5-2013

Leakage resilient authenticated key exchange secure in the auxiliary input model

Guomin YANG
Singapore Management University, gmyang@smu.edu.sg

Yi MU

Willy SUSILO

Duncan S. WONG

Leakage Resilient Authenticated Key Exchange Secure in the Auxiliary Input Model*

Guomin Yang[1], Yi Mu[1], Willy Susilo[1], and Duncan S. Wong[2]

[1] Centre for Computer and Information Security Research
School of Computer Science and Software Engineering
University of Wollongong, Australia
{gyang,ymu,wsusilo}@uow.edu.au
[2] Department of Computer Science
City University of Hong Kong
duncan@cityu.edu.hk

Abstract. Authenticated key exchange (AKE) protocols allow two parties communicating over an insecure network to establish a common secret key. They are among the most widely used cryptographic protocols in practice. In order to resist key-leakage attacks, several leakage resilient AKE protocols have been proposed recently in the bounded leakage model. In this paper, we initiate the study on leakage resilient AKE in the auxiliary input model. A promising way to construct such a protocol is to use a digital signature scheme that is *entropically-unforgeable under chosen message and auxiliary input attacks*. However, to date we are not aware of any digital signature scheme that can satisfy this requirement. On the other hand, we show that in the *random oracle model*, it is sufficient to use a digital signature scheme that is secure under *random message and auxiliary input attacks* in order to build a secure AKE protocol in the auxiliary input model, while the existence of such a digital signature scheme has already been proven. We will also give a comparison between the existing public-key encryption based and digital signature based leakage resilient AKE protocols. We show that the latter can provide a higher level of security than the former.

Keywords: Leakage resilient cryptography, authenticated key exchange, auxiliary input model.

1 Introduction

Leakage Resilient Cryptography. Traditional cryptographic systems always assume that the user secret keys are absolutely secure and out of the adversary's reach. However, in recent years, various kinds of side-channel attacks [21,9,22,19] have shown that we can extract some partial information of the user secret keys stored in a computing device by observing the physical output of a computation (e.g. running time, power consumption, radiation, etc.). In order

* This work is supported by the ARC Future Fellowship (FT0991397).

R.H. Deng and T. Feng (Eds.): ISPEC 2013, LNCS 7863, pp. 204–217, 2013.

to defend against different types of side-channel (or more general, key leakage) attacks, *leakage resilient cryptography* have become a popular research topic in recent years.

Before constructing a leakage resilient cryptosystem, we must first build an appropriate security model to define the information an adversary can learn in a key leakage attack. There are several leakage models that have been defined in the literature. In the *relative leakage* model [2], the leakage function h can be any polynomial-time computable function with bounded output length. More specifically, let k denote the size of a user secret key sk, then size of $h(sk)$ must be significantly smaller than k (e.g. the size of $h(sk)$ is less than $k/2$). Later, in [26], the restriction on the size of $h(sk)$ is relaxed by requiring that the secret key sk should still have a sufficient amount of min-entropy left after the adversary has observed $h(sk)$.

Another leakage model that has been extensively studied in the literature is the *bounded retrieval* model (BRM) [13,17,3]. In BRM, the size of the leakage can be arbitrarily large, however, users can increase their secret key size flexibly so as to allow for a large amount of leakage. The main goal of this setting is to ensure that increasing the size of the user secret key should not result in significant increase in the computation or communication cost.

Recently, Dodis et al. [16,14] defined another leakage model named the *auxiliary input* model. In this model, an adversary is allowed to see a computationally hard-to-invert function (e.g. a one-way permutation) of the secret key. In other words, the auxiliary input model has eliminated the leakage bound, and therefore can capture a larger class of leakage functions.

Authenticated Key Exchange. Authenticated Key Exchange (AKE) protocols are mechanisms that allow two parties communicating over an insecure network to establish a common secret key. They are a central piece for building secure communication channels. The design and analysis of AKE protocols have been extensively studied in the last three decades for different network settings (e.g. [6,7,5,11,1,24,27,10,29,28]). The first formal security model for AKE was proposed by Bellare and Rogaway [6]. The Bellare-Rogaway (or BR, for short) model and its variants are nowadays the *de facto* standard for analyzing the security of an AKE protocol. In particular, the Canetti-Krawczyk (CK) model [11], which can be considered as a combination of the BR model and the Bellare-Canetti-Krawczyk (BCK) model [4], has been used to prove the security of many practical AKE protocols such as the ISO protocol [20] (named SIG-DH in [11]) and the Internet Key Exchange (or SIGMA) protocol [12,23]. In FC'11, Yang et al. [28] extended the CK model to consider AKE under bad randomness. Two new models were proposed in [28], one formalized the reset attacks, and the other one formalized the bad randomness attacks. Some generic methods for enhancing the security of existing AKE protocols (such as ISO and SIGMA) were also proposed.

