Key Exchange with Anonymous Authentication using DAA-SIGMA Protocol

Jesse Walker and Jiangtao Li

Intel Labs
{jesse.walker, jiangtao.li}@intel.com

Abstract. Anonymous digital signatures such as Direct Anonymous Attestation (DAA) and group signatures have been a fundamental building block for anonymous entity authentication. In this paper, we show how to incorporate DAA schemes into a key exchange protocol between two entities to achieve anonymous authentication and to derive a shared key between them. We propose a modification to the SIGMA key exchange protocol used in the Internet Key Exchange (IKE) standards to support anonymous authentication using DAA. Our key exchange protocol can be also extended to support group signature schemes instead of DAA. We present a secure model for key exchange with anonymous authentication derived from of the Canetti-Krawczyk key-exchange security model. We formally prove that our DAA-SIGMA protocol is secure under our security model.

1 Introduction

Anonymous digital signatures such as group signatures [19, 1, 3, 4], Direct Anonymous Attestation (DAA) [5, 6, 20, 21, 12], and anonymous credentials [13–15] play an important role in privacy enhanced technologies. They allow an entity (e.g., a user or a hardware device) to create a signature without revealing its identity. Anonymous signatures also enable anonymous entity authentication.

The concept of group signature scheme was first introduced by Chaum and van Heyst [19] in 1991. In a group signature scheme, all the group members share a group public key, yet each member has a unique private key. A group signature created by a group member is anonymous to the verifiers but traceable to a trusted group manager. Many group signature schemes have been proposed, e.g., in [1, 3, 4, 23]. Direct Anonymous Attestation (DAA) was first introduced by Brickell, Camenisch, and Chen [5] for remote anonymous authentication of a Trusted Platform Module (TPM). DAA can be seen as a special group signature scheme without the traceability feature. DAA has received a lot of attentions in the trusted computing community, and many DAA schemes have been developed recently, e.g., in [6, 20, 22, 21, 12].

Anonymous digital signatures have attracted industry attentions in recent years. For example, the Trusted Computing group (TCG), a global industrial standard body [28], adopted the original DAA scheme [5] and standardized it in the TCG TPM Specification Version 1.2 [27]. The same DAA scheme has recently been adopted by ISO/IEC as an international standard. DAA has been implemented and already shipped in millions of TPMs today. Intel has implemented an extension of DAA called Enhanced Privacy ID [8–10] in the Intel P55 Ibex Peak chipsets [11]. Recently, ISO/IEC has started to develop two new international standards: one on anonymous digital signatures including group signature schemes and DAA schemes, the other on anonymous entity authentication using anonymous digital signatures.

DAA can be used for anonymous authentication in a straight-forward manner: a verifier sends a challenge message to a group member; the group member can authenticate to the verifier anonymously using his private key to create a DAA signature on the message. After verifying the DAA

signature, the verifier is convinced that the group member is a valid DAA signer, but does not learn who created the signature. TPM uses this method for anonymous authentication in order to obtain an attestation identity key credential from an issuer. This authentication method is limited because the group member did not verify the verifier's credential and there is no shared key derived from the authentication.

A more generic way to use DAA in anonymous authentication is to embed DAA in a Diffie-Hellman key exchange protocol, so that two entities can authenticate to each other and derive a shared session key for future communication. There have been several proposals to incorporate DAA into key exchange protocols. Balfe et al. [2] proposed anonymous authentication in peer-to-peer networks by embedding DAA with TLS and IPsec. Leung and Mitchell [26] introduced an anonymous authentication protocol based on DAA. Recently, Cesena et al. [18] proposed an anonymous authentication protocol based on TLS and DAA including a reference implementation.

1.1 Our Contribution

Although there have been several proposals to use DAA in Diffie-Hellman key exchange protocols [2, 26, 18] in the literature, to the best of our knowledge, there is no formal security model provided and none of these protocols has formal security proof to prove the security of these protocols. This is the motivation of this paper. There are two contributions of this paper. We describe each contribution briefly as follows.

A SECURITY MODEL FOR KEY EXCHANGE WITH ANONYMOUS AUTHENTICATION. We give a rigorous treatment to anonymous authentication and introduce a new security model for key exchange with anonymous authentication. Our security model derives from the Canetti-Krawczyk key-exchange security model [16, 17]. In the Canetti-Krawczyk security model, identity plays an important role in the proof of security. However in anonymous authentication, the identity of the entity who wants to be anonymous cannot be revealed in the key exchange. This creates a significant challenge in the definition of secure model and in the design of the key exchange protocol.

A SECURE KEY EXCHANGE PROTOCOL WITH ANONYMOUS AUTHENTICATION. We develop a new key exchange protocol with anonymous authentication based on DAA and the SIGMA family of key exchange protocols from IPsec [25] and the Internet Key Exchange (IKE) standards [24]. We call our protocol the DAA-SIGMA protocol. We present a formal security analysis of our DAA-SIGMA protocol based on our security model.

1.2 Organization of the Paper

Rest of this paper is organized as follows. We first introduce our notation and briefly review the DAA schemes and the SIGMA key exchange protocol in Section 2. We then propose the security model of key exchange protocol with anonymous authentication in Section 3. We present our DAA-SIGMA protocol in Section 4 and give the security proof in Section 5. We discuss some extensions of our DAA-SIGMA protocol in Section 6. We conclude our paper in Section 7.

2 Background and Building Blocks

In this section, we first review our notations and terminologies then briefly review the basic concepts of DAA. Next we review the SIGMA key exchange protocol.

2.1 Notations

We use the following notations in this paper. Let P and Q to be two entities of the key exchange protocol.

- $\text{MAC}_k(m)$: a message authentication code of m computed using the key k.
- $\text{SIG}_P(m)$: a signature of m created by the entity P.
- ID_P: an identity of the entity P.
- CERT_P: a public key certificate of the entity P.
- PRF: a pseudo-random function.

2.2 Review of DAA

In this section, we review the specification and security model of DAA proposed in [7]. The security model in [7] is simpler than the original DAA definition [5] and easier to understand the security properties of DAA. There are four types of players in a DAA scheme: an issuer $\mathcal{I}$, a TPM $\mathcal{M}_i$, a host $\mathcal{H}_i$ and a verifier $\mathcal{V}_j$. $\mathcal{M}_i$ and $\mathcal{H}_i$ form a platform in the trusted computing environment and share the role of a DAA signer $\mathcal{S}_i$. To simplify our notation, we treat the DAA signer as a single entity in the rest of this paper. A DAA scheme has following polynomial-algorithms or interactive protocols (Setup, Join, Sign, Verify, Link):

Setup : On input of a security parameter 1^k, $\mathcal{I}$ uses this randomized algorithm to produce a pair ($\texttt{gpk}$, $\texttt{isk}$), where $\texttt{isk}$ is the issuer's secret key, and $\texttt{gpk}$ is the public key including the global public parameters.

