Affiliation-Hiding Authentication with Minimal Bandwidth Consumption

Mark Manulis and Bertram Poettering

Cryptographic Protocols Group, TU Darmstadt & CASED, Germany
mark@manulis.eu, bertram.poettering@cased.de

Abstract. Affiliation-Hiding Authentication (AHA) protocols have the seemingly contradictory property of enabling users to authenticate each other as members of certain groups, without revealing their affiliation to group outsiders. Of particular interest in practice is the *group-discovering* variant, which handles multiple group memberships per user. Corresponding solutions were only recently introduced, and have two major drawbacks: high bandwidth consumption (typically several kilobits per user and affiliation), and only moderate performance in scenarios of practical application.

While prior protocols have $O(n^2)$ time complexity, where n denotes the number of affiliations per user, we introduce a new AHA protocol running in $O(n \log n)$ time. In addition, the bandwidth consumed is considerably reduced. We consider these advances a major step towards deployment of privacy-preserving methods in constraint devices, like mobile phones, to which the economization of these resources is priceless.

1 Introduction

In cryptography, Authenticated Key Establishment (AKE) protocols are an essential building block for creation of secure communication channels. Such schemes offer both the establishment of a strong session key and, simultaneously, mutual authentication of respective protocol partners. Usually, this authentication step is PKI-based and explicitly reveals to other users (including adversarial eavesdroppers) the identities and certificates of participants. This behavior can be considered a breach of users' privacy. To tackle this issue, Affiliation-Hiding Authentication (AHA) in form of Secret Handshakes (SH) [1, 2, 9, 15–17, 29–31] and key establishment protocols (AHA/KE) [13, 14, 19, 21, 22] emerged in the last decade.

Generally, in AHA protocols, users authenticate each other on basis of their affiliation to certain groups, and do so in a privacy-preserving manner: In the classical 'exact matching' approach [2, 9, 15, 17, 29–31], the own affiliation is revealed to the protocol partner if and only if the protocol partner is member of the same group. Users become members of groups by registering with the respective group's authority (GA). On admission of a new user, GA generates a corresponding membership certificate and gives it to the user. This credential allows the user to authenticate itself to other group members in later so-called

'Handshake' sessions. We stress that the attempt to authenticate to non-members using such a group membership certificate not only fails, but in addition does not reveal any evidence or even hint about the group membership to given non-member. This is the pivotal property in affiliation-hiding security.

The main difference between the notions of Secret Handshakes and AHA/KE protocols is that the former are pure authentication protocols, i.e. are limited to perform the affiliation-hiding authentication of users, while the latter also succeed in the generation of a secure session key that may be used to protect further communication and exchange of digital data. In particular, AHA/KE protocols guarantee the usual key security properties of AKE protocols [3, 8] (including forward secrecy, etc).

These properties make AHA/KE protocols very attractive in various settings where privacy-preserving communication is needed. Their deployment in practice, especially on resource constraint devices and networks, requires, however, further research on efficient solutions. As we elaborate in the next sections, current proposals have efficiency limitations and are, therefore, less suitable in a mobile setting. To overcome these limitations we propose a novel AHA/KE protocol that outperforms existing approaches and minimizes the consumed communication bandwidth.

1.1 Linkable vs. Unlinkable AHA

Affiliation-Hiding Authentication protocols are either linkable or unlinkable. In *linkable* schemes [2, 9, 13, 14], users hold identities or have assigned pseudonyms which they actively reveal in protocol runs. Still, hiding of affiliations is considered valuable nonetheless, and remains an explicit security goal of those protocols. Linkable protocols are usually deployed in cases where participants are addressed by their identities anyway, e.g. in instant messaging, social networks, etc.

In *unlinkable* affiliation-hiding protocols [1, 15, 17, 31], however, sessions of users cannot be linked back to them. Obviously, these schemes offer a higher level of privacy. The challenging part in their design is the support of revocation, i.e. exclusion of members from the group: even though users do not have explicit identities, the schemes must provide methods for their individual exclusion.

In practice, linkable AHA protocols enjoy very efficient revocation by blacklisting pseudonyms on public revocation lists, while unlinkable AHA protocols support revocation either by restricting the number of unlinkable sessions of users [31], by regularly updating unrevoked membership credentials [15], or by the considerably costly verification of revocation tokens [17].

1.2 The Challenge of Group Discovery

Classical AHA schemes [1, 2, 9, 15, 17, 29–31] are mostly *single-group* protocols, i.e. it is assumed that the participating users are member of one group each, and the protocol execution checks whether these groups are identical or not. We argue, however, that this restriction to only one group may not be acceptable in practice. Consider, for instance, a social network where users are member of many, say n, groups. Now, when two participants of the network meet they may want to investigate in a privacy-preserving manner whether they have any group in common or not. If they used a classical AHA scheme, they would have to run $n \times n$ protocol instances in parallel to cover all possible combinations, spotting and reporting only the matching ones. This overhead is clearly unacceptable in practice for being inefficient, and justifies the need for special group-discovering (but still affiliation-hiding) protocols. The plurality of existing AHA schemes, however, ignores the *group-discovery* problem by design, and only two schemes, namely [16] and [19], support deployment of credentials for multiple groups in a single protocol run.

For AHA protocols that support multiple credentials, we need to define what we consider a successful authentication. It seems that the most useful notion is the following: If both users provide credentials for various groups in a specific protocol execution, the authentication is considered successful if there is at least one group in common. Output of the protocol would be this indication and, optionally, a list of all matching groups.

1.3 Related Work

The Secret Handshakes in [2, 9, 30] are linkable protocols that have been designed for the main purpose of authentication. While some of them, additionally, offer the generation of a session key, security of the latter is neither formally analyzed nor does it reach an adequate level of security in practice. In contrast, the schemes from [14, 21, 22] incorporate a secure key establishment protocol that satisfies accepted models of key security [3, 8]. Group Secret Handshakes are presented by Jarecki et al. in [12, 13], where the two-party setting is extended to multiparty authentication and key agreement. Both works achieve session group key establishment through a variant of the Burmester-Desmedt technique [7]. In [18, 21, 22], the impact of corrupt GAs on users' privacy is explored. In particular, while Manulis et al. [21, 22] act conservatively and harden protocols to withstand GA attacks, Kawai [18] deviates from the traditional setting by splitting the GA's role among an issue authority and a tracing authority (that are assumed not to collude).