Leakage Resilient AKE. Several leakage resilient AKE protocols have been proposed recently. In [3], Alwen et al. extended the CK model to the bounded retrieval setting, and showed that in BRM, a leakage resilient AKE protocol can

be constructed from an entropically-unforgeable digital signature scheme secure under chosen-message attacks. Later, in [15], Dodis et al. showed that we can also construct leakage resilient AKE protocols based on public-key encryption (PKE) schemes secure in the bounded leakage model. In Sec. 4, we will review these two constructions and give a comparison between PKE-based and signature-based leakage resilient AKE protocols. In ASIACCS'11, based on the eCK security model proposed by LaMacchia, Lauter, and Mityagin [24], Moriyama and Okamoto [25] presented a new bounded leakage model for AKE protocols. They also proposed a two-pass implicitly-authenticated leakage resilient AKE protocol and proved its security in their security model.

Our Contributions. In this paper, we initiate the study on leakage resilient AKE in the auxiliary input model. Based on the result in [3], we can expect that such a protocol can be built by using a digital signature scheme that is *entropically-unforgeable under chosen message and auxiliary input attacks*. However, a problem arises when using this approach: to date we are not aware of any digital signature scheme that can satisfy the requirement. Although Faust et al. [18] have recently proposed a signature scheme that is secure under chosen-message and auxiliary input attacks, they assume the adversary can only see an *exponentially hard-to-invert function*, rather than a *computationally hard-to-invert function*, of the user secret key.

In this paper, we show that in the *random oracle model* [8], it is sufficient to use a digital signature scheme that is secure under *Random Message and Auxiliary Input Attacks* in order to build a secure AKE protocol in the auxiliary input model, while the existence of such a digital signature scheme has recently been proved in [18]. The key to ensure that this condition is sufficient comes from the specific design requirement for AKE protocols: an AKE participant not only receives but also generates random challenges in each AKE session. We will elaborate on this in Sec. 5.

It is worth noting that we may also use public-key encryption schemes secure in the auxiliary input model to achieve our goal. However, as we will show in Sec. 4, in the CK-model, Signature-based AKE protocols will offer better security than PKE-based protocols when the adversary can reveal the session state of a party during a protocol execution.

2 Preliminaries

Notations. We only consider probabilistic polynomial time (PPT) algorithms in this paper. In general, all the PPT algorithms have a security parameter 1^k as input, however, this input is usually omitted. We use $x \leftarrow S$ to denote the operation of randomly selecting x from a set S, and $y \leftarrow A(x)$ to indicate that y is the output of running an algorithm A on input x.

Leakage Functions. We follow the work of Dodis et al. [14] to define the class of admissible leakage functions $\mathcal{H}$ with regard to a public-key cryptosystem. We define $\mathcal{H}_{\mathsf{pkow}}(\ell(k))$ as the class of polynomial time computable functions

$h : \{0,1\}^{|pk|+|sk|} \to \{0,1\}^*$ such that given $(pk, h(pk, sk))$, no PPT adversary can find sk with probability greater than $\ell(k) \geq 2^k$, where (pk, sk) denote a random key pair generated by running the key generation algorithm of the public-key cryptosystem. In the rest of the paper, we will simply use $\mathcal{H}_{\mathsf{pkow}}$ to denote $\mathcal{H}_{\mathsf{pkow}}(\mathsf{negl}(k))$ where $\mathsf{negl}(\cdot)$ can be any negligible function.

Digital Signature. A digital signature scheme $\mathcal{DS}$ consists of three polynomial time algorithms.

- $\mathcal{DS}.\mathsf{SKG}(1^k)$: the key generation algorithm takes a security parameter 1^k as input and outputs a private signing key sk and a public verification key vk.
- $\mathcal{DS}.\mathsf{Sig}(sk, m)$): the signing algorithm takes a signing key sk and a message m from the message space $\mathcal{M}$ as input and outputs a signature σ.
- $\mathcal{DS}.\mathsf{Ver}(vk, m, \sigma)$): the verification algorithm takes a verification key vk, a message m, and a signature σ as input and outputs a bit '1' or '0'.

