Secret Handshakes from CA-Oblivious Encryption

Claude Castelluccia, Stanisław Jarecki and Gene Tsudik
School of Information and Computer Science,
UC Irvine, CA 92697, USA
{ccastell,stasio,gts}@ics.uci.edu

Abstract

Secret handshake protocols were recently introduced [1] to allow members of the same group to authenticate each other *secretly*, in the sense that someone who is *not* a group member cannot tell, by engaging some party in the handshake protocol, whether that party is a member of this group. On the other hand, any two parties who *are* members of the same group will recognize each other as members. Thus, a secret handshake protocol can be used in any scenario where group members need to identify each other without revealing their group affiliations to outsiders.

The work of [1] constructed a secret handshake protocol secure under the *Bilinear Diffie-Hellman* (BDH) assumption in the Random Oracle Model (ROM). We show how to build secret handshake protocols secure under more standard cryptographic assumptions, using a novel tool of *CA-oblivious* public key encryption, which is an encryption scheme s.t. neither the public key nor the ciphertext reveal any information about the Certification Authority (CA) which certified the public key. We construct CA-oblivious encryptions and handshake schemes based (in ROM) on either the Computational Diffie-Hellman or the RSA assumption.
keywords: secret handshake, authentication, privacy, anonymity, encryption

1 Introduction

Problem Exposition. A secret handshake scheme, introduced by Balfanz et al. [1], allows two members of the same group to identify each other *secretly*, in the sense that each party reveals his/her affiliation to the other only if the other party is also a group member. For example, a CIA agent Alice might want to authenticate herself to Bob, but only if Bob is also a CIA agent. Moreover, if Bob is *not* a CIA agent, the protocol should not help Bob in determining whether Alice is a CIA agent or not. This secrecy property can be extended to ensure that group members's affiliations are revealed only to members who hold specific *roles* in the group. For example, Alice might want to authenticate herself as a CIA agent with security level one if *and only if* Bob is a CIA agent with security clearance two, and vice versa.

In other words, if A is a member of group G_a with role r_a and B is a member of G_b with role r_b, a secret handshake scheme guarantees the following [1]:

- A and B authenticate each other if and only if $G_a = G_b$.[1]
- If $G_a \neq G_b$ then the only thing that either party learns is *the sole fact* that $G_a \neq G_b$.
- A can choose not to reveal anything about herself unless B is a member with particular role r_b (and vice versa).[2]
- An eavesdropper or a man in the middle learn nothing from the protocol.

[1]However, as noted by [1], a handshake protocol cannot be *fair* in the sense that if $G_a = G_b$ then one party is going to learn about it first and could abort the protocol and thus withold their group affiliation from the counterparty.

[2]To simplify the presentation, we will ignore roles for most of the paper. However, as we show in appendix C.1, they can be easily handled by our schemes.

As observed in [1], secret handshakes seem to require **new cryptographic protocols** since they can not be easily obtained from existing tools in the "cryptographic toolbox". For example,[3] group signatures [2, 3] might appear to be an attractive building block for secret handshakes. However, they offer anonymity and unlinkability of group members' signatures, not secrecy of membership itself. In the interactive variant of group signatures, called *identity escrow* [4], one party can prove to another its membership in a group in an anonymous fashion. However, what turns out to be quite difficult is the seemingly simple issue of two parties proving group membership to each other simultaneously, in such a way that one party never reveals its group membership to another unless the former is also a member of the same group.

Secret Handshake Scheme as a "CA-oblivious PKI". To be usable in practice, a secret handshake scheme must provide efficient revocation of any group member by the Group Authority (GA) which administers the group. To support this functionality we will consider secret handshake schemes which look very much like PKI's (Public Key Infrastructures), where the role of a group authority corresponds to that of a Certification Authority (CA) in a PKI.[4] Namely, to become a member of a group a party needs the GA to issue a certificate on an ID bitstring which the CA agrees to assign to this party. The certificate must include a CA-specific *trapdoor* which corresponds to this ID.[5] To revoke some party, the CA puts that party's ID on a revocation list. To perform a handshake, two parties first exchange their ID's, and then proceed only if the ID of the other party is not on the revocation list of their CA. Since the secret handshake protocol must hide one's group affiliation from outsiders, the IDs will be random strings picked from the same domain by all the CA's.[6]

In this setting, constructing a secret handshake scheme amounts to solving the following protocol problem: For a given CA, Alice wants to prove to Bob that she posseses a trapdoor t_A issued by this CA on her ID_A, but only if Bob posseses a trapdoor t_B issued by *the same* CA on his ID_B (and vice versa). Moreover, the protocol must be "CA-oblivious" in the sense that if a cheating Bob is not in the group administered by a given CA, and hence does not hold a CA-specific trapdoor t_B associated with ID_B, then his interaction with Alice must not help him in guessing if Alice belongs to this group or not. (And vice versa for an honest Bob and a cheating Alice.) While this protocol problem can be solved in principle with general 2-party secure computation techniques, the issue remains whether it can be solved with a *practical* protocol, at a cost comparable to standard authentication protocols.

Existing solution based on bilinear maps. The secret handshake protocol of [1] is based on bilinear maps, which can be constructed using Weil pairings on elliptic curves [5, 6]. The protocol of [1] builds on the non-interactive key-agreement scheme of [7], and works as follows. As in the identity based encryption scheme of [8], A and B can compute each other's public keys from each other's ID's and from the public parameters associated with the CA. If Alice is a group member, she can use her trapdoor t_A corresponding to PK_A to non-interactively compute a session key from (t_A, PK_B). Similarly, if Bob is a group member he can compute *the same* session key from (t_B, PK_A). The two parties can then verify if they computed the same key via a standard MAC-based challenge-response protocol. Under the *Bilinear Diffie-Hellman* (BDH) assumption, it is easy to show (in the Random Oracle Model) that an attacker who does not hold the correct trapdoor cannot compute the session

[3] See [1] for a discussion of the unsuitability of other seemingly related constructs. See also Related Work below.

[4] The SH scheme of [1] also falls in this category.

[5] For example, in an identity based encryption scheme, the trapdoor is a secret key corresponding to the public key which can be recovered from ID and the public parameters associated with the CA. In a standard PKI system, this correspondence has an added level of indirection: The trapdoor t is a secret key corresponding to the public key PK which is in turn bound to the ID string by a signature of CA on the $(ID|PK)$ pair.

[6] Optionally, to make protocol runs executed by the same party *unlinkable*, Balfanz et al. [1] propose that a single ID and its certificate could be used in only one instance of the handshake protocol. The CA would then issue multiple (ID,certificate) pairs to each member.

key. Moroever, the MAC-based challenge response confirmation protocol has the needed property that without the knowledge of the key, one learns nothing from the counterparty's responses.

Thus, the "CA-obliviousness" property of the protocol of [1] follows from two properties of cryptosystems built on bilinear maps: (1) that the receiver's public key can be recovered by the sender from the receiver's ID, and thus the receiver does not need to send any information revealing his CA affiliation to the sender, and (2) knowing their public keys, the two parties can establish a session key non-interactively, and thus they again do not reveal any CA-specific information. Given that the first property relies on identity based encryption, and that the only practical IBE known so far is based on bilinear maps [8],[7] it seems that BDH is indeed needed for secret handshakes.

Our contributions. In this paper we show that efficient secret handshake (SH) schemes can be built using weaker and more standard assumptions than the BDH, namely the *Computational Diffie Hellman* (CDH) and the RSA assumptions, in the so-called Random Oracle Model (ROM).

