Privacy-Preserving Group Discovery with Linear Complexity

Mark Manulis[1], Benny Pinkas[2,⋆], and Bertram Poettering[1]

[1] Cryptographic Protocols Group, TU Darmstadt & CASED, Germany
mark@manulis.eu, bertram.poettering@cased.de
[2] Department of Computer Science, University of Haifa, Israel
benny@pinkas.net

Abstract. Affiliation-Hiding Authenticated Key Exchange (AH-AKE) protocols enable two distrusting users, being in possession of membership credentials for some group, to establish a secure session key without leaking any information about this group to non-members. In practice, users might be members of several groups, and such protocols must be able to generate session keys between users who have one or more groups in common. Finding efficient solutions for this *group discovery* problem has been considered an open research problem, inherent to the practical deployment of these protocols.

We show how to solve the privacy-preserving group discovery problem with *linear* computational and communication complexity, namely $O(n)$ complexity where n is the number of groups per user. Our generic solution is based on a new primitive — *Index-Hiding Message Encoding (IHME)*, for which we provide definitions and an unconditionally secure construction. Additionally, we update the syntax and the security model of AH-AKE protocols to allow multiple input groups per participant and session. Furthermore, we design a concrete multi-group AH-AKE protocol by applying IHME to a state-of-the-art single-group scheme.

1 Introduction

Privacy-Preserving Key Establishment. Affiliation-Hiding Authenticated Key Exchange (AH-AKE) protocols [12, 13] combine the privacy-preserving authentication properties of Secret Handshakes (SH) [2, 6, 21, 20, 19, 1, 14, 16] with secure establishment of session keys. The typical setting of AH-AKE assumes that participants are registered members of some groups, whereby each group is administrated by a separate Group Authority (GA). Each GA is responsible for the admission of users to its group and for the issue of corresponding membership credentials. Most protocols also support revocation of users, which is also performed by the GA. The actual privacy of authentication stems from the requirement to hide the affiliations (i.e. groups) of users participating in a handshake protocol from outsiders, and also from each other, as long as their

⋆ Research partially supported by EU project CACE and by the ERC project SFEROT.

J. Zhou and M. Yung (Eds.): ACNS 2010, LNCS 6123, pp. 420–437, 2010.

groups do not satisfy some predefined matching relationship, such as equality or dynamic matching [1].

Linkability vs. Unlinkability. AH-AKE and SH protocols come in two flavors: Linkable protocols such as [2, 6, 12, 13] are realized using pseudonyms and allow users to recognize each other across multiple sessions. In many applications linkability is essential, e.g. in social networks it is necessary for distinguishing amongst different members of the community. Linkable protocols enjoy very efficient forms of revocation where pseudonyms are simply added to revocation lists authenticated by the GAs. In contrast, unlinkable AH-AKE and SH protocols such as [21, 14, 1, 16] preclude the ability to link communication sessions of the same user. To handle revocation these protocols usually deploy less efficient group management schemes. In addition, the practical usage of unlinkable AH-AKE protocols for many envisioned group-oriented applications seems further away than that of protocols that support linkability.

Group Discovery Problem. The affiliation-hiding property provided by AH-AKE protocols is meaningful only if multiple groups are present in the system. Since users may belong to several groups at the same time, the inherent problem in practice is not to decide whether two given users are members of the same single group, but rather whether there is a non-empty intersection between the two sets of groups to which the users belong. Current AH-AKE and SH protocols ignore the latter problem by design, i.e. the handshake execution is performed typically with respect to only a *single* input group per participant and session. Little attention has been paid so far to possible solutions for the more general problem, termed as the *group discovery* problem in [13, p. 356]. A protocol that solves the group discovery problem would take as input a *(sub)set of groups* per participant and session, output the intersection of these sets, and, in the case that this intersection is not empty, provide a session key to the users for their subsequent communication. One of the main challenges here is to prevent the event that participants inadvertently reveal non-matching groups from their input sets to each other or to outsiders. A trivial solution for group discovery is to execute a single-group protocol for all possible combinations of membership groups, and whenever some session is successful its input group is added to the intersection set. Clearly, this solution is highly inefficient (see also discussion in Section 2). Another challenge would be the computation of the session key in a way that ensures that leakage of this key does not reveal any information about groups in the intersection set. Motivated by the importance of group discovery for the practical use of AH-AKE protocols we highlight in this paper the main challenges and explore various solutions.

1.1 Related Work

The concept of linkable Secret Handshakes [2,6] evolved into linkable Affiliation-Hiding Key Exchange [13]. These protocols commonly consider user pseudonyms as part of their group membership credentials. These pseudonyms are sent in the

clear and allow for linking the sessions of the same user. Membership credentials in [2] are further bound to secret elements of a bilinear group, in [6] they are derived from private/public key pairs of a CA-oblivious PKI-enabled encryption that is realized via Schnorr signatures, and in [20, 13] they are given by full domain hash RSA signatures on the pseudonyms. All linkable protocols can be tweaked to support unlinkability through the use of one-time credentials (and pseudonyms). Due to the impracticality of this approach several unlinkable Secret Handshakes [1, 14, 19, 16] based on reusable credentials have been proposed. However, these schemes suffer from a rather complicated group management, e.g. [1] does not support revocation, [14] requires synchronization of revocation epochs amongst members, [19] is a heavy-weight framework scheme that involves the use of group signatures and broadcast encryption techniques, while the state-of-the-art scheme in [16] uses group signatures with verifier-local revocation for group management and private conditional oblivious transfer for the handshake session in the pairing-based setting. In Section 2 we discuss in more detail why known flavors of Private Set Intersection (PSI) protocols do not give solutions to the group discovery problem.

1.2 Contribution and Organization

To this end, our goal is to explore implicit and efficient solutions for the group discovery problem in AH-AKE protocols (and consequently also in pure Secret Handshake schemes where the computation of secure session keys is not amongst the necessary requirements). We start in Section 2 by commenting on the relationship of this problem to different flavors of Private Set Intersection protocols. We show that group discovery has a more complicated setting compared to the latter (although this may not appear so at first sight). We then continue by describing potential solutions with varying efficiency and security based on probabilistic hashing.

In Section 3 we introduce *Index-Hiding Message Encoding (IHME)* — our new technique that allows for efficient group discovery in AH-AKE protocols with linear communication and computation complexity. We give a perfectly secure construction of IHME using polynomial interpolation and relate the performance of IHME-based group discovery to the mentioned hashing-based approaches.

In Section 4 we then introduce a group-discovering linkable AH-AKE protocol by applying IHME to the state-of-the-art linkable AH-AKE protocol from [13], which admits only one input group per user and session. The index-hiding property is hereby essential to preserve the affiliation-hiding property in the multi-group setting.

In Section 5 we formalize the security of linkable AH-AKE protocols by adopting a model from [13] to the setting that admits group discovery. In particular, in the definition of *affiliation hiding* we resolve the challenge of relating the outcome of the protocol containing the intersection of input group (sub)sets to the knowledge of the adversary that may have valid membership credentials for some of these groups. Following the model we finally give the security analysis of our scheme in Section 6.

