# Anonymity-Based Authenticated Key Agreement with Full Binding Property

Jung Yeon Hwang, Sungwook Eom, Ku-Young Chang, Pil Joong Lee, and DaeHun Nyang

*Abstract:* **In this paper, we consider some aspects of binding properties that bind an anonymous user with messages. According to whether all the messages or some part of the messages are bound with an anonymous user, the protocol is said to satisfy the *full binding* property or the partial binding property, respectively. We propose methods to combine binding properties and anonymity-based authenticated key agreement protocols. Our protocol with the full binding property guarantees that while no participant's identity is revealed, a participant completes a key agreement protocol confirming that all the received messages came from the other participant. Our main idea is to use an anonymous signature scheme with a signer-controlled yet partially enforced linkability. Our protocols can be modified to provide additional properties, such as revocable anonymity. We formally prove that the constructed protocols are secure.**

*Index Terms:* **Anonymity, authentication, full binding property, key agreement, session hijacking, session turnover.**

## I. INTRODUCTION

PRIVACY enhancing technologies such as group signature schemes and direct anonymous attestation (DAA) are playing an important role in the anonymous service structure; however, considering that today's web-oriented service structures require confidential sessions to facilitate the communication, it is sufficient to rely soley on anonymous signatures. Therefore, it is necessary to have an anonymity-based key agreement protocol using those primitives.

An effective way to achieve a secure session establishment is to use a key agreement (KA) protocol with an identity. A protocol achieving key agreement and authentication is called an authenticated key agreement (AKA) protocol and it can be used to establish a secure session [1], [2]. In practice, Kerberos [3], secure socket layer (SSL) [4], and X.509—an authentication framework [5]—have been used for user identification and key agreement. An agreed key via AKA is used not only for confidentiality and message integrity, but also for *binding* communication messages with the identity.

Recently, Walker *et al*. have proposed an anonymous authenticated key exchange protocol with DAA [6] called DAA-SIGMA. They showed a formal security proof using the Canetti-Krawcyzk model, and the protocol can be used as a way to establish an anonymous confidential session. Also, by applying a group signature instead of a normal signature to an AKA protocol, one can easily obtain an anonymous AKA protocol. Though DAA-SIGMA and group signature-based AKA protocols are enough when a verifier does not mind each individual user's behavior, but it is interested only in the group behavior. It is also desirable if the verifier is able to be sure that a user who has been authenticated is the same user who has just sent a message during the established session. In that sense, DAA-SIGMA and group signature-based AKA protocols are not strong enough.

Assume, as is the case with AKA, that an anonymous authentication protocol is dependent upon a session key established during the authentication for binding each message with a user. Then a user may transfer his/her session key to another user, but the verifier cannot recognize it. Consequently, a user is able to *turn over its session* to another user while the verifier is not aware. For example, while a user wants to remain anonymous after authentication, he/she might also want the verifier to maintain his/her rewarding points. However, the misbinding might disturb the verifier's record of rewarding points, or a verifier might want to record an individual user's misbehavior while not knowing the user's identity sothat a verifier could block the misbehaving user later. Thus, the misbinding might cause some confusion in the verifier's record of misbehaviors. The rewarding/penalty system needs to singulate an anonymous user while not being aware of the user's identity. Unlike with anonymous AKA, session turnover may not be critical in AKA, by using a normal signature scheme, an individual user is clearly known to a verifier, thus, he/she does not have much motivation to turn over the session by transferring the session key. On the other hand, in an anonymous authenticated key agreement protocol, anonymous users cannot be singulated among group members, thus an anonymous user is likely to abuse the misbinding when it is possible.

Singulating a user correctly is important not only to *prevent a session turnover* but also to *prevent a session hijacking* caused from a stolen session key. Assume that a user was watching a video content from a server anonymously, and changed his device to continue to watch the video at home on a large screen. While changing the devices, his session key was stolen. Then, neither a server nor a user could recognize this session hijacking. This type of hijacking from a stolen session key is always possible considering that a session key resides in RAM that is

Manuscript received June 4, 2013; approved for publication by Taekyoung Kwon, Division I Editor, November 29, 2015.

A preliminary version of this paper was presented at WISA 2012.

This work was supported by the ICT R&D program of MSIP/IITP [B1206-15-1007, Development of Universal Authentication Platform Technology with Context-Aware Multi-Factor Authentication and Digital Signature].

J. Y. Hwang and K. Jang are with Electronics and Telecommunication Research Institute, 218 Gajeong-ro Yuseong-gu, Daejeon, 305-700, Republic of KOREA, email: {videmot, jang1090}@etri.re.kr.

S. Eom and P. J. Lee are with the Department of Electrical Engineering, POSTECH, Hyoja-dong, Namgu, Pohang, Kyungbuk, 790-784, Republic of KOREA, email: {sweom, pjl}@postech.ac.kr.

D. Nyang is the corresponding author and is with the School of Computer and Insformation Engineering, INHA Univ., Republic of KOREA, email: nyang@inha.ac.kr.

Digital object identifier 10.1109/JCN.2016.000028

not secure at all, while a long-term key is stored in a secured area such as a tamper-proof-module.

For example, we can imagine an anonymous content streaming service that allows device roaming. In the service, digital contents such as video and music are provided anonymously. The service provider does not want subscribers to share their services with non-subscribers by any means such as session turnover or session hijacking. It does want to singulate a subscriber anonymously to manage his/her service history. Meanwhile, a subscriber of the service wants to get the streaming service on any of his/her devices, which requires a device roaming capability while blocking session hijacking. Our anonymous authenticated key agreement (AAKA) protocol fits quite well into these purposes.

In this paper, we consider aspects of binding properties that bind an anonymous user with messages. According to whether all the messages or some part of the messages are bound with an anonymous user, the protocol is said to satisfy the *full binding* property or the partial binding property, respectively. We propose methods to combine binding properties and anonymity-based authenticated key agreement protocols. Our full binding property protocol guarantees that while no participant's identity is revealed, a participant completes a key agreement protocol confirming that all the received messages came from the other participant. Our main idea is to use an anonymous signature scheme with a signer-controlled yet partially enforced linkability. We formally prove that the constructed protocols are secure. Because dependency on a session key has been eliminated, our protocols can resolve the session turnover and the session hijacking effectively.

Finally, we present AAKA protocols with a relaxed binding property. Here not all of the outgoing messages-but only the messages that might be necessary for singulation of the anonymous user later-are anonymously signed with DAA. These protocols are advantageous in terms of computational overhead because some messages can be authenticated by message authentication code (MAC), which may be more efficiently computed than DAA. Our protocols can be modified to achieve additional properties such as revocable anonymity or unilateral anonymity.

**Related works.** In order to deal with anonymity-based authentication, various anonymous digital signature schemes such as group signatures [7], DAA [8], and anonymous credentials [9] have been proposed. These schemes mainly focus on how to control two basic privacy properties, unlinkability on signatures and anonymity on signer identity.

The notion of a group signature was first introduced in [7]. It allows a member of a group to sign a message anonymously on behalf of the group without revealing a user's identity to unauthorized entities. However, it provides *revocable anonymity*, that is, a signer can be identified by a trusted opener. Formal definitions and security properties for a group signature are well presented for both of static and dynamic groups [10], [11]. A number of group signature schemes have been constructed from various mathematical assumptions [12], [8], [13]. Recently, Hwang *et al*. proposed a short group signature scheme that provides the revocable anonymity and controllable linkability for dynamic membership. Controllable linkability enables an entity that possesses a special linking key to check whether two signatures are generated by the same signer or not [14]. Direct anonymous attestation (DAA) was first introduced in order to provide the remote anonymous authentication of a trusted platform module (TPM) [8]. In a DAA scheme, no trusted opener exists, and thus any signer's identity is not revealed from a DAA signature. Since the initial construction of DAA in [8], many improvement methods have been proposed in the aspect of privacy and performance [15]. Recently several groups of researchers have constructed pairing-based DAA schemes to achieve better efficiency such as a reduction of TPM resources or an increase in flexibility with asymmetric pairings [16]–[18]. Since the two-party key agreement protocol proposed by Diffie and Hellman [19], many AKA protocols combined with various authentication have been suggested. However, in contrast to ordinary AKA protocols in which participants can be identified [20], [21], AAKA protocols have not been well studied formally, and only a few of secure AAKA protocols have been constructed. Leung *et al*. introduced an anonymous AKA protocol using DAA without considering a binding property [22]. Cesena *et al*. proposed an anonymous AKA protocol to protect sharing credentials [23]. Chen *et al*. proposed an AAKA protocol for mobile embedded devices [24]. Walker and Li proposed a formal security model for an anonymous AKA protocol to treat the security against so-called misbinding attacks [6].

**Organization.** The rest of this paper is organized as follows: In Section II, we review some preliminaries for our construction; in Section III, we present a security model for an AAKA protocol and binding properties; in Section IV, we propose various AAKA protocols with the binding properties and prove their security; and finally, we conclude in Section V.

## II. PRELIMINARIES

### A. Computational Assumptions

Let $\mathbb{G}$ be a cyclic group of order $q$ and $g$ a random generator of $\mathbb{G}$ where $q$ is a large prime number.

**Decisional Diffie-Hellman (DDH) Problem**. A DDH algorithm $\mathcal{A}$ for $\mathbb{G}$ is a probabilistic polynomial time (PPT) algorithm satisfying, for some fixed $\alpha > 0$ and sufficiently large $n$ : $\left|\Pr\left[\mathcal{A}(p,g,g^a,g^b,g^{ab})=1\right]-\Pr\left[\mathcal{A}(p,g,g^a,g^b,g^c)=1\right]\right| > 1/n^\alpha$, where $a, b$, and $c$ are selected from $\mathbb{Z}_q^*$ uniformly at random. We say that $\mathbb{G}$ satisfies the DDH assumption if there is no DDH algorithm for $\mathbb{G}$.