We remark that unlinkable AHA schemes can generically be obtained from linkable protocols by using one-time pseudonyms; however, this approach is clearly impractical for not being scalable. Due to this restriction, several unlinkable Secret Handshakes [1, 15, 17, 29] based on reusable credentials have been proposed. Here, the challenging part is revocation of protocol participants: Ateniese's protocol [1] does not support revocation at all, Jarecki [15] presents a protocol in which participants need to regularly contact the GA for an update of users' internal state, Tsudik [29] introduces a heavy-weight framework that involves the use of group signatures and broadcast encryption techniques, while the state-of-the-art scheme [17] by Jarecki et al. uses group signatures with verifier-local revocation for group management and private conditional oblivious transfer for the handshake session, in the pairing-based setting.

Exclusively the protocols in [16, 19] offer support for multiple credentials per user, i.e. they solve the problem of efficient group-discovery. Still, the scheme in [16] by Jarecki et al. has the somehow weird and unusual property that GAs are in the position to fully impersonate any user that is registered to them: First step of the registration process to a group is that user sends its complete private key material to the corresponding GA, which then computes the appropriate user credential from it. Manulis' scheme in [19] is a rather efficient RSA-based protocol with two exponentiations per user and group, and can be considered the state-of-the-art protocol for group-discovering AHA/KE.

In the efficiency analyses, both [16] and [19] distinguish between the so-called asymmetric and symmetric workload of their protocols. While the former reflects the amount of (expensive) public key operations like exponentiations and pairing computations, the latter covers the remaining (relatively cheap) parts, including block cipher and hash function evaluations. Claiming that protocols' efficiency can be adequately estimated by taking into account only their asymmetric overhead, [16] and [19] promote their schemes as $O(n)$ protocols, where n is the number of group affiliations per user, although in both cases the real number of operations is $O(n^2)$. Contradicting this reasoning, recent results in [20] reveal that, in practice, the symmetric overhead of [16, 19] may not be neglected and limits protocols' applicability. We stress that our new protocol presented in Section 3 offers real $O(n \log n)$ performance, counting *all* computations, while the asymmetric workload remains to be $O(n)$, as in [16, 19].

1.4 Contributions and Organization

The contribution of this work is the construction of a new and highly-efficient linkable group-discovering AHA/KE protocol that outclasses existing schemes [16, 19] in several aspects: First, our protocol is the first real $O(n \log n)$ solution (consisting of $O(n)$ public key operations plus a simple sorting step in $O(n \log n)$). Second, the protocol's bandwidth requirements are impressively low. Specifically, as we will show in Section 3.3, our protocol consumes only 4% of the bandwidth when compared to [19].

We consider these improvements as a major step forward to make privacy-preserving techniques deployable in practice. In particular, on mobile devices, which are usually restricted in at least computing power or available bandwidth, without our improvements, execution of AHA protocols would be hardly feasible.

As an application, we envision users managing and exploring their social network relationships through their mobile phones that form ad hoc wireless networks of constraint range. In these scenarios, privacy-preserving techniques are of highest importance.

We start our work by giving insight into our main building block, a *Non-Interactive Key Distribution Scheme* (NIKDS), in Section 2. In Section 3 we present our new protocol, and discuss its efficiency and the selection of reasonable parameters for deployment in practice. We support the security of our protocol by giving a formal analysis in form of a model specification (Section 4) and proof of security (Section 5), in respect to this model.

2 Non-Interactive Key Distribution

In a multi-user setting, the purpose of a Non-Interactive Key Distribution Scheme (NIKDS) [5, 11, 23] is the assignment of a (fixed) symmetric key to each pair of users. The intrinsic property and advantage of NIKDS over (authenticated) key establishment protocols is that NIKDS are non-interactive, i.e. users can compute the particular keys shared with other users without any (prior) communication with them.

Typically, NIKDS are identity based schemes, i.e. users are 'addressed' by their identities, which may be arbitrary strings. In NIKDS, users first register their particular identity $id \in \{0,1\}^*$ with an authority called *Key Generation Center* (KGC) to obtain their specific user credential $sk[id]$. With this credential they can compute, without any interaction, a key shared between id and id', for any other user identity $id' \in \{0,1\}^*$. The notion of NIKDS and its security properties are formalized next.

2.1 Definition and Security Model of NIKDS

Definition 1. *A* Non-Interactive Key Distribution Scheme *is a tuple* NIKDS $=$ (Setup, Register, GetKey) *of efficient algorithms as follows:*

Setup(1^λ) :
This algorithm initializes a KGC. *On input security parameter* 1^λ*, it outputs a secret key* sk.

Register(sk, id) :
On input KGC*'s secret key* sk *and user identity* $id \in \{0,1\}^*$*, this algorithm outputs private user credential* $sk[id]$.

GetKey$(sk[id], id')$:
On input user credential $sk[id]$ *and user identity* $id' \in \{0,1\}^*$*, this algorithm outputs a key* $K \in \{0,1\}^\lambda$.

A NIKDS *is called* correct *if for all* $\lambda \in \mathbb{N}$*, all* $sk \leftarrow \mathsf{Setup}(1^\lambda)$*, all user identities* $id, id' \in \{0,1\}^*$*, all* $sk[id] \leftarrow \mathsf{Register}(sk, id)$ *and* $sk[id'] \leftarrow \mathsf{Register}(sk, id')$*, and all* $K \leftarrow \mathsf{GetKey}(sk[id], id')$ *and* $K' \leftarrow \mathsf{GetKey}(sk[id'], id)$ *we have* $K = K'$*. We consider this key as a* shared key *between parties id and id'. For convenience, we denote it by* $\mathsf{SharedKey}(sk; id, id')$.

The following security property adopts the classical key indistinguishability requirements of interactive key agreement protocols [3] to the non-interactive setting. Note that in [11] an even stronger but less natural computational variant of this model is analyzed.