First, we generalize the IBE-based secret handshake solution described above by showing that an efficient four-rounds secret handshake protocol can be built using any *PKI-enabled* encryption with the additional property of *CA-obliviousness*. We define the notion of (chosen-*plaintext* secure) PKI-enabled encryption, which generalizes both the Identity Based Encryption schemes, and the standard encryption schemes used in the context of a PKI system like X.509. We define the CA-obliviousness property for a PKI-enabled encryption, which says that both the public-key-related information which the receiver provides to the sender, and the ciphertext sent from the sender to the receiver, do not reveal which CA issued the receiver's certificate. We then show that every CA-oblivious PKI-enabled encryption leads to a four-round secret handshake protocol whose cost is one decryption and one encryption for each party.

Next, we combine ElGamal encryption and Schnorr signatures to construct a practical CA-oblivious PKI-enabled encryption secure under the CDH assumption (in ROM), which thus leads to secret handshakes secure under CDH. We also note that the recent "oblivious envelope" construction of [10], which is secure under the RSA assumption (in ROM), can be seen as a secure PKI-enabled encryption, and that with some minor changes it can also be made CA-oblivious and can thus be used to build an RSA-based SH scheme.

Both SH schemes constructed in this manner take four rounds, compared to three rounds in the SH scheme of [1]. However, we show that our CDH-based construction can be simplified to run in three rounds. Moreover, its computational cost is roughly **three to five** times lower than that of the BDH-based SH scheme of [1].

Finally, we show that our SH schemes handle roles just as easily as the SH of [1]. We also show that our CDH-based SH schemes can support "blinded" issuance of the member certificates in the sense that the CA does not learn the trapdoors included in the certificate, and thus, in contrast to the BDH-based SH scheme of [1], the CA cannot impersonate that member.

Related Work. As described in [1], existing anonymity tools such as anonymous credentials, group signatures, matchmaking protocols, or accumulators, have different goals than secret handshakes, and it is indeed unclear how to achieve a secret handshake scheme from any of them. However, the recent work of [10] proposes a notion of "oblivious signature-based envelopes", which is closely related to the secret handshake problem. In fact, the oblivious envelope notion they define is equivalent to a PKI-enabled encryption with *half* of our CA-obliviousness property. Namely, they only require that the identity of the CA is not revealed by the receiver-related information the sender needs to encrypt, while we also require the ciphertext produced by the sender to hide the identity of the CA.

[7]The IBE scheme of [9], whose security rests on the quadratic residuosity decisional assumption, needs one modular group element to encode a single bit of the plaintext. Moreover, we do not know how to establish a shared session key in a CA-oblivious way using this IBE scheme.

The one-sided CA-obliviousness turns out to be enough for the application considered in [10], in which two parties in a regular PKI system announce to each other the cleartexts of their public key certificates, which include the identities of the CA's which signed them, but do not want to reveal to one another the CA's *signatures* on their certificates unless the other party posseses the corresponding signature too. They thus offer a lower level of secrecy protection than a secret handshake would, since, in their protocol a party announces what CA issued its certificate, and they only want to hide the signature which *proves* the possession of this certificate. Nevertheless, the RSA-based encryption scheme given in [10] can be modified to provide the CA-obliviousness in both directions, and thus by our general transformation, it leads to an RSA-based Sescret Handshake scheme.

Organization. In section 2 we revise the definitions of an SH scheme [1], restricting them to "PKI-like" SH schemes we consider here. In section 3 we define the notion of a *PKI-enabled encryption*, and the *CA-obliviousness* property of such encryption. In section 4 we give a general construction of an SH scheme from any CA-oblivious encryption. In section 5 we construct a CA-oblivious encryption secure under CDH in ROM, and we show how to adapt the RSA-based encryption of [10] to make it CA-oblivious. In appendix B we show how to simplify the CDH-based SH scheme so that it is about 3 to 5 times more efficient than the BDH-based SH scheme of [1]. In appendix C we show how to support roles and blinded issuing of CA certificates.

2 Secret Handshake Definition

We adapt the definition of a secure Secret Handshake [SH] scheme from [1] to what we call "PKI-like" SH schemes. Our definitions might potentially restrict the notion of a Handshake Scheme, but both the SH schemes of [1] and ours fall into this category. An SH scheme is a tuple of probabilistic algorithms Setup, CreateGroup, AddMember, and Handshake s.t.

- Setup is an algorithm executed publicly on the high-enough security parameter k, to generate the public parameters params common to all subsequently generated groups.
- CreateGroup is a key generation algorithm executed by a GA, which, on input of params, outputs the group public key G, and the GA's private key t_G.
- AddMember is a protocol executed between a group member and the GA on GA's input t_G and shared inputs: params, G, and the bitstring ID (called a *pseudonym* in [1]) of size regulated by params. The group member's private output is the trapdoor t produced by GA for the above ID.
- Handshake is the authentication protocol, i.e. the SH protocol itself, executed between players A, B on public input ID_A, ID_B, and params. The private input of A is (t_A, G_A) and the private input of B is (t_B, G_B). The output of the protocol for either party is either a *reject* or *accept*.

We note that AddMember can be executed multiple times for the same group member, resulting in multiple (ID, t) authentication tokens for that member. We also note that in all the SH schemes discussed here the output of the Handshake protocol can be extended to include an authenticated session key along with the "accept" decision.

2.1 Basic Security Properties

A secure SH scheme must be complete, impersonator resistant, and detector resistant:[8]

[8]Once we restrict the notion of SH Schemes to the PKI-like SH schemes, the security properties defined originally in [1] can be stated in a simpler way. Specifically, their properties of *impersonator resistance* and *impersonator tracing* are subsumed by our *impersonator resistance*, and their *detector resistance and tracing* is subsumed by what we call *detector resistance*.

Completeness. If two honest memebers A, B of the same group engage in $\mathsf{Handshake}$ with valid trapdoors t_A, t_B generated for their ID strings ID_A, ID_B and for the same group $G_A = G_B$, then both parties output "accept" at the end of the protocol.

Impersonator Resistance. Intuitively, the impersonator resistance property is violated if an honest party V who is a member of group G authenticates an adversary $\mathcal{A}$ as a group member, even though the $\mathcal{A}$ **is not** a member of G. Formally, we say that an SH scheme is *impersonator resistant* if every polynomially bounded adversary $\mathcal{A}$ has negligible probability of winning in the following game, for *any* string ID_V which models the ID string of the victim in the impersonation attack:

1. The Setup and the $\mathsf{CreateGroup}$ algorithms are executed: $\mathsf{params} \leftarrow \mathsf{Setup}(1^k)$, $(G, t_G) \leftarrow \mathsf{CreateGroup}(\mathsf{params})$.

2. $\mathcal{A}$, on input (G, ID_V), invokes the $\mathsf{AddMember}$ algorithm on any number of group members ID_i of his choice. (The GA's inputs are ID_i's, G, and t_G.)

3. $\mathcal{A}$ announces a new $ID_{\mathcal{A}}$ string, different from all the ID_i's above. (This models a situation where the ID_i's belong to group members who are malicious but who might be revoked.)

4. $\mathcal{A}$ interacts with the honest player V in the $\mathsf{Handshake}$ protocol, on common inputs $(ID_{\mathcal{A}}, ID_V)$, and on V's private inputs G and t_V, where $t_V \leftarrow \mathsf{AddMember}((G, ID_V), t_G)$.

We say that $\mathcal{A}$ *wins* if V outputs "accept" in the above $\mathsf{Handshake}$ instance.