### B. Cryptographic Primitives

**Message authentication code (MAC).** An MAC consists of two algorithms: MAC-Gen and MAC-Vrfy. The algorithms are associated to key space $\mathcal{K}$, which is typically defined by $\{0,1\}^\kappa$. MAC-Gen takes as input a message $m$ and a key $s \in \mathcal{K}$ to generate a MAC-tag tag. MAC-Vrfy takes as input a message $m$, a MAC-tag tag and a key $s$, and verifies the MAC-tag. We say that a MAC scheme is $(t, \epsilon)$-secure if, for any PPT adversary $\mathcal{A}$ running in time at most $t$, we have $\Pr\left[s \leftarrow \mathcal{K}; \langle M, \mathtt{tag}\rangle \leftarrow \mathcal{A}^{MAC_s(\cdot)} : \mathsf{MAC\text{-}Vrfy}(M, \mathtt{tag}) = 1, M \notin\right.$

$\mathcal{M}] \leq \epsilon$, where $\mathcal{M}$ is the set of messages that $\mathcal{A}$ submitted to its oracle $MAC_s(\cdot)$.

**Pseudorandom function.** Let $F : \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^*$ be an efficient, length-preserving, keyed function. We say that $F$ is a pseudorandom function if for all PPT distinguishers $D$, there exists fixed $\alpha > 0$ for sufficiently large $n$ such that: $\left| \Pr\left[D^{F_k(\cdot)}(1^n) = 1\right] - \Pr\left[D^{f(\cdot)}(1^n) = 1\right] \right| \leq 1/n^{\alpha}$, where $k \in \{0,1\}^n$ is chosen uniformly at random and $f$ is chosen uniformly at random from the set of functions mapping $n$-bit strings to $n$-bit strings.

### C. DAA Review

In this section we review a DAA scheme [8] that is used as the main building block for constructing our AAKA protocols with the full binding property.

In general, a DAA scheme has three types of players, i.e., an issuer $\mathcal{I}$, a signer $\mathcal{S}_i$[1], and a verifier $\mathcal{V}_j$. Assume that $\mathcal{I}$ is a trusted third party who honestly issues a key. A DAA scheme consists of the following polynomial-time algorithms or an interactive protocol:

- Setup: It takes as input a security parameter $1^{\lambda}$ and then outputs a pair $(\mathtt{gpk}, \mathtt{mik})$, where $\mathtt{mik}$ is the master issuing key secretly managed by the Issuer $\mathcal{I}$ and $\mathtt{gpk}$ is the group public key including the global public parameters. We assume that anyone can access $\mathtt{gpk}$.
- Join: It is an interactive protocol run between a player $\mathcal{S}_i$ who wants to join a group and the issuer $\mathcal{I}$. For the joining process, $\mathcal{S}_i$ generates its own secret key $\mathtt{usk}_i$ in advance. After the completion of the protocol, $\mathcal{S}_i$ obtains a full DAA signing key $\mathtt{sk}_i = (\mathtt{usk}_i, \mathtt{cre}_i)$ where $\mathtt{cre}_i$ is a membership credential issued by $\mathcal{I}$. The value of $\mathtt{usk}_i$ (and so $\mathtt{sk}_i$) is unknown to the issuer.
- Sign: It takes as input $\mathtt{gpk}$, $\mathtt{sk}_i = (\mathtt{usk}_i, \mathtt{cre}_i)$, a basename $\mathtt{bsn}$, and a message $m$, and then outputs a signature $\sigma$ on $m$. The basename $\mathtt{bsn}$ can be either an arbitrary string, such as the name of a verifier or a special symbol $\perp$ to represent a randomly selected string. It can be used for controlling the linkability between signatures.
- Vrfy: It takes as input $\mathtt{gpk}$, $\mathtt{bsn}$, $m$, and $\sigma$, and then outputs 0 ('invalid') or 1 ('valid').
- Link: It takes as input $\mathtt{gpk}$, $(m_0, \sigma_0)$, $(m_1, \sigma_1)$, and $\mathtt{bsn}$, and then outputs 0 ('unlinked') or 1 ('linked').

We say that a DAA scheme is secure if the following properties hold (for the concrete formal definition of the security of DAA, refer to [8], [16], [6], [25]):

- Correctness. A signature generated with a valid signing key should be verified correctly. In addition, two signatures generated with a valid signing key and a same non-empty basename should be linked.
- Anonymity. It must be hard for an adversary to learn the identity of a signer of a signature.
- Unforgeability. It must be hard for an adversary to generate a signature if he/she does not hold a valid (or non-revoked) signing key.
- User-controlled linkability. Signatures created by the same set of $\mathtt{sk} = (\mathtt{usk}, \mathtt{cre})$ and $\mathtt{bsn}$ must be linked. However, signatures cannot be linked if they are created using different basenames (or if $\mathtt{bsn} = \perp$). This property guarantees that no adversary can output two unlinkable signatures generated by a same secret key and a same basename.
- Non-frameability. It is required that no adversary can frame an honest user, that is, claim that the user has generated a DAA signature on a message if he/she has not done so [25]. For non-frameability, the following two properties must be satisfied: (1) It is hard for an adversary (even one who can access $\mathtt{mik}$) to create a signature maliciously traced to an honest user's secret key; and (2) It is hard for an adversary to create two different yet linkable signatures using two different user secret keys and a single basename. This property means that the adversary cannot frame an honest user via the Link algorithm. For this property, we consider a powerful attack such that the adversary can control over all users and the issuer.

As mentioned in the introduction, DAA signature schemes have been constructed using various mathematical assumptions.

## III. SECURITY MODEL

In this section we present a formal security model for a two-party AAKA protocol by modifying the security model of [20], [21], [6]. In particular, this model captures the notion of a *binding property* to guarantee that a participant has only one anonymous partner in a run of a two-party AAKA protocol.

### A. Security Model for a Two-Party AAKA Protocol

**Participants and initialization.** Let $\alpha$ be a polynomial on a security parameter and $\mathcal{U} = \{U_1, \cdots, U_\alpha\}$ denote a set of players for an AAKA protocol. A pair of users in $\mathcal{U}$ is able to invoke the AAKA protocol to share a session key. They can run the protocol concurrently. We assume that there exists a trusted authority, *Issuer*, denoted by $\mathcal{I}$, who is responsible for issuing a secret key to a user. The issuer is not included in $\mathcal{U}$.

During the initialization phase, a key generation process is executed to generate a group public key $\mathtt{gpk}$ and its corresponding secret master issuing key, $\mathtt{mik}$. Assume that the group public key can be accessed publicly by all users and also by an adversary. The group public key is not bound to a specific user but to the whole set $\mathcal{U}$. In addition, a join process is executed between a joining user $U_i$ and $\mathcal{I}$, in order to generate a long-term secret key $\mathtt{sk}_{U_i}$ for the user. Each $U_i$ generates a user private key, $\mathtt{usk}_{U_i}$ and uses it during the join process while keeping it private to $\mathcal{I}$. For example, $\mathtt{sk}_{U_i}$ can be a secret key for DAA, which consists of $\mathtt{usk}_i$ and $\mathtt{cre}_i$. Assume that complete transcripts from each execution of the joining process can identity a user uniquely. Thus, it is assumed that each $\mathtt{usk}_{U_i}$ can be used only for the single user, $U_i$, not other users. In the initialization phase, all parameters and keys are honestly generated and no users are not corrupted.

**Session IDs, partner IDs, and related notions.** To model the

[1] Actually a DAA signer should be defined by the combination of TPM $\mathcal{M}_i$ and a host $\mathcal{H}_i$ because $\mathcal{M}_i$ and $\mathcal{H}_i$ form a platform in the trusted computing environment and share the role of a DAA signer $\mathcal{S}_i$. For simplicity, the DAA signer is treated as a single entity in the paper.

multiple executions of user $U \in \mathcal{U}$, we make use of instances. More concretely, instance $i$ of user $U$ is denoted by $\Pi_U^i$, and the index $i$ increases sequentially according to the number of executions of $U$ [20], [21]. $\Pi_U^i$ runs the protocol with an instance of different participants.

A session identity (ID) of instance $\Pi_U^i$ (denoted by $\mathsf{sid}_U^i$) is defined as the set of the common transcripts generated by the instance and its partner instance executing the protocol. Note that the identity of a user is concealed during the execution of the protocol with anonymity-based authentication. Accordingly, a partner ID is not defined explicitly when a user $U$ (or instance $\Pi_U^i$) first initiates the protocol. Instead, the partner ID for instance $\Pi_U^i$ can be defined with temporal *pseudo-identities* of anonymous players (including $U$ itself) who take part in a run of the protocol to establish a session key. We simply view the partner ID as the session ID. However, to clarify the description of the model, we define $\mathsf{pid}_U^i$ as the set of the identities of the real players taking part in the protocol.

We say that an instance $\Pi_U^i$ *accepts* when it computes a session key $\mathsf{skey}_U^i$. To show the acceptance status, we use a Boolean parameter, $\mathsf{acc}_U^i$. That is, $\mathsf{acc}_U^i = 1$ means that $\Pi_U^i$ accepts, and otherwise, it does not accept. If an instance computes a session key $\mathsf{skey}_U^i$, we assume that it outputs $(\mathsf{sid}_U^i, \mathsf{skey}_U^i)$. Finally, we say that a pair of instances $\Pi_U^i$ and $\Pi_V^j$ (with $U \neq V$) are (anonymously) *partnered* if and only if (1) $\mathsf{sid}_U^i = \mathsf{sid}_V^j$ and (2) they have both accepted.

**Adversarial model.** Behaviors of an adversary are modeled through the following oracle queries and the corresponding responses:

- $\mathsf{Execute}(U, i, V, j)$. The oracle executes the protocol between instances $\Pi_U^i$ and $\Pi_V^j$ and outputs the a collection of all transcripts generated from the honest execution of the AAKA protocol.
- $\mathsf{Send}(U, i, M)$. The oracle sends message $M$ to instance $\Pi_U^i$ and outputs the response generated by this instance according to the protocol.
- $\mathsf{Corrupt}(U)$. The oracle outputs the long-term secret key $\mathtt{sk}_U$ of user $U$.
- $\mathsf{Reveal}(U, i)$. The oracle outputs session key $\mathsf{sk}_U^i$ for a terminated instance $\Pi_U^i$. This query models the leakage of a session key via break-ins, side-channel attacks, etc.
- $\mathsf{Test}(U, i)$. This oracle query is allowed only when $\Pi_U^i$ has accepted. The oracle picks a bit $b \in \{0, 1\}$ uniformly at random. If $b = 0$, it outputs a random session key, and otherwise, i.e., if $b = 1$, it outputs the session key $\mathsf{sk}_U^i$ computed by $\Pi_U^i$. The oracle query can be issued only once.