2 The Group Discovery Problem and Potential Solutions

In this section we focus on various solutions for the group discovery problem in AH-AKE protocols. One of these solutions is trivial but inefficient, whereas some other solutions, while being more sophisticated, may be less secure.

In order to illustrate different approaches we briefly introduce the setting for the group discovery problem in AH-AKE protocols. Consider N different groups $G_1, \ldots, G_N$ managed by distinct group authorities. We assume that user U_1 is a registered member of n_1 groups, i.e. U_1 holds a set of n_1 different membership credentials $\{(G_i, \mathsf{cred}_i^1)\}_i$. Similarly, U_2 is a registered member of n_2 groups holding own set of n_2 credentials $\{(G_j, \mathsf{cred}_j^2)\}_j$. To simplify the exposition, let us further assume that $n_1 = n_2$ and use the notation $n = n_1 = n_2$. At a high level, the goal of the group discovery problem in AH-AKE protocols is to execute a handshake session between U_1 and U_2 such that at the end of the session (a) the users identify the subset of groups for which both have respective membership credentials (without disclosing information about any other credentials they possess), and (b) if this subset of groups is non-empty, then the two users agree on a secret key. Current AH-AKE (and SH) protocols admit only one input group per handshake participant and session, basically allowing for privacy-preserving matching of some input groups G_i and G_j used by U_1 and U_2, respectively. In our description we will utilize this ability of U_1 and U_2 to execute such single-group AH-AKE protocols using any of their membership credentials.

2.1 Relationship to (Authorized) Private Set Intersection

We investigate first whether the group discovery problem can be solved in a more general way using Private Set Intersection (PSI) protocols such as [9, 17, 10, 15, 8, 11]. In a typical PSI setting users execute the protocol using sets of elements as inputs. Each user has its own input set and the main goal of a PSI protocol is to allow users to learn the intersection of their input sets without disclosing any information about further elements. One might attempt to design a group discovery protocol by simply letting group credentials be random nonces from a large domain (but identical for all members of the group) and use a PSI protocol to check if the two users have nonces of the same group. With this solution, however, a number of problems arise: (a) providing the same credential to all group members precludes member revocation since users can trivially create and admit new members to their groups without the GA noticing it, simply by revealing their nonces to other users, (b) the proposed technique leads to an AH-AKE protocol which is not affiliation-hiding in the sense defined in Section 5 as shown in our full version, and (c) although PSI protocols with linear computation overhead are known [8, 11], our group discovery protocol presented in Section 4 can be implemented more efficiently as it relies on simpler building blocks.

A related class of protocols [7, 8], called Authorized PSI (APSI), strengthens the requirements of PSI protocols in that the computed intersection of the users' inputs must contain authorized elements only, i.e. elements that have been previously certified by some trusted authority. A technique for computing the

intersection of certified sets has been introduced in [4]. One may think that authorization of elements in APSI corresponds to the registration process of users to groups in AH-AKE protocols. However, the APSI setting assumes that the *same* authority certifies all elements in the input sets. In contrast, the AH-AKE setting explicitly requires existence of *multiple* independent group authorities providing users with membership certificates. In addition, problem (a) in the PSI setting (support for revocation) also applies here. Since neither PSI nor APSI protocols help to solve the group discovery problem directly, we discuss in the following several alternatives before coming to our most efficient solution.

2.2 Possible Solutions

Naïve Approach. The trivial solution is for U_1 and U_2 to use a single-group AH-AKE protocol for any possible combination of their membership groups. This requires n^2 different AH-AKE sessions, which might be too high in practice.

Reducing the Overhead by Using Hashing. A possible improvement that decreases the overhead involves the usage of hashing. We describe here only the basic ideas of this solution, since the focus of this paper is on a more efficient solution based on a new encoding technique, which we introduce in Section 3.

In the hashing-based approach, the parties use a random hash function h, which is either chosen in advance or jointly defined by the two parties. The hash function maps arbitrary values to an output in the range $[1, B], B \in \mathbb{N}$, namely into one of B bins. Each party then assigns its membership credentials in group G_i into bin $h(i)$. Now, when U_1 and U_2 meet, they need not run the AH-AKE protocol between each of the n^2 combinations of their potential input groups. Instead, the protocol must only be run between the membership credentials that were mapped by both parties to the same bin. Indeed, for every group G_i for which both U_1 and U_2 have membership credentials, both parties map these credentials to the same bin $h(i)$ and will, therefore, run the AH-AKE protocol with these credentials.

The basic idea described above succeeds in finding every match between membership credentials of the two parties. However, in order to protect privacy, a protocol which is based on this approach must hide from each party how many credentials were mapped by the other party to each of the bins (otherwise some data is leaked; for example, if the first bin of U_1 is empty then U_2 learns that U_1 in not a member of any group G_i for which $h(i) = 1$). Hiding the number of items in every bin can be done (following [9]) by finding a bound M such that the following property holds with high probability: when n items are mapped by a random hash function to B bins, then no more than M items are mapped to any single bin. Given this bound M, each party first maps its credentials to the B bins, and then adds to each bin that has less than M credentials, additional "dummy" values, which are indistinguishable from real credentials, so that the total number of items in the bin is M. The protocol now requires to run M^2 handshakes of the single-group AH-AKE protocols for every bin, resulting in

the total of BM^2 sessions. In order to set the right parameters, we can use the following well known fact [18, Theorem 1]:

Fact 1. *If n items are mapped at random to $B = n/\log n$ bins, then the probability that there is a bin with more than $M = O(\log n)$ items is $o(1)$.*

The communication overhead of the protocol is obviously $O(BM^2)$. Also, the computation requires each party to run the single-group handshake BM^2 times. Plugging in the parameters $B = n/\log n$ and $M = O(\log n)$, we get that the communication and computation overheads are both $O(n \log n)$. (An even more efficient variant of this technique, based on *balanced allocation hashing*, is explored in the full version of this paper. Its communication and computation complexity is $O(n \log \log n)$. That variant leaks however some additional information about group affiliations.)

3 Index-Hiding Message Encoding

The main tool we will use in our protocol is a new primitive called *Index-Hiding Message Encoding (IHME)*. By this term we understand a technique that pools a set of input messages $m_1, \ldots, m_n \in \mathcal{M}$ (where $\mathcal{M}$ is a message space) into a single data structure $\mathcal{S}$. Any message can be recovered from $\mathcal{S}$ individually by addressing it via its *index* which is arbitrarily chosen from an index set $\mathcal{I}$ and specified at encoding-time. If it is impossible for an adversary to reveal information about the deployed indices by inspecting $\mathcal{S}$ then the scheme is called *index-hiding*. This notion is now formalized by first presenting the syntax of the related concept of *index-based message encoding* and its correctness definition, and then by giving a game-based definition of the index-hiding property.