Join : A signer $\mathcal{S}_i$ and the issuer $\mathcal{I}$ run an interactive join protocol. In the end of the protocol, $\mathcal{S}_i$ outputs a secret key $\texttt{sk}_i$ and a membership credential $\texttt{cre}_i$ issued by $\mathcal{I}$. We denote the $\texttt{sk}_i$ and $\texttt{cre}_i$ pair as the signing key of $\mathcal{S}_i$. Note that the value of $\texttt{sk}_i$ is unknown to $\mathcal{I}$.

Sign : On input of $\texttt{gpk}$, $\texttt{sk}_i$, $\texttt{cre}_i$, a basename $\texttt{bsn}_j$, and a message m, $\mathcal{S}_i$ uses this algorithm to produce a signature σ on m. The basename $\texttt{bsn}_j$ can be either the name string of the verifier $\mathcal{V}_j$ or a special symbol $\perp$ and it is used for controlling the linkability.

Verify : On input of $\texttt{gpk}$, $\texttt{bsn}_j$, m, a candidate signature σ on m, and a revocation list $\texttt{RL}$, $\mathcal{V}_j$ uses this deterministic algorithm to determine whether σ is valid. The revocation is out of the scope of this paper.

Link : On input of two signatures σ_0 and σ_1, $\mathcal{V}_j$ uses this deterministic algorithm to return linked, unlinked, or invalid signatures.

The formal security definition of DAA can be found in [5, 7]. For completeness of this paper, we review the formal security model of DAA in [7] in the Appendix A. Informally, a DAA scheme is secure if the following properties hold:

- *Correctness.* A signature created using a valid signing key can be verified by any verifiers correctly.
- *Anonymity.* An adversary who are not in possession of a secret key $\texttt{sk}$ cannot learn the identity of the signer of signatures created using $\texttt{sk}$.
- *Unforgeability.* An adversary cannot forge a valid signature if he does not have a signing key or his signing keys have been all revoked.
- *User-controlled linkability.* Signatures created by the same set of $\texttt{sk}$, $\texttt{cre}$, $\texttt{bsn}$ can be linked, if $\texttt{bsn} \neq \perp$. However, signatures cannot be linked if they are created using different $\texttt{bsn}$ or if $\texttt{bsn} = \perp$.

If a signature σ was created using $\mathtt{bsn} = \perp$, then we call σ a random base signature; otherwise, we call σ a name base signature. For the DAA-SIGMA protocol, we first focus on random base signatures. We shall discuss how to use name base signatures in Section 6.

Let I be a DAA issuer. Let P be a DAA signer who has a valid signing key issued by I. We use the following notations in the rest of the paper:

- ID_I: the identity of the issuer I. For simplicity, we assume I creates only one group public key $\mathtt{gpk}$. One can figure out the corresponding $\mathtt{gpk}$ from ID_I.
- CERT_I: a public key certificate of I issued by a certificate authority to certify the group public key $\mathtt{gpk}$.
- $\text{DAA-SIG}_P(m)$: a DAA signature on message m created by the entity P. Let $\mathtt{sk}_P$ be P's secret key and $\mathtt{cre}_P$ be P's membership credential, then $\text{DAA-SIG}_P(m)$ denotes $\mathsf{Sign}(\mathtt{gpk}, \mathtt{sk}_P, \mathtt{cre}_P, \mathtt{bsn}, m)$ where $\mathtt{bsn} = \perp$ if $\mathtt{bsn}$ is not provided as input.

2.3 Review of SIGMA Key Exchange Protocol

We now review the SIGMA key exchange protocol. This key exchange protocol is one of the Internet Key-Exchange (IKE) protocols [24] used to establish secret shared keys for use in the Internet Protocol Security (IPsec) standards [25]. The SIGMA key exchange protocol uses Diffie-Hellman key exchange with "sign-and-mac" mechanism. It is presented as follows: Let P and Q be two entities of the key exchange protocol, where P is the protocol initiator who activates the session and Q is the intended peer of the session. Let G be a cyclic group of prime order q where the Decisional Diffie-Hellman (DDH) problem is hard and g be a generator of G. Let s be a unique session identifier chosen by P or Q. The SIGMA protocol has the following messages.

$$\begin{aligned}
&\text{Message 1 } (P \rightarrow Q): \ s, g^x \\
&\text{Message 2 } (P \leftarrow Q): \ s, g^y, \text{ID}_Q, \text{MAC}_{k_1}(s, \text{ID}_Q), \text{SIG}_Q(s, g^x, g^y) \\
&\text{Message 3 } (P \rightarrow Q): \ s, \text{ID}_P, \text{MAC}_{k_1}(s, \text{ID}_P), \text{SIG}_P(s, g^y, g^x)
\end{aligned}$$

In the above protocol, P chooses a random integer x and computes the ephemeral DH public key g^x. Analogously, Q chooses a random integer y and compute its ephemeral DH public key g^y. By exchanging g^x and g^y, both entities can compute g^{xy}, but "man-in-the-middle" attackers cannot. Both entities then derive a session key k_0 and a MAC key k_0 from g^{xy}. In message 2, Q computes a MAC of its identity using k_1 and signs the ephemeral DH public keys (g^x, g^y) using its signing key. P can verify the signature to ensure that the message 2 indeed comes from entity Q and verify the MAC to ensure that Q knows k_1, thus knows the session key k_0 as well. Similarly, P computes a MAC of its identity and a signature of the DH public keys in message 3. After Q verifies the message 3, both entities have established a session key k_0 and can use k_0 for further communications.

3 Security Model

In this section, we first review theCanetti-Krawczyk key exchange model [16, 17], and then describe how to extend this key exchange model to support anonymous authentication.

3.1 Review of Canetti-Krawczyk Key Exchange Model

This section reviews the Canetti-Krawczyk model [16] for authenticated key establishment protocols. A *session key adversary*, or *SK-adversary*, $\mathcal{A}$ is a probabilistic polynomial time algorithm that controls all communication between any parties using the set of queries specified in Table 1 below. An SK-adversary accomplishes this control by issuing queries to a set of *oracles* $\mathcal{O}_{s,P}$. An oracle $\mathcal{O}_{s,P}$ represents the protocol instance identified by s initiated by principal P. This communications instance is called a session. Each oracle responds to the queries, and the outputs of all the oracles $\mathcal{O}_{s,P}$ collectively represent the history of the protocol. The SK-adversary receives all output, and the uses this output to decide which oracles to activate next.