**Correctness.** We say that an AAKA protocol is *correct* if the following conditions hold: For each pair of users $(U, V)$ and positive integers $i, j$, if $\mathsf{sid}_U^i = \mathsf{sid}_V^j$ and $\mathsf{pid}_U^i = \mathsf{pid}_V^j$, then $\mathsf{acc}_U^i = \mathsf{acc}_V^j = 1$ and $\mathsf{skey}_U^i = \mathsf{skey}_V^j \neq \mathtt{NULL}$.

**AKA security.** Let $\pi$ be an AKA protocol and $\mathcal{A}$ an adversary attacking $\pi$. We say that user $U$ is *corrupted* if a $\mathsf{Corrupt}(U)$ query has been issued. A session is *corrupted* if there exists an anonymous player $U$ who is corrupted while there is an instance $\Pi_{U'}^i$ who has not terminated. We say that instance $\Pi_U^i$ is *fresh* if (1) the adversary has never queried $\mathsf{Reveal}(U', j)$ for any instance $\Pi_{U'}^j$, and (2) the session $\mathsf{sid}$ is not corrupted. The adversary succeeds if it queries the $\mathsf{Test}$ oracle regarding a fresh instance, and correctly guesses the value of the bit $b$ used by the $\mathsf{Test}$ oracle. We denote this event by Succ and define the advantage of adversary $\mathcal{A}$ attacking protocol $\pi$ to be $\mathsf{Adv}_{\mathcal{A},\pi}^{\mathsf{AKA}} = \mid \Pr[\mathrm{Succ}] - 1/2 \mid$. We say that $\pi$ is *AKA-secure* if, for any PPT adversary $\mathcal{A}$, the advantage $\mathsf{Adv}_{\mathcal{A},\pi}^{\mathsf{AKA}}$ is negligible.

**Anonymity.** Let $(U, V)$ be a pair of two players to execute a two-party KA protocol, $\pi$. Note that the roles of the players may differ according to whether they are initiating the protocol or not. To capture the notion of each player's anonymity more precisely, we define two unilateral anonymities for $U$ and $V$. For distinction, they are called $L$ (left-hand side)-anonymity and $R$ (right-hand side)-anonymity games, respectively. The games are played between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ attacking the protocol.

The games consist of the following four phases.

- QUERY I. First, $\mathcal{A}$ issues $\mathsf{Send}$, $\mathsf{Corrupt}$, and $\mathsf{Reveal}$ queries adaptively.
- FIND. $\mathcal{A}$ chooses users $\hat{U}_0$, $\hat{U}_1$, and $V$ from $\mathcal{U}$ in the $L$-anonymity game (resp. $U$ and $\hat{V}_0$, $\hat{V}_1$ in the $R$-anonymity game). A bit $b$ is chosen uniformly at random. The two-party KA protocol $\pi$ is run using $(\hat{U}_b, V)$ (resp. $(U, \hat{V}_b)$ as a pair of two players. A set of all transcripts generated from the protocol is given to adversary $\mathcal{A}$.
- GUESS. $\mathcal{A}$ outputs a guessed bit, $b'$.

Assume that no corrupt query has been made on the target player, i.e., either $\hat{U}_0$ or $\hat{U}_1$ (resp. either $\hat{V}_0$ or $\hat{V}_1$). We say that the adversary wins the game if $b = b'$. We denote by $\mathrm{CGus}_L$ (resp. $\mathrm{CGus}_R$) the event that the adversary correctly guesses the bit in the $L$-anonymity game (resp. $R$-anonymity game). The advantage of $\mathcal{A}$ attacking $\pi$ in the $L$-anonymity game (resp. $R$-anonymity game) is defined by $\mathsf{Adv}_{\mathcal{A},\pi}^{\mathsf{L\text{-}Anon}} = \mid \Pr[\mathrm{CGus}_L] - 1/2 \mid$ (resp. $\mathsf{Adv}_{\mathcal{A},\pi}^{\mathsf{R\text{-}Anon}} = \mid \Pr[\mathrm{CGus}_R] - 1/2 \mid$). We say that $\pi$ is *L-anonymous* (resp. *R-anonymous*) if, for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A},\pi}^{\mathsf{L\text{-}Anon}}$ (resp. $\mathsf{Adv}_{\mathcal{A},\pi}^{\mathsf{R\text{-}Anon}}$) is negligible.

We say that a KA protocol $\pi$ is *(fully or bilaterally) anonymous* if, for any PPT adversary $\mathcal{A}$, both advantages, i.e., $\mathsf{Adv}_{\mathcal{A},\pi}^{\mathsf{L\text{-}Anon}}$ and $\mathsf{Adv}_{\mathcal{A},\pi}^{\mathsf{R\text{-}Anon}}$, are negligible.
We say that $\pi$ is a secure AAKA protocol if $\pi$ is AKA-secure and anonymous.

**Binding property.** Let $(U, V)$ be a pair of two different players for a two-party KA protocol, $\pi$. Assume that two instances $\Pi_U^i$ and $\Pi_V^j$ are *anonymously partnered* in a session after performing $\pi$. Assume that the corresponding session identities, $\mathsf{sid}_U^i$ and $\mathsf{sid}_V^j$ are defined from partnering. By definition, we have $\mathsf{sid}_U^i = \mathsf{sid}_V^j$. Let $\mathcal{T}_V^j$ denote a collection of all transcripts that $\Pi_V^j$ transmitted for the session. If it is guaranteed that $\mathsf{sid}_V^j \setminus \mathcal{T}_V^j$, i.e., the remaining transcripts except $\mathcal{T}_V^j$ are generated by $\Pi_U^i$ using its secret key, $\mathtt{sk}_U$ then we can say that a unilateral binding property (BP) holds for $\Pi_U^i$. Similarly, we can define another

unilateral BP for $\Pi_V^j$.

To capture the notion of a binding property[2] for each player more precisely, we define two unilateral BP games for $\Pi_V^j$ and $\Pi_U^i$, respectively. For distinction, they are called $L$(left-hand side)-BP and $R$(right-hand side)-BP games, respectively. The games are played between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ attacking the protocol. In the game, $\mathcal{A}$ is given all secret keys and outputs a pair of instances $\Pi_U^i$ and $\Pi_V^j$ (with $U \neq V$) and $\mathsf{sid}_U^i = \mathsf{sid}_V^j$. The adversary succeeds in the $L$-BP game (resp. $R$-BP game) if the following conditions hold. (1) $\Pi_U^i$ and $\Pi_V^j$ are *anonymously partnered* for a session; and (2) Some of $\mathsf{sid}_V^j \setminus \mathcal{T}_V^j$ have not been generated by $\Pi_U^i$ using $\mathtt{sk}_U$ (resp. some of $\mathsf{sid}_U^i \setminus \mathcal{T}_U^i$ have not been generated by $\Pi_V^j$ using $\mathtt{sk}_V$).

If the binding properties hold for both of the instances then we say that $\pi$ achieves the (bilateral) full BP. Intuitively, the full BP guarantees that a party must have only one partnered party of an anonymous session.

As noted in [26], the above notion can be relaxed to capture a *partial* BP (pBP) to guarantee that some part of the messages in a session are generated using a long-term key $\mathtt{sk}_U$, and the other parts are generated with a short-term key. In a session, players compute *a private short-term key* (for example, Diffie-Hellman key) from public transcripts (for example, Diffie-Hellman ephemeral public keys). The short-term key is valid only during the session, and it is independent across sessions. Owing to the key, a communicating player can bind all the messages that belong to the session. In terms of security, the partial binding is guaranteed in the sense that an attacker cannot hijack the session without knowing the short-term key, but a player can turn his/her session over to another player.

Similar to the (bilateral) full BP, we consider two unilateral games for $\Pi_V^j$ and $\Pi_U^i$, respectively. For distinction, they are called $L$(left-hand side)-pBP and $R$(right-hand side)-pBP, respectively. The games are played between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ attacking $\pi$.

The games consist of the following phases.

- QUERY. According to a strategy, $\mathcal{A}$ issues Send, Corrupt, and Reveal queries, adaptively.
- FIND. $\mathcal{A}$ outputs a pair of instances $\Pi_U^i$ and $\Pi_V^j$ (with $U \neq V$) and $\mathsf{sid}_U^i = \mathsf{sid}_V^j$.

The adversary succeeds in the $L$-pBP game (resp. $R$-pBP game) if the following conditions hold: (1) $\Pi_U^i$ and $\Pi_V^j$ are *anonymously partnered* for a session; (2) $\mathcal{A}$ has not made any corrupt query on $U$ (and resp. $V$); and (3) Some of $\mathsf{sid}_V^j \setminus \mathcal{T}_V^j$ have been generated by $\Pi_U^i$ using $\mathtt{sk}_U$ (resp. some of $\mathsf{sid}_U^i \setminus \mathcal{T}_U^i$ have been generated by $\Pi_V^j$ using $\mathtt{sk}_V$).

If the $L$-pBP and $R$-pBP hold then we say that $\pi$ achieves the (bilateral) pBP.

## IV. OUR AAKA PROTOCOLS

In this section, we propose two-party AAKA protocols with the full binding property. The main idea to achieve our goal is to use a DAA scheme. Basically anonymous authentication can be provided under the anonymity of the underlying DAA scheme. Furthermore, because of the user-controlled linkability of DAA, we can build a proper structure to achieve the full binding property while preserving anonymity for participants.

Assume that a DAA scheme, $\mathrm{DAA} = (\mathsf{Setup}, \mathsf{Join}, \mathsf{Sign}, \mathsf{Vrfy}, \mathsf{Link})$ is given. Before running the proposed protocols, Setup is performed to generate a group public key, $\mathtt{gpk}$ and a secret master issuing key, $\mathtt{mik}$. Each user $U$ performs Join with the Issuer who can access $\mathtt{mik}$, to obtain a signing key, $\mathtt{sk}_U = (\mathtt{usk}_U, \mathtt{cre}_U)$ for DAA. Here $\mathtt{cre}_U$ is a membership credential and $\mathtt{usk}_U$ is a secret key that is generated by the user and is unrevealed to Issuer. Assume that $\mathtt{usk}_U$ is generated to be a value of which randomness is contributed by both a user and Issuer. It can be easily done by using known secure multi-party computation methods. This feature is necessary to achieve the full binding property in our protocols.