We note that stronger versions of the impersonator resistance property are possible. For example, the attacker can be allowed to ask for trapdoors on additional $ID_i \neq ID_A$ strings *during* the $\mathsf{Handshake}$ protocol against the intended victim V. Also, the attacker can be allowed to have several attempts against V and be able to ask for additional trapdoors after each attempt, before he announces that he is ready for the true challenge. We adopt the simpler definition here to reduce the level of formalism, although the schemes we propose remain secure under these stronger notions as well.

Detector Resistance. Intuitively, an adversary $\mathcal{A}$ violates the detector resistance property if it can decide whether some honest party V is a member of some group G, even though $\mathcal{A}$ **is not** a member of G. Formally, we say that an SH scheme is *detector resistant* if there exists a probabilistic polynomial-time algorithm SIM, s.t. any polynomially bounded adversary $\mathcal{A}$ cannot distinguish between the following two games with the probability which is non-negligibly higher than 1/2, for *any* target ID string ID_V:

1-3. Steps 1-3 proceed as in the definition of *Impersonator Resistance*, i.e. on input ID_V and a randomly generated G, $\mathcal{A}$ querries GA on adaptively chosen ID_i's and announces some challenge string $ID_{\mathcal{A}}$, $ID_{\mathcal{A}} \neq ID_i$ for all i.

4-1. In game 1, $\mathcal{A}$ interacts with an algorithm for the honest player V in the $\mathsf{Handshake}$ protocol, on common inputs $(ID_{\mathcal{A}}, ID_V)$, and on V's private inputs G and $t_V = \mathsf{AddMember}((G, ID_V), t_G)$.

4-2. In game 2, $\mathcal{A}$ interacts with the SIM algorithm on common inputs $(ID_{\mathcal{A}}, ID_V)$. (Note that SIM has no private inputs.)

5. $\mathcal{A}$ can query GA on additional strings $ID_i \neq ID_{\mathcal{A}}$.

6. $\mathcal{A}$ outputs "1" or "2", making a judgment about which game he participated in.

Again, stronger notions of the detector resistance are possible. However, we adopt this relatively simple definition in order to reduce the level of formalism.

2.2 Other Security Properties

Authenticated Key Exchange. In practice one would like to extend the notion of a secret handshake from mere authentication, where participants' outputs are binary decisions "accept"/"reject", to authenticated key exchange, where parties output instead either "reject" or an identifier of an authenticated session and a session key. Our SH schemes, just like the original SH protocol of [1] are in fact easily extendible to AKE protocols. (Formal arguments of AKE security would require extending the AKE formalism of [11, 12], which is beyond the scope of this paper.)

Group-Affiliation Secrecy against Eavesdroppers. Our schemes also protect secrecy of participants' group affiliations against eavesdroppers, even if the eavesdropper is a malicious member of the same group. An observer of our SH protocols does not even learn if the participants belong to the same group or not. We do not formally define security against eavesdroppers, because it is very similar to the security against active attackers which we do define, the impersonator and detector resistance. Moreover, if the protocol participants first establish a secure anonymous session, e.g. using SSL or IKE, and then run the SH protocol over it, the resulting protocol is trivially secure against eavesdroppers.

Unlinkability. A potentially desirable property identified by Balfanz et al. [1], is *unlinkability*, which extends privacy protection for group members by requiring that instances of the handshake protocol performed by the same party cannot be efficiently linked. This can be achieved trivially (but inefficiently) by issuing to each group member a list of one-time certificates, each issued on a randomly chosen ID, to be discarded after a single use. Unfortunately, an honest member's supply of one-time certificates can be depleted by an active attacker who initiates the handshake protocol enough times. Indeed, while one can run our SH schemes using multiple certifciates to offer some heuristic protections against linking, contructing an efficient and perfectly unlinkable SH scheme remains an open problem.

3 PKI-enabled encryption

We define the notion of *PKI-enabled* encryption, which models the use of standard encryption in the context of a PKI system, and also generalizes Identity Based Encryption. We define *one-way security* for PKI-enabled encryption, adapting a standard (although weak) notion of one-way security of encryption to our context, and we define a novel *CA-obliviousness* property for such schemes.

A PKI-enabled encryption is defined by the following algorithms:

- Initialize is run on a high-enough security parameter, k, to generate the public parameters params common to all subsequently generated Certification Authorities (CAs).
- CAInit is a key generation algorithm executed by a CA. It takes as inputs the system parameters params and returns the public key G and the private key t_G of the CA.
- Certify is a protocol executed between a CA and a user who needs to be certified by this CA. It takes CA's private input t_G, and public inputs G (assume that G encodes params) and string ID which identifies the user, and returns *trapdoor* t and *certificate* ω as the user's outputs.
- Recover is an algorithm used by a *sender*, a party who wants to send an encrypted message to a user identified by some string ID, to recover that user's public key. It takes as inputs G, ID, and ω, and outputs a public key PK.
- Encrypt is the actual encryption algorithm which takes inputs message m and the public key PK (assume that PK encodes params and G), and outputs a ciphertext c.

- Decrypt is the decryption algorithm which takes as inputs the ciphertext c and the trapdoor t (as well as possibly params, G, ID, and ω, all of which can be encoded in t), and returns m.

The above algorithms must satisfy the obvious *correctness* property that the decryption procedure always inverts encryption correctly.

It is easy to see (see Appendix A) that this notion of encryption indeed models both regular encryption schemes in the PKI context as well as the Identity Based encryption schemes.

One-Way Security. We define the security of PKI-enabled encryption only in the relatively weak sense of so-called *one-way* security, namely that the attacker who does not own a trapdoor for some public key cannot decrypt an encryption of a random message. This is a weaker notion than the standard *semantic* security for an encryption, but we adopt it here because (1) it simplifies the definition of security, (2) one-way security is all we need in our construction of a secure SH scheme, and (3) in the Random Oracle Model, it is always possible to convert a one-way secure encryption into a semantically secure encryption, or even a CCA-secure encryption using the method of Fujisaki and Okamoto [13].

The definition of security for PKI-enabled encryption is very simliar to the definition of security of an IBE scheme: We say that a PKI-enabled encryption scheme is *One-Way* (OW) secure on message space $\mathcal{M}$ under *Chosen-Plaintext Attack* (CPA), if every polynomially-bounded adversary $\mathcal{A}$ has only negligible probability of winning the following game:

1. The Initialize and CAInit algorithms are run, and the resulting public key G is given to $\mathcal{A}$.
2. $\mathcal{A}$ repeatedly triggers the Certify protocol under the public key G, on ID strings ID_i of $\mathcal{A}$'s choice. In each instance $\mathcal{A}$ receives (t_i, ω_i) from the CA.
3. $\mathcal{A}$ announces a pair $(ID_{\mathcal{A}}, \omega)$, where $ID_{\mathcal{A}} \neq ID_i$ for all ID_i's querried above.
4. $\mathcal{A}$ receives $c = \mathsf{Encrypt}_{PK}(m)$ for a random message $m \in \mathcal{M}$ and $PK = \mathsf{Recover}(G, ID_{\mathcal{A}}, \omega)$.
5. $\mathcal{A}$ is allowed to trigger the Certify algorithm on new $ID_i \neq ID_{\mathcal{A}}$ strings of his choice, getting additional (t_i, ω_i) pairs from the CA.
6. $\mathcal{A}$ outputs a message m'.

We say that $\mathcal{A}$ *wins* in the above game if $m = m'$.

