Abstract—Nowadays, users are increasingly concerned about individual privacy in cyberspace and Internet. In this paper, we propose the concept of private handshakes with optional accountability, which allows the two users in handshaking to decide real time whether or not to make their interactions accountable. Such optionally accountable private handshaking protocols are a more flexible privacy-preserving authentication primitive than unlinkable secret handshakes and private handshakes. We formulate a formal definition for optionally accountable private handshakes, and propose a concrete scheme based on bilinear pairings.

I. INTRODUCTION

With the prevalence of Internet, more and more services and transactions are becoming electronic, e.g., e-commerce and e-banking. Users, on one hand, fervently embrace the use of Internet and enjoy the great convenience it brings, while on the other hand, become increasingly conservative in disclosing individual information when using Internet. As a consequence, privacy-preserving techniques that can make users accomplish the desired functionalities and at the same time maintain their individual privacy are expected to play a key part in a wider adoption of Internet applications. Secret handshaking protocols are a privacy-preserving authentication primitive that enables a pair of users from the same group, each holding a group credential, to authenticate each other, while guaranteeing that 1) non-members learn nothing on the handshake including whether the two users recognize each other; 2) a non-member cannot pretend to be a member, and in turn perform handshakes with members. Secret handshaking protocols turn out to be quite useful, especially at the time when users are more and more concerned about individual privacy, reluctant to reveal their activities and preferences over Internet. Therefore, since first formulated by Balfanz et al. [2], secret handshake has attracted enormous attention.

In its original form, secret handshakes are linkable, since a user needs to repeatedly use the same pseudonym (or ID) in different handshake sessions. However, unlinkability is often a pursued goal for privacy-preserving protocols. Unlinkable secret handshakes were thus proposed, e.g., [8]. We should point out that it trivially achieves unlinkability using any secret handshaking protocol, as long as a user can have in possession an indefinite number of pseudonyms (and credentials). Note that to use a pseudonym for secret handshakes, there must be a group credential bound to that pseudonym. Hence this trivial approach is not practical, since it requires a user to apply for an ample number of credentials from the group administrator. In contrast, the interesting part of unlinkable secret handshakes is that unlinkability is attained by each user using a single credential. We here should also clarify the differences between unlinkable secret handshaking protocols and other extensively studied privacy-preserving authentication primitives such as group signature (e.g., [1]) and anonymous credential (e.g., [6]): unlinkable secret handshakes are affiliation-hiding in the sense that non-members cannot learn to which group the communicating users belong; but group signature and anonymous credential do not intend to hide the users’ affiliation at all.

Among others, traceability is a property offered by unlinkable secret handshakes. Traceability allows the group administrator to find out the users who have engaged in a secret handshake process (e.g., in the case of certain exceptional events), based on the protocol transcript. Although traceability is often a desired feature in a unlinkable privacy-preserving system, possession of it actually weakens privacy protection of the underlying system: anyhow a certain party (group administrator in the setting of secret handshakes), however trustful it is, can violate unlinkability. Recently, Hoepman [9] proposed the notion of private handshakes, which are essentially unlinkable secret handshakes without traceability. Offering no traceability whatsoever, private handshakes may be useful in certain circumstances, e.g., a user in some applications may not want anyone to identify him in any means.

In this work, we propose the concept of private handshakes with optional accountability, which allows the two users in handshaking to negotiate real time whether to make their interactions accountable. To make it clearer, such optionally accountable private handshakes can be viewed as a primitive between private handshakes and unlinkable secret handshakes, as shown in Figure 1. In particular, if the users agree to achieve no accountability, then the protocol is private handshakes at one end of the spectrum; on the contrary, if the users decide to achieve accountability, then the protocol boils down to be similar to unlinkable secret handshakes at the other end of the spectrum (more precisely, we call it accountable private handshakes). Optionally accountable private handshakes clearly offer the users the flexibility to choose the level of

1Here we somehow abuse the notion of accountability, which in our current context refers to the removal of anonymity.
2We stress that accountable private handshakes are not precisely unlinkable secret handshakes. The distinction is that in the former, the communicating users directly violate unlinkability, while in the latter only the group administrator can violate unlinkability.
privacy protection in their handshakes, i.e., accountability or no accountability at all.