Definition 2 (Indistinguishability of NIKDS). *A* NIKDS $=$ (Setup, Register, GetKey) *is called* indistinguishable under adaptive chosen-identity attacks *(IND-CIA), if for all efficient adversaries* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *the advantage function*

$$\mathbf{Adv}^{\mathsf{ind\text{-}cia}}_{\mathsf{NIKDS},\mathcal{A}}(\lambda) = \left|\Pr\left[\mathbf{Exp}^{\mathsf{ind\text{-}cia},1}_{\mathsf{NIKDS},\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathbf{Exp}^{\mathsf{ind\text{-}cia},0}_{\mathsf{NIKDS},\mathcal{A}}(\lambda) = 1\right]\right|$$

is negligible in λ*, where* $\mathbf{Exp}^{\mathsf{ind\text{-}cia},b}_{\mathsf{NIKDS},\mathcal{A}}$ *denotes the following experiment:*

$\mathbf{Exp}^{\mathsf{ind\text{-}cia},b}_{\mathsf{NIKDS},\mathcal{A}}(\lambda)$*:*

- $sk \leftarrow \mathsf{Setup}(1^\lambda)$
- $(id, id', state) \leftarrow \mathcal{A}_1^{\mathsf{Register}(sk,\cdot)}(1^\lambda)$
- $K_0 \xleftarrow{\$} \{0,1\}^\lambda$ *and* $K_1 \leftarrow \mathsf{SharedKey}(sk; id, id')$
- $b' \leftarrow \mathcal{A}_2^{\mathsf{Register}(sk,\cdot)}(state, K_b)$
- *output* 0 *if* $\mathcal{A}$ *queried* $\mathsf{Register}(sk, id)$ *or* $\mathsf{Register}(sk, id')$ *to its oracle*
- *else output* b'

2.2 A Construction of NIKDS Based on Bilinear Maps (Pairings)

The first efficient NIKDS was constructed in [23] and analyzed in [11] (although the notion of NIKDS was introduced to cryptography about 20 years earlier, in [25]). The scheme is described as follows:

$\mathsf{Setup}(1^\lambda)$:
Specify cyclic groups $G = \langle g \rangle$ and G_T of prime order q, for which an efficient non-degenerate bilinear pairing $\hat{e} : G \times G \to G_T$ is known (see also [5, Chapter X]). In addition, specify hash functions $H : \{0,1\}^* \to G$ and $H^* : G_T \to \{0,1\}^\lambda$. Pick $s \xleftarrow{\$} \mathbb{Z}_q \setminus \{0\}$ randomly and return secret key $sk = s$.

$\mathsf{Register}(sk, id)$:
On input secret key $sk = s$ and identity $id \in \{0,1\}^*$, user credential $sk[id] = H(id)^s$ is output.

$\mathsf{GetKey}(sk[id], id')$:
This algorithm outputs key $K = H^*\big(\hat{e}(sk[id], H(id'))\big)$.

Proof of Correctness. For arbitrary $id, id' \in \{0,1\}^*$, let $h \leftarrow H(id)$ and $h' \leftarrow H(id')$. We then have $sk[id] = h^s$ and $sk[id'] = (h')^s$, and correctness is implied by $\hat{e}(h^s, h') = \hat{e}(h, h')^s = \hat{e}(h', h)^s = \hat{e}((h')^s, h)$. Note that $\hat{e}(h, h') = \hat{e}(h', h)$ follows for all $h, h' \in G$ from $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab} = \hat{e}(g^b, g^a)$. □

Security of this scheme was established in [11] as follows:

Theorem 1. NIKDS *is IND-CIA secure under the Bilinear Diffie-Hellman assumption (BDH) [6] in the Random Oracle Model (ROM) [4].*

3 Our Affiliation-Hiding Authentication Protocol

In this section, we present our Affiliation-Hiding Authentication and Key Agreement protocol (AHA/KE). We kept the scheme's syntax consistent with [19], where the first practical AHA in the multi-group setting is proposed. Still, we improve considerably on that protocol in both asymptotic computational performance and bandwidth consumption. In particular, while in both [19] and our protocol the number of public key operations is linear in the number of affiliations n, the remaining 'symmetric' workload of [19] is $O(n^2)$, in contrast to $O(n \log n)$ in our protocol.

3.1 Syntax of AHA

An AHA scheme $\mathsf{AHA} = (\mathsf{CreateGroup}, \mathsf{AddUser}, \mathsf{Handshake}, \mathsf{Revoke})$ consists of four efficient algorithms and protocols:

$\mathsf{CreateGroup}(1^\lambda)$:
This algorithm is executed by a Group Authority (GA) to set up a new group G. On input security parameter 1^λ, a public/secret group key pair $(G.pk, G.sk)$ is generated, the group's pseudonym revocation list $G.prl$ is initialized to $\emptyset$, and public group parameters $G.par = (G.pk, G.prl)$ and private key $G.sk$ are output.

$\mathsf{AddUser}(G, id)$:
This algorithm is executed by the GA of group G to add user pseudonym $id \in \{0,1\}^*$ to its group. A private membership credential $sk_G[id]$ is created and confidentially handed over to the particular user. Note that users are allowed to register the same pseudonym id in different groups.

$\mathsf{Handshake}(U_i \leftrightarrow U_j)$:
This is the key exchange protocol (handshake), executed between two users U_i and U_j, that have pseudonyms id_i and id_j, respectively. Input of U_i is a set $\mathcal{G}_i$ of pairs of the form $(sk_G[id_i], G.prl)$, where all $sk_G[id_i]$ are credentials on pseudonym id_i obtained from the GA of particular G (computed by the AddUser algorithm), and $G.prl$ is the corresponding revocation list. For user U_j, the protocol's input is $\mathcal{G}_j$, analogously.