CA-Obliviousness. Informally, PKI-enabled encryption is CA-oblivious if (1) the receiver's message to the sender, i.e., the pair (ID, ω), hides the identity of the CA which certified this ID; and (2) the sender's messages to the receiver, i.e., ciphertexts, do not leak any information about the CA which the *sender* assumed in computing the receiver's public key. Consequentely, in a standard exchange of messages between the receiver and the sender, neither party can guess which CA is assumed by the other one. Formally, we call a PKI-enabled encryption scheme *CA-oblivious* under two conditions:

(I) It is *receiver CA-oblivious*, i.e., if there exists a probabilistic polynomial-time algorithm $SIM_{(R)}$, s.t. no polynomially-bounded adversary $\mathcal{A}$ can distinguish between the following two games with probability non-negligibly higher than 1/2, for *any* target ID string ID_R:

1. The Initialize and CAInit algorithms are executed, and the resulting parameters params and the public key G is given to $\mathcal{A}$.
2. $\mathcal{A}$ can trigger the Certify protocol on any number of ID_i's.

3-1. In game 1, $\mathcal{A}$ gets (ID_R, ω_R), where ω_R is output by the $\mathsf{Certify}$ protocol on G and ID_R.

3-2. In game 2, $\mathcal{A}$ gets (ID_R, r) where $r = SIM_{(R)}(\mathsf{params})$.

4. $\mathcal{A}$ can trigger the $\mathsf{Certify}$ protocol some more on any $ID_i \neq ID_R$.

5. $\mathcal{A}$ outputs "1" or "2", making a judgment about which game he participated in.

(II) It is *sender CA-oblivious*, i.e., if there exists a probabilistic polynomial-time algorithm $SIM_{(S)}$ s.t. no polynomially-bounded adversary $\mathcal{A}$ can distinguish between the following two games, with probability non-negligibly higher than 1/2:

1. The $\mathsf{Initialize}$ and CAInit algorithms are executed, and the resulting parameters params and the public key G is given to $\mathcal{A}$.

2. $\mathcal{A}$ can trigger the $\mathsf{Certify}$ protocol any number of times, for public key G and group members ID_i's of $\mathcal{A}$'s choice.

3. $\mathcal{A}$ announces pair (ID_R, ω_R) on which he wants to be tested, where $ID_R \neq ID_i$ for all i.

4-1. In game 1, $\mathcal{A}$ gets $c = \mathsf{Encrypt}_{PK_R}(m)$ for random $m \in \mathcal{M}$ and $PK_R = \mathsf{Recover}(G, ID_R, \omega_R)$.

4-2. In game 2, $\mathcal{A}$ gets $c = SIM_{(S)}(\mathsf{params})$.

5. $\mathcal{A}$ can query GA on some more ID_i's s.t. $\forall_i, ID_i \neq ID_R$.

5. $\mathcal{A}$ outputs "1" or "2", making a judgment about which game he participated in.

4 Secret Handshakes from CA-Oblivious Encryption

We show how to built a secure SH scheme using CA-oblivious PKI-enabled encryption. Given a CA-oblivious one-way secure PKI-enabled encryption scheme ($\mathsf{Initialize}$, CAInit, $\mathsf{Certify}$, $\mathsf{Recover}$, $\mathsf{Encrypt}$, $\mathsf{Decrypt}$), and a hash function $H : \{0,1\}^* \rightarrow \{0,1\}^k$ modeled as a random oracle, we specify a secret handshake scheme as follows: Algorithms Setup, $\mathsf{CreateGroup}$, and $\mathsf{AddMember}$, are simply set to $\mathsf{Initialize}$, CAInit, and $\mathsf{Certify}$, respectively, while algorithm $\mathsf{Handshake}$ proceeds as follows. A's inputs are (ID_a, ω_a, t_a) and B's inputs are (ID_b, ω_b, t_b).[9]

1. SH-1 ($A \longleftarrow B$): ID_b, ω_b

   A does the following:

   a) obtains $PK_b = \mathsf{Recover}(G, ID_b, \omega_b)$

   b) picks $r_a \leftarrow \mathcal{M}$ and $ch_a \leftarrow \{0,1\}^k$

   c) computes $C_a = \mathsf{Encrypt}_{PK_b}(r_a)$

2. SH-2 ($A \longrightarrow B$): ID_a, ω_a, C_a, ch_a

   B does the following:

   a) obtains $PK_a = \mathsf{Recover}(G, ID_a, \omega_a)$

   b) obtains $r_a = \mathsf{Decrypt}_{t_b}(C_a)$

   c) picks $r_b \leftarrow \mathcal{M}$ and $ch_b \leftarrow \{0,1\}^k$

   d) computes $C_b = \mathsf{Encrypt}_{PK_a}(r_b)$

   e) computes $resp_b = H(r_a, r_b, ch_a)$

[9] Group member's trapdoor on string ID in this SH scheme is a *pair* (ω, t) produced by the $\mathsf{Certify}$ protocol. We can also assume that (ID_a, ID_b) are public inputs.

3. SH-3 ($A \longleftarrow B$): C_b, $resp_b$, ch_b

   A does the following:

   a) obtains $r_b = \mathsf{Decrypt}_{t_a}(C_b)$

   b) if $resp_b \neq H(r_a, r_b, ch_a)$, outputs FAIL; otherwise outputs ACCEPT.

   c) computes $resp_a = H(r_a, r_b, ch_b)$

4. SH-4 ($A \longrightarrow B$): $resp_a$

   B does the following:

   a) if $resp_a \neq H(r_a, r_b, ch_b)$, outputs FAIL; otherwise outputs ACCEPT.

We note that the above protocol can be easily turned into an Authenticated Key Exchange (AKE) protocol (secure in the ROM model) if the two parties compute their authenticated session key as $K = H(r_a, r_b)$.

Theorem 1 *If the PKI-enabled encryption is CA-oblivious and One-Way (Chosen-Plaintext Attack) Secure, the above construction yields a Secret Handshake scheme secure in the Random Oracle Model (ROM).*

Proof of Impersonator Resistance (sketch): Assume that $\mathcal{A}$ violates with non-negligible probability ϵ the impersonator resistance property against some honest member V identified by ID_V. Assume that $\mathcal{A}$ plays the role of A and V plays the role of B (the other case is easier because B has to speak first). Therefore with prob. ϵ, $\mathcal{A}$ sends a valid $resp_a = H(r_a, r_b, ch_b)$ response to B. In the ROM model, that can happen with non-negligible probability only if $\mathcal{A}$ querries the oracle for $H(\cdot)$ on the input (r_a, r_b, ch_b) s.t., in particular, r_b was the value picked by V and sent to $\mathcal{A}$ in the form of a ciphertext $C_b = \mathsf{Encrypt}_{PK_a}(r_b)$ for $PK_a = \mathsf{Recover}(G, ID_a, \omega_a)$, where (ID_a, ω_a) are sent by $\mathcal{A}$ in its first message to V. Therefore, in ROM, we can use $\mathcal{A}$ to create a break $\mathcal{A}'$ against the one-way security of the encryption scheme:

On input G, $\mathcal{A}'$ passes the public key G to $\mathcal{A}$. When $\mathcal{A}$ can makes a querry ID_i, so does $\mathcal{A}'$, passing back (t_i, ω_i) to $\mathcal{A}$. When $\mathcal{A}$ announces that he is ready for the impersonation challenge against V, $\mathcal{A}'$ passes as *his* encryption challenge the pair (ID_a, ω_a) sent by $\mathcal{A}$ in his first message to V. On encryption challenge $c = \mathsf{Encrypt}_{PK_a}(m)$ where m is chosen at random in $\mathcal{M}$, $\mathcal{A}'$ passes the same challenge as its response $C_b = c$ to $\mathcal{A}$, together with a random challenge value ch_b and $resp_b$ picked at random. The only way $\mathcal{A}$ can tell between this communication and a conversation with an honest V is by querying H on (r_a, r_b, ch_a) for $r_b = \mathsf{Decrypt}_{t_a}(C_b) = m$. Otherwise, as we argued above, he queries H on (r_a, r_b, ch_b) with probability almost ϵ. In either case, since $\mathcal{A}$ can make only polynomially-many queries to H, $\mathcal{A}'$ can pick one such query at random, and $\mathcal{A}'$ will have a non-negligible chance of outputing $r_b = m$. Thus $\mathcal{A}'$ breaks the one-wayness of the encryption scheme. □

Proof of Detector Resistance (sketch): We will show a simulator SIM s.t. if $\mathcal{A}$ distinguishes between interactions with SIM and interactions with a group member, we can break the one-way security of the encryption scheme. Assume again that the adversary $\mathcal{A}$ plays the role of A and V plays the role of B. Assume that the underlying encryption scheme is CA-oblivious, and therefore there exist simulators $SIM_{(S)}$ and $SIM_{(R)}$ which satisfy the two CA-obliviousness criteria. We define a simulator SIM, running on input $(ID_A, ID_V, \mathsf{params})$, as follows: (1) To simulate V's first message SH-1, SIM sends $ID_b = ID_V$ together with $\omega_b = SIM_{(R)}(\mathsf{params})$, (2) To simulate B's second message SH-3, SIM sends $resp_b$ and ch_b picked at random, and $C_b = SIM_{(S)}(\mathsf{params})$.

If $\mathcal{A}$ can distinguish a conversation with such SIM from a conversation with a true group member V, then by a standard hybrid argument, since the $SIM_{(S)}$ and $SIM_{(R)}$ simulators produce messages which are indistinguishable from the messages of an honest B, it must be that $\mathcal{A}$ distinguishes random values $resp_b$ chosen by SIM from values $resp_b = H(r_a, r_b, ch_a)$ computed by a real player. But this can happen only if $\mathcal{A}$ makes an oracle query on the triple (r_a, r_b, ch_a), in which case we can use $\mathcal{A}$, exactly in the same manner as we did in the proof of impersonator resistance, to attack the one-way security of the underlying encryption scheme. □

5 CA-Oblivious Encryption Schemes: Constructions

We show two constructions for CA-oblivious PKI-enabled encryption schemes, based on the CDH and RSA assumptions, respectively, all in the Random Oracle Model. The CDH-based scheme is novel, while the RSA-based scheme is a simple modification of the RSA-based encryption envelope proposed in [10]. By theorem 1, both constructions lead to secret handshake schemes secure under the respective assumptions.[10]

5.1 CDH-based encryption scheme

- Initialize picks the standard discrete logarithm parameters (p, q, g) of security k, i.e., primes p, q of size polynomial in k, s.t. g is a generator of a subgroup in $\mathbb{Z}_p^*$ of order q. Initialize also defines hash functions $H : \{0,1\}^* \to \mathbb{Z}_q$ and $H' : \{0,1\}^* \to \{0,1\}^k$. (Both hash functions are modeled as random oracles, but we note that H' is not essential in this construction and can be easily removed.)

- CAInit picks random private key $x \in \mathbb{Z}_q$ and public key $y = g^x \bmod p$.

- In Certify on public inputs (y, ID), the CA computes the Schnorr signature on string ID under the key y [14], i.e., a pair $(\omega, t) \in (\mathbb{Z}_p^*, \mathbb{Z}_q)$ s.t. $g^t = \omega y^{H(\omega, ID)} \bmod p$. The user's outputs are the trapdoor t and the certificate ω. The signature is computed as $\omega = g^r \bmod p$ and $t = r + xH(\omega, ID) \bmod q$, for random $r \leftarrow \mathbb{Z}_q$.

- Recover(y, ID, ω) outputs $PK = \omega y^{H(\omega, ID)} \bmod p$.

- Encrypt$_{PK}(m)$ is an ElGamal encryption of message $m \in \{0,1\}^k$ under the public key PK: It outputs a ciphertext $[c_1, c_2] = [g^r \bmod p, m \oplus H'(PK^r \bmod p)]$, for random $r \in \mathbb{Z}_q$.

- Decrypt is an ElGamal decryption, outputing $m = c_2 \oplus H'(c_1^t \bmod p)$.

Theorem 2 *The above encryption scheme is CA-oblivious and One-Way (CPA) Secure under the CDH assumption in the Random Oracle Model.*

Proof of One-Way Security (sketch): Assume that an adversary $\mathcal{A}$ breaks one-wayness of this encryption scheme. Then $\mathcal{A}$ receives n Schnorr signatures (t_i, w_i) on ID_i's of his choice. Then $\mathcal{A}$ sends some tuple (ID, w) for $ID \neq ID_i$ for all the above ID_i's. If $\mathcal{A}$ breaks one-wayness then it must be computing $c_1^t \bmod p$ where $g^t = \omega y^{H(\omega, ID)} \bmod p$.

Therefore, if $\mathcal{A}$ succeeds, then she can exponentiate a random element c_1 to t. Hence, what we need to argue that, even though $\mathcal{A}$ receives n signatures (t_i, w_i) on her ID_i's, she cannot produce a new pair (ID, w) s.t. she can exponentiate a random elements c_1 to exponent t where $g^t = w * y^{h(t, ID)}$. Now, this is very similar to proving the chosen message attack security of the underlying Schnorr signature scheme, where one argues that, after receiving n signatures, $\mathcal{A}$ cannot produce a new triple

[10] We point out that the Identity Based Encryption scheme of [8] is also a CA-oblivious PKI-based encryption scheme, and therefore our general SH construction applied to that encryption scheme leads to another Weil-pairing based SH scheme.

(ID, t, w) s.t. $g^t = w * y^{h(w,ID)}$. Hence, our proof is very similar to the forking-lemma proof for Schnorr signature security in [15]. However, here we reduce the successful attack not to computing discrete logarithm, but to breaking the CDH assumption by computing m^x on input $y = g^x$ and a random value m.

To reduce $\mathcal{A}$'s ability to succeed in this protocol to computing m^x on the Diffie-Hellman challenge (g, g^x, m), we first simulate, as in the proof of Schnorr signature security, the signatures (t_i, w_i) that $\mathcal{A}$ gets on her ID_i's, by taking random t_i, c_i, computing $\omega_i = g^{t_i} * y^{-c_i} \bmod p$, and assigning $H(ID_i, w_i)$ to c_i. Since the verification equation is satisfied and t_i, c_i are picked at random, this is indistinguishable from receiving real signatures. As in the forking lemma argument of [15], we can argue that if $\mathcal{A}$'s probability of success is ϵ, the probability that $\mathcal{A}$ executed twice in a row succeeds in *both* executions *and* sends the *same* (ID, w) pair in both of them, is at least ϵ^2/q_h where q_h is the number of queries $\mathcal{A}$ makes to the hash function H (see [15]). The forking lemma used in the security proof of the Schnorr signature scheme shows that if two conversations with an adversary produce triples (ID, t, w) and $(ID, \hat{t}, w)$, where in first conversation $H(ID, r) = c$ and in the second $H(ID, r) = \hat{c}$ for some random $c, \hat{c}$, then $x = DL_g(y)$ can be computed as $x = (s-\hat{s})/(c-\hat{c}) \bmod q$, because $g^t = w*y^c$ and $g^{\hat{t}} = w*y^{\hat{t}}$. By the same forking lemma applied to our case, adversary $\mathcal{A}$ produces two *exponentiations* m^t and $m^{\hat{t}}$, instead of forgeries $t, \hat{t}$, but still we have that $x = DL_g(y) = (t-\hat{t})/(c-\hat{c})$. Therefore, with probability ϵ^2/q_h we can break the CDH challenge and compute $m^x = m^{(t-\hat{t})/(c-\hat{c})} = (m^t/m^{\hat{t}})^{1/(c-\hat{c})} \bmod p$.