Definition 1 (Index-Based Message Encoding). *An* index-based message encoding *scheme* $(\mathsf{iEncode}, \mathsf{iDecode})$ *over an index space $\mathcal{I}$ and a message space $\mathcal{M}$ consists of two efficient algorithms:*

$\mathsf{iEncode}(\mathcal{P})$ *On input consisting of a tuple of index-message pairs $\mathcal{P} = \{(i_1, m_1), \ldots, (i_n, m_n)\} \subseteq \mathcal{I} \times \mathcal{M}$ with distinct indices $i_1, \ldots, i_n$, this algorithm outputs an encoding $\mathcal{S}$.*

$\mathsf{iDecode}(\mathcal{S}, i)$ *On input of an encoding $\mathcal{S}$ and an index $i \in \mathcal{I}$ this algorithm outputs a message $m \in \mathcal{M}$.*

An index-based message encoding scheme is correct *if $\mathsf{iDecode}(\mathsf{iEncode}(\mathcal{P}), i_j) = m_j$ for all $j \in \{1, \ldots, n\}$ and all tuples $\mathcal{P} = \{(i_1, m_1), \ldots, (i_n, m_n)\} \subseteq \mathcal{I} \times \mathcal{M}$ with distinct indices i_j.*

An index-based message encoding scheme is called *index-hiding* if it hides the indices in which the messages are encoded. That is, it ensures that an attacker, who sees an encoding $\mathcal{S}$ and might even know some of the indices and corresponding messages, cannot identify any other indices in which messages are encoded.

Given this property, which is formalized below, an index-hiding message encoding can be used for solving group discovery in AH-AKE in the following way:

The first party, U_1, IHME-encodes its first messages m_j of each of the AH-AKE protocols in the groups it belongs to, using fixed indices i_j assigned to these groups. The resulting encoding $\mathcal{S}$ is sent to the second party, U_2, which can retrieve and answer the messages encoded in the indices corresponding to U_2's credentials. However, messages corresponding to other groups cannot be recognized by U_2, since it does not have the credentials required to participate in the AH-AKE protocols of the corresponding groups, and therefore it is in the same situation as someone participating in a one-on-one AH-AKE protocol without being a member of the relevant group. U_2 therefore cannot identify neither these messages nor the deployed indices.

We note that our construction of IHME, presented below, has the attractive property that encodings $\mathcal{S}$ are of the same size (in bits) as the sets of embedded messages. Given this property one can pass n messages associated with n specific indices in an encoding $\mathcal{S}$ which is only as large as the original set of messages, and with the following two properties: (a) for each index known to the receiving party, decoding is possible in a single attempt (since the receiving party knows where to look for the message), and (b) the indices of messages which the other party is not authorized to decode are kept hidden.

Definition 2 (Index-Hiding Message Encoding (IHME)). *Let* $\mathsf{IHME} = (\mathsf{iEncode}, \mathsf{iDecode})$ *denote a correct index-based message encoding scheme over index space* $\mathcal{I}$ *and message space* $\mathcal{M}$*. Let* $b \in \{0,1\}$ *be a randomly chosen bit and let* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *be a PPT adversary that participates in the following game.*

$\mathsf{Game}^{\mathsf{ihide},b}_{\mathcal{A},\mathsf{IHME}}(\kappa)$:

- $(I_0, I_1, M', St) \leftarrow \mathcal{A}_1(1^\kappa)$ *such that* $I_0, I_1 \subseteq \mathcal{I}$ *with* $|I_0| = |I_1| = n$*, and* $M' = (m'_1, \ldots, m'_{|I_0 \cap I_1|})$ *with each* $m'_j \in \mathcal{M}$*; (the adversary chooses two subsets of* n *indices each, as well as a message* m'_j *for each index* i_j *in the intersection of the sets);*
- *denote the indices in* $I_b \backslash I_{1-b}$ *as* $\{i_1, \ldots, i_r\}$ *and define* $m_1, \ldots, m_r \xleftarrow{\$} \mathcal{M}$*, (additional* $r = n - |I_0 \cap I_1|$ *messages are chosen uniformly at random in the message space),*
  let $\mathcal{S} \leftarrow \mathsf{iEncode}(\{(i_j, m'_j) \mid i_j \in I_0 \cap I_1\} \cup \{(i_j, m_j) \mid i_j \in I_b \setminus I_{1-b}\})$*, (the messages are encoded for the indices in* I_b *);*
- $b' \leftarrow \mathcal{A}_2(St, \mathcal{S})$ *(the adversary is given* S *and attempts to find* b*);*
- *if* $b' = b$ *then return 1 else return 0.*

The advantage of $\mathcal{A}$ *is defined as*

$$\mathsf{Adv}^{\mathsf{ihide}}_{\mathcal{A},\mathsf{IHME}}(\kappa) := \left| \Pr[\mathsf{Game}^{\mathsf{ihide},0}_{\mathcal{A},\mathsf{IHME}}(\kappa) = 1] - \Pr[\mathsf{Game}^{\mathsf{ihide},1}_{\mathcal{A},\mathsf{IHME}}(\kappa) = 1] \right|.$$

By $\mathsf{Adv}^{\mathsf{ihide}}_{\mathsf{IHME}}(\kappa)$ *we denote the maximum advantage over all PPT adversaries* $\mathcal{A}$*. We say that* IHME *provides* index-hiding *if this advantage is negligible in* κ*. Moreover, if* $\mathsf{Adv}^{\mathsf{ihide}}_{\mathsf{IHME}}(\kappa) = 0$ *for all* κ*, the* IHME*-scheme is called* perfect*.*

The definition above enables the adversary to choose the sets of indices, and the messages corresponding to the intersection of the sets. The other messages are

chosen at random. The adversary is given one of the two sets and its goal is to identify which set it is. The definition requires that the adversary's success probability be negligible.

An Implementation of Perfect IHME. One way to efficiently implement IHME is by using polynomial interpolation in a finite field $\mathbb{F}$. The index and message spaces are then specified as $\mathcal{I} = \mathcal{M} = \mathbb{F}$. Algorithms iEncode and iDecode are specified as follows:

iEncode($\mathcal{P}$) On input of $\mathcal{P} = \{(i_1, m_1), \ldots, (i_n, m_n)\} \subseteq \mathcal{I} \times \mathcal{M} = \mathbb{F}^2$, the encoding is defined as the list $\mathcal{S} = (a_{n-1}, \ldots, a_0)$ of coefficients of the (unique) polynomial $f = \sum_{k=0}^{n-1} a_k x^k \in \mathbb{F}[x]$ which satisfies $\forall (i_j, m_j) \in \mathcal{P} : f(i_j) = m_j$. This polynomial can easily be determined by Lagrange interpolation.

iDecode($\mathcal{S}, i$) On input of $\mathcal{S} = (a_{n-1}, \ldots, a_0)$ and index $i \in \mathcal{I}$ this algorithm outputs the evaluation $m = f(i) = \sum_{k=0}^{n-1} a_k i^k$ of f at position i.