The proposed protocols make use of a message authentication code MAC, a pseudorandom function PRF, and a cryptographic hash function $H : \{0,1\}^* \to \{0,1\}^\ell$. Let $\mathbb{G}$ be a cyclic group of order $q$ and $g$ a random generator of $\mathbb{G}$, where $q$ is a large prime number. The collection of public public parameters is set to be $((\mathbb{G}, g, q), \mathtt{gpk}, H, \mathrm{PRF})$.

### A. Three-Pass AAKA Protocol I

The basic structure of this protocol is similar to the KA protocol [27] that runs in three rounds. However, our protocol performs a mutual anonymity-based authentication process. For explicit key confirmation, our protocol uses a DAA signature. See Fig. 1. For convenience, we call the proposed AAKA protocol 3AKA-FB.

**Theorem 1** *The proposed protocol is a secure AAKA protocol that achieves the full binding property if the underlying DAA signature scheme is secure.*

*Proof.* In order to prove the theorem, we should show that the protocol achieves three security properties defined in Section III: the AKA security, the anonymity, and the full binding property. The proof of the theorem can be completed from the following lemmas. ■

**Lemma 1** *The proposed protocol achieves the AKA security if the DDH assumption in $\mathbb{G}$ holds and the underlying DAA signature scheme is secure.*

*Proof.* Let $\mathcal{A}$ be an adversary attacking the proposed AAKA protocol. Assume that the adversary makes at most $q_{ex}$ and $q_s$ calls to the Execute and the Send oracles. We consider two independent events, $\mathtt{Forge}$ and $\mathtt{Repeat}$ that are mainly related to the advantage of the adversary. Let $\mathtt{Forge}$ be the event that $\mathcal{A}$ generates a forged DAA signature for user $U$ before making a query of the form $\mathsf{Corrupt}(U)$ to obtain a secret DAA key of $U$. More precisely, $\mathtt{Forge}$ is the event that $\mathcal{A}$ makes a query of the form $\mathsf{Send}(V, j, [X, Sign_U])$, $\mathsf{Send}(V, j, [Sign'_U])$ or $\mathsf{Send}(U, i, [Y, Sign_V])$ where $Sign_U$, $Sign'_U$, and $Sign_V$ are valid, i.e., pass the verification test of the DAA scheme as defined in Fig. 1, but these were not previously output by any instance of the as-yet-uncorrupted user as DAA signatures. Let

[2]This binding property prevents so-called misbinding attacks in [6].

| $U(\mathtt{sk}_U)$: $\mathtt{gpk}$ of DAA | | $V(\mathtt{sk}_V)$: $\mathtt{gpk}$ of DAA |
|---|---|---|
| $x \leftarrow_R \mathbb{Z}_q$; $X \leftarrow g^x$ | | |
| $\mathtt{bsn}_U \leftarrow H(X)$ | | |
| $Sign_U \leftarrow \mathsf{Sign}_{\mathtt{sk}_U}(\mathtt{gpk}, \mathtt{bsn}_U, X)$ | $\xrightarrow{[X, Sign_U]}$ | $\mathtt{bsn}_U \leftarrow H(X)$<br>If $1 \neq \mathsf{Vrfy}(\mathtt{gpk}, \mathtt{bsn}_U, X, Sign_U)$,<br>abort. Otherwise<br>$y \leftarrow_R \mathbb{Z}_q$; $Y \leftarrow g^y$<br>$K \leftarrow X^y = g^{xy}$<br>$k_0 \leftarrow \mathrm{PRF}_K(0)$, $k_1 \leftarrow \mathrm{PRF}_K(1)$<br>$\mathtt{bsn}_V \leftarrow H(Y)$; $h_v = H(Y, X, k_0)$ |
| $K \leftarrow Y^x = g^{xy}$<br>$k_0 \leftarrow \mathrm{PRF}_K(0)$, $k_1 \leftarrow \mathrm{PRF}_K(1)$<br>$h_v = H(Y, X, k_0)$<br>If $1 \neq \mathsf{Vrfy}(\mathtt{gpk}, \mathtt{bsn}_V, h_v, Sign_V)$,<br>abort. Otherwise<br>$h_u = H(X, Y, k_0)$ | $\xleftarrow{[Y, Sign_V]}$ | $Sign_V \leftarrow \mathsf{Sign}_{\mathtt{sk}_V}(\mathtt{gpk}, \mathtt{bsn}_V, h_v)$ |
| $Sign'_U \leftarrow \mathsf{Sign}_{\mathtt{sk}_U}(\mathtt{gpk}, \mathtt{bsn}_U, h_u)$ | $\xrightarrow{[Sign'_U]}$ | $h_u = H(X, Y, k_0)$<br>If $1 \neq \mathsf{Vrfy}(\mathtt{gpk}, \mathtt{bsn}_U, h_u, Sign'_U)$,<br>abort.<br>If $1 \neq \mathsf{Link}(\mathtt{gpk}, (X, Sign_U), (h_u, Sign'_U), \mathtt{bsn}_U)$, abort. |
| Session key: $\mathsf{skey} = k_1$ | | Session key: $\mathsf{skey} = k_1$ |

Fig. 1. Three-pass AAKA protocol.

$\varepsilon_{\text{DAA}}$ be the maximum advantage of an adversary in forging a DDA signature on a new message. Straightforwardly, we have $\Pr[\mathtt{Forge}] \leq |\mathcal{U}| \cdot \varepsilon_{\text{DAA}}$ because the underlying DAA scheme is assumed to be (existentially) unforgeable and $\mathcal{A}$ can use at most $|\mathcal{U}|$ DAA signatures where $|\mathcal{U}|$ is the number of all protocol users, which is polynomially bounded in a security parameter. Let $\mathtt{Repeat}$ be the event that an honest user generates an ephemeral DH public key $g^r$ twice. Let $q$ be the prime order of the (mathematical) group $\mathbb{G}$ and $\lambda$ be the bit-length of the order. We can show $\Pr[\mathtt{Repeat}] \leq q_s \cdot (q_s + q_{ex})/2^{\lambda}$ from the well-known birthday paradox because a user selects a random element $r \in \mathbb{Z}_q$ at most $q_s + q_{ex}$ times. Note that $\Pr[\mathtt{Repeat}]$ is negligible because $\lambda$ is assumed to be sufficiently large.

Next, we show that, unless $\mathtt{Forge}$ and $\mathtt{Repeat}$ occur, we can construct an efficient algorithm $\mathcal{D}$ solving the DDH problem in $\mathbb{G}$ using $\mathcal{A}$, who attacks the proposed protocol. Note that $\mathcal{A}$'s $\mathsf{Test}$ query can be asked to an instance that was initialized via an $\mathsf{Execute}$ query or a $\mathsf{Send}$ query. We denote by $\mathtt{Test}_{\mathsf{E}}$ the event that $\mathcal{A}$ asks its $\mathsf{Test}$ query to an instance that was initialized via an $\mathsf{Execute}$ query. We also denote by $\mathtt{Test}_{\mathsf{S}}$ the event that $\mathcal{A}$ asks its $\mathsf{Test}$ query to an instance that was initialized via a $\mathsf{Send}$ query. Since $\mathcal{A}$'s $\mathsf{Test}$ query can be made only from the two events and the events are disjoint, we have $\mathtt{Test}_{\mathsf{S}} = \overline{\mathtt{Test}_{\mathsf{E}}}$. We will separably bound the probabilities of $\Pr^{\mathsf{AKA}}_{\mathcal{A},\text{3AKA-FB}}[\text{Succ} \wedge \mathtt{Test}_{\mathsf{E}}]$ and $\Pr^{\mathsf{AKA}}_{\mathcal{A},\text{3AKA-FB}}[\text{Succ} \wedge \mathtt{Test}_{\mathsf{S}}]$ by constructing DDH solvers, $\mathcal{D}_1$ and $\mathcal{D}_2$ properly.

Initially, $\mathcal{D}_1$ and $\mathcal{D}_2$ perform the following same process. The DDH solvers, $\mathcal{D}_1$ and $\mathcal{D}_2$ generate all private DAA signing keys and a group public key by running $\mathsf{Setup}$ and $\mathsf{Join}$ algorithms of the underlying DAA scheme. The group public key is given to the adversary $\mathcal{A}$. Assume that the DDH solvers are given a random DDH instance $(g, A = g^a, B = g^b, Z) \in \mathbb{G}^4$. The goal of the DDH solvers is to determine whether $Z = g^{ab}$ or not.

We first describe the simulation for $\mathcal{A}$'s oracle queries provided by $\mathcal{D}_1$ as follows.

- $\mathsf{Execute}$ queries. First, $\mathcal{D}_1$ guesses $\alpha \in \{1, ..., q_{ex}\}$ uniformly at random such that $\mathcal{A}$ makes its $\mathsf{Test}$ query to an instance that was initialized via the $\alpha^{\text{th}}$ $\mathsf{Execute}$ query. $\mathcal{D}_1$ responds to each query of the form $\mathsf{Execute}(U, i, V, j)$ as follows:
  - On the $\alpha^{\text{th}}$ $\mathsf{Execute}$ query of the form $\mathsf{Execute}(U^*, i^*, V^*, j^*)$, $\mathcal{D}_1$ generates $T = \{(X = g^a, Sign_{U^*}), (Y = g^b, Sign_{V^*}), (Sign'_{U^*})\}$ using the given DDH instance, where $\{Sign_{U^*}, Sign'_{U^*}\}$ and $Sign_{V^*}$ are generated with DAA signing keys of $U^*$ and $V^*$, respectively. Note that $\mathcal{D}_1$ can generate the signatures easily using DAA signing keys that it generated. $\mathcal{D}$ defines the session key $\mathsf{skey}$ corresponding to the $\alpha^{\text{th}}$ $\mathsf{Execute}$ query by $Z$. Then $\mathcal{D}_1$ returns $T$.
  - On any other $\mathsf{Execute}$ query except the $\alpha^{\text{th}}$ query, $\mathcal{D}_1$ honestly executes the protocol and then returns the resulting transcripts. This is possible because $\mathcal{D}_1$ is aware of the private DAA keys of all users.
- $\mathsf{Send}$ queries. $\mathcal{D}_1$ responds to a $\mathsf{Send}$ query using the transcripts that it obtains after executing the protocol itself.
- $\mathsf{Corrupt}$ queries. On a $\mathsf{Corrupt}(U)$ query, $\mathcal{D}_1$ responds with the corresponding private DAA signing key of user $U$. This is obviously possible because it generated all DAA signing keys at the initial step.
- $\mathsf{Reveal}$ queries. On a $\mathsf{Reveal}$ query, $\mathcal{D}_1$ proceeds as follows: If this is related to the $\alpha^{\text{th}}$ $\mathsf{Execute}$ query, i.e., the query has the form, $\mathsf{Reveal}(U^*, i^*)$ or $\mathsf{Reveal}(V^*, j^*)$ then $\mathcal{D}_1$ aborts; otherwise, $\mathcal{D}_1$ finds an instance associated with an $\mathsf{Execute}$ or a $\mathsf{Send}$ query, and then $\mathcal{D}_1$ returns the $\mathsf{skey}$ generated (by $\mathcal{D}_1$) for the instance.

