.
17. S. Goldwasser, S. Micali, and R. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Computing*, 17(2):281–308, 1988.
18. D. Harkins and D. Carrel. *RFC 2409: The Internet Key Exchange (IKE)*. Internet Activities Board, 1998.
19. S. Jarecki and J. Kim and G. Tsudik. Authentication for Paranoids: Multi-Party Secret Handshakes. In *Proc. of ACNS'06*.
20. J. Katz and M. Yung. Complete characterization of security notions for probabilistic private-key encryption. In *Proc. of ACM STOC'00*, pp 245–254.
21. H. Krawczyk. Sigma: The 'sign-and-mac' approach to authenticated diffie-hellman and its use in the ike-protocols. In *Proc. of CRYPTO'03*, pp 400–425.
22. M. Naor. Deniable ring authentication. In *Proc. of CRYPTO'02*, pp 481–498.
23. C. Rackoff and D. R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Proc. of CRYPTO'91*, pp 433–444.
24. R. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *Proc. of ASIACRYPT'01*, pp 552–565.
25. L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):557–570, 2002.
26. G. Tsudik and S. Xu. A flexible framework for secret handshakes. In *Proc. of PET'06* (a one-page abstract appeared in ACM PODC'05).
27. S. Xu and M. Yung. k-anonymous secret handshakes with reusable credentials. In *Proc. ACM CCS'04*, pp 158–167.
28. S. Xu and M. Yung. k-anonymous multi-party secret handshakes. Full version of the present paper available at `http://www.cs.utsa.edu/~shxu`, 2007.