The correctness of the proposed IHME scheme is obvious. Its index-hiding ability is assured by the following theorem, proven in the full version of this paper based on the fact that the distributions of the encodings of I_0 and I_1 are identical.

Theorem 1 (Index-Hiding Property of our IHME Construction). *The proposed IHME scheme provides perfect index-hiding.*

In our solution for the group discovery problem in AH-AKE protocols, the index set $\mathcal{I}$ is identified with the set of all possible groups. U_1 follows, for each of the n groups that it is affiliated with, the computation rules of a single-group AH-AKE handshake. The computed first messages from all these handshake instances are then encoded into a single structure using the IHME approach by considering the identifiers of the n groups (e.g. hashes of the public parameters) as indices for the corresponding messages. The encoding is sent over to U_2 which extracts the handshake messages for only the groups it is affiliated with. Note that for all matching groups G_i (i.e. groups in which both U_1 and U_2 are members) the first messages of handshake instances are correctly transferred from U_1 to U_2. The IHME technique is then applied independently to all subsequently exchanged handshake messages. Observe that for the secure deployment of IHME (as per Definition 2) it is essential that messages exchanged between users in the given single-group AH-AKE handshake are polynomially indistinguishable from random in $\mathcal{M} = \mathbb{F}$. This property is satisfied by some protocols, in particular by the linkable AH-AKE protocol from [13] that is underlying our construction with implicit group discovery, as presented in Section 4.

On General Performance of IHME. The IHME technique suggested above has the nice property of zero message expansion: in iEncode( ), the length of the input messages is equal to the length of the output IHME encoding $\mathcal{S}$. The communication complexity of our protocol is therefore $O(n)$. Since users perform computations of a single-group handshake only for groups in which they are

members, they need to participate in only n group handshakes. Thus, in contrast to possible solutions from Section 2.2 our IHME technique with perfect index-hiding can solve the group discovery problem with *linear* communication and computation complexity[1] as summarized in Table 1.

Table 1. Solutions for the group discovery problem with n input groups per participant

Technique	Computation (AH-AKE invocations)	Communication	Remarks
Naïve approach	$O(n^2)$	$O(n^2)$	
Hashing into bins	$O(n \log n)$	$O(n \log n)$	
Balanced allocation hashing	$O(n \log \log n)$	$O(n \log \log n)$	not privacy preserving
Our perfect IHME	$O(n)$	$O(n)$	

We notice that the general technique for solving group discovery in AH-AKE schemes based on our IHME construction clearly outperforms all other solutions suggested above.

4 Affiliation-Hiding Key Exchange with Group Discovery

In this section we illustrate how the IHME technique can be used to construct a concrete AH-AKE protocol with *implicit* solution to the group discovery problem. Our protocol is based on the state-of-the-art linkable AH-AKE (LAH-AKE) scheme from [13], which allows to privately check the match of a single group per user and handshake session only. This protocol has the nice property that its messages remain indistinguishable from random strings to anyone who is not a group member. Therefore, we can directly apply IHME to compress the complexity of group discovery without assuming any further building blocks. However, in order to address the group discovery problem implicitly we have to update the syntax of LAH-AKE protocols.

4.1 Syntax of LAH-AKE with Implicit Group Discovery

A LAH-AKE scheme, which admits *multiple* group credentials as input per user and session, denoted $\mathsf{MLAH\text{-}AKE}$, consists of four algorithms:

$\mathsf{CreateGroup}(1^\kappa)$**.** This probabilistic algorithm sets up a new group G and is executed by the corresponding group authority (GA). On input of security parameter 1^κ it generates a public/private group key pair $(G.\mathsf{pk}, G.\mathsf{sk})$, initializes the group's pseudonym revocation list $G.\mathsf{prl}$ to $\emptyset$ and outputs public group parameters $G.\mathsf{par} = (G.\mathsf{pk}, G.\mathsf{prl})$, and private key $G.\mathsf{sk}$. It is assumed that public key $G.\mathsf{pk}$ may serve as a non-ambiguous identifier for group G.

[1] The overhead of interpolating the polynomial in the IHME method is $O(n^2)$ multiplications. However, the overhead of these operations is negligible compared with the overhead of computing exponentiations in the AH-AKE handshake protocols.

$\mathsf{AddUser}(U, G)$**.** This is a protocol that is executed between a prospective group member U and the group authority of G. User U presents a pseudonym id and is issued a private membership credential $\mathsf{cred}_{G.\mathsf{pk}}$ for id in group G. It is legitimate for users to register the same pseudonym id in different groups. The communication channel between U and G is assumed to be authentic.

$\mathsf{Handshake}(U_i, U_j)$**.** This is the key exchange protocol (handshake), executed between two users U_i and U_j. The input for U_i is $(\mathsf{id}_i, \mathcal{G}_i, r_i)$ where id_i is his pseudonym, $\mathcal{G}_i$ is a set of triples of the form $(G.\mathsf{pk}, \mathsf{cred}_{G.\mathsf{pk}}, G.\mathsf{prl})$, and $r_i \in \{\mathsf{init}, \mathsf{resp}\}$. All values $\mathsf{cred}_{G.\mathsf{pk}}$ in $\mathcal{G}_i$ are credentials previously registered in the respective group $G.\mathsf{pk}$ by using the $\mathsf{AddUser}$ algorithm on pseudonym id_i. By $G.\mathsf{prl}$ we denote the respective pseudonym revocation list. For user U_j the protocol's input $(\mathsf{id}_j, \mathcal{G}_j, r_j)$ is defined analogously.

Users keep track of the state of created $\mathsf{Handshake}(\mathsf{id}, \mathcal{G}, r)$ protocol sessions π through session variables that are initialized as follows: $\pi.\mathsf{state} \leftarrow \mathsf{running}$, $\pi.\mathsf{id} \leftarrow \mathsf{id}$, $\pi.\mathcal{G} \leftarrow \mathcal{G}$ 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 κ), the handshake partner's pseudonym is assigned to $\pi.\mathsf{partner}$, and $\pi.\mathsf{groups}$ holds a non-empty set of group identifiers.

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

Definition 3 (Correctness of $\mathsf{MLAH\text{-}AKE}$**).** *Assume that users U_i and U_j participate in a* $\mathsf{Handshake}$ *protocol with inputs $(\mathsf{id}_i, \mathcal{G}_i, r_i)$ and $(\mathsf{id}_j, \mathcal{G}_j, r_j)$, 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 groups' revocation lists. The* $\mathsf{MLAH\text{-}AKE}$ *scheme is called* correct *if (1) π_i and π_j complete in the same state, which is* $\mathsf{accepted}$ *iff $(\mathcal{G}_\cap \neq \emptyset \wedge r_i \neq r_j)$, and (2) if both sessions accept then $(\pi_i.\mathsf{key}, \pi_i.\mathsf{partner}, \pi_i.\mathsf{id}) = (\pi_j.\mathsf{key}, \pi_j.\mathsf{id}, \pi_j.\mathsf{partner})$ and $\pi_i.\mathsf{groups} = \pi_j.\mathsf{groups} = \mathcal{G}_\cap$.*

4.2 Number-Theoretic Assumptions and Building Blocks

Assumptions. The security of our protocol builds on the hardness of the following RSA problem, which is a standard RSA assumption for *safe* moduli, also used in the design of AH-AKE and SH protocols from [20, 12, 13].