- Test queries. When $\mathcal{A}$ issues $\mathsf{Test}(U,t)$ for a terminated instance, $\mathcal{D}$ finds the related session ID, $\mathsf{sid}_U^t$ for the instance, and the corresponding session key generated by the instance. If $\mathsf{sid}_U^t$ is not the same to the set of all transcripts generated via the $\alpha^{\text{th}}$ Execute query, then $\mathcal{D}$ aborts and outputs a random bit; otherwise, $\mathcal{D}_1$ sets $K = Z$, computes $k_1 = \text{PRF}_K(1)$, and returns $\mathsf{skey} = k_1$ to $\mathcal{A}$. $\mathcal{D}_1$ outputs the bit that $\mathcal{A}$ outputs.

Let REAL denote the transcript distribution of the real protocol and its resulting session key skey as defined in Fig. 2.

When $\texttt{Test}_\mathsf{E}$ occurs, $\mathcal{D}_1$ does not abort with probability $1/q_{ex}$. In addition, unless $\mathcal{D}_1$ aborts, there exist two possible cases in the above simulation as follows: If $Z = g^{ab}$, i.e., $(g, g^a, g^b, Z)$ is a valid DDH tuple, then the simulation is a correct execution of the proposed protocol, that is, sk is a real session key. Otherwise, we have $Z = g^z$ for random $z \in \mathbb{Z}_q^*$. Thus $\mathsf{sk} = \text{PRF}_Z(1)$ is indistinguishable from a random key because PRF is a pseudo-random function and $Z = g^z$ is distributed uniformly at random. In this case, however, $k_0 = \text{PRF}_{Z=g^z}(0)$ is used in the simulation, instead of $k_0 = \text{PRF}_{g^{ab}}(0)$ which is originally defined by the proposed construction.

We can show that any difference from this modification is negligible. To show this, we consider distributions modified from the original one, REAL. For distinction, let FAKE and RAND denote the modified distributions as defined in Fig. 2. Let $\varepsilon_{\text{PRF}}$ be the maximum advantage that an adversary can obtain when attacking the given PRF. Then we have $|\Pr[(\mathcal{T}, \mathsf{skey}) \leftarrow \mathsf{REAL} : \mathcal{A}(\mathcal{T}, \mathsf{skey}) = 1] - \Pr[(\mathcal{T}, \mathsf{skey}) \leftarrow \mathsf{FAKE} : \mathcal{A}(\mathcal{T}, \mathsf{skey}) = 1]| \leq \mathsf{Adv}_{\mathbb{G}}^{\mathsf{DDH}}$ and $|\Pr[(\mathcal{T}, \mathsf{skey}) \leftarrow \mathsf{FAKE} : \mathcal{A}(\mathcal{T}, \mathsf{skey}) = 1] - \Pr[(\mathcal{T}, \mathsf{skey}) \leftarrow \mathsf{RAND} : \mathcal{A}(\mathcal{T}, \mathsf{skey}) = 1]| \leq \mathsf{Adv}_{\mathbb{G}}^{\mathsf{DDH}} + \varepsilon_{\text{PRF}}$. Note that the distribution of parameters related to DAA and the hash function are independent of the DDH parameter. Thus we have $\Pr[(\mathcal{T}, \mathsf{skey} = \mathsf{sk}_0) \leftarrow \mathsf{RAND}; \mathsf{sk}_1 \leftarrow_R \{0,1\}^n; b \leftarrow \{0,1\} : \mathcal{A}(\mathcal{T}, \mathsf{sk}_b) = b] = 1/2$.

The advantage of the adversary $\mathcal{A}$ is upper-bounded by $2 \cdot (2\mathsf{Adv}_{\mathbb{G}}^{\mathsf{DDH}} + \varepsilon_{\text{PRF}})$. In other words, $\mathcal{D}_1$ can solve the given DDH instance using the adversary $\mathcal{A}$ with a meaningful probability.

Let $\texttt{Gus}_\mathsf{E}$ be the event that $\mathcal{D}_1$ guesses $\alpha$ correctly. The simulation provided by $\mathcal{D}_1$ is perfect, i.e., perfectly identical to the security game with the proposed protocol unless $\mathcal{A}$ aborts. Hence we obtain $4 \cdot \mathsf{Adv}_{\mathbb{G}}^{\mathsf{DDH}} + 2 \cdot \varepsilon_{\text{PRF}} = 2 \cdot |1/q_{ex} \cdot \Pr_{\mathcal{A}}[\mathsf{Succ} \wedge \texttt{Test}_\mathsf{E}] + 1/2 \cdot \Pr_{\mathcal{A}}[\overline{\texttt{Gus}_\mathsf{E}} \vee \texttt{Test}_\mathsf{S}] - 1/2|$. Since $\Pr_{\mathcal{A}}[\overline{\texttt{Gus}_\mathsf{E}} \vee \texttt{Test}_\mathsf{S}] = (1 - 1/q_{ex}) \Pr_{\mathcal{A}}[\texttt{Test}_\mathsf{E}] + \Pr_{\mathcal{A}}[\texttt{Test}_\mathsf{S}]$ and $\Pr[\texttt{Test}_\mathsf{E}] = 1 - \Pr[\texttt{Test}_\mathsf{S}]$, we have $4 \cdot \mathsf{Adv}_{\mathbb{G}}^{\mathsf{DDH}} + 2 \cdot \varepsilon_{\text{PRF}} = 2 \cdot |1/q_{ex} \Pr_{\mathcal{A}}[\mathsf{Succ} \wedge \texttt{Test}_\mathsf{E}] + (1/2q_{ex}) \Pr_{\mathcal{A}}[\texttt{Test}_\mathsf{S}] - (1/2q_{ex})|$.

Next we describe the simulation for $\mathcal{A}$'s oracle queries provided by $\mathcal{D}_2$ as follows.

- Execute queries. On an Execute query, $\mathcal{D}_2$ responds with the transcripts that it obtains after executing the protocol itself.
- Send queries. First, $\mathcal{D}_2$ guesses $\beta_1, \beta_2 \in \{1, \cdots, q_s\}$ ($\beta_1 < \beta_2$) uniformly at random such that $\mathcal{A}$ makes its Test query to an instance that was initialized via the $\beta_1^{\text{th}}$ Send query or the $\beta_2^{\text{th}}$ Send query. $\mathcal{D}_2$ responds to each Send query as follows.
  - On the $\beta_1^{\text{th}}$ Send query, if it does not have the form $\mathsf{Send}(U^*, i^*, \text{Start})$, $\mathcal{D}_2$ aborts; otherwise, $\mathcal{D}_2$ returns $(X = A = g^a, Sign_{U^*})$ where $Sign_{U^*}$ is generated with DAA signing key of $U^*$ with $\texttt{bsn}_{U^*} = H(X)$.
  - On the $\beta_2^{\text{th}}$ Send query, if it does not have the form $\mathsf{Send}(V^*, j^*, (X, Sign_{U^*}))$, $\mathcal{D}_2$ aborts; otherwise, $\mathcal{D}_2$ returns $(Y = B = g^b, Sign_{V^*})$ where $Sign_{V^*}$ is generated with a DAA signing key of $V^*$ with $\texttt{bsn}_{V^*} = H(Y)$.
  - On any other Send query except the $\beta_1^{\text{th}}$ and the $\beta_2^{\text{th}}$ queries, $\mathcal{D}_2$ proceeds as follows.
    (a) If the query has the form $\mathsf{Send}(U^*, i^*, (Y, Sign_{V^*}))$ then $\mathcal{D}_2$ sets $K = Z$ and computes $k_0 = \text{PRF}_K(0)$, $k_1 = \text{PRF}_K(1)$, $h_u = H(X, Y, k_0)$ and $Sign'_U \leftarrow \mathsf{Sign}_{\mathsf{sk}_U}(\texttt{gpk}, \texttt{bsn}_U, h_u)$. Then $\mathcal{D}_2$ returns $Sign'_U$;
    (b) otherwise, $\mathcal{D}_2$ honestly executes the corresponding procedure of the protocol and then returns the resulting transcripts. This is possible because $\mathcal{D}_2$ is aware of the private DAA keys of all users.
- Corrupt queries. On a $\mathsf{Corrupt}(U)$ query, $\mathcal{D}_2$ responds with the corresponding private DAA signing key of user $U$. This is obviously possible because it generated all DAA signing key at the initial step.
- Reveal$(U, i)$ queries. On a Reveal query, $\mathcal{D}_2$ proceeds as follows. If this is related to the $\alpha^{\text{th}}$ send query then $\mathcal{D}_2$ aborts; otherwise, $\mathcal{D}_2$ finds an instance associated with an Execute or a Send query, and then $\mathcal{D}_2$ returns the skey generated (by $\mathcal{D}_2$) for the instance.
- Test queries. When $\mathcal{A}$ issues $\mathsf{Test}(U,t)$ for a terminated instance, $\mathcal{D}_2$ finds related session ID, $\mathsf{sid}_U^t$ for the instance, and the corresponding session keys that are generated by the instance. If $\mathsf{sid}_U^t$ is not the same to the set of all transcripts generated by the instances that are initialized via the $\beta_1^{\text{th}}$ and the $\beta_2^{\text{th}}$ Send queries, then $\mathcal{D}_2$ aborts and outputs a random bit; otherwise, $\mathcal{D}_2$ sets $K = Z$, computes $\mathsf{skey} = k_1 = \text{PRF}_K(1)$, and returns skey to $\mathcal{A}$. $\mathcal{D}_2$ outputs the bit that $\mathcal{A}$ outputs.