Table 1. Adversarial Queries

Query	Description
Send(P, Q, s, m)	Oracle $\mathcal{O}_{s,P}$ responds to message m to Q according to the protocol specification and decides whether to accept, reject, or continue. Accept means the oracle has completed the protocol with partner Q, a session id s, session key K, and $\mathcal{O}(s, P)$ signals its acceptance by including (P, s, Q) with its public output. Reject means the protocol failed. Continue means the oracle is waiting for its next protocol step. $\mathcal{O}_{s,P}$ returns its decision to the adversary A.
Session-Key-Reveal(P, Q, s)	If an oracle $\mathcal{O}_{s,P}$ has accepted and holds a session key K, the oracle returns K to SK-adversary $\mathcal{A}$. Otherwise $\mathcal{O}_{s,P}$ ignores this query. This query models a key leaking via break-ins, cryptanalysis, careless disposal of keys, and the like.
State-Reveal(P)	If the oracle $\mathcal{O}_{s,P}$ has accepted or holds a session key, it ignores this query. Otherwise the oracle returns its internal ephemeral session state, but no long-lived secrets. This query models the compromise of a single session.
Corrupt(P)	P responds to this query by yeielding its entire internal state, including all of the state held by all its oracles $\mathcal{O}_{s,P}$, and any long-lived keys. Subsequently the SK-adversary controls all of P's actions.
Expire-Session(P, Q, s)	This query causes oracle $\mathcal{O}_{s,P}$ to delete its session state, and oracle $\mathcal{O}_{s,P}$ will henceforth ignore any subsequent Session-Key-Reveal query, because it no longer has the targeted session key.
Test(P, Q, s)	An SK-adversary $\mathcal{A}$ may use the Test query only once. This query is used to measure that adversary's efficacy at distinguishing a real session key from a randomly generated one. If $\mathcal{O}_{s,P}$ has accepted with a session key K, then it selects a bit value b randomly; if $b = 0$, then it returns K and otherwise generates a random value K, which it returns instead. If the oracle has not accepted or does not have the session key, it does not respond to the Test query.

A session is called *exposed* if an SK-adversary issues a Session-Key-Reveal query against its oracle or an State-Reveal or Corrupt query against one of the session's principals.

As noted, the model uses the Test query to formalize the efficacy of an attack. The adversary may apply the Test query to any session that is not exposed. The test session returns a real or randomly generated key each with probability half and the adversary must guess whether the returned key is real. The adversary is called successful if it can distinguish the real key with probability significantly larger than 1/2.

An instance of protocol execution results in a partnership or agreement between two oracle instances of principals' communication. Partnership is a tricky notion, because the identity of the parties can be unknown before the protocol instance completes. In [16] Canetti and Krawcyzk

accommodate this by allowing the communicating oracles to learn the identity of the other party from the protocol instance.

Definition 1. *Suppose oracle $\mathcal{O}_{s,P}$ has accepted with output (P, s, Q). The oracle $\mathcal{O}_{s,R}$ is called the matching session of $\mathcal{O}_{s,P}$ if either*

1. *$\mathcal{O}_{s,R}$ has accepted with output (R, s, S), where $R = Q$ and $S = P$, or,*
2. *$\mathcal{O}_{s,R}$ has not completed.*

With this background, it is possible to define security in the model.

Definition 2. *A protocol π is called SK-secure if for all SK-adversaries $\mathcal{A}$ the following holds:*

1. *If two uncorrupted parties P and Q complete matching sessions $\mathcal{O}(s, P)$ and $\mathcal{O}(s, Q)$ of π with public outputs (P, s, Q) and (Q, s, P), respectively, then the session key K for these sessions is the same except with negligible probability.*
2. *$\mathcal{A}$ succeeds in distinguishing the output from its test query with probability not more than 1/2 plus a negligible fraction.*

3.2 Security Model for Key Exchange with Anonymous Authentication

In this paper, we primarily consider unilateral anonymous authentication. Mutual anonymous authentication shall be discussed in Section 6. Let P and Q be the two parties of the key exchange protocol with anonymous authentication, where P is the protocol initiator who activates the session and wants to anonymously authenticate to Q, the intended peer of the session. We use group-based anonymous digital signature schemes such as DAA or group signatures as the fundamental building block for anonymous authentication. Note that anonymous authentication in our model is not equivalent to anonymity without authentication, where P does not show any identity or signatures to Q. Anonymous authentication in our context means that P authenticates to Q as a member of a group created by an issuer I, but Q does not learn the actual identity of P.

Canetti and Krawcyzk prove that the SIGMA protocol in Section 2.3 is secure in the key exchange model described above. However, this security model assumes concrete identity of each party revealed during the protocol, so we will need to extend their model to accommodate anonymous authentication where the actual identity of the protocol initiator cannot be disclosed. The technical need is to have a specific identity to use to establish the matching sessions property. In particular, each party P must verify that the peer Q is the same entity throughout an instance of the protocol. To this end we make the following

Definition 3. *Suppose P is a member of a group identified by ID_I, and let $f(x)$ denote a one-way function. A pseudo identity of P is a pair $(\text{ID}_I, f(n))$, where n is a randomly selected nonce.*

We substitute anonymous identity instances in place of concrete identities in the SIGMA protocol. A principal P outputs $(\hat{P}, s, Q)$ where $\hat{P} = (\text{ID}_I, f(n))$, and argue that this is an appropriate identity for the Canetti-Krawcyzk key exchange model. Indeed, P's DAA signature proves its membership in a specific group, but this by itself does not prove that the messages of the protocol instance are exchanged with one single group member. To accomplish this, the definition of pseudo identity requires P to send a commitment $f(n)$ of the random value n to its peer Q, and then P proves it knows n as part of authentication. The allows the matching sessions property to be

established, because an adversary cannot arrange for a second group member R to interject itself into the protocol, because R does not know the value of n. We now define security of key exchange with anonymous authentication as follows.

Definition 4. *A key exchange protocol with anonymous authentication π is called* secure *if for all polynomial-time SK-adversaries $\mathcal{A}$ the following holds:*

1. *If two uncorrupted parties P and Q complete matching sessions $\mathcal{O}(s, P)$ and $\mathcal{O}(s, Q)$ of π with public outputs $(\hat{P}, s, Q)$ and $(Q, s, \hat{P})$, respectively, where $\hat{P}$ is the pseudo identity of P, then the session key K for these two sessions is the same except with negligible probability.*
2. *$\mathcal{A}$ succeeds in distinguishing the output from its test query with probability not more than 1/2 plus a negligible fraction.*
3. *An uncorrupted party P remains anonymous to its peers in all its sessions. The definition of anonymity follows the definition of the underlying anonymous digital signature schemes. If DAA is used, then anonymity of party P refers to user-controlled-anonymity in Appendix A.*

4 Proposed DAA-SIGMA Protocol

We now describe our DAA-SIGMA key exchange protocol as follows. Our key exchange protocol builds on top of the SIGMA protocol with the following changes: (1) P uses its DAA private key to signs the ephemeral DH public keys and (2) P uses (ID_I, g^x) as its pseudo identity in the protocol where the one-way function is defined as $f(x) = g^x$.