Correctness. For any $k \in \mathbb{N}$, $(vk, sk) \leftarrow \mathcal{DS}.\mathsf{SKG}(1^k)$, and $m \in \mathcal{M}$, we have

$$1 \leftarrow \mathcal{DS}.\mathsf{Ver}(vk, m, \mathcal{DS}.\mathsf{Sig}(sk, m)).$$

Unforgeability under Random Message and Auxiliary Input Attacks [18]. We say $\mathcal{DS}$ satisfies Random Message Unforgeability under Random Message and Auxiliary Input Attacks (RU-RMAA) with respect to a class of admissible leakage functions $\mathcal{H}$ if for any polynomial time algorithm $\mathcal{F}$, and any function $h \in \mathcal{H}$,

$$\mathbf{Adv}^{\mathsf{RU\text{-}RMAA}}_{\mathcal{DS},\mathcal{H},\mathcal{F}}(k) = \Pr\left[\begin{array}{c}(vk, sk) \leftarrow \mathcal{DS}.\mathsf{SKG}(1^k); m^* \leftarrow \mathcal{M}; \\ \sigma^* \leftarrow \mathcal{F}^{\mathcal{O}(sk,\cdot)}(vk, h(vk, sk), m^*) : \\ \mathcal{DS}.\mathsf{Ver}(vk, m^*, \sigma^*) = 1\end{array}\right]$$

is negligible in k, where the oracles $\mathcal{O}(sk, \cdot)$ is defined as

$$\text{Oracle } \mathcal{O}(sk, \cdot): \quad m \leftarrow \mathcal{M}; \quad \text{return } (m, \mathcal{DS}.\mathsf{Sig}(sk, m)).$$

Decisional Diffie-Hellman (DDH) Assumption: Let g denote a generator of a cyclic group $\mathbb{G}$ with prime order q. The DDH assumption says for any polynomial time algorithm $\mathcal{D}$,

$$\mathbf{Adv}^{\mathsf{DDH}}_{\mathcal{D}}(k) = \Pr[\mathcal{D}(g, g^a, g^b, Z) = 1 | Z = g^{ab}] - \Pr[\mathcal{D}(g, g^a, g^b, Z) = 1 | Z = g^r]$$

is negligible in k where a, b, r are randomly selected from $\mathbb{Z}_q$.

3 Security Model and Definition

3.1 System Model

An Authenticated Key Exchange (AKE) protocol consists of two probabilistic polynomial time algorithms: the Long-Lived Key generation algorithm SKG and a protocol execution algorithm P. In this paper, we focus on the public key

setting where the algorithm SKG returns a public key and the corresponding private key upon each invocation.

Protocol Participants. Let $\mathcal{U} = \{U_1, U_2, \cdots, U_n\}$ denote the set of users. Each user $U \in \mathcal{U}$ holds a public/private key pair (pk_U, sk_U) that is generated by honestly executing the Long-Lived Key generation algorithm SKG. A user may run many instances concurrently. We denote instance i of user U by Π_U^i.

Protocol Execution. A protocol execution algorithm P determines how an instance behaves in response to messages from the environment. Upon receiving an incoming message M_{in}, an instance executes the protocol P and generates

$$(M_{\text{out}}, \mathsf{dec}, \mathsf{sid}_U^i, \mathsf{pid}_U^i, \text{ssk}, St_U^i) \leftarrow \text{P}(U, pk_U, sk_U, St_U^i, M_{\text{in}}).$$

The first component M_{out} corresponds to the responding message, and the second component dec denotes the *decision* of the instance. A session id sid_U^i and partner id pid_U^i may be generated during the protocol execution. When the decision is acc, the instance holds a session key ssk which is to be used by upper layer applications. The instance may also update its internal state St_U^i.

Partnership. The partnership between two instances is defined via parter ID (pid) and session ID (sid). The pid names the party with which the instance believes it has just exchanged a key, and the sid is an identifier which uniquely labels the AKE session. We say two instances Π_U^i and Π_V^j are partners if $\mathsf{pid}_U^i = V, \mathsf{pid}_V^j = U$ and $\mathsf{sid}_U^i = \mathsf{sid}_V^j$.

3.2 Security Model

We consider an adversary $\mathcal{A}$ with full control over the routing and scheduling of network messages. Our adversarial model is defined via a game between the adversary $\mathcal{A}$ and a game simulator $\mathcal{SIM}$. $\mathcal{SIM}$ first tosses a random coin b which will be used later in the game. $\mathcal{SIM}$ then generates for each $U \in \mathcal{U}$ a public/secret key pair (pk_U, sk_U) and gives pk_U and auxiliary input $h_U(pk_U, sk_U)$ to $\mathcal{A}$ where $h_U \in \mathcal{H}_{\mathsf{pkow}}$. $\mathcal{A}$ is allowed to make the following oracle queries to the simulator:

- Send(U, i, m): This query allows the adversary to send a message m to an instance Π_U^i. If the message m is sent by another instance $\Pi_{U'}^j$ with the intended receiver U, then this query models a passive attack. Otherwise, it models an active attack by the adversary. The simulator then simulates the reaction of Π_U^i upon receiving the message m by running $\text{P}(U, pk_U, sk_U, St_U^i, m)$, and returns to $\mathcal{A}$ the response (if there is any) that Π_U^i would generate.
- Corrupt(U): This query allows the adversary to corrupt a party U. By making this query, the adversary learns the long-term secret key sk_U of user U.
- StateReveal(U, i): This query allows the adversary to learn the current state information St_U^i held by the instance Π_U^i.