Users keep track of the state of created $\mathsf{Handshake}(\mathcal{G})$ sessions π through session variables that are initialized as follows: $\pi.\mathsf{state} \leftarrow \mathsf{running}$, $\pi.id \leftarrow id$, $\pi.\mathcal{G} \leftarrow \mathcal{G}$, and $(\pi.\mathsf{key}, \pi.\mathsf{partner}, \pi.\mathsf{groups}) \leftarrow (\perp, \perp, \emptyset)$. At some point the protocol will complete and $\pi.\mathsf{state}$ is then updated to either $\mathsf{rejected}$ or $\mathsf{accepted}$. In the latter case, $\pi.\mathsf{key}$ is set to the established session key (of length λ), the handshake partner's pseudonym is assigned to $\pi.\mathsf{partner}$, and $\pi.\mathsf{groups}$ holds a non-empty set of group identifiers.

$\mathsf{Revoke}(G, id)$:
This algorithm is executed by the GA of G and results in the update of G's pseudonym revocation list $G.prl$.

Definition 3 (Correctness of AHA). *Assume that two users with pseudonyms id_i and id_j participate in a* Handshake *protocol on inputs $\mathcal{G}_i$ and $\mathcal{G}_j$, respectively, and let π_i and π_j denote the corresponding sessions. By $\mathcal{G}_\cap$ we denote the set of groups that appear in both $\mathcal{G}_i$ and $\mathcal{G}_j$ with the restriction that neither id_i nor id_j are contained in the respective group's revocation lists $G.prl$. The* AHA *scheme is called* correct *if (1) π_i and π_j complete in the same state, which is* accepted *iff $\mathcal{G}_\cap \neq \emptyset$, and (2) if both sessions accept then $\pi_i.\mathsf{key} = \pi_j.\mathsf{key}$, $(\pi_i.\mathsf{partner}, \pi_j.\mathsf{partner}) = (id_j, id_i)$, and $\pi_i.\mathsf{groups} = \pi_j.\mathsf{groups} = \mathcal{G}_\cap$.*

3.2 Protocol Definition

We are ready to specify our new AHA protocol with implicit group discovery. As a major building block it uses a generic NIKDS. In particular, the scheme presented in Section 2 is suitable. Recall that the algorithms of NIKDS are denoted Setup, $\mathsf{Register}$, and GetKey.

$\mathsf{CreateGroup}(1^\lambda)$ Algorithm. To create a new group, the Group Authority (GA) sets up a new KGC of a NIKDS by running $sk \leftarrow \mathsf{Setup}(1^\lambda)$. In addition, the group's pseudonym revocation list $G.prl$ is emptied. This algorithm outputs $G.par = (G.prl)$ and $G.sk = sk$. Note that a group's public key is not needed, and it hence is not specified.

$\mathsf{AddUser}(G, id)$ Algorithm. A new user with pseudonym id is added to group G by registering id to the NIKDS's KGC: the user's private membership credential in group G will be $sk_G[id] \leftarrow \mathsf{Register}(sk, id)$, where $sk = G.sk$.

$\mathsf{Handshake}(\mathcal{G})$ Protocol. The specification of our $\mathsf{Handshake}$ protocol is given in Figure 1. The protocol makes use of the following building blocks:

- To achieve forward security of the established session key, a standard Diffie-Hellman key exchange is incorporated into the protocol (cf. lines 1 and 3). Hence, we require existence of a cyclic group $\mathbb{G} = \langle g \rangle$ of prime order q in which the Computational Diffie-Hellman Problem (CDH) is hard (in respect to security parameter λ).
- By $H : \{0,1\}^* \rightarrow \{0,1\}^\ell$, where $\ell = \ell(\lambda)$ is polynomially dependent on security parameter λ, we denote a hash function. It will be modeled as random oracle in the security analysis of the protocol.
- By $Sort(\mathcal{M})$, for a set $\mathcal{M} \subseteq \{0,1\}^\ell$ of strings of length ℓ, we denote the lexicographic ordering of $\mathcal{M}$. It is well-known that $Sort(\,)$ is an $O(n \log n)$ algorithm (e.g. 'Quicksort'), and that look-up in an ordered set is an $O(\log n)$ operation.

We briefly explain the design principles of the protocol from the point of view of user U_i. For all groups G in which id_i is registered (line 5) and in which id_j is not revoked (line 6), the NIKDS key K_G shared by id_i and id_j is computed (line 7) and used to derive two authentication tags $c_{G,0}, c_{G,1}$ in lines 8 and 9 (that also will serve for key confirmation). One of the tags is sent to U_j (line 12), while the other one is stored in state variable $\mathcal{S}_i$ for later use (line 10). Note that U_j computes the same tags for all groups that both U_i and U_j are member of. This intersection set (named groups) is determined in lines 13–16, by recording all matches of group-specific authentication tags. If U_i and U_j have at least one group in common (line 17), then the protocol accepts with a secure session key (lines 1, 3, and 18). Observe that the purpose of the sorting step (center of line 12) is not only to enable an $O(\log n)$ look-up of authentication tags in line 15, but also to hide the order in which these tags have been computed. This is an important prerequisite to make the protocol affiliation-hiding.