We assume the following initial information. Let P and Q be the two parties of the DAA-SIGMA key exchange protocol, where P is the protocol initiator who activates the session and Q is the intended peer of the session.

- P has a DAA signing key issued by an issuer I whose identity is ID_I. P can use the DAA signing algorithm DAA-SIG to create a DAA signature.
- Q has an identity ID_Q and a public-private key pair. Q can use his private key and a signature algorithm SIG to create a signature.
- Both entities agree on G, a cyclic group of prime order q in which the DDH problem is hard, and g, a generator of G. For example, one can choose primes p and q such that $q|p-1$, and then choose a number g whose multiplicative order modulo p is q.
- The protocol uses a message authentication code family MAC and a pseudorandom function family PRF.

The DAA-SIGMA protocol has the following messages and steps.

$$\begin{aligned}
&\text{Message 1 } (P \rightarrow Q): \ s, g^x \\
&\text{Message 2 } (P \leftarrow Q): \ s, g^y, \text{ID}_Q, \text{MAC}_{k_1}(s, \text{ID}_Q), \text{SIG}_Q(s, g^x, g^y) \\
&\text{Message 3 } (P \rightarrow Q): \ s, \text{ID}_I, g^x, \text{MAC}_{k_1}(s, \text{ID}_I, g^x), \text{DAA-SIG}_P(s, g^y, g^x)
\end{aligned}$$

Sending Message 1

1. P chooses a session id s.
2. P picks a random $x \leftarrow \mathbb{Z}_q$ and computes its ephemeral DH public key g^x.
3. P sends $\{s, g^x\}$ to Q.

Sending Message 2

1. Q picks a random $y \leftarrow \mathbb{Z}_q$ and computes its ephemeral DH public key g^y.
2. Q computes $g^{xy} = (g^x)^y$.
3. Q derives two keys $k_0 = \text{PRF}_{g^{xy}}(0)$, and $k_1 = \text{PRF}_{g^{xy}}(1)$.
4. Q securely erases y and g^{xy} from its memory.
5. Q computes $\text{MAC}_{k_1}(s, \text{ID}_Q)$.
6. Q computes $\text{SIG}_Q(s, g^x, g^y)$ using its private key.
7. Q sends $\{s, g^y, \text{ID}_Q, \text{MAC}_{k_1}(s, \text{ID}_Q), \text{SIG}_Q(s, g^x, g^y)\}$ to P.

Receiving Message 2

1. P computes $g^{xy} = (g^y)^x$.
2. P derives two keys $k_0 = \text{PRF}_{g^{xy}}(0)$, and $k_1 = \text{PRF}_{g^{xy}}(1)$.
3. P securely erases x and g^{xy} from its memory.
4. P verifies $\text{MAC}_{k_1}(s, \text{ID}_Q)$ using derived key k_1.
5. P retrieves the public key of Q from ID_Q.
6. P verifies $\text{SIG}_Q(s, g^x, g^y)$ using Q's public key.
7. If one of the above verification steps fails, the session is aborted and outputs "failure".

Sending Message 3

1. P computes $\text{MAC}_{k_1}(s, \text{ID}_I, g^x)$.
2. P uses its DAA signing key to compute $\text{DAA-SIG}_P(s, g^y, g^x)$.
3. P sends $\{s, \text{ID}_I, g^x, \text{MAC}_{k_1}(s, \text{ID}_I, g^x), \text{DAA-SIG}_P(s, g^y, g^x)\}$ to Q.
4. P completes the session with public output $((\text{ID}_I, g^x), s, \text{ID}_Q)$ and secret session key k_0.

Receiving Message 3

1. Q verifies $\text{MAC}_{k_1}(s, \text{ID}_I, g^x)$ using derived key k_1.
2. Q retrieves the DAA group public key of the issuer I from ID_I.
3. Q verifies $\text{DAA-SIG}_P(s, g^y, g^x)$ using the DAA group public key.
4. If one of the above verification steps fails, the session is aborted and outputs of "failure".
5. Q completes the session with public output $(\text{ID}_Q, s, (\text{ID}_I, g^x))$ and secret session key k_0.

5 Security Proof of the DAA-SIGMA Protocol

In Theorem 6 of [17] Canetti and Krawczyk prove that the SIGMA protocol is secure in their model. Our security proof of the DAA-SIGMA protocol follows their security proof. We replace the signing algorithm of the protocol initiator P with the DAA signing algorithm and P's identity with the pseudo identity (ID_I, g^x), where ID_I is the identity of the issuer and g^x is an ephemeral Diffie-Hellman value P generates. To prove the security of our DAA-SIGMA protocol, we begin with a review of the definition of the Decisional Diffie-Hellman (DDH) assumption, which is the assumption underlying the security of all Diffie-Hellman based key exchange protocols.

Assumption 1 *Let G, generated by g, be a cyclic group of prime order q. The DDH assumption in G holds if the probability distributions of $Q_0 = \{\langle g, g^x, g^y, g^{xy}\rangle : x, y \leftarrow \mathbb{Z}_q\}$ and $Q_1 = \{\langle g, g^x, g^y, g^r\rangle : x, y, r \leftarrow \mathbb{Z}_q\}$ are computationally indistinguishable.*

In addition to the DDH assumption, the security of the key exchange protocol in Section 4 also depends on the security of the underlying cryptographic primitives, namely, digital signatures, message authentication codes, pseudo-random functions, and DAA. We have the following theorem.

Theorem 1. *Under the DDH assumption in G and assuming the security of the underlying cryptographic functions* SIG, MAC, PRF, DAA-SIG, *the DAA-SIGMA protocol in Section 4 is secure in the model defined in Section 3.*

To prove the above theorem, we need to prove the three properties of secure key exchange protocol with anonymous authentication. We use the term Σ-attacker to denote an SK-adversary attacking against the DAA-SIGMA protocol.

P1. If two uncorrupted parties P and Q complete matching sessions $((\text{ID}_P, g^x), s, \text{ID}_Q)$ and $(\text{ID}_Q, s, (\text{ID}_P, g^x))$ respectively under the DAA-SIGMA protocol then, except for a negligible probability, the session key output in these sessions is the same.

P2. No efficient Σ-attacker can distinguish a real response from a random response with non-negligible probability. More precisely, if for any Σ-attacker we define

1. $P_{\text{real}}(\mathcal{A}) = \Pr[\mathcal{A}$ outputs 1 when given the real test session key$]$
2. $P_{\text{random}}(\mathcal{A}) = \Pr[\mathcal{A}$ ouptus 1 when given a random test session key$]$

then we need to prove that for any Σ-attacker $\mathcal{A}$ that $|P_{\text{real}}(\mathcal{A}) - P_{\text{random}}(\mathcal{A})|$ is negligible.