- $\textsc{Reveal}(U, i)$: This query allows the adversary to learn the session key that has been generated by the instance Π_U^i. If the instance Π_U^i does not hold any session key, then a special symbol $\perp$ is returned to the adversary.
- $\textsc{Test}(U^*, i^*)$: This query can only be made to a *fresh* instance $\Pi_{U^*}^{i^*}$ (as defined below). If the instance $\Pi_{U^*}^{i^*}$ holds a session key $\text{ssk}_{U^*}^{i^*}$, then $\mathcal{SIM}$ does the following
  - if the coin $b = 1$, $\mathcal{SIM}$ returns $\text{ssk}_{U^*}^{i^*}$ to the adversary;
  - otherwise, a random session key is drawn from the session key space and returned to the adversary.

  Otherwise, a special symbol $\perp$ is returned to the adversary.

SK-security without PFS. We define session key security without perfect forward secrecy as follows.

We say an instance Π_U^i is *fresh* if

- $\mathcal{A}$ has never made a $\textsc{Corrupt}$ query to U or pid_U^i; and
- $\mathcal{A}$ has never made a $\textsc{Reveal}$ query to Π_U^i or its partner; and
- $\mathcal{A}$ has never made a $\textsc{StateReveal}$ query to Π_U^i or its partner.

At the end of the game, the adversary outputs a bit b' as her guess for b. The adversary's advantage in winning the game is defined as

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{AKE}}(k) = |2\Pr[b' = b] - 1|.$$

Definition 1. *We say an AKE protocol is SK-Secure without perfect forward secrecy in the auxiliary input model if the following conditions hold.*

1. *If two uncorrupted parties complete matching sessions then they both output the same key.*
2. *For any PPT adversary A, and any $\{h_U \in \mathcal{H}_{\mathsf{pkow}}\}_{U \in \mathcal{U}}$, $\mathbf{Adv}_{\mathcal{A}}^{\mathsf{AKE}}(k)$ is a negligible function of k.*

SK-security with PFS. In order to define perfect forward secrecy, we follow the approach of Canetti and Krawczyk [11] by introducing a new type of oracle query

- $\textsc{Expire}(U, i)$: Upon receiving this query, the simulator erases all the state information St_U^i and the session key ssk_U^i held by the instance Π_U^i.

The *freshness* of an instance Π_U^i is now redefined as follows:

- $\mathcal{A}$ makes a $\textsc{Corrupt}(U)$ query only after an $\textsc{Expire}(U, i)$ query; and
- $\mathcal{A}$ has never made a $\textsc{Reveal}$ query to Π_U^i; and
- $\mathcal{A}$ has never made a $\textsc{StateReveal}$ query to Π_U^i; and
- if Π_U^i has a partner instance Π_V^j, then $\mathcal{A}$ also obeys the above rules with respect to Π_V^j; otherwise, $\mathcal{A}$ has never made a $\textsc{Corrupt}(\mathsf{pid}_U^i)$ query.

Defined the adversary's advantage in winning the PFS game as

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{AKE\text{-}PFS}}(k) = |2\Pr[b' = b] - 1|.$$

Definition 2. *We say an AKE protocol is* SK-Secure *with perfect forward secrecy in the auxiliary input model if the following conditions hold.*

1. *If two uncorrupted parties complete matching sessions then they both output the same key.*
2. *For any PPT adversary A, and any $\{h_U \in \mathcal{H}_{\mathsf{pkow}}\}_{U\in\mathcal{U}}$, $\mathbf{Adv}_{\mathcal{A}}^{\mathsf{AKE\text{-}PFS}}(k)$ is negligible in k.*

4 SIG-DH vs PKE-DH

Several leakage resilient AKE protocols [3,15,25] have been proposed recently in the bounded leakage/retrieval model. In this section, we briefly review the Signature-based Diffie-Hellman protocol (eSIG-DH) [3] and the PKE-based Diffie-Hellman protocol (Enc-DH) [15], and give a comparison between them.