Notice that the scheme is displayed as four-message protocol for reasons of better readability. By combining messages m_j and $Sort(\mathcal{M}_j)$ into a single datagram, the scheme can be relieved by one message transmission.

```
    USER $U_i$ ON INPUT $id_i$ AND $\mathcal{G}_i$:                                USER $U_j$ ON INPUT $id_j$ AND $\mathcal{G}_j$:

1   $r_i \leftarrow_R \mathbb{Z}_q$                  $\xrightarrow{m_i = (id_i, g^{r_i})}$      $r_j \leftarrow_R \mathbb{Z}_q$
                                                   $\xleftarrow{m_j = (id_j, g^{r_j})}$
2   $\mathsf{sid} \leftarrow m_i \| m_j$                                                 $\mathsf{sid} \leftarrow m_i \| m_j$
3   $K \leftarrow \mathsf{sid} \| g^{r_i r_j}$                                             $K \leftarrow \mathsf{sid} \| g^{r_i r_j}$
4   $\mathcal{M}_i \leftarrow \emptyset$, $\mathcal{S}_i \leftarrow \emptyset$                                          $\mathcal{M}_j \leftarrow \emptyset$, $\mathcal{S}_j \leftarrow \emptyset$
5   FOR ALL $(sk_G[id_i], G.prl) \in \mathcal{G}_i$:                         FOR ALL $(sk_G[id_j], G.prl) \in \mathcal{G}_j$:
6     IF $id_j \notin G.prl$:                                         IF $id_i \notin G.prl$:
7       $K_G \leftarrow \mathsf{GetKey}(sk_G[id_i], id_j)$                        $K_G \leftarrow \mathsf{GetKey}(sk_G[id_j], id_i)$
8       $c_{G,0} \leftarrow H(K_G \| K \| 0)$                                 $c_{G,0} \leftarrow H(K_G \| K \| 0)$
9       $c_{G,1} \leftarrow H(K_G \| K \| 1)$                                 $c_{G,1} \leftarrow H(K_G \| K \| 1)$
10      $\mathcal{S}_i \leftarrow \mathcal{S}_i \cup \{(G, c_{G,1})\}$                            $\mathcal{S}_j \leftarrow \mathcal{S}_j \cup \{(G, c_{G,0})\}$
11    ELSE: $c_{G,0} \leftarrow_R \{0,1\}^\ell$                               ELSE: $c_{G,1} \leftarrow_R \{0,1\}^\ell$
12    $\mathcal{M}_i \leftarrow \mathcal{M}_i \cup \{c_{G,0}\}$                                $\mathcal{M}_j \leftarrow \mathcal{M}_j \cup \{c_{G,1}\}$
                                                   $\xrightarrow{Sort(\mathcal{M}_i)}$
13  $\mathsf{groups}_i \leftarrow \emptyset$                          $\xleftarrow{Sort(\mathcal{M}_j)}$        $\mathsf{groups}_j \leftarrow \emptyset$
14  FOR ALL $(G, c_{G,1}) \in \mathcal{S}_i$:                                    FOR ALL $(G, c_{G,0}) \in \mathcal{S}_j$:
15    IF $c_{G,1} \in Sort(\mathcal{M}_j)$:                                  IF $c_{G,0} \in Sort(\mathcal{M}_i)$:
16      $\mathsf{groups}_i \leftarrow \mathsf{groups}_i \cup \{G\}$                            $\mathsf{groups}_j \leftarrow \mathsf{groups}_j \cup \{G\}$

17  IF $\mathsf{groups}_i \neq \emptyset$ THEN                                   IF $\mathsf{groups}_j \neq \emptyset$ THEN
18    $\mathsf{key}_i \leftarrow H(K)$                                            $\mathsf{key}_j \leftarrow H(K)$
19    $\mathsf{partner}_i \leftarrow id_j$                                        $\mathsf{partner}_j \leftarrow id_i$
20    TERMINATE WITH "ACCEPT"                                    TERMINATE WITH "ACCEPT"
21  ELSE                                                       ELSE
22    TERMINATE WITH "REJECT"                                    TERMINATE WITH "REJECT"
```

Fig. 1. Specification of $\mathsf{Handshake}(U_i \leftrightarrow U_j)$. We consider the left party as initiator and the right party as responder. We intentionally left out indices i, j for variables $\mathsf{sid}, K, K_G, c_{G,0}, c_{G,1}$ as they are expected to have the same value in both U_i's and U_j's computations.

$\mathsf{Revoke}(G, id)$ Algorithm. By setting $G.prl \leftarrow G.prl \cup \{id\}$, the group's pseudonym revocation list $G.prl$ is extended by the new entry. The updated *prl* is distributed authentically to all group members.

3.3 Correctness, Efficiency, and Parameter Selection

Our AHA scheme is correct in the sense of Definition 3. This follows from correctness of deployed NIKDS and inspection of Figure 1. Recall also the exposition of design rationale in Section 3.2.

Asymptotically, the protocol is an $O(n \log n)$ protocol, where $n = |\mathcal{G}|$ denotes the number of groups per user. This is due to the fact that both the sorting step (line 12) and the tag-matching step (lines 14–16) are $O(n \log n)$. However, the number of expensive public key operations (i.e. pairing evaluations in the NIKDS) is linear in the number of affiliations. More precisely: A user that deploys credentials for n groups has to compute n pairings to complete the protocol (or even less, when considering the possibility of revoked users). Note that the AHA schemes from [16] and [19] have $O(n^2)$ workload of 'symmetric operations'. Although these can be considered rather fast in comparison to big-integer exponentiations or pairing evaluations, for large n (e.g. $n \gg 100$), the quadratic overhead of [16, 19] will be clearly noticed [20].

Especially in respect to bandwidth consumption, our protocol impressively outperforms state-of-the-art protocol [19]. In the latter, for being an RSA-based protocol, more than 4000 bits have to be sent and received per user, affiliation, and session. In our protocol, however, this number drops to about 160 bits (80 bits for each authentication tag), for a comparable level of security. Hence, our protocol consumes only 4% of the bandwidth, when compared to [19].

For practical security, we suggest to use Diffie-Hellman and pairing groups of about 2^{160} elements, authentication tags of length 80 bit (lines 8–9), and a 128 bit KDF for key derivation (line 18).

The selection of parameters for an efficient pairing suitable in practice will not be too complicated. Note that in NIKDS group elements are never transmitted from one party to the other. Hence, care does not have to be taken to find pairing-friendly curves with 'nice' element representations. Although, for reasons of convenience, only symmetric pairings were considered in Section 2 to build NIKDS, they can be built from asymmetric pairings as well [11]. At the time of writing this article, ηT-pairing evaluations on desktop machines in under 500 μs were feasible [26, 27], i.e. for a user with about 100 affiliations[1] we estimate the total running time of a Handshake execution below 50 ms. Recall from Section 2 that in our NIKDS scheme the first input element to the pairing is always $sk[id]$, which can be considered a fixed long-term parameter. See [10, 24] for considerable optimizations on fixed-argument pairing evaluations. Finally note that all NIKDS computations are session-independent and can be cached: If the same two users run the Handshake protocol multiple times they can fall back to previously computed K_G to considerably save computation time.