Let $\texttt{Bad} = \texttt{Forge} \vee \texttt{Repeat}$. Then we have $\Pr[\texttt{Bad}] \leq \Pr[\texttt{Forge}] + \Pr[\texttt{Repeat}]$. If neither Bad nor $\texttt{Test}_\mathsf{E}$ occur, then $\mathcal{D}_2$ does not abort at least with probability $2/q_s(q_s - 1)$, because he can correctly guess $(\beta_1, \beta_2)$ with the probability. Let $\texttt{Gus}_\mathsf{S}$ be the event that $\mathcal{D}_2$ correctly guesses $(\beta_1, \beta_2)$. Unless $\mathcal{A}_2$ aborts, we note that there exist two possible cases in the above simulation as follows: If $Z = g^{ab}$, i.e., $(g, g^a, g^b, Z)$ is a valid DDH-tuple then the simulation is a correct execution of the proposed protocol, and so skey is a real key shared between $\Pi_{U^*}^{i^*}$ and $\Pi_{V^*}^{j^*}$. Otherwise, we have $Z = g^z$ for a random $c \in \mathbb{Z}_q^*$. $\mathsf{skey} = \text{PRF}_Z(1)$ is indistinguishable from a random key because PRF is a pseudo-random function. In this case, however, $k_0 = \text{PRF}_{Z=g^z}(0)$ is used in the simulation instead of $k_0 = \text{PRF}_{g^{ab}}(0)$, which is originally defined by the proposed construction. Using a similar argument for $\texttt{Test}_\mathsf{E}$ with the transcript distributions in Fig. 2, we can show that any difference from the modification is negligible. The advantage of the adversary $\mathcal{A}$ is upper-bounded by $2 \cdot (2\mathsf{Adv}_{\mathbb{G}}^{\mathsf{DDH}} + \varepsilon_{\text{PRF}})$.

Hence, we obtain $4 \cdot \mathsf{Adv}_{\mathbb{G}}^{\mathsf{DDH}} + 2 \cdot \varepsilon_{\text{PRF}} = 2 \cdot |1/(q_s/2(q_s/2 - 1)) \cdot \Pr_{\mathcal{A}}[\mathsf{Succ} \wedge \texttt{Test}_\mathsf{S} \wedge \neg\mathsf{Bad}] + 1/2 \cdot \Pr_{\mathcal{A}}[\overline{\texttt{Gus}_\mathsf{S}} \vee \texttt{Test}_\mathsf{E} \vee \mathsf{Bad}] - 1/2|$. Since $\Pr_{\mathcal{A}}[\neg\texttt{Gus}_\mathsf{S} \vee \texttt{Test}_\mathsf{E} \vee \mathsf{Bad}] = \Pr_{\mathcal{A}}[\texttt{Test}_\mathsf{E} \vee \mathsf{Bad}] + (1 - 1/(q_s/2(q_s/2 - 1))) \Pr_{\mathcal{A}}[\texttt{Test}_\mathsf{S} \wedge \overline{\mathsf{Bad}}]$ and $\Pr[\texttt{Test}_\mathsf{S} \wedge \overline{\mathsf{Bad}}] + \Pr[\texttt{Test}_\mathsf{E} \vee$

$$\mathsf{REAL} \stackrel{\text{def}}{=} \left[\begin{array}{c} x \leftarrow_R \mathbb{Z}_q, X = g^x, \mathtt{bsn}_U = H(X), Sign_U \leftarrow \mathsf{Sign}_{\mathtt{sk}_U}(\mathtt{gpk}, \mathtt{bsn}_U, X); \\ y \leftarrow_R \mathbb{Z}_q; Y = g^y, K = g^{xy}, k_0 = \mathrm{PRF}_K(0), \\ \mathtt{bsn}_V = H(V), h_v = H(Y, X, k_0), Sign_V \leftarrow \mathsf{Sign}_{\mathtt{sk}_V}(\mathtt{gpk}, \mathtt{bsn}_V, h_v); \\ h_u = H(X, Y, k_0), Sign'_U \leftarrow \mathsf{Sign}_{\mathtt{sk}_U}(\mathtt{gpk}, \mathtt{bsn}_U, h_u) \quad : (\mathcal{T}, \mathsf{skey}) \\ \mathcal{T} = (X, Sign_U, Y, Sign_V, Sign'_U) \\ \mathsf{skey} = k_1 = \mathrm{PRF}_K(1) \end{array}\right]$$

$$\mathsf{FAKE} \stackrel{\text{def}}{=} \left[\begin{array}{c} x \leftarrow_R \mathbb{Z}_q, X = g^x, \mathtt{bsn}_U = H(X), Sign_U \leftarrow \mathsf{Sign}_{\mathtt{sk}_U}(\mathtt{gpk}, \mathtt{bsn}_U, X); \\ y, z \leftarrow_R \mathbb{Z}_q, Y = g^y, K = g^{xy}, Z = g^z, k_0 = \mathrm{PRF}_Z(0), \\ \mathtt{bsn}_V = H(V), h_v = H(Y, X, k_0), Sign_V \leftarrow \mathsf{Sign}_{\mathtt{sk}_V}(\mathtt{gpk}, \mathtt{bsn}_V, h_v); \\ h_u = H(X, Y, k_0), Sign'_U \leftarrow \mathsf{Sign}_{\mathtt{sk}_U}(\mathtt{gpk}, \mathtt{bsn}_U, h_u) \quad : (\mathcal{T}, \mathsf{skey}) \\ \mathcal{T} = (X, Sign_U, Y, Sign_V, Sign'_U) \\ \mathsf{skey} = k_1 = \mathrm{PRF}_Z(1) \end{array}\right]$$

$$\mathsf{RAND} \stackrel{\text{def}}{=} \left[\begin{array}{c} x \leftarrow_R \mathbb{Z}_q, X = g^x, \mathtt{bsn}_U = H(X), Sign_U \leftarrow \mathsf{Sign}_{\mathtt{sk}_U}(\mathtt{gpk}, \mathtt{bsn}_U, X); \\ y, z \leftarrow_R \mathbb{Z}_q, Y = g^y, K = g^{xy}, Z = g^z, k_0 = \mathrm{PRF}_Z(0), \\ \mathtt{bsn}_V = H(V), h_v = H(Y, X, k_0), Sign_V \leftarrow \mathsf{Sign}_{\mathtt{sk}_V}(\mathtt{gpk}, \mathtt{bsn}_V, h_v); \\ h_u = H(X, Y, k_0), Sign'_U \leftarrow \mathsf{Sign}_{\mathtt{sk}_U}(\mathtt{gpk}, \mathtt{bsn}_U, h_u) \quad : (\mathcal{T}, \mathsf{skey}) \\ \mathcal{T} = (X, Sign_U, Y, Sign_V, Sign'_U) \\ \mathsf{skey} = k_1 = \mathrm{PRF}_Z(1) \end{array}\right]$$

Fig. 2. Distribution of parameters for the real protocol and its conceptually modified protocols.

| $U(\mathtt{sk}_U)$: $\mathtt{gpk}$ of DAA | | $V(\mathtt{sk}_V)$: $\mathtt{gpk}$ of DAA |
|---|---|---|
| $x \leftarrow_R \mathbb{Z}_q; X \leftarrow g^x$ | | $y \leftarrow_R \mathbb{Z}_q; Y \leftarrow g^y$ |
| $\mathtt{bsn}_U \leftarrow H(X)$ | | $\mathtt{bsn}_V \leftarrow H(Y)$ |
| $Sign_U \leftarrow \mathsf{Sign}_{\mathtt{sk}_U}(\mathtt{gpk}, \mathtt{bsn}_U, X)$ | $\xrightarrow{[X, Sign_U]}$ | $Sign_V \leftarrow \mathsf{Sign}_{\mathtt{sk}_V}(\mathtt{gpk}, \mathtt{bsn}_V, Y)$ |
| | $\xleftarrow{[Y, Sign_V]}$ | |
| If $1 \neq \mathsf{Vrfy}(\mathtt{gpk}, \mathtt{bsn}_V, Y, Sign_V)$, abort. Otherwise | | If $1 \neq \mathsf{Vrfy}(\mathtt{gpk}, \mathtt{bsn}_U, X, Sign_U)$, abort. Otherwise |
| $K \leftarrow Y^x = g^{xy}$ | | $K \leftarrow X^y = g^{xy}$ |
| $k_0 \leftarrow \mathrm{PRF}_K(0), k_1 \leftarrow \mathrm{PRF}_K(1)$ | | $k_0 \leftarrow \mathrm{PRF}_K(0), k_1 \leftarrow \mathrm{PRF}_K(1)$ |
| $h_u = H(X, Y, k_0)$ | | $h_v = H(Y, X, k_0)$ |
| $Sign'_U \leftarrow \mathsf{Sign}_{\mathtt{sk}_U}(\mathtt{gpk}, \mathtt{bsn}_U, h_u)$ | $\xrightarrow{[Sign'_U]}$ | $Sign'_V \leftarrow \mathsf{Sign}_{\mathtt{sk}_V}(\mathtt{gpk}, \mathtt{bsn}_V, h_v)$ |
| $h_v = H(Y, X, k_0)$ | $\xleftarrow{[Sign'_V]}$ | $h_u = H(X, Y, k_0)$ |
| If $1 \neq \mathsf{Vrfy}(\mathtt{gpk}, \mathtt{bsn}_V, h_v, Sign'_V)$, abort. | | If $1 \neq \mathsf{Vrfy}(\mathtt{gpk}, \mathtt{bsn}_U, h_u, Sign'_U)$, abort. |
| If $1 \neq \mathsf{Link}(\mathtt{gpk}, (Y, Sign_V), (h_V, Sign'_V), \mathtt{bsn}_V)$, abort. | | If $1 \neq \mathsf{Link}(\mathtt{gpk}, (X, Sign_U), (h_u, Sign'_U), \mathtt{bsn}_U)$, abort. |
| Session key: $\mathsf{skey} = k_1$ | | Session key: $\mathsf{skey} = k_1$ |

Fig. 3. Two Pass Parallel AAKA Protocol.