Definition 4 (RSA Assumption on Safe Moduli). *Let* $\mathsf{RSA\text{-}G}(\kappa')$ *be a probabilistic algorithm that outputs pairs (n, e) where (a) $n = pq$ for random κ'-bit primes $p \neq q$, (b) $p = 2p' + 1, q = 2q' + 1$ for primes p', q', and (c) $e \in \mathbb{Z}_{\varphi(n)}$ is coprime to $\varphi(n)$. The RSA-success probability of a PPT solver $\mathcal{A}$ is defined as*

$$\mathsf{Succ}^{\mathsf{rsa}}_{\mathcal{A}}(\kappa') = \Pr\big[(n, e) \leftarrow \mathsf{RSA\text{-}G}(\kappa');\ z \xleftarrow{\$} \mathbb{Z}^*_n;\ m \leftarrow \mathcal{A}(n, e, z) \text{ with } m^e = z\big].$$

The RSA assumption on safe moduli *states that the maximum RSA-success probability* $\mathsf{Succ}^{\mathsf{rsa}}(\kappa')$ *(defined over all PPT solvers $\mathcal{A}$) is negligible in κ'.*

Perfect IHME. Let κ, κ' be security parameters (where κ' is polynomially dependent on κ) and p be the smallest prime number satisfying $p > 2^{2\kappa'+\kappa}$. A central building block of our MLAH-AKE protocol is the perfect IHME scheme presented in Section 3, defined over finite field $\mathbb{F} = GF(p)$. (As alternative, finite field $GF(2^{2\kappa'+\kappa})$ could be used. However, in this paper we use $GF(p)$ to simplify notations.)

Hash Functions. The protocol makes use of three different hash functions (modeled as random oracles):

$$H : \{0,1\}^* \to [0, p-1] \qquad H' : \{0,1\}^* \to \{0,1\}^\kappa \qquad H^* : \{0,1\}^* \to [0, p-1]$$

For convenience, for each $n \in \mathbb{N}$ of length $2\kappa'$ we define

$$H_n : \{0,1\}^* \to \mathbb{Z}_n;\ x \mapsto H^*(n \,\|\, x) \bmod n.$$

4.3 The Protocol Specification

$\mathsf{CreateGroup}(1^\kappa)$ *Algorithm.* Group authorities setup new group parameter sets as follows: in a first step, two κ'-bit primes $p, q \in \mathbb{N}$ with $p = 2p' + 1$ and $q = 2q' + 1$ for prime numbers p' and q' are picked. For $n = pq$ an exponent $e \in \mathbb{Z}_{\varphi(n)}$ which is coprime to $\varphi(n) = (p-1)(q-1) = 4p'q'$ is chosen, and $d = e^{-1} \pmod{\varphi(n)}$ is computed.

As $\mathbb{Z}_n^* \cong \mathbb{Z}_p^* \times \mathbb{Z}_q^*$ the largest element order in $\mathbb{Z}_n^*$ is $\mathrm{lcm}(\varphi(p), \varphi(q)) = 2p'q' = \varphi(n)/2$, and hence $\mathbb{Z}_n^*$ is not cyclic. Let the CreateGroup algorithm pick an element $g \in \mathbb{Z}_n^*$ with $\mathrm{ord}(g) \geq p'q'$ and $g^{p'q'} \neq \pm 1$. It follows that $\mathrm{ord}(g) = 2p'q'$ and that $-1 \notin \langle g \rangle$, and we may infer $\mathbb{Z}_n^* \cong \langle -1 \rangle \times \langle g \rangle$. As about a half of the elements in $\mathbb{Z}_n^*$ have the desired two properties [13], g can easily be found by random sampling and testing.

The algorithm sets $G.\mathsf{pk} \leftarrow (n, g, e)$, $G.\mathsf{prl} \leftarrow \emptyset$ and $G.\mathsf{sk} \leftarrow d$, and outputs $G.\mathsf{par} = (G.\mathsf{pk}, G.\mathsf{prl})$ and $G.\mathsf{sk}$ as public and private key, respectively.

$\mathsf{AddUser}(U, G)$ *Protocol.* To admit a new member to a group the corresponding GA issues as credential $\mathsf{cred}_{G.\mathsf{pk}}$ the full domain hash RSA signature [3] on the pseudonym id specified by the user, i.e. $\mathsf{cred}_{G.\mathsf{pk}} = H_n(\mathsf{id})^{G.\mathsf{sk}} \bmod n$. The communication between U and G is assumed to be authentic and confidential.

$\mathsf{Handshake}((\mathsf{id}_i, \mathcal{G}_i, \mathsf{init}), (\mathsf{id}_j, \mathcal{G}_j, \mathsf{resp}))$ *Protocol.* The handshake protocol is executed between two users U_i and U_j holding corresponding pseudonyms id_i and id_j as well as membership lists $\mathcal{G}_i$ and $\mathcal{G}_j$, respectively. Each user's membership list contains triples $(G.\mathsf{pk}, \mathsf{cred}_{G.\mathsf{pk}}, G.\mathsf{prl})$ for all affiliations of that user. The handshake protocol is detailed in Figure 1.

The lines where the numbering is formatted in bold face coincide with those from [13]; in particular this includes the calculation of the $\theta = (-1)^b g^t \mathsf{cred} + kn$ values (lines 4–8), the partial keys $r = (\theta^e H_n(\mathsf{id})^{-1})^{2t}$ and the confirmation messages c (lines 19–20). Lines 17 and 22 effectively implement user revocation.

Innovative in this protocol is the parallel transmission of multiple θ and c values encoded as IHME-structures $\mathcal{S}$ and $\mathcal{S}'$, respectively. Note the usage of RSA moduli n as group specific indices. The lists $\mathcal{T}$ and $\mathcal{R}$ are not transmitted, but hold the inner state of the protocol.

Note that protocol correctness demands that the string X in the third code block (lines 26–32) is mounted in the same order for both U_i and U_j. This can be achieved by letting the corresponding FOR-loop iterate in the order of ascending n_i.

$\mathsf{Revoke}(G, \mathsf{id})$ *Algorithm.* The revocation of pseudonyms is handled by the particular group authority of G by including the pseudonym id into the corresponding pseudonym revocation list $G.\mathsf{prl}$. We assume that this list is distributed authentically to all members of the group.

4.4 Protocol Correctness, Efficiency Analysis, and Optimizations

In the following we show why the protocol is correct and discuss its concrete efficiency, including possible optimizations. The actual security analysis of the protocol is postponed to Section 6 after the specification of the security model with regard to the group discovery problem.