Note that if the success probability ϵ is higher than negligible, and if $\mathcal{A}^*$ is an efficient algorithm and hence the number of queries q_h is polynomial, then the probability of CDH break ϵ^2/q_h is non-negligible as well. □

Proof of CA Obliviousness: It is easy to see that w and the ciphertext $C = [c_1, c_2]$ does not reveal any information about the CA i.e., that the proposed CDH-based encryption is CA-oblivious. Since $\omega = g^r$ for random r, ω is independent from CA's public key y, and hence the scheme is receiver CA-oblivious. Ciphertext $C = [c_1, c_2]$ on a random message m is also independent from the group key y, because $c_1 = g^r$ for random r and c_2 is computed by xoring $H'(PK^r)$ with the random m. □

5.2 RSA-based encryption scheme

- Initialize defines k' polynomial in the security parameter k and two hash functions $H : \{0,1\}^* \rightarrow \{0,1\}^{2k'}$ and $H' : \{0,1\}^* \rightarrow \{0,1\}^k$.

- CAInit picks $e = 3$ and an RSA modulus $n = pq$ for $|p| = |q| = k'$, with additional restriction $2^{2k'} < n < 2^{2k'+1}$. The private RSA key is $d = e^{-1} \bmod \phi(n)$.

- Certify computes the RSA signature on ID under the key (d, n), i.e., $s = H(ID)^d \bmod n$. It then picks $t \leftarrow \mathbb{Z}_n$ and computes $\omega = s * (H(ID))^t \bmod n$. For CA-obliviousness, if $\omega > 2^{2k'}$ then another t is chosen at random and ω is recomputed, until $\omega \leq 2^{2k'}$.

- Recover outputs $PK = \omega^e / H(ID) \bmod n$.

- Encrypt is an ElGamal encryption of k-bit message m under the public key PK modulo n. The sender picks $t \in \mathbb{Z}_n$ and computes the ciphertext $[c_1, c_2] = [(h^e)^r \bmod n, m \oplus H'(PK^r \bmod n)]$, where $h = H(ID)$. Again, for CA obliviousness, if $c_1 > 2^{2k'}$ then another r is chosen at random and ciphertext is recomputed, until $c_1 \leq 2^{2k'}$.

- Decrypt returns $m = c_2 \oplus H'(c_1^t \bmod n)$.

Theorem 3 *The above encryption scheme is CA-oblivious and One-Way (CPA) Secure under the RSA assumption in the Random Oracle Model.*

Proof of One-Way Security: When H' is modelled as a random oracle, then an attack against one-wayness of this encryption scheme means that there exists a polynomially bounded adversary $\mathcal{A}$ s.t. $\mathcal{A}$ gets valid pairs (t_i, w_i) on ID_i's of his choice from the CA, announces a pair (w, ID), s.t. $ID_i \neq ID$, and then the challenger randomly picks $r \leftarrow \mathbb{Z}_n$, and sends $c_1 = (h^e)^r$ and some c_2 to $\mathcal{A}$, where $h = H(ID) \mathrm{mod} n$. The adversary can then receive additional (t_i, w_i) pairs on ID_i's of his choice, but finally $\mathcal{A}$ must make a query $PK^r = (w^e/h)^r \bmod n$ to H, because otherwise, in ROM, he has a negligible chance of outputing the correct decryption of $[c_1, c_2]$.

Given an attacker $\mathcal{A}$ that wins the above game with probability ϵ, we can construct another attacker $\mathcal{B}$ that can successfully forge the RSA signature $H(ID)^d \mathrm{mod} n$ with probability ϵ', where $|\epsilon - \epsilon'|$ is negligible. $\mathcal{B}$ does the following: (1) $\mathcal{B}$ computes $H(ID)$, picks a randon value z in $\mathbb{Z}_n$ and sends $h^{(1+ez)}$ to $\mathcal{A}$. Note that $h^{(1+z)} = h^{(ed+ez)} = h^{e(d+z)}$. $\mathcal{A}$ returns $r = w * e^{(d+z)} * h^{-(d+z)}$. $\mathcal{B}$ can then computes $h^d = w * e^{(d+z)} * h^z / r$.

$\mathcal{B}$ succeeds in forging an RSA signature iff $\mathcal{A}$ wins the above game. This happens with probability ϵ'. What $\mathcal{A}$ receives from the Challenger in the first game is the distributed family $\{h^{er} | r \in \mathbb{Z}_n\}$. What $\mathcal{A}$ receives from $\mathcal{B}$ in the second game is the distributed family $\{h^{e(d+z)} | z \in \mathbb{Z}_n\}$. Since these two distribution families are statistically indistinguishable, $|\epsilon - \epsilon'|$ is negligible. $\square$

Proof of CA Obliviousness: It is easy to see that w and the ciphertext $C = [c_1, c_2]$ does not reveal any information about the CA modulus n. If fact since w is choosen such that its value is always smaller than $2^{2k'}$, the distribution families defined by w for two different values of n, n_1 and n_2, are statistically indistinguishable. The same argument can be used for c_1 and c_2. $\square$

References

[1] D. Balfanz, G. Durfee, N. Shankar, D.K. Smetters, J. Staddon, and H.C. Wong, "Secret handshakes from pairing-based key agreements," in *IEEE Symposium on Security and Privacy*, 2003.

[2] D. Chaum and E. Van Heyst, "Group signatures," in *Advances in Cryptography - EUROCRYPT'91*, Springer-Verlag, Ed., 1991, vol. 547, pp. 257–265.

[3] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik, "A practical and provably secure coalition-resistant group signature scheme," in *CRYPTO'2000*, 2000.

[4] J. Kilian and E. Petrank, "Identity escrow," in *Advances in Cryptography - CRYPT0 1998*, Santa Barbara, CA, August 1998.

[5] A. Joux, "The weil and tate pairings as building blocks for public key cryptosystems," in *Proceedings of the 5th International Symposium on Algorithmic Number Theory*, 2002.

[6] Martin Gagne, "Applications of bilinear maps in cryptography," M.S. thesis, University of Waterloo, 2002.

[7] R. Sakai, K. Ohgishi, and M. Kasahara, "Cryptosystems based on pairing," in *Symposium in Cryptography and Information Security*, Okinawa, Japan, January 2000.

[8] D. Boneh and M. Franklin, "Identity based encryption from weil pairing," in *Advances in Cryptography - CRYPT0 2001*, Santa Barbara, CA, August 2001.

[9] C. Cocks, "An identity based encryption scheme based on quadratic residues," in *Eighth IMA International Conference on Cryptography and Coding*, Decembre 2001.

[10] N. Li, W. Du, and D. Boneh, "Oblivious signature-based enevelope," in *Proceedings of 22nd ACM Symposium on Principles of Distributed Computing (PODC 2003)*, Boston, Massachusetts, July 13-16 2003.