Organization. The rest of the paper is organized as follows. In Section II, we review related work. We then formulate a model for optionally accountable private handshakes in Section III, followed by our proposed scheme and security analysis in Section IV. Section V concludes the paper.

II. RELATED WORK

The notion of secret handshake traces back to private match making [4], where users with the same “target” can locate and authenticate each other secretly. However, in private match making, it is likely that any user can identify members if he correctly guesses the “target”.

Balfanz et al. [2] first formulated the notion of secret handshake and revived the interest in such a privacy-preserving authentication primitive. Their protocols are based on bilinear pairings, and secure under the bilinear computational Diffie-Hellman assumption [3] and the random oracle model. Subsequently, Castelluccia et al. [5] proposed secret handshaking protocols, with security under computational Diffie-Hellman assumption. RSA-based secret handshaking protocols were due to Jarecki et al. [7] and Vergnaud [11].

The above secret handshaking protocols are inherently linkable, unless using distinct credentials each time. Achieving unlinkability directly using these protocols requires a user to have in possession an indefinite number of credentials, which is unlikely to be feasible in practice. A more satisfactory solution is that a user is able to reuse his credential while attaining unlinkability. Xu and Yung [12] realized the use of reusable credentials in secret handshakes. However, their scheme only achieves a weak version of unlinkability, i.e., \( k \)-anonymity, which allows the attacker to learn that a communicating user is from one of the \( k \) publicly known groups. Tsudik and Xu [10] proposed a protocol achieving (full) unlinkability, but all members from the same group are required to share a group secret. One of the main drawbacks of sharing secret is that the propagation of revocation information must be strictly synchronized; otherwise, some members will fail to authenticate. Jarechi and Liu’s unlinkable secret handshaking protocol [8] does not rely on group members sharing secret, and it also tolerates to some extent unsynchronized propagation of revocation information.

Unlinkable secret handshakes allow the group administrator to violate unlinkability, such that the administrator can find out the members who have engaged in handshakes, based on the handshaking protocol transcripts. Traceability is often a useful feature in a privacy-preserving system. Nevertheless, in some applications users may not be happy to be traced by any one. This motivated Hoepman [9] to propose the concept of private handshakes, which are essentially unlinkable secret handshakes, but do not have traceability to any party.

As we discussed earlier, private handshakes with optional accountability should be viewed as a primitive between private handshakes and unlinkable secret handshakes, allowing the two communicating users to decide real time whether to proceed their handshake as private one or accountable one. It is thus a more flexible handshake tool than the existing ones.

III. MODEL

An optionally accountable private handshaking system consists of a set \( G \) of groups, a set \( U \) of users, and a set of \( A \) of group administrators who create groups and enrol users in groups. A user may or may not be affiliated to a group (for simplicity, we assume that a user belongs to at most one group). If a user belongs to a group, then he is a member of that group; otherwise, he is non-member of that group. An optionally accountable private handshaking system is composed of the following algorithms.

- **CreateGroup** \((1^n)\): On input a security parameter \( 1^n \), the algorithm, executed by a group administrator \( A \in A \), outputs a group secret \( s_G \in \{0,1\}^* \) for a group \( G \in G \).

- **EnrolUser** \((G, u)\): On input a group \( G \in G \) and a user \( u \in U \), the algorithm, executed by the group administrator \( A_G \) of \( G \), outputs a secret credential \( x_u \in \{0,1\}^* \) bound to the user’s identity (or pseudonym) \( u \). Note that \( x_u \) is generated using the group secret \( s_G \).

- **PrivHandshake** \((u_1, u_2, b)\): This is an interactive process between two users \( u_1, u_2 \in U \), governed by \( b \in \{0,1\} \). If \( b = 1 \), the interactions between \( u_1 \) and \( u_2 \) constitute a private handshake protocol; otherwise, the interactions are an accountable private handshake protocol. In either case, the algorithm outputs a shared key \( sk \in \{0,1\}^* \) between \( u_1 \) and \( u_2 \) if they belong to the same group, or \( \perp \) otherwise.

- **RevokeUser** \((G, u)\): On input a group \( G \in G \) and a user \( u \in U \), the algorithm, executed by the group administrator \( A_G \) of \( G \), revokes the membership of \( u \), and inserts \( u \) into the RevokedUserList of \( G \).

A. Security Requirements

An optionally accountable private handshaking system should satisfy the following security requirements.