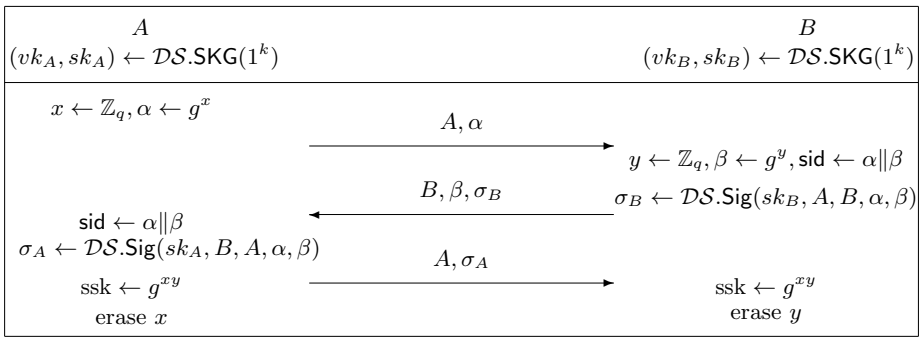

Fig. 1. The eSIG-DH Protocol [3]

The eSIG-DH protocol [3] is presented in Fig. 1. It is an extension of the SIG-DH protocol [11] in the bounded retrieval setting. The protocol makes use of a digital signature scheme $\mathcal{DS}$ that is entropically-unforgeable under chosen message attacks in the bounded retrieval model to achieve mutual authentication. In contrast, the Enc-DH protocol [15] (Fig. 2) is based on a leakage resilient PKE scheme supporting labels. The idea behind Enc-DH is that only the real user who has the decryption key can decrypt a ciphertext and answer the challenge.

Deniable Authentication. As pointed out by Dodis et al. in [15], due to the non-repudiation property of the digital signatures, it is obvious that the eSIG-DH protocol cannot provide the feature of deniable authentication. On the other hand, the Enc-DH protocol can achieve such a property, since the messages generated by user A in fact can be simulated by user B, and vice versa.

Security under StateReveal *Query.* There is actually another big difference between the eSIG-DH and Enc-DH protocols: if we consider the full CK model where the adversary is able to make StateReveal queries, then an adversary can launch the following attack against the Enc-DH protocol:

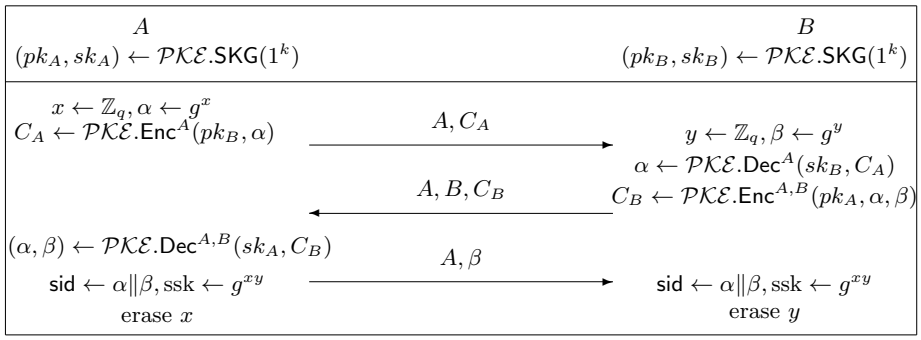

Fig. 2. The Enc-DH Protocol [15]

1. The adversary activates an instance of A to start a new AKE session with B, and faithfully delivers the first message (A, C_A) to B.
2. Upon receiving the response (A, B, C_B) from B, the adversary makes a StateReveal query to B and obtains α.
3. The adversary then generates $\beta' = g^{y'}$ and $C'_B \leftarrow \mathcal{PKE}.\mathsf{Enc}^{A,B}(pk_A, \alpha, \beta')$, and sends (A, B, C'_B) to user A.
4. User A would accept the session, send the third message (A, β'), and output the session key $\mathrm{ssk}_A = g^{xy'}$.

Since the adversary knows the value of y', she can derive the session key and win the game. It is worth noting that in the above attack, the instance of user A does not have a partner, but it is still fresh according to the definition. In other words, the adversary can successfully break the authentication mechanism employed under the Enc-DH protocol if she can make StateReveal queries. On the other hand, it is easy to check that such a problem does not exist in the eSIG-DH protocol.

5 A Leakage Resilient AKE Protocol Secure in the Auxiliary Input Model

In this section, we present a leakage resilient AKE protocol that is secure in the auxiliary input model. The scheme is based on a signature scheme that is *random message unforgeable under random message and auxiliary input attacks* (RU-RMAA) [18].

5.1 The aSIG-DH AKE Protocol

The only difference between our new protocol and the eSIG-DH protocol (Fig. 1) resides in the computation of the digital signatures. In the eSIG-DH protocol [3], an entropically-unforgeable signature scheme secure in the bounded-retrieval

model is used, while in the aSIG-DH protocol, each party will first compute a hash digest of the message, and then sign the hash digest using an RU-RMAA secure digital signature scheme [18]. Another important remark we should make is that same as the SIG-DH [11] and eSIG-DH [3] protocols, we assume that the signing operation is an atomic operation done by an independent module.