4 Security Model for AHA

In this section we present the security model for AHA protocols. It takes into account the challenges implied by the group discovery problem and bases on the current state-of-the-art model from [19]. Essentially, there exist two central security properties for AHA: Linkable Affiliation-Hiding security, and Authenticated Key Exchange security (with forward secrecy). Both requirements are defined with regard to multiple input groups per user and session. As the model for the latter goal is very similar to standard definitions of AKE security [3, 8], and only minor modifications are necessary to fit the LAH setting, we abstain from giving a full description of the model in this article, and refer to [19] for a detailed exposition. In contrast, LAH security is a non-standard goal and was only recently modeled [19]. We describe it in full detail below.

[1] Note that an average Facebook user is member of about 80 communities or groups [28].

4.1 Adversary Model

The adversary $\mathcal{A}$ is modeled as a PPT machine that interacts with protocol participants and can mount attacks via the following set of queries.

$\mathsf{Handshake}(id, \mathcal{G}, r)$: This query lets the holder of pseudonym id start a new session π of the $\mathsf{Handshake}$ protocol. It receives as input a set $\mathcal{G}$ of groups G and a role identifier $r \in \{\mathsf{init}, \mathsf{resp}\}$ that determines whether the session will act as protocol initiator or responder. If there is a group G listed in $\mathcal{G}$ for which id has no private credential $sk_G[id]$ then this query is ignored. Optionally, this query returns a first protocol message M.

$\mathsf{Send}(\pi, M)$: Message M is delivered to session π. After processing M, the eventual output is given to $\mathcal{A}$. This query is ignored if π is not waiting for input.

$\mathsf{Reveal}(\pi)$: If $\pi.\mathsf{state} = \mathsf{running}$ then this query is ignored. Otherwise $(\pi.\mathsf{state}, \pi.\mathsf{key}, \pi.\mathsf{groups})$ is returned.

$\mathsf{Corrupt}(id, G)$: Membership credential $sk_G[id]$ of pseudonym id in group G is passed to the adversary. Note that this query models the possibility of selective corruptions.

$\mathsf{Revoke}(G, id)$: This query lets the GA of G include id in its revocation list $G.prl$.

4.2 Linkable Affiliation-Hiding Security

We now define the property of Linkable Affiliation-Hiding (LAH). At a high level, the goal here is to protect from disclosure of non-shared affiliations to handshake partners. We model LAH-security using the indistinguishability approach (similar to that used for encryption schemes). The goal of the adversary is to decide which of two sets of affiliations, $\mathcal{G}_0^*$ or $\mathcal{G}_1^*$, some challenge session π^* is running on. The adversary specifies these sets himself, and, additionally, is allowed to invoke any number of handshake sessions, and ask Reveal and Corrupt queries at will. This intuition is formalized as follows.

Definition 4 (LAH-Security). *Let* $\mathsf{AHA} = \{\mathsf{CreateGroup}, \mathsf{AddUser}, \mathsf{Handshake}, \mathsf{Revoke}\}$, *$b$ be a randomly chosen bit, and* $\mathcal{Q} = \{\mathsf{Handshake}, \mathsf{Send}, \mathsf{Reveal}, \mathsf{Corrupt}, \mathsf{Revoke}\}$ *denote the set of queries the adversary $\mathcal{A}$ has access to. We consider the following experiment between a challenger and an efficient adversary $\mathcal{A}$:*

$\mathbf{Exp}^{\mathsf{lah},b}_{\mathsf{AHA},\mathcal{A}}(\lambda, n, m)$:

- *the challenger creates users $U_1, \ldots, U_n$ and pseudonyms $\mathsf{ID} = \{id_1, \ldots, id_n\}$;*
- *the challenger creates m groups $\mathcal{G} = \{G_1, \ldots, G_m\}$ and registers user U_i with pseudonym id_i in group G_j for all $(i, j) \in [1, n] \times [1, m]$;*
- *$\mathcal{A}^{\mathcal{Q}}$ interacts with all participants using the queries in $\mathcal{Q}$; at some point $\mathcal{A}^{\mathcal{Q}}$ outputs a tuple $(id^*, \mathcal{G}_0^*, \mathcal{G}_1^*, r^*)$ where $id^* \in \mathsf{ID}$, $\mathcal{G}_0^*, \mathcal{G}_1^* \subseteq \mathcal{G}$ with $|\mathcal{G}_0^*| = |\mathcal{G}_1^*|$, and $r^* \in \{\mathsf{init}, \mathsf{resp}\}$. The set $\mathcal{D}^* = (\mathcal{G}_0^* \setminus \mathcal{G}_1^*) \cup (\mathcal{G}_1^* \setminus \mathcal{G}_0^*) = (\mathcal{G}_0^* \cup \mathcal{G}_1^*) \setminus (\mathcal{G}_0^* \cap \mathcal{G}_1^*)$ is called the* distinguishing set*;*
- *the challenger invokes a $\mathsf{Handshake}(id^*, \mathcal{G}_b^*, r^*)$ session π^* (and provides all needed credentials);*

– *$\mathcal{A}^{\mathcal{Q}}$ continues interacting via queries (including on session π^*) until it terminates and outputs bit b';*
– *the output of the game is b' if all of the following hold; else the output is 0:*
  (a) if π^* accepted and there is a Handshake *session π' with $\mathcal{D}^* \cap \pi'.\mathcal{G} \neq \emptyset$ which was in state* running *while π^* was in state* running*, then no* Reveal(π^*) *query was asked, and*
  (b) no Reveal(π') *query was asked for any* Handshake *session π' with $\mathcal{D}^* \cap \pi'.\mathcal{G} \neq \emptyset$ and $\pi'.\mathsf{partner} = id^*$ that was in state* running *while π^* was in state* running*, and*
  (c) no Corrupt(id, G) *query with $(id, G) \in \mathsf{ID} \times \mathcal{D}^*$ was asked.*

We define $\mathbf{Adv}^{\mathsf{lah}}_{\mathsf{AHA},\mathcal{A}}(\lambda, n, m) :=$

$$\left|\Pr\left[\mathbf{Exp}^{\mathsf{lah},0}_{\mathsf{AHA},\mathcal{A}}(\lambda, n, m) = 1\right] - \Pr\left[\mathbf{Exp}^{\mathsf{lah},1}_{\mathsf{AHA},\mathcal{A}}(\lambda, n, m) = 1\right]\right|$$

and denote with $\mathbf{Adv}^{\mathsf{lah}}_{\mathsf{AHA}}(\lambda, n, m)$ the maximum advantage over all PPT adversaries $\mathcal{A}$. We say that AHA *is* LAH-secure *if this advantage is negligible in λ (for all n, m polynomially dependent on λ).*