Correctness. Let $(n, g, e) = G.\mathsf{pk}$ denote a group to which U_i and U_j are both registered, i.e. U_i owns a credential $\mathsf{cred}_{G.\mathsf{pk}} = H_n(\mathsf{id}_i)^d \bmod n$ for pseudonym id_i, while U_j possesses credential $H_n(\mathsf{id}_j)^d \bmod n$ for id_j. Then, by construction, the value $\mathsf{iDecode}(\mathcal{S}_i, n)$ has the form $\theta_i = (-1)^{b_i} g^{t_i} H_n(\mathsf{id}_i)^d \bmod n$. The value r_j computed by U_j for group $G.\mathsf{pk}$ is

$$r_j = (\theta_i{}^e H_n(\mathsf{id}_i)^{-1})^{2t_j} = (((-1)^{b_i} g^{t_i} H_n(\mathsf{id}_i)^d)^e H_n(\mathsf{id}_i)^{-1})^{2t_j} = g^{2et_it_j} \pmod n.$$

Symmetrically, for group $G.\mathsf{pk}$ user U_i computes the same value $r_i = g^{2et_it_j}$ from $\mathsf{iDecode}(\mathcal{S}_j, n)$ and id_j. The protocol's correctness is now verifiable by inspection. The case that $H_n(\mathsf{id})^{-1}$ is not defined occurs only with negligible probability.

Efficiency Analysis. The computational costs of the $\mathsf{Handshake}$ protocol mainly consist of the exponentiations by t_i (resp. t_j), which are executed twice per group. Precisely, the computational effort a user U_i starting a $\mathsf{Handshake}(\mathsf{id}_i, \mathcal{G}_i, \mathsf{init})$ protocol session has to stem can be estimated by $2|\mathcal{G}_i|$ exponentiations with modulus size $2\kappa'$, where $|\mathcal{G}_i|$ denotes the number of groups U_i is member in. That is, the computational overhead scales linearly with the number of affiliations. Also the size of IHME-encodings $\mathcal{S}, \mathcal{S}'$ grows linearly with the number of affiliations. More precisely, the total communication complexity of the handshake amounts to $2(2\kappa' + \kappa)(|\mathcal{G}_i| + |\mathcal{G}_j|)$ bits (without considering pseudonyms that can be short).

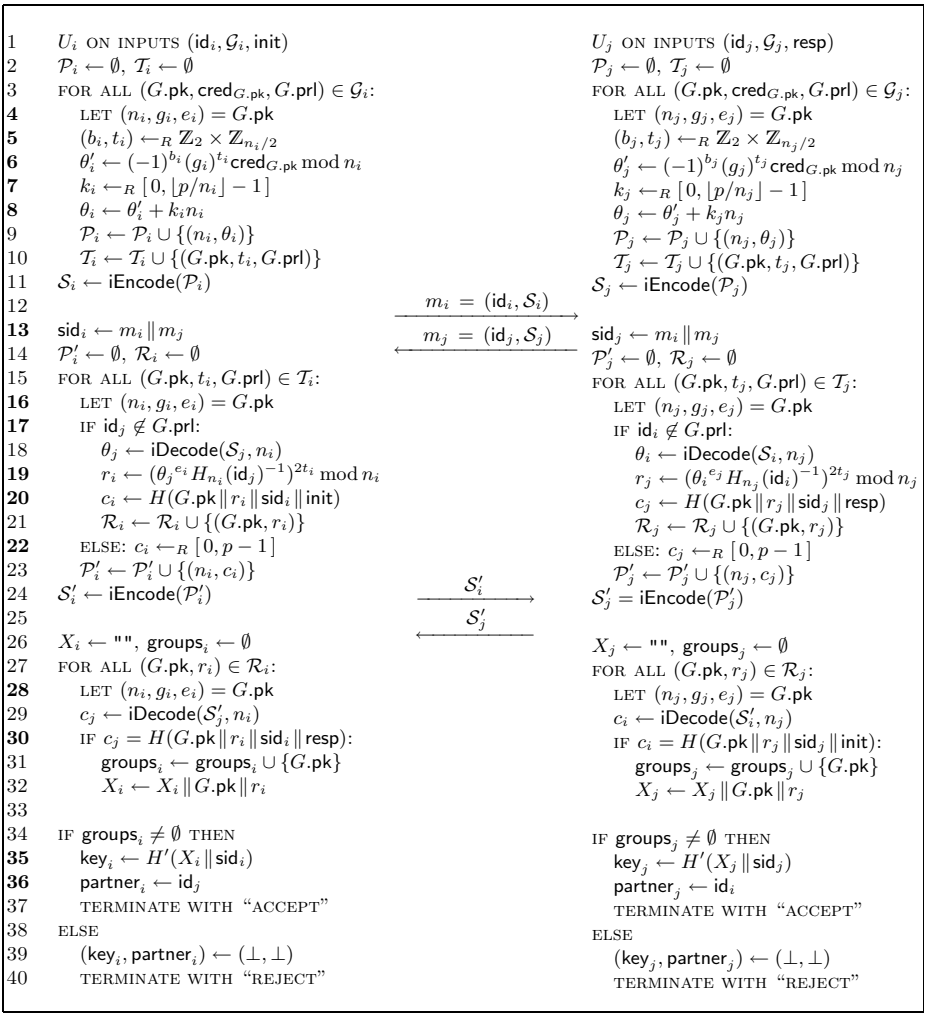

Fig. 1. Specification of $\mathsf{Handshake}((\mathsf{id}_i, \mathcal{G}_i, \mathsf{init}), (\mathsf{id}_j, \mathcal{G}_j, \mathsf{resp}))$

Further Optimizations. The following optimization idea would render the Handshake protocol slightly more efficient. It is based on the observation that in the Handshake protocol both the θ values and the confirmation tags c are transferred by IHME using the same finite field $\mathbb{F} = GF(p)$. The protocol would stay secure if the confirmation tags c would be shortened from $2\kappa' + \kappa$ bits to just κ bits. To implement this, the confirmation tags c have to be computed by an auxiliary hash function whose range is $\mathbb{F}'$, where $\mathbb{F}'$ is a finite field of order $\approx 2^\kappa$, and the IHME scheme for transferring $\mathcal{S}'$ would have to be defined over $\mathbb{F}'$ instead of $\mathbb{F}$. The deployment of this idea would save $2\kappa'(|\mathcal{G}_i| + |\mathcal{G}_j|)$ communication bits, resulting in the total complexity of $(2\kappa' + 2\kappa)(|\mathcal{G}_i| + |\mathcal{G}_j|)$.

5 Security Model for LAH-AKE with Group Discovery

In this section we introduce the security model for LAH-AKE protocols while taking into account various challenges implied by the group discovery problem and the updated syntax of such protocols, by modifying the current state-of-the-art model from [13].