Why RU-RMAA Security is Sufficient. Readers may wonder that given an RU-RMAA secure digital signature scheme $\mathcal{DS} = (\mathsf{SKG}, \mathsf{Sig}, \mathsf{Ver})$, if we first hash the message M and then sign the hash digest $H(M)$ using $\mathcal{DS}.\mathsf{Sig}$, will we obtain an entropically-unforgeable signature scheme secure under chosen message attacks in the random oracle model (i.e. H is modelled as a random oracle)? Unfortunately, in general the answer is No! Consider that $\mathcal{DS}.\mathsf{Sig}$ is a randomized algorithm, then in a chosen message attack, when the adversary makes two signing queries with the same message m, two distinct yet valid signatures should be returned to the adversary. However, such signing queries cannot be answered by using the signing oracle defined in the RU-RMAA security game. Fortunately, in an AKE protocol, each participant will generate a fresh challenge in an AKE session. To impersonate a participant A without knowing A's signing key, the adversary needs to forge a valid signature on the fresh challenge sent by B. On the other hand, to answer the SEND queries made by the adversary to the participant A, the simulator can make use of the signing oracle in the RU-RMAA security game since the signed message will also contain a fresh challenge generated by A (i.e. a message to be signed by A would not appear in two different sessions).

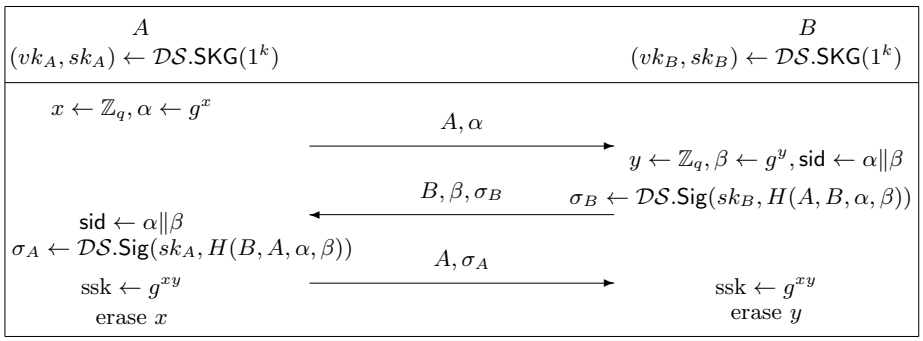

Fig. 3. The aSIG-DH Protocol

Theorem 1. *The aSIG-DH protocol is SK-secure with perfect forward secrecy in the auxiliary input model if the digital signature scheme $\mathcal{DS}$ is RU-RMAA secure w.r.t. $\mathcal{H}_{\mathsf{pkow}}$, the DDH assumption holds in group $\mathbb{G}$, and H is a random oracle.*

Proof. The first condition in the definition of SK-security is easy to see. Below we prove that the aSIG-DH protocol also satisfies the second condition. We define a sequence of games $G_i (i \geq 0)$ where G_0 is the original game defined in our security model with PFS. We also define $\mathbf{Adv}_i$ as the advantage of the adversary in game G_i (i.e. $\mathbf{Adv}_0 = \mathbf{Adv}_{\mathcal{A}}^{\mathsf{AKE\text{-}PFS}}(k)$).

Game G_1. Let forge denote the event that $\mathcal{A}$ successfully forges a valid signature of a user U (i.e. an instance Π_V^j receives a message/signature pair $((V, U, \alpha, \beta), \sigma_U)$ in a SEND query such that $\mathcal{DS}.\mathsf{Ver}(pk_U, (V, U, \alpha, \beta), \sigma_U) = 1$ and there is no instance of U which has sent a valid signature on (V, U, α, β) to the $\mathcal{A}$) before corrupting U. If a forge event happens, then the simulator aborts the game and outputs a random bit b'. Then we have

$$\Pr[b' = b \text{ in } G_0 | \neg \mathsf{forge}] = \Pr[b' = b \text{ in } G_1 | \neg \mathsf{forge}]$$

and

$$\Pr[b' = b \text{ in } G_0] - \Pr[b' = b \text{ in } G_1] \leq \Pr[\mathsf{forge}].$$

Therefore, we have

$$\mathbf{Adv}_0 \leq \mathbf{Adv}_1 + 2\Pr[\mathsf{forge}].$$

In the following, we show that the event forge happens only with a negligible probability.

CLAIM. The event forge happens only with a negligible probability if $\mathcal{DS}$ is RU-RMAA secure with respect to $\mathcal{H}_{\mathsf{pkow}}$ and H is a random oracle.

Proof. Suppose there exists an adversary $\mathcal{A}$ and a set of leakage functions $\mathbb{S} = \{h_1, h_2, \cdots, h_n\} \subset \mathcal{H}_{\mathsf{pkow}}$ w.r.t. the set of users $\mathcal{U} = \{U_1, U_2, \cdots, U_n\}$ such that a forge event would occur with a non-negligible probability, we show that there exists another algorithm $\mathcal{F}$ and a leakage function $h \in \mathcal{H}_{\mathsf{pkow}}$ such that $\mathcal{F}$ can win the RU-RMAA security game also with a non-negligible probability.