[11] R. Canetti and H. Krawczyk, "Universally composable notions of key exchange and secure channels," in *Advances in Cryptology - EUROCRYPT 2002*, 2002.

[12] Shoup V., "On formal models for secure key exchange," Tech. Rep. RZ3120, IBM, April 1999.

[13] E. Fujisaki and T. Okamoto, "Secure integration of asymmetric and symmetric encryption schemes," in *Advances in Cryptology-CRYPTO'99*, August 1999, pp. 537–554.

[14] C. Schnorr, "Efficient identification and signatures for smart cards," in *Advances in Cryptography - CRYPT0 1989*, Santa Barbara, CA, August 1989.

[15] D. Pointcheval and J. Stern, "Security proofs for signatures," in *Eurocrypt'96*, 1996, pp. 387 – 398.

[16] P. Barreto, H. Kim, B. Lynn, and M. Scott, "Efficient algorithms for pairing-based cryptosystems," in *Advances in Cryptography - CRYPT0 2002*, Santa Barbara, CA, August 2002.

A PKI-enabled encryption generalizes both standard PKI and IBE

The notion of the PKI-enabled encryption generalizes both the standard use of encryption in the PKI context and the Identity Based Encryption schemes. An IBE scheme can be seen as a special case of PKI-enabled encryption where certificates ω computed in the Certify protocol are empty strings. Therefore, a given user's public key PK is computable solely from G (CA's public key) and ID (the user's identity).

Similarly, if we want to model the standard PKI setting, the Initialize procedure would simply output as params its input 1^k. This output sets the *minimal* security parameter to be used by any CA participating in this PKI system. The CAInit generates the standard public/private signature keypair (CA, t_{CA}) of the CA. The Certify procedure generates a private/public keypair (Pr, PK) for the user, and outputs the user's trapdoor $t = Pr$ and a certificate $\omega = (PK, sig_{CA}[PK, ID])$, which binds the user's ID to his public key. Then Recover(CA, ID, ω) outputs PK if the signature in ω verifies under public key CA on message $[PK, ID]$. Otherwise Recover outputs null. Encryption and decryption proceed in a standard way.

B More Efficient CDH-based Secret Handshake

This section presents a secure extension of the *CDH-based* SH scheme presented above that uses "CA-oblivious Diffie-Hellman Key Agreement" instead of "CA-oblivious Diffie-Hellman encryption".

This extension improves efficiency so that the scheme has only 3 rounds (instead of 4) and requires only one (multi)exponentiation per player. This results in a SH scheme which has the same number of rounds but requires at least 3 times computationnally more efficient than the *BDH-based* SH scheme of [1]. In fact as shown in [16], for the same level of security, a Weil pairing computation is about 5 to 6 times more expensive than a modular exponentation computation, i.e. at least 3 times more expensive than a modular multi-exponentation.

The faster secret handshake protocol is as follows:

1. Initialize picks the standard discrete logarithm parameters (p, q, g) of security k, i.e., primes p, q of size polynomial in k, s.t. g is a generator of a subgroup in $\mathbb{Z}_p^*$ prime order q. Initialize also defines hash functions $H : \{0,1\}^* \rightarrow \mathbb{Z}_q$ and $H' : \{0,1\}^* \rightarrow \{0,1\}^k$.

2. CAInit picks random private key $x \in \mathbb{Z}_q$ and public key $y = g^x \mathrm{mod} p$.

3. In Certify on public inputs (y, ID), the CA computes the Schnorr signature of ID under the key y [14], i.e., the signature on ID is a pair $(\omega, t) \in (\mathbb{Z}_p^*, \mathbb{Z}_q)$ s.t. $g^t = \omega * y^{H(\omega, ID)} \mathrm{mod} p$. The user's outputs are the trapdoor t and the certificate ω.

4. Recover(y, ID, ω) outputs $PK = \omega * y^{H(\omega, ID)} \mathrm{mod} p$. The message space $\mathcal{M} = \{0,1\}^k$.

5. The Handshake protocol:

   The secret handshake protocol between Alice on inputs t_A, (w_A, ID_A), and y, and Bob on inputs t_B, (w_B, ID_B), and y, proceeds as follows:

   (a) msg1 ($A \longrightarrow B$): w_A, ID_A, N_1
   B computes $M_A = w_A y^{h(w_A, ID_A)}$ (from Recover) and $K_B = M_A^{t_B} \bmod p$, computes $v_B = mac(K_B, \tau)$ where $\tau = (w_B || ID_B || N_2 || msg1)$

   (b) msg2 ($B \longrightarrow A$): w_B, ID_B, N_2, v_B
   $\mathcal{A}$ computes $M_B = w_B y^{h(w_B, ID_B)}$ and $K_A = M_B^{t_A} \bmod p$, $v_A = mac(K_A, \tau')$ where $\tau' = (msg1 || msg2)$, and accepts the authentication protocol if $v_B = mac(K_A, \tau)$.

   (c) msg3 ($A \longrightarrow B$): v_A
   B accepts the authentication protocol if $v_A = mac(K_B, \tau')$

We show that the above scheme achieves the *detector resistance and impersonator resistance* properties under the CDH assumption in the Random Oracle Model, and therefore we have:

Theorem 4 *In the Random Oracle Model, under the CDH assumption, the above secret handshake protocol is* secure.

Intuitively, we show that to remain *secure* a malicious party $\mathcal{A}$ would have to use a $w_{\mathcal{A}}$ which was not issued by the CA. We argue that in this case, the ability of $\mathcal{A}$ to authenticate itself (in the Random Oracle Model) is equivalent to his ability to compute the Diffie-Hellman key $K = M_B^{t_A^*}$ where t_A is defined by $g^{t_A^*} = \mathsf{Recover}(w_A^*, y)$ and $M_B = \mathsf{Recover}(w_B, y)$. We then show that his ability to compute such exponentiation is equivalent to solving a computational Diffie-Hellman challenge and computing z^x on input g, y, z, where $y = g^x$.

Interestingly, the proof of the secrecy property is very similar to the proof of security. The ability of an adversary $\mathcal{A}$, who does not have a valid group member certificate under the group key y, to tell if its participant in the secret handshake protocol is a member of the group under this public key, is equivalent to $\mathcal{A}$ being able himself to compute the proper authentication tag $tag = H(K, \tau)$, after sending some value w_A^* which is again not issued by the CA. But in the Random Oracle Model this is again equivalent to $\mathcal{A}$'s ability to compute the Diffie-Hellman key K as above. Hence the same proof as the one in the Appendix for theorem 4 shows that existance of such adversary is equivalent to breaking the computational Diffie Hellman problem.