5.1 Adversary Model

After describing the basic set of adversarial queries we define two security properties: 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. The following definition is helpful to keep track on executed handshake sessions:

Definition 5 (Session IDs and Partnered Session). *For a* Handshake *session* π *with* $\pi.\mathsf{state} = \mathsf{accepted}$ *the* session id $\pi.\mathsf{sid}$ *is a value that uniquely identifies* π *in the set of all protocol sessions started by* $\pi.\mathsf{id}$*. Two sessions* π, π' *are called* partnered *if* $\pi.\mathsf{state} = \pi'.\mathsf{state} = \mathsf{accepted}$ *and* $(\pi.\mathsf{sid}, \pi.\mathsf{id}, \pi.\mathsf{partner}) = (\pi'.\mathsf{sid}, \pi'.\mathsf{partner}, \pi'.\mathsf{id})$.

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.

$\mathsf{Handshake}(\mathsf{id}, \mathcal{G}, r)$**.** This query lets the holder of pseudonym id start a new session π of the Handshake protocol. It receives as input a set $\mathcal{G}$ of public group keys $G.\mathsf{pk}$ and a role identifier $r \in \{\mathsf{init}, \mathsf{resp}\}$ that determines whether the session will act as protocol initiator or responder. Session variable $\pi.\mathsf{revealed}$ is initialized to false. If there is a group $G.\mathsf{pk}$ listed in $\mathcal{G}$ for which id has no private credential $\mathsf{cred}_{G.\mathsf{pk}}$ 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 the flag $\pi.\mathsf{revealed}$ is set to true and $(\pi.\mathsf{state}, \pi.\mathsf{key}, \pi.\mathsf{groups})$ is returned.

$\mathsf{Corrupt}(\mathsf{id}, G)$**.** Membership credential $\mathsf{cred}_{G.\mathsf{pk}}$ of pseudonym id in group G is passed to the adversary. Note that this query models the possibility of selective corruptions.

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

5.2 Linkable Affiliation-Hiding Security

We start with the property of Linkable Affiliation-Hiding (LAH). At a high level the goal here is to protect the disclosure of non-matching affiliations of handshake participants. 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 the two sets of affiliations $\mathcal{G}_0^*$ or $\mathcal{G}_1^*$ some challenge session π^* is using. The adversary can also invoke any number of handshake sessions, and ask Reveal and Corrupt queries at will. This intuition is formalized as follows.

Definition 6 (LAH-Security). *Let* MLAH-AKE $= \{$CreateGroup, AddUser, Handshake, Revoke$\}$*, b be a randomly chosen bit, and $\mathcal{Q} = \{$Handshake, Send, Reveal, Corrupt, Revoke$\}$ denote the set of queries the adversary $\mathcal{A}$ has access to. We consider the following game between a challenger and the adversary $\mathcal{A}$:*

$\mathsf{Game}^{\mathsf{lah},b}_{\mathcal{A},\mathsf{MLAH\text{-}AKE}}(\kappa,n,m)$:

- *the challenger creates users $U_1,\ldots,U_n$ and pseudonyms $ID = \{\mathsf{id}_1,\ldots,\mathsf{id}_n\}$;*
- *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 $(\mathsf{id}^*,\mathcal{G}_0^*,\mathcal{G}_1^*,r^*)$ where $\mathsf{id}^* \in ID$, $\mathcal{G}_0^*,\mathcal{G}_1^* \subseteq \mathcal{G}$ 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 $\mathsf{Handshake}(\mathsf{id}^*,\mathcal{G}_b^*,r^*)$ 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 1 if all of the following hold; else the output is 0:*
  - *(a) $b = b'$,*
  - *(b) 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 $\mathsf{Reveal}(\pi^*)$ query was asked,*
  - *(c) no $\mathsf{Reveal}(\pi')$ query was asked for any* Handshake *session π' with $\mathcal{D}^* \cap \pi'.\mathcal{G} \neq \emptyset$ and $\pi'.\mathsf{partner} = \mathsf{id}^*$ that was in state* running *while π^* was in state* running*,*
  - *(d) no $\mathsf{Corrupt}(\mathsf{id},G)$ query with $(\mathsf{id},G) \in ID \times \mathcal{D}^*$ was asked before π^* left* running *state.*

We define $\mathsf{Adv}^{\mathsf{lah}}_{\mathcal{A},\mathsf{MLAH\text{-}AKE}}(\kappa,n,m) :=$

$$\left|\Pr[\mathsf{Game}^{\mathsf{lah},0}_{\mathcal{A},\mathsf{MLAH\text{-}AKE}}(\kappa,n,m)=1] - \Pr[\mathsf{Game}^{\mathsf{lah},1}_{\mathcal{A},\mathsf{MLAH\text{-}AKE}}(\kappa,n,m)=1]\right|$$

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

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

5.3 Authenticated Key Exchange Security

Authenticated Key Exchange (AKE) security of MLAH-AKE schemes is modeled similarly to [13] where the goal of the adversary $\mathcal{A}$ is to distinguish the session key computed by some test session π^* from a random value. $\mathcal{A}$ may invoke any number of handshake sessions, corrupt pseudonyms, and reveal established sessions keys at will as long as it does not obtain the session key computed by π^* in some trivial way. For the formal definition of AKE-security we introduce two further queries Reveal' and Test (the latter with secret parameter $b \in \{0,1\}$) and the new session variable $\pi.\mathsf{tested}$, which is initially set to false.

$\mathsf{Reveal}'(\pi)$**.** This query is like the original Reveal query except that it is ignored if $\pi.\mathsf{tested} = \mathsf{true}$ or $\pi'.\mathsf{tested} = \mathsf{true}$ for any session π' partnered with π.

$\mathsf{Test}(\pi)$**.** If π is fresh (see Definition 7) then $\pi.\mathsf{tested}$ is set to true and a key K is returned, where $K = \pi.\mathsf{key}$ if $b = 1$ and $K \xleftarrow{\$} \{0,1\}^\kappa$ otherwise. In addition, $\pi.\mathsf{groups}$ is returned. This query may be asked at most once.

Definition 7 (Session Freshness). *A session π invoked in response to a* $\mathsf{Handshake}(\mathsf{id}, \mathcal{G}, r)$ *query is called* fresh *if all of the following hold:*

(a) $\pi.\mathsf{state} = \mathsf{accepted}$ *and* $\pi.\mathsf{revealed} = \mathsf{false}$.
(b) $\pi'.\mathsf{revealed} = \mathsf{false}$ *for any session π' that is partnered with π.*
(c) there exists a group $G \in \pi.\mathsf{groups}$ *such that neither* $\mathsf{Corrupt}(\pi.\mathsf{id}, G)$ *nor* $\mathsf{Corrupt}(\pi.\mathsf{partner}, G)$ *has been asked before* $\pi.\mathsf{state}$ *was set to* accepted.

Conditions (a)–(c) imply the usual constraints of key secrecy models that include forward secrecy [5]. Condition (c) demands that a single group G for which $\pi.\mathsf{id}$ and $\pi.\mathsf{partner}$ remain uncorrupted until protocol acceptance suffices for the tested session to be considered fresh.