Let h^* denote a leakage function randomly selected from $\mathbb{S}$, and $\mathcal{F}$ is given a challenge $(vk^*, h^*(vk^*, sk^*), m^*)$ as input where $(vk^*, sk^*) \leftarrow \mathcal{DS}.\mathsf{SKG}(1^k)$ and m^* is randomly selected from the message space $\mathcal{M}$ of $\mathcal{DS}$. Wlog, assume h^* is the i-th function in the set $\mathbb{S}$ (i.e. $h^* = h_i$). $\mathcal{F}$ then sets the challenge public key vk^* as the public key of the user U_i. $\mathcal{F}$ then generates the long-term keys for all the remaining users in $\mathcal{U}$ by running $\mathcal{DS}.\mathsf{SKG}(1^k)$. In addition, $\mathcal{F}$ randomly selects $\zeta \leftarrow [1, q_H]$ where q_H denotes the number of hash queries $\mathcal{A}$ would make in the game. $\mathcal{F}$ then passes $\{vk_j, h_j(vk_j, sk_j)\} (1 \leq j \leq n)$ to $\mathcal{A}$ and answers $\mathcal{A}$'s oracle queries as follows.

$\mathcal{F}$ answers $\mathcal{A}$'s hash oracle queries as follows: when $\mathcal{A}$ submits a hash query, $\mathcal{F}$ first checks if the same input has been queried before. If yes, then the same output is returned to $\mathcal{A}$. Otherwise, $\mathcal{F}$ checks if the input has the format $(\cdot, U_i, \cdots)$. If not, a random element in $\mathcal{M}$ is selected and returned to $\mathcal{A}$; otherwise, $\mathcal{F}$ issues a signing query to its signing oracle to obtain (m, σ), and sets m as the hash value of $(\cdot, U_i, \cdots)$. When $\mathcal{A}$ makes the ζ-th hash query, $\mathcal{F}$ sets m^* as the hash value and returns m^* to $\mathcal{A}$.

When $\mathcal{A}$ makes a SEND query to an instance of U_i, if a signature of U_i on the hash value of $(U_j, U_i, \alpha, \beta)$ is required in order to answer this query, $\mathcal{F}$ first

checks if a signature σ corresponding to $H(U_j, U_i, \alpha, \beta)$ has been obtained from its signing oracle before. If yes, σ is returned to $\mathcal{A}$. Otherwise, $\mathcal{F}$ first makes a signing query to obtain (m, σ), then sets m as the hash value of $(U_j, U_i, \alpha, \beta)$ and uses σ as the corresponding signature to answer the SEND query. Since each instance of U_i will generate a fresh Diffie-Hellman component (either α or β), with overwhelming probability, $(U_j, U_i, \alpha, \beta)$ would never repeat in difference instances of U_i.

$\mathcal{F}$ simulates other operations performed by each instance honestly, and answers all the REVEAL and STATEREVEAL queries as usual. If $\mathcal{A}$ makes a CORRUPT (U_i) query during the game, $\mathcal{F}$ aborts the game and outputs nothing.

If $\mathcal{A}$ successfully forges a signature σ^* of U_i on the hash value m^*, then $\mathcal{F}$ outputs σ^* and halts. Otherwise, $\mathcal{F}$ outputs nothing and halts when $\mathcal{A}$ halts. Since U_i and ζ are randomly selected, it is clear that

$$\mathbf{Adv}^{\mathsf{RU\text{-}RMAA}}_{\mathcal{DS},\mathcal{H}_{\mathsf{pkow}},\mathcal{F}}(k) = \frac{1}{n \cdot q_H}\Pr[\mathsf{forge}].$$

Hence, we have

$$\mathbf{Adv}_0 \leq \mathbf{Adv}_1 + 2nq_H\mathbf{Adv}^{\mathsf{RU\text{-}RMAA}}_{\mathcal{DS},\mathcal{H}_{\mathsf{pkow}},\mathcal{F}}(k).$$