P3. An uncorrupted party P with a valid DAA private key remains anonymous to its peers in all its sessions. In other words, no Σ-attacker can break the user-controlled-anonymity property of DAA by engaging with P in the key exchange protocol.

For the proof they assume that the Σ-attacker $\mathcal{A}$ never exposes the test session, which would represent an attack that does not undermine the protocol directly.

5.1 Proof of Property P1

The proof for the first property in [17] goes over unchanged if we substitute our notion of a pseudo identity for the identity of the initiator and if we replace the initiator's signature scheme with DAA.

Indeed, the validation required of the initiator to establish Property P1 is unchanged, as message 2 changes in no way from the original version of the protocol. Message 3 differs in that it conveys a pseudo identity instead of an identity and is signed with a DAA signature. The pseudo identity (ID_I, g^x) specifies the DAA group, which the responder to the test session can use to retrieve the correct DAA verification key. The reasoning that this signature proves that the message originated from (ID_I, g^x) except with negligible probability is a standard reduction argument as before.

5.2 Proof of Property P2

The strategy in [17] to prove the second property is to show that any attacker who can distinguish the real from a random key with non-negligible probability can be used to construct a second attacker against one of the underlying primitives – signature, MAC, or the pseudo-random function family. In our setting we must replace the initiator's signature with a DAA signature, and its identity with a pseudo identity.

1. If adversary does not have any valid DAA private keys (all his DAA private keys have been revoked), then Lemma 7.4 in [17] can be used to break the unforgeability property of DAA.

2. If adversary does control some valid DAA private keys, then the commitment/pseudo identity guarantees that adversary cannot break the session key of two uncorrupted parties unless he can break DDH assumption.

5.3 Proof of Property P3

Suppose that an adversary $\mathcal{A}$ of the DAA-SIGMA protocol can extract the identity of the initiator from a protocol instance with non-negligible probability. We construct a DAA adversary $\mathcal{D}$ to invoke $\mathcal{A}$ as a subprogram, to return the identity ID_U of the initiator of a test session with non-negligible probability. $\mathcal{D}$ asks its DAA signing oracle to sign the parameters that are signed in message 3 and then uses $\mathcal{A}$ to execute an instance of the protocol, playing the role of initiator and responder to produce g^x and g^y. When $\mathcal{A}$ completes, it outputs its value ID_U for the initiator, and $\mathcal{D}$ outputs the same value. Since $\mathcal{D}$'s response uses the same signature as in message 3 of the test session from $\mathcal{A}$'s simulation, the probability that ID_U is correct is the same as for the simulation, i.e., is non-negligible. This contradicts the assumed anonymity property of DAA signatures.

6 Discussions and Extensions

In this section, we discuss several variants and possible extensions of the our DAA-SIGMA protocol. We begin with how to use group signature in the SIGMA key exchange protocol.

6.1 Using Group Signatures Instead of DAA

Although this paper primarily focuses on DAA, our DAA-SIGMA protocol can be easily extended to support group signatures [1, 3, 4, 23]. Both DAA and group signatures are group-based anonymous signature schemes. That is, there is an issuer in both schemes who creates a group of members and a group public key. Each group member has an individual private key. A DAA signature or a group signature can be verified using the corresponding group public key. Group signatures are different from DAA in that group signatures can be opened by a trusted authority and the identity of the actual signer can be extracted, whereas DAA signature cannot be opened.

To use a group signature scheme in the key exchange protocol for anonymous authentication, we can replace the DAA signature in the DAA-SIGMA protocol with a group signature. We called the modification as GS-SIGMA protocol. More specifically, we assume party P has a group private key issued by an issuer I. We use ID_I to denote the identity of the issuer and $\text{GS-SIG}_P(m)$ to denote a group signature on m created by P. The GS-SIGMA protocol has the following messages

$$\begin{aligned}
&\text{Message 1 } (P \rightarrow Q): \ s, g^x \\
&\text{Message 2 } (P \leftarrow Q): \ s, g^y, \text{ID}_Q, \text{MAC}_{k_1}(s, \text{ID}_Q), \text{SIG}_Q(s, g^x, g^y) \\
&\text{Message 3 } (P \rightarrow Q): \ s, \text{ID}_I, g^x, \text{MAC}_{k_1}(s, \text{ID}_I, g^x), \text{GS-SIG}_P(s, g^y, g^x)
\end{aligned}$$

This GS-SIGMA protocol is secure in the key exchange model for anonymous authentication defined in Section 3, if the DDH assumption holds and if the underlying SIG, MAC, PRF, and GS-SIG functions are secure. The security proof is straight-forward from Section 5.

6.2 Use Certificates Instead of Identities

In the DAA-SIGMA protocol presented in Section 4, ID_I and ID_Q represent the real identities of the issuer and the party Q, respectively. We assume that P can obtain Q's public key using ID_Q, and similarly Q can obtain P's DAA public key from the identity of the issuer ID_I. In practice, we can replace the identities with full certificates signed by a trusted Certificate Authority (CA). More specifically, assume all the parties share the public key of the trusted CA. If an entity has a regular public and private key pair, we assume it has a public-key certificate issued by the trusted CA. If an entity has a DAA private key, we assume it has CERT_I, a certificate issued by the trusted CA to certify the issuer including the DAA public key. The certificate-based DAA-SIGMA protocol is presented as follows.

$$\begin{aligned}
&\text{Message 1 } (P \rightarrow Q): \ s, g^x \\
&\text{Message 2 } (P \leftarrow Q): \ s, g^y, \text{CERT}_Q, \text{MAC}_{k_1}(s, \text{CERT}_Q), \text{SIG}_Q(s, g^x, g^y) \\
&\text{Message 3 } (P \rightarrow Q): \ s, \text{CERT}_I, g^x, \text{MAC}_{k_1}(s, \text{CERT}_I, g^x), \text{DAA-SIG}_P(s, g^y, g^x)
\end{aligned}$$

Note that, in the above protocol, the protocol participants P and Q do not need to retrieve public keys from the identities. Instead, they can use the public keys embedded in the certificate to verify the signatures in messages 2 and 3. Canetti and Krawczyk proved the certificate-based SIGMA protocol is secure in [17]. The security proofs for our certificate-based DAA-SIGMA protocol should work as well.

6.3 Supporting User-Controlled-Traceability and Revocation of DAA

As we mentioned earlier, the DAA signing algorithm takes the verifier's basename $\texttt{bsn}$ as an optional input. If the basename is not given in the signing algorithm, we treat $\texttt{bsn} = \perp$. DAA signatures are unlinkable if $\texttt{bsn} = \perp$ and linkable otherwise. This feature is called user-controlled-traceability in [7], as the signer and verifier can negotiate whether to use basename. The DAA-SIGMA protocol in Section 4 only supports unlinkable DAA signatures, but it can be easily extended to support user-controlled-traceability by passing the basename value in the protocol messages.