Proof Assume that an adversary $\mathcal{A}$ receives n authentication tokens (t_i, w_i) from a group of malicious colluding group members $\{A_i\}$. This is equivalent to $\mathcal{A}$ receving n modified-Schnorr (message,signature) pairs $(ID_i, (t_i, w_i))$. Suppose that $\mathcal{A}$ successfully authenticates in the secret handshake protocol to some honest party B. If $\mathcal{A}$ sends one of the received legimitate certificates (w_i, ID_i) to B in the authentication protocol, the CA will immediately identify this certificate. Suppose then that $\mathcal{A}$ sends some (w, ID) different from any of the received certificates. We show that in that case, under the CDH assumption, the probability that $\mathcal{A}$ the authentication protocol is negligible. The reason is that if $\mathcal{A}$ passes the authentication protocol then he must compute the right authentication tag value $v = h(K, \tau)$ where $K = M^{t*t'} \mathrm{mod} p$, where t' is the Diffie-Hellman secret held by B, and

$M = g^t \bmod p$ is defined as $\mathsf{Recover}((w, ID), y) = wy^{h(w,ID)} \bmod p$. In the Random Oracle Model of analysis, computing such value is possible, except for negligible guessing probability, only by quering the hash function on the correctly computed argument K. Therefore, in the ROM model, an adversary $\mathcal{A}$ who succeeds in this protocol must also output $K = M^{t'} = g^{t*t'}$ as a query to the hash function. Note that $\mathcal{A}$ receives information about t' in the form of B's certificate w', ID') which defines t' because it defines $M' = \mathsf{Recover}((w', ID'), y)$ where $M' = g^{t'}$. In other words, $\mathcal{A}$ commits himself to t by sending $M = g^t$ (in the form of values (w, ID)), receives a random M' (in the form of values (w', ID')), and outputs $K = M'^t$.

Therefore, if $\mathcal{A}$ succeeds then he can exponentiate a random element M' to value t committed to in M. Hence, what we need to argue in the proof of traceability is that even though $\mathcal{A}$ receives n (message,signature) pairs $(m_i, (w_i, t_i))$ he cannot produce a new element (t, m) s.t. he can exponentiate random elements M' to exponent t where $g^t = M = wy^{h(w,m)}$. Now, this is very similar to proving the chosen message attack security of the underlying modified-Schnorr signature scheme, where one argues that, after receiving n (message,signature) pairs, $\mathcal{A}$ cannot produce a new triple (t, w, m) s.t. $g^t = wy^{h(w,m)}$. The proof is very similar to the forking-lemma proof for Schnorr signature security in [15], but here we reduce the successfull attack not to computing discrete logarithm, but to breaking the CDH assumption by computing M'^x on input $y = g^x$ and a random value M'. (We will later on with the fact that our $\mathcal{A}$ does not get M' as such, but in the form of w', m'. For now assume that $\mathcal{A}$ receives from B a random value M' in the authentication protocol.)

To reduce $\mathcal{A}$'s ability to succeed in this protocol to computing M'^x on the Diffie-Hellman challenge (g, y, M'), we first simulate, as in the proof of Schnorr signature security, the signatures (t_i, w_i, m_i) that $\mathcal{A}$ gets by taking random t_i, m_i, c_i, computing $w_i = g^{t_i} * y^{-c_i} \bmod p$, and assigning $h(w_i, m_i)$ to c_i. Since the verification equation is satisfied and c_i is picked at random, this is indistinguishable from receiving random (message,signature) pairs. As in the forking lemma argument of [15], we can argue that if $\mathcal{A}$'s probability of success is ϵ, the probability that $\mathcal{A}$ executed twice in a row succeed in *both* executions *and* he sends the *same* $C = (w, m)$ certificate in both of them, is at least ϵ^2/q_h where q_h is the number of queries $\mathcal{A}$ makes to the hash function h (see [15] for the argument). The signature-scheme security argument shows that if with probability at least ϵ^2/q_h, two conversations with $\mathcal{A}$ produce triples (m, t, w) and $(m, \hat{t}, w)$, where in first conversation $h(m, w) = c$ and in the second $h(m, w) = \hat{c}$ for some random $c, \hat{c}$, then $x = DL_g(y)$ can be computed as $x = (t - \hat{s})/(c - \hat{c})$ because $g^s = wy^c$ and $g^{\hat{t}} = wy^{\hat{c}}$, and therefore $y = g^{(t-\hat{t})/(c-\hat{c})}$. Here, however, by the same forking lemma $\mathcal{A}$ does not produce two forgeries t and $\hat{t}$, but two *exponentiations* $K = M'^t$ and $\hat{K} = M'^{\hat{t}}$, but still we have that $x = DL_g(y) = (t - \hat{t})/(c - \hat{c})$ because here again wea have that $M = g^t = wy^c$ and $\hat{M} = g^{\hat{t}} = wy^{\hat{c}}$. Therefore, with probability ϵ^2/q_h we break the CDH challenge and compute M'^x as $(K/\hat{K})^{1/(c-\hat{c})} = (M'^{t-\hat{t}})^{1/(c-\hat{c})}$.

Note that if the success probability ϵ is higher than negligible, and if $\mathcal{A}$ is an efficient algorithm and hence the number of queries q_h is polynomial, then the probability of CDH break ϵ^2/q_h is non-negligible as well.

Finally, we have to deal with the fact that $\mathcal{A}$ does not compute $K = M'^t$ on any random M' he is given, but on $M' = w'y^{h(m',w')}$ where (m', w') is sent to $\mathcal{A}$ by B. Therefore the reduction from CDH is slightly different but still succesful: For example, we can still break a CDH input challenge g, y, w' and compute w'^x by running the above simulation twice, using the same values m', w' in the two simulations, but different values for $h(m', w')$: c' and $\hat{c'}$. From the first succesful simulation we get $M'^x = w'^x y^{xc'}$ (because a conversation with $\mathcal{A}$ can be used to recover $\alpha = M'^x$ in the same way as argued above), and from the second we get $\hat{\alpha} = \hat{M}'^x = w'^x y^{x\hat{c}'}$. Therefore $\alpha^{\hat{c}'}/\hat{\alpha}^{c'} = w'^{x\hat{c}'}/w'^{xc'} = (w'^x)^{(\hat{c}'-c')}$, and hence we can break CDH and output w'^x by exponentiating the above entity to $1/(\hat{c}' - c')$. □

C Achieving Additional Security Properties

This section presents a few simple enhancements of the above SH schemes. We show that our SH schemes handle roles just as easily as the SH of [1]. We show that our CDH-based SH schemes can support "blinded" issuance of the member certificates in the sense that the CA does not learn the trapdoors included in the certificate, and thus the CA cannot impersonate that member.

C.1 Roles

Our schemes can easily be extended to handle group member roles (as in the SH scheme of [1]), in a way that a member can choose not to reveal anything about herself unless the other party is a member with a particular role r (and vice versa). This functionality can be provided by modifiying the AddMember and Recover procedures as follows:

- AddMember: takes as inputs $params$, G, t_G and an arbitrary string $ID \in \{0,1\}^*$ and returns (t, ω) where t is a trapdoor and ω is a public parameter. (t, ω) are constructed using the string $ID|r$ (instead of ID as in the original procedure), where r is the role that the CA is assigning to the user.

- Recover: takes as input $params$, G, ID and ω (provided by another user B). It outputs a public key PK using as input $ID|r$ (instead of ID as in the original Recover procedure), where r is the role that $\mathcal{A}$ chooses to have a secret hanshake with.

C.2 Trapdoor Secrecy

Note that in the RSA-based solution, the trapdoor can be computed by the user as part of AddMember and does not have to be known to the CA.

In the CDH-based solution, however, the trapdoor t is computed by the CA. As a result, the CA can impersonate the user or eavesdrop on his communications. Would that be problematic, AddMember can easily be modified to blind the trapdoor as follows:
AddMember: takes as inputs $params$, G, t_G (only known to the CA), $b = g^{\delta}$, where δ is a secret only known by the user, and an arbitrary string $ID \in \{0,1\}^*$. The CA computes $w = g^k * b$, where k is a random value in Z_{p-1}, and $t' = k + H(ID|w) * t_G$. The user then computes his trapdoor $t = t' + \delta$ (note that (t, ω) is a valid ElGamal signature of ID).