Conditions (a)–(c) exclude some trivial attacks on affiliation hiding. Condition (a) thwarts the attack where $\mathcal{A}$ starts a Handshake($id', \mathcal{G}', r'$) session π' with $\mathcal{G}' \cap \mathcal{D}^* \neq \emptyset$, relays all messages between π^* and π' and finally asks Reveal(π^*). By protocol correctness $\pi^*.\mathsf{groups}$ would contain elements from $\mathcal{D}^*$ and it would be trivial to correctly decide about b. Condition (b) handles the same attack, but from the point of view of π'. Condition (c) prevents $\mathcal{A}$ to corrupt a pseudonym in a group in $\mathcal{D}^*$, to impersonate that user, and to decide about bit b from the output of its protocol run.

5 Security Analysis of Our Protocol

Following the definitions in Section 4, we claim that our AHA protocol from Section 3 satisfies the desired security goals.

Theorem 2 (Linkable Affiliation-Hiding Security). *The* AHA *protocol presented in Section 3.2 is LAH-secure in the Random Oracle Model (ROM) [4], given that* NIKDS *is IND-CIA secure.*

Proof (Sketch). We prove LAH security of our AHA protocol by using the 'game-hopping' technique, i.e. by presenting a sequence of games that are 'neighbor-wise' computationally indistinguishable from adversary's point of view. The first game, $G_{0,b}$, is identical with $\mathbf{Exp}^{\mathsf{lah},b}_{\mathsf{AHA},\mathcal{A}}(\lambda, n, m)$.

Let $G_{1,b}$ denote the game that is identical with $G_{0,b}$, except that the challenger, before even starting the simulation, makes guesses for $\mathcal{A}$'s (future) choice of attacked identity id^* and protocol partner $\pi^*.\mathsf{partner}$. The simulation is aborted if these guesses are not consistent with adversary's actions, i.e. with probability at most $1/n^2$, as n denotes the number of simulated users.

Let $G_{2,b}$ denote the game that is identical with $G_{1,b}$, except that, for all groups in $\mathcal{G}_b^* \cap \mathcal{D}^*$, the NIKDS keys K_G shared between id^* and $\pi^*.\mathsf{partner}$ are replaced by random values in $\{0,1\}^\lambda$ (see Figure 1, line 7). As condition (c) in Definition 4 assures that adversary $\mathcal{A}$ did not obtain a corresponding user credential $sk_G[id^*]$ or $sk_G[\pi^*.\mathsf{partner}]$ by corruption, the probability for $\mathcal{A}$ to detect a difference between $G_{2,b}$ and $G_{1,b}$ can be bound by $|\mathcal{G}_b^* \cap \mathcal{D}^*| \cdot \mathbf{Adv}^{\mathsf{ind\text{-}cia}}_{\mathsf{NIKDS},\mathcal{A}}(\lambda)$ (see Definition 2), which is assumed to be negligible in λ.

Now note that in messages and keys of the protocol simulated in game $G_{2,b}$ no information about the groups in $\mathcal{G}_b^* \cap \mathcal{D}^*$ remains (all relevant keys K_G have been replaced by random strings). We hence argue that experiments $G_{2,b}$ and $G_{2,(1-b)}$ are not distinguishable (the stochastic distance is zero).

We conclude the proof by noticing that we have shown

$$G_{0,b} \approx G_{1,b} \approx G_{2,b} = G_{2,(1-b)} \approx G_{1,(1-b)} \approx G_{0,(1-b)},$$

where relation '$\approx$' expresses that two games are only computationally distinguishable by an adversary with negligible probability. It follows that

$$\Pr\left[\mathbf{Exp}^{\mathsf{lah},1}_{\mathsf{AHA},\mathcal{A}}(\lambda,n,m)=1\right] \approx \Pr\left[\mathbf{Exp}^{\mathsf{lah},0}_{\mathsf{AHA},\mathcal{A}}(\lambda,n,m)=1\right],$$

and hence $\mathbf{Adv}^{\mathsf{lah}}_{\mathsf{AHA},\mathcal{A}}(\lambda,n,m)$ is negligible in λ (cf. Definition 4). □

As we abstained from formally defining AKE-security in Section 4, we here only give a qualitative result about key security of our protocol. We refer the interested reader to [19] for the state-of-the-art key security model that considers the affiliation-hiding setting. The proof given for the protocol of [19] can easily be adapted to fit our new scheme.

Theorem 3 (Authenticated Key Exchange Security). *The* AHA *protocol presented in Section 3.2 is AKE-secure [19] under the CDH assumption in the Random Oracle Model (ROM) [4], given that* NIKDS *is IND-CIA secure.*

6 Conclusion

We gave a construction of a new and impressively efficient Affiliation-Hiding Authentication scheme with included Key Establishment (AHA/KE). Its asymptotic computational performance of $O(n \log n)$ compares very favorably to $O(n^2)$ of its predecessors [16, 19]. The same holds for bandwidth consumption, which amounts to only 4% of that of [19]. Still, the protocol's syntax and security properties remain in consistency with accepted security notions for AHA [19].

We consider this work crucial in respect to deployment of privacy-preserving techniques in devices with limited resources, such as mobile phones in ad hoc wireless networks. Without the improvements described in the preceding sections, implementations of AHA protocols would hardly run at acceptable speed on such equipment.

Acknowledgments

The authors are supported in part by the DAAD PPP grant Nr. 50743263 (Project PACU) and by the BMBF grant AUS 10/046 (Project POC).