Another important feature of DAA is revocation. A revocation method in DAA proposed in [8, 10] is called signature-based revocation, where the issuer can revoke a DAA signer without knowing the signer's private key. In this revocation method, the issuer put DAA signatures that are suspicious or involved in malicious transaction a revocation list denoted as $\texttt{sRL}$. The verifier can request the signer to prove that his DAA signing key did not generate those signatures in the revocation list. To support this type of revocation, the verifier needs to sends the revocation list to the signer so that signer can prove innocent. To support this revocation method in the DAA-SIGMA protocol, the revocation list $\texttt{sRL}$ needs to be passed in the protocol messages.

We now extend the DAA-SIGMA protocol to support the above two functionalities of DAA as follows. In the following protocol, variables in brackets are optional fields.

$$\begin{aligned}
&\text{Message 1 } (P \rightarrow Q): \ s, g^x, [\text{ID}_I] \\
&\text{Message 2 } (P \leftarrow Q): \ s, g^y, \text{ID}_Q, [\texttt{bsn}_Q, \texttt{sRL}], \text{MAC}_{k_1}(s, \text{ID}_Q, [\texttt{bsn}_Q, \texttt{sRL}]), \text{SIG}_Q(s, g^x, g^y) \\
&\text{Message 3 } (P \rightarrow Q): \ s, \text{ID}_I, g^x, \text{MAC}_{k_1}(s, \text{ID}_I, g^x), \text{DAA-SIG}_P(s, g^y, g^x)
\end{aligned}$$

Note that without the optional fields (i.e., ID_I in the message 1 and $\texttt{bsn}_Q$ and $\texttt{sRL}$ in the message 2), the above protocol is exact the same DAA-SIGMA protocol as in Section 4. To support user-control-traceability feature of DAA, Q can send its basename $\texttt{bsn}_Q$ in the message 2 so that P can

create the DAA signature with $\texttt{bsn}_Q$ as input. To support signature-based revocation, P first reveals its DAA group (in this case, the identity of the issuer ID_I) to Q who can find the corresponding revocation list sRL of the DAA group. Q then sends sRL to P in the message 2 so that P can prove he has not been revoked in the message 3 as part of the DAA signature.

6.4 Mutual Anonymous Authentication using DAA

The DAA-SIGMA protocol presented in Section 4 is a key exchange protocol with unilateral anonymous authentication. This is a natural model for computation over the Internet. For example, a computer platform wants to have a key exchange with a web server with mutual authentication and yet remains anonymous. It is natural to have the protocol initiator as the anonymous entity. It may not make sense for mutual anonymous authentication over the Internet, as the protocol initiator needs to who is the protocol responder in order to know where to send the first message. Mutual anonymous authentication may be useful for ubiquitous computing environments or for vehicle communication where two entities (e.g., mobile devices or cars) are already physically close to each other and want to authenticate to each other anonymously.

We can extend the DAA-SIGMA protocol to support mutual anonymous authentication as follows. Assume each protocol party have a DAA private key for computing anonymous signatures, i.e., P has a DAA key issued by an issuer I and Q has a DAA key issued by an issue I', where I and I' could be the same entity or two different entities. The messages 1 and 3 are the same as in the Section 4. In message 2, Q uses $(\text{ID}_{I'}, g^y)$ as its pseudo identity instead of revealing its actual identity ID_Q in order to be anonymous.

$$\begin{aligned}
&\text{Message 1 } (P \rightarrow Q): \ s, g^x \\
&\text{Message 2 } (P \leftarrow Q): \ s, g^y, \text{ID}_{I'}, \text{MAC}_{k_1}(s, \text{ID}_{I'}, g^y), \text{DAA-SIG}_Q(s, g^x, g^y) \\
&\text{Message 3 } (P \rightarrow Q): \ s, \text{ID}_I, g^x, \text{MAC}_{k_1}(s, \text{ID}_I, g^x), \text{DAA-SIG}_P(s, g^y, g^x)
\end{aligned}$$

7 Conclusions and Future Work

We presented in this paper a new security model for key exchange with anonymous authentication derived from the Canetti-Krawczyk key exchange model. We proposed a modification to the SIGMA key exchange protocol with DAA incorporated and with the notion of pseudo identity to achieve anonymous authentication. We proved our DAA-SIGMA key exchange protocol is secure under the new security model. Future related research topics include

1. Provide a formal security model for mutual anonymous authentication and prove our mutual anonymous authentication protocol in Section 6 is secure.
2. Study how to incorporate DAA into SSL/TLS protocol with provable security under our new security model.

References

1. Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *Advances in Cryptology — CRYPTO '00*, volume 1880 of *LNCS*, pages 255–270. Springer, 2000.