- **Non-member exclusiveness.** A user Alice who does not belong to a group pretending to be a member is not able to successfully perform handshake with a genuine group.
member Bob; she cannot even learn anything on her counterpart, including whether he belongs to that group or not.

- **Unlinkability.** A non-member eavesdropper cannot tell apart the protocol runs involving the same user from those involving different users.

- **Optional accountability.** The two communicating users can decide real time whether to engage in countable or non-countable handshakes: if the former (i.e., accountable private handshakes), the two learn their respective counterparts’ identity; otherwise, the handshake is entirely unlinkable to them, without any traceability whatsoever (i.e., private handshakes).

IV. OUR CONSTRUCTION

In this section, we present a concrete optionally accountable private handshaking protocol. We first review the preliminary knowledges underlying our construction.

A. Preliminaries

Our construction is based on bilinear parings. Let $G_1, G_2$ be two cyclic groups of a large prime order $q$, then $\hat{e} : G_1 \times G_1 \to G_2$ is a bilinear map if for any $a, b \in \mathbb{Z}_q, P, Q \in G_1$, we have $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$. A bilinear map should be non-generate, i.e., not all $P, Q$ make $\hat{e}(P, Q)$ the generator in $G_2$. Weil and Tate pairings on supersingular elliptic curves are examples of such bilinear maps. In bilinear pairings, the Bilinear Computational Diffie-Hellman (BCDH) problem is assumed hard, which states that it is hard to compute $\hat{e}(P, P)^{abc}$ from $aP, bP, cP$ for random $a, b, c \in \mathbb{Z}_q$ and $P \in G_1$. Formally, for all probabilistic polynomial time (PPT) $A$, we define $\text{Adv}_{A}^{BCDH} = \Pr[A(aP, bP, cP) = \hat{e}(P, P)^{abc}]$ to be the advantage $A$ solves the BCDH problem, which is assumed negligible in the security parameter. The security of our proposed protocol is based on this BCDH assumption.

B. Details of Protocol

The system parameters include $\hat{e} : G_1 \times G_1 \to G_2$ as above, cryptographic hash functions $H_0 : \{0, 1\}^* \to G_1, H_1, H_2 : \{0, 1\}^* \to K$, and a semantically secure symmetric key encryption $E : \{0, 1\}^* \times K \to \{0, 1\}^*$ (the decryption algorithm is $D : \{0, 1\}^* \times K \to \{0, 1\}^*$), where $K$ denotes the appropriate key space.

- **CreateGroup($1^\kappa$):** Given the security parameter $1^\kappa$, a group administrator $A$ creates a group $G$ by selecting a random secret $s_G \in \mathbb{Z}_q$, specific to $G$. Note that the size of $q$ polynomially relates to $\kappa$. Then $A$ is the group administrator of $G$, denoted as $A_G$.

- **EnrolUser($G, u$):** To enrol user $u$ into group $G$ whose group secret is $s_G$, the corresponding group administrator $A_G$ computes and issues $u$ a credential $x_u = s_GH_0(u)$, where $u$ denotes the identity (or pseudonym) of the user.

- **PrivHandshake($u_1, u_2, b$):** The interactions between $u_1, u_2$, whose credentials are $x_{u_1} = s_GH_0(u_1)$ and $x_{u_2} = s_GH_0(u_2)$, respectively, are the following, aiming to authenticate each other and establish a common key for their subsequent communication (Figure 2 summarizes the details of the interactions).

**Step 1.** Let $u_1$ be the initiator. $u_1$ chooses a random number $r_1 \in \mathbb{Z}_q$, and computes and sends $R_1 = r_1H_0(u_1)$ to $u_2$, together with a bit $b$ (if $b = 1$, then it means $u_1$ wants to engage in private handshake with $u_2$; otherwise it means $u_1$ wants an accountable private handshake): $u_1 \xrightarrow{} u_2 : R_1, b$

**Step 2.** Upon receiving the message, if $u_2$ does not agree with $u_1$ on the type of handshake indicated by $b$, $u_2$ aborts. Otherwise, $u_2$ chooses a random number $r_2 \in \mathbb{Z}_q$, computes $R_2 = r_2H_0(u_2)$ and $V_2 = H_1(\hat{e}(R_1, r_2x_{u_2}, b)) = H_1(\hat{e}(H_0(u_1), H_0(u_2))^q)$, and responds to $u_1$ with $R_2, V_2$: $u_2 \xrightarrow{} u_1 : R_2, V_2$