References

1. Ateniese, G., Kirsch, J., Blanton, M.: Secret Handshakes with Dynamic and Fuzzy Matching. In: Network and Distributed System Security Symposium (NDSS 2007). The Internet Society, San Diego (2007)
2. Balfanz, D., Durfee, G., Shankar, N., Smetters, D.K., Staddon, J., Wong, H.-C.: Secret Handshakes from Pairing-Based Key Agreements. In: IEEE Symposium on Security and Privacy 2003, pp. 180–196. IEEE CS, Los Alamitos (2003)
3. Bellare, M., Rogaway, P.: Entity Authentication and Key Distribution. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 232–249. Springer, Heidelberg (1994)
4. Bellare, M., Rogaway, P.: Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In: 1st ACM Conference on Computer and Communications Security (CCS 1993), pp. 62–73. ACM, New York (1993)
5. Blake, I., Seroussi, G., Smart, N., Cassels, J.W.S.: Advances in Elliptic Curve Cryptography. London Mathematical Society Lecture Note Series. Cambridge University Press, New York (2005)
6. Boneh, D., Franklin, M.K.: Identity-Based Encryption from the Weil Pairing. SIAM J. Comput. 32(3), 586–615 (2003)
7. Burmester, M., Desmedt, Y.G.: A Secure and Efficient Conference Key Distribution System. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 275–286. Springer, Heidelberg (1995)
8. Canetti, R., Krawczyk, H.: Analysis of Key-Exchange Protocols and their use for Building Secure Channels. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 453–474. Springer, Heidelberg (2001)
9. Castelluccia, C., Jarecki, S., Tsudik, G.: Secret Handshakes from CA-Oblivious Encryption. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 293–307. Springer, Heidelberg (2004)
10. Costello, C., Stebila, D.: Fixed Argument Pairings. In: Abdalla, M., Barreto, P.S.L.M. (eds.) LATINCRYPT 2010. LNCS, vol. 6212, pp. 92–108. Springer, Heidelberg (2010)
11. Dupont, R., Enge, A.: Provably Secure Non-interactive Key Distribution Based on Pairings. Discrete Applied Mathematics 154(2), 270–276 (2006)
12. Jarecki, S., Kim, J.H., Tsudik, G.: Authentication for Paranoids: Multi-party Secret Handshakes. In: Zhou, J., Yung, M., Bao, F. (eds.) ACNS 2006. LNCS, vol. 3989, pp. 325–339. Springer, Heidelberg (2006)
13. Jarecki, S., Kim, J.H., Tsudik, G.: Group Secret Handshakes or Affiliation-Hiding Authenticated Group Key Agreement. In: Abe, M. (ed.) CT-RSA 2007. LNCS, vol. 4377, pp. 287–308. Springer, Heidelberg (2006)
14. Jarecki, S., Kim, J.H., Tsudik, G.: Beyond Secret Handshakes: Affiliation-Hiding Authenticated Key Exchange. In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 352–369. Springer, Heidelberg (2008)
15. Jarecki, S., Liu, X.: Unlinkable Secret Handshakes and Key-Private Group Key Management Schemes. In: Katz, J., Yung, M. (eds.) ACNS 2007. LNCS, vol. 4521, pp. 270–287. Springer, Heidelberg (2007)

16. Jarecki, S., Liu, X.: Affiliation-Hiding Envelope and Authentication Schemes with Efficient Support for Multiple Credentials. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part II. LNCS, vol. 5126, pp. 715–726. Springer, Heidelberg (2008)
17. Jarecki, S., Liu, X.: Private Mutual Authentication and Conditional Oblivious Transfer. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 90–107. Springer, Heidelberg (2009)
18. Kawai, Y., Yoneyama, K., Ohta, K.: Secret Handshake: Strong Anonymity Definition and Construction. In: Bao, F., Li, H., Wang, G. (eds.) ISPEC 2009. LNCS, vol. 5451, pp. 219–229. Springer, Heidelberg (2009)
19. Manulis, M., Pinkas, B., Poettering, B.: Privacy-Preserving Group Discovery with Linear Complexity. In: Zhou, J., Yung, M. (eds.) ACNS 2010. LNCS, vol. 6123, pp. 420–437. Springer, Heidelberg (2010)
20. Manulis, M., Poettering, B.: Practical Affiliation-Hiding Authentication from Improved Polynomial Interpolation. In: ACM Symposium on Information, Computer and Communications Security (ASIACCS 2011). ACM, New York (2011)
21. Manulis, M., Poettering, B., Tsudik, G.: Affiliation-Hiding Key Exchange with Untrusted Group Authorities. In: Zhou, J., Yung, M. (eds.) ACNS 2010. LNCS, vol. 6123, pp. 402–419. Springer, Heidelberg (2010)
22. Manulis, M., Poettering, B., Tsudik, G.: Taming Big Brother Ambitions: More Privacy for Secret Handshakes. In: Atallah, M.J., Hopper, N.J. (eds.) PETS 2010. LNCS, vol. 6205, pp. 149–165. Springer, Heidelberg (2010)
23. Sakai, R., Ohgishi, K., Kasahara, M.: Cryptosystems Based on Pairings. In: Symposium on Cryptography and Information Security, SCIS (2000)
24. Scott, M.: Computing the Tate Pairing. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 293–304. Springer, Heidelberg (2005)
25. Shamir, A.: Identity-Based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
26. Shigeo, M.: A Fast Implementation of ηT Pairing in Characteristic Three on Intel Core 2 Duo Processor. Cryptology ePrint Archive, Report 2009/032 (2009)
27. Takahashi, G., Hoshino, F., Kobayashi, T.: Efficient GF(3^{m}) Multiplication Algorithm for ηT Pairing. Cryptology ePrint Archive, Report 2007/463 (2007)
28. The Facebook (2010), `http://www.facebook.com/press/info.php?statistics`
29. Tsudik, G., Xu, S.: A Flexible Framework for Secret Handshakes. In: Danezis, G., Golle, P. (eds.) PETS 2006. LNCS, vol. 4258, pp. 295–315. Springer, Heidelberg (2006)
30. Vergnaud, D.: RSA-Based Secret Handshakes. In: Ytrehus, Ø. (ed.) WCC 2005. LNCS, vol. 3969, pp. 252–274. Springer, Heidelberg (2006)
31. Xu, S., Yung, M.: k-Anonymous Secret Handshakes with Reusable Credentials. In: 11th ACM Conference on Computer and Communications Security (CCS 2004), pp. 158–167. ACM, New York (2004)