2. Shane Balfe, Amit D. Lakhani, and Kenneth G. Paterson. Trusted computing: Providing security for peer-to-peer networks. In *Proceedings of the 5th IEEE International Conference on Peer-to-Peer Computing*, pages 117–124, August 2005.
3. Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *Advances in Cryptology — CRYPTO '04*, volume 3152 of *LNCS*, pages 41–55. Springer, 2004.
4. Dan Boneh and Hovav Shacham. Group signatures with verifier-local revocation. In *Proceedings of 11th ACM Conference on Computer and Communications Security*, pages 168–177, October 2004.
5. Ernie Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In *Proceedings of the 11th ACM Conference on Computer and Communications Security*, pages 132–145. ACM Press, 2004.
6. Ernie Brickell, Liqun Chen, and Jiangtao Li. A new direct anonymous attestation scheme from bilinear maps. In *Proceedings of 1st International Conference on Trusted Computing*, volume 4968 of *LNCS*, pages 166–178. Springer, 2008.
7. Ernie Brickell, Liqun Chen, and Jiangtao Li. Simplified security notions of direct anonymous attestation and a concrete scheme from pairings. *International Journal of Information Security*, 8(5):315–330, 2009.
8. Ernie Brickell and Jiangtao Li. Enhanced Privacy ID: A direct anonymous attestation scheme with enhanced revocation capabilities. In *Proceedings of the 6th ACM Workshop on Privacy in the Electronic Society*, pages 21–30, October 2007.
9. Ernie Brickell and Jiangtao Li. Enhanced Privacy ID: A remote anonymous attestation scheme for hardware devices. *Intel Technology Journal: Advances in Internet Security*, 13(2), 2009.
10. Ernie Brickell and Jiangtao Li. Enhanced Privacy ID from bilinear pairing for hardware authentication and attestation. In *Proceedings of 2nd IEEE International Conference on Information Privacy, Security, Risk and Trust*, 2010.
11. Ernie Brickell and Jiangtao Li. Enhanced Privacy ID: Hardware attestation beyond TPM. First Workshop on Anonymous Digital Signatures: Mechanisms & Usages, 2010. `http://www.trust2010.org/workshop-anon.html`.
12. Ernie Brickell and Jiangtao Li. A pairing-based DAA scheme further reducing TPM resources. In *Proceedings of 3rd International Conference on Trust and Trustworthy Computing*, volume 6101 of *LNCS*, pages 181–195. Springer, 2010.
13. Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *Advances in Cryptology — EUROCRYPT '01*, volume 2045 of *LNCS*, pages 93–118. Springer, 2001.
14. Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In *Proceedings of the 3rd Conference on Security in Communication Networks*, volume 2576 of *LNCS*, pages 268–289. Springer, 2002.
15. Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In *Advances in Cryptology — CRYPTO '04*, volume 3152 of *LNCS*, pages 56–72. Springer, 2004.
16. Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In *Advances in Cryptology — EUROCRYPT '01*, volume 2045 of *LNCS*, pages 453–474. Springer, 2001.
17. Ran Canetti and Hugo Krawczyk. Security analysis of IKE's signature-based key-exchange protocol. In *Advances in Cryptology — CRYPTO '02*, volume 2442 of *LNCS*, pages 143–161. Springer, 2002.
18. Emanuele Cesena, Hans Löhr, Gianluca Ramunno, Ahmad-Reza Sadeghi, and Davide Vernizzi. Anonymous authentication with tls and daa. In *Proceedings of 3rd International Conference on Trusted and Trustworthy Computing*, volume 6101 of *LNCS*, pages 47–62. Springer, 2010.
19. David Chaum and Eugène van Heyst. Group signatures. In *Advances in Cryptology — EUROCRYPT '91*, volume 547 of *LNCS*, pages 257–265. Springer, 1991.
20. Liqun Chen, Paul Morrissey, and Nigel P. Smart. Pairings in trusted computing. In *Proceedings of the 2nd Internation Conference on Pairing-Based Cryptography*, volume 5209 of *LNCS*, pages 1–17. Springer, 2008.
21. Liqun Chen, Dan Page, and Nigel P. Smart. On the design and implementation of an efficient DAA scheme. In *Proceedings of the 9th Smart Card Research and Advanced Application IFIP Conference*. Springer, 2010.
22. Xiaofeng Chen and Dengguo Feng. Direct anonymous attestation for next generation TPM. *Journal of Computers*, 3(12):43–50, 2008.
23. Jun Furukawa and Hideki Imai. An efficient group signature scheme from bilinear maps. *IEICE Transactions*, 89-A(5):1328–1338, 2006.
24. D. Harkins and D. Carrel. The Internet key exchange (IKE). IETF RFC 2409, November 1998.
25. S. Kent and R. Atkinson. Security architecture for the Internet protocol. IETF RFC 2401, November 1998.
26. Adrian Leung and Chris J. Mitchell. Ninja: Non identity based, privacy preserving authentication for ubiquitous environments. In *Proceedings of 9th International Conference on Ubiquitous Computing*, volume 4717 of *LNCS*, pages 73–90. Springer, 2007.
27. Trusted Computing Group. TCG TPM specification 1.2, 2003. `http://www.trustedcomputinggroup.org`.
28. Trusted Computing Group website. `http://www.trustedcomputinggroup.org`.

A Review Security Model of DAA

In this section, we review the specification and security model of DAA proposed in [7]. There are four types of players in a DAA scheme: an issuer $\mathcal{I}$, a TPM $\mathcal{M}_i$, a host $\mathcal{H}_i$ and a verifier $\mathcal{V}_j$. $\mathcal{M}_i$ and $\mathcal{H}_i$ form a platform in the trusted computing environment and share the role of a DAA signer. A DAA scheme has three polynomial-algorithms $(\mathsf{Setup}, \mathsf{Verify}, \mathsf{Link})$ and two interactive protocols $(\mathsf{Join}, \mathsf{Sign})$:

Setup : On input of a security parameter 1^k, $\mathcal{I}$ uses this randomized algorithm to produce a pair $(\mathtt{gpk}, \mathtt{isk})$, where $\mathtt{isk}$ is the issuer's secret key, and $\mathtt{gpk}$ is the public key including the global public parameters.

Join : This randomized algorithm consists of two sub-algorithms $\mathsf{Join_t}$ and $\mathsf{Join_i}$. $\mathcal{M}_i$ uses $\mathsf{Join_t}$ to produce a pair $(\mathtt{sk}_i, \mathtt{comm}_i)$, where $\mathtt{sk}_i$ is the TPM's secret key and $\mathtt{comm}_i$ is a commitment of $\mathtt{sk}_i$. On input of $\mathtt{comm}_i$ and $\mathtt{isk}$, $\mathcal{I}$ uses $\mathsf{Join_i}$ to produce $\mathtt{cre}_i$, which is a DAA credential associated with $\mathtt{sk}_i$. Note that the value $\mathtt{cre}_i$ is given to both $\mathcal{M}_i$ and $\mathcal{H}_i$, but the value $\mathtt{sk}_i$ is known to $\mathcal{M}_i$ only.

Sign : On input of $\mathtt{sk}_i$, $\mathtt{cre}_i$, a basename $\mathtt{bsn}_j$ (the name string of $\mathcal{V}_j$ or a special symbol $\perp$), and a message m that includes the data to be signed and the verifier's nonce n_V for freshness, $\mathcal{M}_i$ and $\mathcal{H}_i$ use this randomized algorithm to produce a signature σ on m under $(\mathtt{sk}_i, \mathtt{cre}_i)$ associated with $\mathtt{bsn}_j$. The basename $\mathtt{bsn}_j$ is used for controlling the linkability.

Verify : On input of m, $\mathtt{bsn}_j$, a candidate signature σ for m, and a set of revoked secret keys $\mathtt{RL}$, $\mathcal{V}_j$ uses this deterministic algorithm to return either 1 (accept) or 0 (reject). How to build the revocation list is out the scope of the DAA scheme.

Link : On input of two signatures σ_0 and σ_1, $\mathcal{V}_j$ uses this deterministic algorithm to return 1 (linked), 0 (unlinked) or $\perp$ (invalid signatures). Link will output $\perp$ if, by using an empty $\mathtt{RL}$, either $\mathsf{Verify}(\sigma_0) = 0$ or $\mathsf{Verify}(\sigma_1) = 0$ holds. Otherwise, Link will output 1 if signatures can be linked or 0 if the signatures cannot be linked.

A DAA scheme is secure if it is correct, user-controlled-anonymous, and user-controlled-traceable.