$\mathsf{Bad}] = 1$, we have $4 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{\mathbb{G}} + 2 \cdot \varepsilon_{\mathrm{PRF}} = 2 \cdot |1/(q_s/2 \cdot (q_s/2-1)) \Pr_{\mathcal{A}}[\mathsf{Succ} \wedge \mathtt{Test}_{\mathsf{S}} \wedge \overline{\mathsf{Bad}}] - 1/2 \cdot 1/(q_s/2(q_s/2-1)) \cdot \Pr_{\mathcal{A}}[\mathtt{Test}_{\mathsf{S}} \wedge \overline{\mathsf{Bad}}]|$.

Therefore, using the above results, we obtain the desired result for the theorem as follows.

$\mathsf{Adv}^{\mathsf{AKE}}_{\mathcal{A},3\mathrm{AKA-FB}} = 2 \cdot |\Pr_{\mathcal{A}}[\mathsf{Succ}] - 1/2|$

$= 2 \cdot |\Pr[\mathsf{Succ} \wedge \mathtt{Test}_{\mathsf{E}}] + \Pr[\mathsf{Succ} \wedge \mathtt{Test}_{\mathsf{S}} \wedge \mathsf{Bad}] + \Pr[\mathsf{Succ} \wedge \mathtt{Test}_{\mathsf{S}} \wedge \overline{\mathsf{Bad}}] - 1/2|$

$\leq q_{ex} \cdot (4 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{\mathbb{G}} + 2 \cdot \varepsilon_{\mathrm{PRF}}) + 2 \cdot |-(1/2)\Pr[\mathtt{Test}_{\mathsf{S}}] + \Pr[\mathsf{Succ} \wedge \mathtt{Test}_{\mathsf{S}} \wedge \mathsf{Bad}] + \Pr[\mathsf{Succ} \wedge \mathtt{Test}_{\mathsf{S}} \wedge \overline{\mathsf{Bad}}]|$

$\leq q_{ex} \cdot (4 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{\mathbb{G}} + 2 \cdot \varepsilon_{\mathrm{PRF}}) + (q_s(q_s-1))/2 \cdot (4 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{\mathbb{G}} + 2 \cdot \varepsilon_{\mathrm{PRF}})$
$+ |-\Pr[\mathtt{Test}_{\mathsf{S}}] + \Pr[\mathtt{Test}_{\mathsf{S}} \wedge \overline{\mathsf{Bad}}]$
$+ 2 \cdot \Pr[\mathsf{Succ} \wedge \mathtt{Test}_{\mathsf{S}} \wedge \mathsf{Bad}]|$

$\leq (2q_{ex} + q_s(q_s-1))/2 \cdot (4 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{\mathbb{G}} + 2 \cdot \varepsilon_{\mathrm{PRF}}) + \Pr[\mathsf{Bad}]$

$\leq (2q_{ex} + q_s(q_s-1))(2 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{\mathbb{G}} + \varepsilon_{\mathrm{PRF}}) + |\mathcal{U}| \cdot \varepsilon_{\mathrm{DAA}} + (q_s \cdot (q_s + q_{ex}))/2^{\lambda}$. ■

**Lemma 2** *The proposed protocol achieves anonymity if the DDH assumption in $\mathbb{G}$ holds and the underlying DAA signature scheme is secure.*

*Proof.* The AAKA protocol combines the ephemeral Diffie-Hellman KA protocol and a DAA signature scheme. Each message is transmitted together with a DAA signature generated for the message. By assumption, a DAA signature scheme provides anonymity on a signer [6]. Basically, a verifier cannot learn the identity of the signer from a given signature due to the anonymity of the DAA signature. Since the anonymity of the DAA signature is independent of the position of the protocol participants, the proposed protocol achieves the bilateral anonymity, i.e., it is anonymous. ■

**Lemma 3** *The above proposed protocol achieves the full binding property if the underlying DAA signature scheme is secure.*

*Proof.* Let DAA $=$ (Setup, Join, Sign, Vrfy, Link) be a DAA signature scheme. In the proof, we construct an efficient attacker $\mathcal{F}$ to break the non-frameability of the DAA scheme, using an adversary, $\mathcal{A}$ attacking the proposed AAKA protocol, 3AKA-FB, in the $L$-BP game or the $R$-BP game. The goal of $\mathcal{F}$ is to output two linkable signatures with the same basename but from two different secret keys.

As defined in [25], we assume that $\mathcal{F}$ is given all secret keys including mik corresponding to gpk after executing Setup of DAA. In other words, no oracles are required to $\mathcal{F}$ because no honest parties exist. We provide a simulation of the $L$-BP game for 3AKA-FB as follows:

- Initialization. $\mathcal{F}$ generates a cyclic group $\mathbb{G}$ of order $q$ and $g$ a random generator of $\mathbb{G}$ of prime order $q$. In addition, $\mathcal{F}$ generates a message authentication code MAC, a pseudorandom function PRF, and a cryptographic hash function $H : \{0,1\}^* \to \{0,1\}^\ell$. A set of public parameters of 3AKA-FB, $(\mathtt{gpk}, (\mathbb{G}, g, q), \text{MAC}, \text{PRF}, H)$ are given to $\mathcal{A}$.
  In addition, mik corresponding to gpk is given to $\mathcal{A}$. As defined in the model, we consider a strongest attack environment, that is, no oracles are required to $\mathcal{A}$ because all secret keys are known to $\mathcal{A}$.

Assume that $\mathcal{A}$ outputs a pair of instances $\Pi_U^i$ and $\Pi_V^j$ for two different users $U$ and $V$, and $\mathsf{sid}_U^i = \mathsf{sid}_V^j$ which is a collection of all transcripts generated from a complete protocol run between the instances. When $\mathcal{A}$ outputs $\mathsf{sid}_U^i (= \mathsf{sid}_V^j)$, $\mathcal{F}$ outputs $(X, Sign_U, \mathtt{bsn}_U), (h_U, Sign'_U, \mathtt{bsn}_U)$.

Assume that the instances are anonymously *partnered*. Let $\mathsf{sid}_U^i = \mathsf{sid}_V^j = \{(X = g^x, Sign_U), (Y = g^y, Sign_V), (Sign'_U)\}$ where $K = g^{xy}$, $k_0 = \text{PRF}_K(0)$, $k_1 = \text{PRF}_K(1)$ and $Sign_U$, $Sign'_U$, and $Sign_V$ are DAA signatures on the messages, $X = g^x$, $h_u = H(X, Y, k_0)$, and $h_v = H(Y, X, k_0)$, respectively. Let $\mathcal{T}_U^i = \{(X = g^x, Sign_U)_1, (Sign'_U)_2\}$ and $\mathcal{T}_V^j = \{(Y = g^y, Sign_V)_1\}$. Here, $(\cdots)_i$ denotes a transcript sent in the $i^{\text{th}}$ pass. We have $\mathcal{T}_U^i = \mathsf{sid}_V^j \setminus \mathcal{T}_V^j$.

Since $Sign_U$ and $Sign'_U$ are valid DAA signatures by definition of the $L$-BP, we have $\mathsf{Vrfy}(\mathtt{gpk}, \mathtt{bsn}_U, X, Sign_U) = 1$ and $\mathsf{Vrfy}(\mathtt{gpk}, \mathtt{bsn}_U, h_u, Sign'_U) = 1$. In addition, because the both instances accept the session by assumption, we have $\mathsf{Link}(\mathtt{gpk}, (X, Sign_U), (h_U, Sign'_U), \mathtt{bsn}_U) = 1$.

If $Sign_U$ and $Sign'_U$ are generated by two different secret keys then $\mathcal{A}$ succeeds in the $L$-BP game; otherwise, $\mathcal{A}$ fails in the game. Therefore, if $\mathcal{A}$ succeeds in the $L$-BP game then $\mathcal{F}$ succeeds in breaking the non-frameability of DAA. If not, $\mathcal{F}$ fails to break the non-frameability of DAA. The advantage of $\mathcal{F}$ is the same to that of $\mathcal{A}$.

For the $R$-BP, we can obviously show that $\pi$ achieves it as follows: A single transcript is only sent from $V$ to $U$ in the protocol. The transcript consists of $Y$ and $Sign_V$, where $Sign_V$ is a valid DAA signature on $Y$. That is, $\{Y, Sign_V\} = \mathsf{sid}_U^i \setminus \mathcal{T}_U^i$ must be generated only by a single player.

Hence, the proposed protocol achieves the bilateral full BP. ■

### B. Two-Pass Parallel AAKA Protocol

In this section we present a two-pass AAKA protocol. Consider the delay-tolerant network in which the communication delay overwhelms the computation time. For example, the computation time is in milisecond order, but the communication delay is in hours or even in days. The delay-tolerant network also known as the pocket-switching network or opportunistic network, has very high delay due to its inherent nature of very limited connectivity. Because of this high latency, reducing the number of passes in a protocol significantly improves the session establishing time and also in the consequent communication time.

The basic structure of our protocol is similar to the KA protocol of [28]. The protocol runs in two rounds. However, our protocol performs a mutual anonymous authentication process. For explicit key confirmation, this protocol uses a DAA signature. See Fig. 3.

**Theorem 2** *The above protocol is a secure AAKA protocol with the full binding property if the DDH assumption in $\mathbb{G}$ holds and the underlying DAA signature scheme is secure.*

*Proof.* The theorem can be proved using a concept similar to that of the proof of Theorem 1. We omit concrete proof details. ■

### C. MAC-Based AAKA Protocols

We can convert the previous AAKA protocols into MAC-based AAKA protocols using a MAC tag for explicit key confirmation. Since MAC can be efficiently executed compared to an anonymous signature such as a DAA signature, the MAC-based protocols can gain advantages in terms of the computational costs. However, the protocols provide a weakened binding property that has the same strength as that of the property presented in [6], i.e., the partial BP defined in Section 3 instead of the (bilateral) full BP.

We briefly depict these MAC-based protocols in Fig. 4 and Fig. 5. In the figures, $MAC_U \leftarrow \mathsf{MAC\text{-}Gen}(H(g^{xy}), X||Y)$ and $MAC_V \leftarrow \mathsf{MAC\text{-}Gen}(H(g^{xy}), Y||X)$. Our MAC-based protocols are similar to the DAA-SIGMA of [6]. We can prove the security of the above protocols using a similar proof idea as [6] under the hardness of the DDH problem.