Definition 8 (AKE-Security with Forward Secrecy). *Let* $\mathsf{MLAH\text{-}AKE} = \{\mathsf{CreateGroup}, \mathsf{AddUser}, \mathsf{Handshake}, \mathsf{Revoke}\}$*, b be a randomly chosen bit, and* $\mathcal{Q} = \{\mathsf{Handshake}, \mathsf{Send}, \mathsf{Corrupt}, \mathsf{Revoke}, \mathsf{Reveal}', \mathsf{Test}\}$ *denote the set of queries the adversary $\mathcal{A}$ has access to. We consider the following game between a challenger and the adversary $\mathcal{A}$:*

$\mathsf{Game}^{\mathsf{ake},b}_{\mathcal{A},\mathsf{MLAH\text{-}AKE}}(\kappa, n, m)$:

- *the challenger creates users $U_1, \ldots, U_n$ and pseudonyms $\mathsf{id}_1, \ldots, \mathsf{id}_n$;*
- *the challenger creates m groups $G_1, \ldots, G_m$ and registers 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 $\mathcal{A}^{\mathcal{Q}}$ asks $\mathsf{Test}(\pi^*)$ to a fresh session π^*;*
- *$\mathcal{A}^{\mathcal{Q}}$ continues interacting via queries until it terminates and outputs bit b', which is the output of the game.*

We define $\mathsf{Adv}^{\mathsf{ake}}_{\mathcal{A},\mathsf{MLAH\text{-}AKE}}(\kappa, n, m) := \left|2\Pr[\mathsf{Game}^{\mathsf{ake},b}_{\mathcal{A},\mathsf{MLAH\text{-}AKE}}(\kappa, n, m) = b] - 1\right|$ *and denote with* $\mathsf{Adv}^{\mathsf{ake}}_{\mathsf{MLAH\text{-}AKE}}(\kappa, n, m)$ *the maximum advantage over all PPT adversaries $\mathcal{A}$. We say that* MLAH-AKE *is* AKE-secure *if this advantage is negligible in κ (for all n, m polynomially dependent on κ).*

6 Security Analysis of Our Protocol

Following the extended definitions from the previous sections we prove that our MLAH-AKE protocol from Section 4 satisfies the desired goals. The respective proofs are provided in the full version of this paper.

Theorem 2 (Linkable Affiliation-Hiding Security). *The* MLAH-AKE *protocol from Section 4 is LAH-secure under the RSA assumption on safe moduli in the random oracle model.*

Theorem 3 (Authenticated Key Exchange Security). *The* MLAH-AKE *protocol from Section 4 is AKE-secure under the RSA assumption on safe moduli in the random oracle model.*

The dependence on the RSA assumption and the random oracle model stems from the underlying LAH-AKE protocol [13] (which is proven secure under these assumptions). Note that our IHME approach can be deployed independently of any number-theoretical or non-standard assumption.

7 Conclusion

We discussed several solutions to the open problem of *efficient group discovery* in AH-AKE protocols. We stress that without efficient group discovery, existing AH-AKE schemes (that provide support for only one input group per user and protocol session) would remain of very limited use. Throughout the paper we described how to perform group discovery more efficiently than by the naïve combinatorial approach, for which the computational and communication complexity is $O(n^2)$ (where n is the number of input groups per participant). Our most efficient solution came from the use of a new primitive, called *Index-Hiding Message-Encoding (IHME)*. In addition to the definition of IHME and its *index-hiding* property we gave a construction for which the property holds unconditionally. We then demonstrated how IHME can be applied to the state-of-the-art linkable AH-AKE protocol from [13] in order to discover groups in *linear complexity* $O(n)$. Our construction is supported by appropriate definitions of security and proofs.

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 (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.: 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)
4. Camenisch, J., Zaverucha, G.M.: Private Intersection of Certified Sets. In: Dingledine, R., Golle, P. (eds.) FC 2009. LNCS, vol. 5628, pp. 108–127. Springer, Heidelberg (2009)

5. 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)
6. 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)
7. Cristofaro, E., Jarecki, S., Kim, J., Tsudik, G.: Privacy-Preserving Policy-Based Information Transfer. In: Goldberg, I., Atallah, M.J. (eds.) PETS 2009. LNCS, vol. 5672, pp. 164–184. Springer, Heidelberg (2009)
8. Cristofaro, E., Tsudik, G.: Practical Private Set Intersection Protocols with Linear Computational and Bandwidth Complexity. Cryptology ePrint Archive, Report 2009/491. To appear in Financial Cryptography and Data Security. LNCS, vol. 6052. Springer, Heidelberg (2010)
9. Freedman, M.J., Nissim, K., Pinkas, B.: Efficient Private Matching and Set Intersection. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 1–19. Springer, Heidelberg (2004)
10. Hazay, C., Lindell, Y.: Efficient Protocols for Set Intersection and Pattern Matching with Security Against Malicious and Covert Adversaries. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 155–175. Springer, Heidelberg (2008)
11. Hazay, C., Nissim, K.: Efficient Set Operations in the Presence of Malicious Adversaries. Cryptology ePrint Archive, Report 2009/594. In: Nguyen, P.Q., Pointcheval, D. (Eds.) PKC 2010. LNCS, vol. 6056, pp. 312–331. Springer, Heidelberg (2010)
12. Jarecki, S., Kim, J., 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)
13. Jarecki, S., Kim, J., Tsudik, G.: Beyond Secret Handshakes: Affiliation-Hiding Authenticated Key Exchange. In: Malkin, T.G. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 352–369. Springer, Heidelberg (2008)
14. 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)
15. Jarecki, S., Liu, X.: Efficient Oblivious Pseudorandom Function with Applications to Adaptive OT and Secure Computation of Set Intersection. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 577–594. Springer, Heidelberg (2009)
16. Jarecki, S., Liu, X.: Private Mutual Authentication and Conditional Oblivious Transfer. In: Halevi, S. (ed.) Advances in Cryptology - CRYPTO 2009. LNCS, vol. 5677, pp. 90–107. Springer, Heidelberg (2009)
17. Kissner, L., Song, D.X.: Privacy-Preserving Set Operations. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 241–257. Springer, Heidelberg (2005)
18. Raab, M., Steger, A.: Balls Into Bins — A Simple and Tight Analysis. In: Rolim, J.D.P., Serna, M., Luby, M. (eds.) RANDOM 1998. LNCS, vol. 1518, pp. 159–170. Springer, Heidelberg (1998)
19. Tsudik, G., Xu, S.: A Flexible Framework for Secret Handshakes. In: Danezis, G., Golle, P. (eds.) PET 2006. LNCS, vol. 4258, pp. 295–315. Springer, Heidelberg (2006)
20. Vergnaud, D.: RSA-Based Secret Handshakes. In: Ytrehus, Ø. (ed.) WCC 2005. LNCS, vol. 3969, pp. 252–274. Springer, Heidelberg (2006)
21. 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)