Correctness If both the signer and verifier are honest, that implies $\mathtt{sk}_i \notin \mathtt{RL}$, the signatures and their links generated by the signer will be accepted by the verifier with overwhelming probability. This means that the DAA scheme must meet the following consistency requirement.

$$
\begin{aligned}
&(\mathtt{gpk}, \mathtt{isk}) \leftarrow \mathsf{Setup}(1^k),\ (\mathtt{sk}_i, \mathtt{cre}_i) \leftarrow \mathsf{Join}(\mathtt{isk}, \mathtt{gpk}),\\
&(m_b, \sigma_b) \leftarrow \mathsf{Sign}(m_b, \mathtt{bsn}_j, \mathtt{sk}_i, \mathtt{cre}_i, \mathtt{gpk})|_{b=\{0,1\}},\\
&\Longrightarrow 1 \leftarrow \mathsf{Verify}(m_b, \mathtt{bsn}_j, \sigma_b, \mathtt{gpk}, \mathtt{RL})|_{b=\{0,1\}} \ \wedge\ 1 \leftarrow \mathsf{Link}(\sigma_0, \sigma_1, \mathtt{gpk})|_{\mathtt{bsn}_j \neq \perp}.
\end{aligned}
$$

User-Controlled-Anonymity A DAA scheme is user-controlled-anonymous if there is no probabilistic polynomial-time adversary can win the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ as follows:

- Initial: $\mathcal{C}$ runs $\mathsf{Setup}(1^k)$ and gives the resulting $\mathtt{isk}$ and $\mathtt{gpk}$ to $\mathcal{A}$.
- Phase 1: $\mathcal{C}$ is probed by $\mathcal{A}$ who makes the following queries:
  - Sign. $\mathcal{A}$ submits a signer's identity S, a basename $\mathtt{bsn}$ (either $\perp$ or a data string) and a message m of his choice to $\mathcal{C}$, who runs Sign to get a signature σ and responds with σ.

- Join. $\mathcal{A}$ submits a signer's identity S of his choice to $\mathcal{C}$, who runs Join_t with $\mathcal{A}$ to create $\mathtt{sk}$ and to obtain $\mathtt{cre}$ from $\mathcal{A}$. $\mathcal{C}$ verifies the validation of $\mathtt{cre}$ and keeps $\mathtt{sk}$ secret.
- Corrupt. $\mathcal{A}$ submits a signer's identity S of his choice to $\mathcal{C}$, who responds with the value $\mathtt{sk}$ of the signer.

– Challenge: At the end of Phase 1, $\mathcal{A}$ chooses two signers' identities S_0 and S_1, a message m and a basename $\mathtt{bsn}$ of his choice to $\mathcal{C}$. $\mathcal{A}$ must not have made any Corrupt query on either S_0 or S_1, and not have made the Sign query with the same $\mathtt{bsn}$ if $\mathtt{bsn} \neq \perp$ with either S_0 or S_1. To make the challenge, $\mathcal{C}$ chooses a bit b uniformly at random, signs m associated with $\mathtt{bsn}$ under $(\mathtt{sk}_b, \mathtt{cre}_b)$ to get a signature σ and returns σ to $\mathcal{A}$.
– Phase 2: $\mathcal{A}$ continues to probe $\mathcal{C}$ with the same type of queries that it made in Phase 1. Again, it is not allowed to corrupt any signer with the identity either S_0 or S_1, and not allowed to make any Sign query with $\mathtt{bsn}$ if $\mathtt{bsn} \neq \perp$ with either S_0 or S_1.
– Response: $\mathcal{A}$ returns a bit b'. We say that the adversary wins the game if $b = b'$.

Definition 5. *Let $\mathcal{A}$ denote an adversary that plays the game above. We denote by $\mathbf{Adv}[\mathcal{A}^{anon}_{\mathcal{DAA}}] = |\mathbf{Pr}[b' = b] - 1/2|$ the advantage of $\mathcal{A}$ in breaking the user-controlled-anonymity game. We say that a DAA scheme is user-controlled-anonymous if for any probabilistic polynomial-time adversary $\mathcal{A}$, $\mathbf{Adv}[\mathcal{A}^{anon}_{\mathcal{DAA}}]$ is negligible.*

User-Controlled-Traceability A DAA scheme is user-controlled-traceable if there is no probabilistic polynomial-time adversary can win the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ as follows:

– Initial: $\mathcal{C}$ executes $\mathsf{Setup}(1^k)$ and gives the resulting $\mathtt{gpk}$ to $\mathcal{A}$. It keeps $\mathtt{isk}$ secret.
– Probing: $\mathcal{C}$ is probed by $\mathcal{A}$ who makes the following queries:
- Sign. The same as in the game of user-controlled-anonymity.
- Semi-sign. $\mathcal{A}$ submits a signer's identity S along with the data transmitted from $\mathcal{H}_i$ to $\mathcal{M}_i$ in Sign of his choice to $\mathcal{C}$, who acts as $\mathcal{M}_i$ in Sign and responds with the data transmitted from $\mathcal{M}_i$ to $\mathcal{H}_i$ in the Sign protocol.
- Join. There are two cases of this query. Case 1: $\mathcal{A}$ submits a signer's identity S of his choice to $\mathcal{C}$, who runs Join to create $\mathtt{sk}$ and $\mathtt{cre}$ for the signer. Case 2: $\mathcal{A}$ submits a signer's identity S with a $\mathtt{sk}$ value of his choice to $\mathcal{C}$, who runs Join_i to create $\mathtt{cre}$ for the signer and puts the given $\mathtt{sk}$ into $\mathtt{RL}$. $\mathcal{C}$ responds the query with $\mathtt{cre}$. Suppose that $\mathcal{A}$ does not use a single S for both of the cases.
- Corrupt. This is the same as in the game of user-controlled-anonymity, except that at the end $\mathcal{C}$ puts the revealed $\mathtt{sk}$ into the list of $\mathtt{RL}$.

– Forge: $\mathcal{A}$ returns a signer's identity S, a signature σ, its signed message m and the associated basename $\mathtt{bsn}$. We say that the adversary wins the game if
1. $\mathsf{Verify}(m, \mathtt{bsn}, \sigma, \mathtt{gpk}, \mathtt{RL}) = 1$ (accepted), but σ is neither a response of the existing Sign queries nor a response of the existing Semi-sign queries (partially); and/or
2. In the case of $\mathtt{bsn} \neq \perp$, there exists another signature σ' associated with the same identity and $\mathtt{bsn}$, and the output of $\mathsf{Link}(\sigma, \sigma')$ is 0 (unlinked).

Definition 6. *Let $\mathcal{A}$ be an adversary that plays the game above. Let $\mathbf{Adv}[\mathcal{A}^{trace}_{\mathcal{DAA}}] = \mathbf{Pr}[\mathcal{A}\ wins]$ denote the advantage that $\mathcal{A}$ breaks the user-controlled-traceability game. We say that a DAA scheme is user-controlled-traceable if for any probabilistic polynomial-time adversary $\mathcal{A}$, $\mathbf{Adv}[\mathcal{A}^{trace}_{\mathcal{DAA}}]$ is negligible.*