### D. Variants

The AAKA protocols proposed in the previous section can be modified for achieving various properties.

| | | |
|---|---|---|
| Round 1. | $U \rightarrow V$: | $X = g^x, Sign_U$ |
| Round 2. | $U \leftarrow V$: | $Y = g^y, Sign_V, MAC_V$ |
| Round 3. | $U \rightarrow V$: | $MAC_U$ |
| Session key | $U, V$: | $k_1 = \text{PRF}_{g^{xy}}(1)$ |

Fig. 4. Three-pass MAC-based AAKA protocol.

| | | |
|---|---|---|
| Round 1. | $U \rightarrow V$: | $X = g^x, Sign_U$ |
| | $U \leftarrow V$: | $Y = g^y, Sign_U$ |
| Round 2. | $U \rightarrow V$: | $MAC_U$ |
| | $U \leftarrow V$: | $MAC_V$ |
| Session key | $U, V$: | $k_1 = \text{PRF}_{g^{xy}}(1)$ |

Fig. 5. Two-pass parallel MAC-based AAKA protocol.

**Unilateral anonymous authentication**. The AAKA protocols proposed in the previous section were constructed to provide mutual anonymous authentication. These protocols can be easily converted into AAKA protocols to provide unilateral anonymous authentication by replacing an anonymous signature with a standard signature for one party. In general, it is assumed that the standard signature scheme does not guarantee anonymity on signers.

**Signer identification**. By using the group signature scheme supporting the opening and linking capability in [14], the proposed protocols can open signatures to identify signers. Note that DAA does not have the capability of opening signatures.

## V. CONCLUSION

We have presented AAKA protocols that provide binding properties, which guarantee that anonymous messages are bound to a single message sender. The binding properties can be crucially used for not only normal anonymous authentication but also more refined anonymous services such as rewarding/penalty systems. Our approach is based on the use of an anonymous signature scheme with a signer-controlled but partially enforced linkability such as DAA. It would be an open issue to design other efficient methods to achieve the full binding property.

## REFERENCES


[1] A. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. CRC Press, 1996.

[2] J. O. Kwon and I. R. Jeong, "Relations among security models for authenticated key exchange," *ETRI J.*, vol. 36, no. 5, pp. 856–864, 2014.

[3] J. Kohl and C. Neuman, "The kerberos network authentication service (v5)," tech. rep., RFC 1510, Sept, 1993.

[4] A. O. Freier, P. Karlton, and P. C. Kocher, "The SSL protocol — version 3.0," Internet Draft, Transport Layer Security Working Group, Nov. 1996.

[5] ITU-T recommendation X.509: Information technology — open systems interconnection — the directory: Authentication framework, ITU-T, 1997.

[6] J. Walker and J. Li, "Key exchange with anonymous authentication using daa-sigma protocol," in *Proc. INTRUST* (L. Chen and M. Yung, eds.), vol. 6802 of *Lecture Notes in Computer Science*, Springer, 2010, pp. 108–127.

[7] D. Chaum and E. van Heyst, "Group signatures," in *Proc. EUROCRYPT* (D. W. Davies, ed.), vol. 547 of *Lecture Notes in Computer Science*, Springer, 1991, pp. 257–265.

[8] E. F. Brickell, J. Camenisch, and L. Chen, "Direct anonymous attestation," in *Proc. ACM CCS*, pp. 132–145, 2004.

[9] J. Camenisch and A. Lysyanskaya, "Signature schemes and anonymous credentials from bilinear maps," in *Proc. CRYPTO* (M. K. Franklin, ed.), vol. 3152 of *Lecture Notes in Computer Science*, Springer, 2004, pp. 56–72.

[10] M. Bellare, D. Micciancio, and B. Warinschi, "Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions," in *Proc. EUROCRYPT* (E. Biham, ed.), vol. 2656 of *Lecture Notes in Computer Science*, Springer, 2003, pp. 614–629.

[11] M. Bellare, H. Shi, and C. Zhang, "Foundations of group signatures: The case of dynamic groups," in *Proc. CT-RSA* (A. Menezes, ed.), vol. 3376 of *Lecture Notes in Computer Science*, Springer, 2005, pp. 136–153.

[12] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik, "A practical and provably secure coalition-resistant group signature scheme," in *Proc. CRYPTO* (M. Bellare, ed.), vol. 1880 of *Lecture Notes in Computer Science*, Springer, 2000, pp. 255–270.

[13] C.-M. Park and H.-S. Lee, "Pairing-friendly curves with minimal security loss by cheon's algorithm," *ETRI J.*, vol. 33, no. 4, pp. 656–659, 2011.

[14] J. Y. Hwang, S. Lee, B.-H. Chung, H. S. Cho, and D. Nyang, "Short group signatures with controllable linkability," in *Proc. LightSec*, vol. 0, pp. 44–52, 2011.

[15] H. Ge and S. R. Tate, "A direct anonymous attestation scheme for embedded devices," in *Proc. PKC 2007*, Springer, 2007, pp. 16–30.

[16] E. Brickell, L. Chen, and J. Li, "Simplified security notions of direct anonymous attestation and a concrete scheme from pairings," *Int. J. Inf. Sec.*, vol. 8, no. 5, pp. 315–330, 2009.

[17] E. Brickell and J. Li, "A pairing-based daa scheme further reducing tpm resources," in *Proc. TRUST*, Berlin, Germany, June, 2010, pp. 181–195.

[18] L. Chen, P. Morrissey, and N. P. Smart, "Daa: Fixing the pairing based protocols," *IACR Cryptology ePrint Archive*, vol. 2009, p. 198, 2009.

[19] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[20] M. Bellare and P. Rogaway, "Entity authentication and key distribution," in *Proc. CRYPTO* (D. R. Stinson, ed.), vol. 773 of *Lecture Notes in Computer Science*, Springer, 1993, pp. 232–249.

[21] J. Katz and M. Yung, "Scalable protocols for authenticated group key exchange," in *CRYPTO* (D. Boneh, ed.), vol. 2729 of *Lecture Notes in Computer Science*, pp. 110–125, Springer, 2003.

[22] A. Leung and C. J. Mitchell, "Ninja: Non identity based, privacy preserving authentication for ubiquitous environments," in *Ubicomp* (J. Krumm, G. D. Abowd, A. Seneviratne, and T. Strang, eds.), vol. 4717 of *Lecture Notes in Computer Science*, pp. 73–90, Springer, 2007.

[23] E. Cesena, H. Löhr, G. Ramunno, A.-R. Sadeghi, and D. Vernizzi, "Anonymous authentication with tls and daa," in *Proc. TRUST*, Berlin, Germany, June. 2010, pp. 47–62.

[24] L. Chen *et al.*, "Lightweight anonymous authentication with tls and daa for embedded mobile devices," *IACR Cryptology ePrint Archive*, vol. 2011, p. 101, 2011.

[25] D. Bernhard, G. Fuchsbauer, E. Ghadafi, N. Smart, and B. Warinschi, "Anonymous attestation with user-controlled linkability," *Int. J. Inf. Sec.*, vol. 12, no. 3, pp. 219–249, 2013.

[26] J. Y. Hwang, S. Eom, K. Chang, P. J. Lee, and D. Nyang, "Anonymity-based authenticated key agreement with full binding property," in *Proc. WISA*, vol. 7690, pp. 177–191, 2012.

[27] D. Harkins and D. Carrel, "The Internet Key Exchange (IKE)." RFC 2409 (Proposed Standard), Nov. 1998. Obsoleted by RFC 4306, updated by RFC 4109.

[28] ISO/IEC 9798-3 Information Technology — Security techniques — Entity Authentication Mechanisms — Part 3: Mechanisms using digital signature techniques, 1998. 2nd ed.

[29] A. Acquisti, S. W. Smith, and A.-R. Sadeghi, eds., *Trust and Trustworthy Computing, Third International Conference, TRUST 2010,* Berlin, Germany, June 21-23, 2010. Proceedings, vol. 6101 of *Lecture Notes in Computer Science*, Springer, 2010.

**Jung Yeon Hwang** received the B.Sc. degree in mathematics from Korea University, Seoul, Republic of Korea, in 1999, and the M.S.and Ph.D. degrees in information security from Korea University, in 2006 respectively. He had worked as a post-doctoral researcher at Korea University from 2006 to 2009. Since 2009, he has been a senior member of the engineering staff at Electronics and Telecommunications Research Institute (ETRI), Korea. His research interests include fuzzy cryptography, broadcast encryption, authentication, and privacy-enhancing techniques, and their applications.

**Sungwook Eom** received the B.S. degree in electronic and electrical engineering from Kyungpook National University, Republic of Korea in 2005. He is currently a Ph.D. student in electronic and electrical engineering at POSTECH, Pohang, Kyungbuk, Republic of Korea. His research interests include cryptography and information security.

**Ku-Young Chang** received the B.S., M.S., and Ph.D. degrees in mathematics from Korea University, Seoul, Republic of Korea in 1995, 1997, and 2000, respectively. He is currently a Principal Researcher of Cryptography Research Section at ETRI, Daejeon, Korea. His research interests include cryptography, data privacy, and finite field theory.

**Pil Joong Lee** received the B.S. and M.S. degrees from Seoul National University and the Engineer degree and Ph.D. degree from University of California at Los Angeles. Before joining POSTECH, he worked for Jet Propulsion Laboratory from 1980 to 1985, and for Bell Communications Research from 1985 to 1990. He was the Dean of Graduate School for Information Technology at POSTECH, Pohang, Kyungbuk, Republic of Korea, and the Director of POSTECH Information Research Lab from 2000 to 2003, and also served as the president of the Korea Institute of Information Security and Cryptology in 2004. He is currently with the Department of Electrical Engineering at POSTECH. His research interests include cryptography and information security.

**DaeHun Nyang** received the B.Eng. degree in Electronic Engineering from Korea Advanced Institute of Science and Technology in 1994 and the M.S. and Ph.D. degrees in Computer Science from Yonsei University, Seoul, Republic of Korea, in 1996 and 2000, respectively. He was a Senior Member of the engineering staff at ETRI, Korea, from 2000 to 2003. Since 2003, he has been a Full Professor of the school of Computer and Information Engineering at Inha University, Korea, where he is also the Founding Director of Information Security Research Laboratory. He is also a consultant for the Korean Information Security Agency and a member of the board of directors and editorial board of the Korean Institute of Information Security and Cryptology. His research interests include public key cryptography and information security, privacy, biometrics and its applications to authentication, and SDN network security. He is also interested in traffic measurement technology.