**Step 3a.** $u_1$ first checks whether $H_1(\hat{e}(r_1x_{u_1}, R_2), b) = V_2$. If the equation does not hold, $u_1$ aborts. Otherwise, $u_1$ computes and sends $V_1 = H_1(\hat{e}(r_1, x_{u_1}, R_2)) = H_1(b, \hat{e}(H_0(u_1), H_0(u_2))^r) \rightarrow u_2$, and at the same time computes $sk_{u_1} = H_2(\hat{e}(r_1x_{u_1}, R_2), R_1, R_2)$: $u_1 \xrightarrow{} u_2 : V_1$

**Step 3b.** Upon receipt of $V_1$, $u_2$ checks whether $H_1(\hat{e}(R_1, r_2x_{u_2})) = V_1$. If the equation does not hold, $u_2$ aborts. Otherwise, $u_2$ computes $sk_{u_2} = H_2(\hat{e}(r_1, r_2x_{u_2}), R_1, R_2)$. Note that, at his point $u_1$ and $u_2$ have authenticated each other if they belong to the same group, and the protocol accomplishes private handshake. Thus if $b = 1$, then the protocol stops, and $sk_{u_1}, sk_{u_2}$ are the shared key between $u_1$ and $u_2$. It is easily verified that $sk_{u_1} = sk_{u_2} = H_2(\hat{e}(H_0(u_1), H_0(u_2))^q, R_1, R_2)$.

However, if $b = 0$, for accountability purposes $u_1$ and $u_2$ continue to send to each other their identity and the random numbers $r_1$ and $r_2$, encrypted by $sk_{u_1}, sk_{u_2}$, respectively. In particular:

**Step 4a.** $u_1 \xrightarrow{} u_2 : C_1 = E_{sk_{u_1}}(r_1, u_1)$ Upon receiving $u_1$’s message, $u_2$ decrypts it to get $r_1', u_1'$, and then checks whether $R_1 = r_1'H_0(u_1')$. If it holds, $u_2$ accepts and computes a new key $sk_{u_1} = H_2(\hat{e}(R_1, r_2x_{u_2}), u_1', u_2)$. Finally, $u_2$ sends to $u_1$: $u_2 \xrightarrow{} u_1 : C_2 = E_{sk_{u_2}}(r_2, u_2)$

Likewise, $u_2$ decrypts and gets $r_2', u_2'$, then checks whether $R_2 = r_2'H_0(u_2')$. If it holds, $u_1$ accepts and computes a new key $sk_{u_2} = H_2(\hat{e}(r_1x_{u_1}, R_2), u_1, u_2')$. Finally, $u_1$ sends to $u_2$:

- **RevokedUser($G, u \rightarrow \{0, 1\}^*$:** To enable user revocation, we assume that in CreateGroup, the group administrator $A_G$ has additionally chosen a $t$-degree polynomial $f(x) \in \mathbb{F}_q[x]$, where $t$ is the maximum possible number of users in the group, and each enrolled user $u$ has been issued $f(u)$. As such, to revoke a user $\bar{u}$ from $G$ (suppose there were already $m$ revoked users $\bar{u}_1, \bar{u}_2, \cdots, \bar{u}_m$...
Technique between unlinkable secret handshakes and private interactions accountable. Optionally accountable private handshakes to negotiate real-time whether or not to make their private handshakes, which allow the two users in the security analysis to the full version of the paper.

For limited space, we leave the formal accountability, in the case of $H_0()$, $H_1()$, $H_2()$ are random oracles; 2) unlinkability is achieved if hash functions $H_0()$, $H_1()$, $H_2()$ are random oracles; 3) on optional accountability, in the case of $b = 1$, private handshake with respect to each of the two communicating users is achieved unconditionally. For limited space, we leave the formal security analysis to the full version of the paper.

V. CONCLUSION

In this paper, we proposed the concept of optionally accountable private handshakes, which allow the two users in handshaking to negotiate real-time whether or not to make their interactions accountable. Optionally accountable private handshakes represent a flexible privacy-preserving authentication technique between unlinkable secret handshakes and private handshakes. We presented the formal definition together with a concrete scheme for optionally accountable private handshakes.

REFERENCES