Game G_2. In game G_2, we change game G_1 as follows: the simulator randomly chooses an instance (say the i-th instance) $\Pi^{i^*}_{U^*}$ among all the instances created in the game, if the TEST query is not performed on $\Pi^{i^*}_{U^*}$, the simulator aborts and outputs a random bit b'. Let n_I denote the number of instances created in the game, then we have

$$\begin{aligned}\Pr[b' = b] &= \Pr[b' = b|\textsc{Test}(U^*, i^*)]\Pr[\textsc{Test}(U^*, i^*)] \\ &\quad +\Pr[b' = b|\neg\textsc{Test}(U^*, i^*)]\Pr[\neg\textsc{Test}(U^*, i^*)] \\ &= \Pr[b' = b \text{ in } G_1]\frac{1}{n_I} + \frac{1}{2}(1 - \frac{1}{n_I}) \\ &= \frac{1}{2} + \frac{1}{n_I}(\Pr[b' = b \text{ in } G_1] - \frac{1}{2})\end{aligned}$$

and

$$\mathbf{Adv}_1 = n_I\mathbf{Adv}_2.$$

Game G_3. In game G_3, we change game G_2 by replacing the Diffie-Hellman key $g^{x^*y^*}$ in the test session with a random element $g^r \in \mathbb{G}$. Below we show that if the adversary's advantage changes significantly in game G_3 , we can construct a distinguisher $\mathcal{B}$ to break the Decisional Diffie-Hellman (DDH) assumption.

$\mathcal{B}$ is given a challenge (g^a, g^b, Z), in which with equal probability, Z is either g^{ab} or a random element of $\mathbb{G}$. $\mathcal{B}$ simulates game G_2 honestly by generating all the long-term secret keys for all the users. When simulating the i-th instance $\Pi^{i^*}_{U^*}$ and its partner, $\mathcal{A}$ sets $g^{x^*} = g^a, g^{y^*} = g^b$ and Z as the corresponding session key. Finally, if $\mathcal{A}$ wins the game, $\mathcal{B}$ outputs 1, otherwise, $\mathcal{B}$ outputs 0.

Since a forge event would not happen on $\mathsf{pid}_{U^*}^{i^*}$ before $\mathsf{pid}_{U^*}^{i^*}$ is corrupted, we can guarantee that a partner instance of $\Pi_{U^*}^{i^*}$ must exist. So the Diffie-Hellman components in the test session must be g^a and g^b. If $Z = g^{ab}$, then $\mathcal{A}$ is in game G_2; otherwise, if Z is a random element of $\mathbb{G}$, then $\mathcal{A}$ is in game G_3. Therefore we have

$$\begin{aligned}\mathbf{Adv}_{\mathcal{B}}^{\mathsf{DDH}}(k) &= \Pr[\mathcal{B} \text{ outputs } 1 | Z = g^{ab}] - \Pr[\mathcal{B} \text{ outputs } 1 | Z = g^r] \\ &= \Pr[\mathcal{A} \text{ wins the game} | Z = g^{ab}] - \Pr[\mathcal{A} \text{ wins the game} | Z = g^r] \\ &= \frac{1}{2}(\mathbf{Adv}_2 - \mathbf{Adv}_3)\end{aligned}$$

and

$$\mathbf{Adv}_2 \leq \mathbf{Adv}_3 + 2\mathbf{Adv}_{\mathcal{B}}^{\mathsf{DDH}}(k).$$

It is clear that the adversary $\mathcal{A}$ has no advantage than random guess in game G_3 (i.e. $\mathbf{Adv}_3 = 0$). Hence, we have

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{AKE\text{-}PFS}}(k) \leq 2n_I\mathbf{Adv}_{\mathcal{B}}^{\mathsf{DDH}}(k) + 2nq_H\mathbf{Adv}_{\mathcal{DS},\mathcal{H}_{\mathsf{pkow}},\mathcal{F}}^{\mathsf{RU\text{-}RMAA}}(k).$$

□

6 Conclusion

In this paper, we initiated the study on leakage resilient authenticated key exchange in the auxiliary input model. We showed that in the random oracle model, we can build an AKE protocol secure under auxiliary input attacks based on a digital signature scheme that is random message unforgeable under random message and auxiliary input attacks (RU-RMAA). We also showed the differences between signature-based and public-key encryption-based Diffie-Hellman protocols and concluded that signatured-based protocols can offer a higher level of security than encryption-based ones when the adversary is allowed to learn the state information of a protocol participant. We leave the construction of a secure AKE against auxiliary input attacks without random oracles as our future work.

References

1. Aiello, W., Bellovin, S.M., Blaze, M., Canetti, R., Ioannidis, J., Keromytis, A.D., Reingold, O.: Just fast keying: Key agreement in a hostile Internet. ACM Trans. Inf. Syst. Secur. 7(2), 242–273 (2004)
2. Akavia, A., Goldwasser, S., Vaikuntanathan, V.: Simultaneous hardcore bits and cryptography against memory attacks. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 474–495. Springer, Heidelberg (2009)
3. Alwen, J., Dodis, Y., Wichs, D.: Leakage-resilient public-key cryptography in the bounded-retrieval model. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 36–54. Springer, Heidelberg (2009)

4. Bellare, M., Canetti, R., Krawczyk, H.: Modular approach to the design and analysis of key exchange protocols. In: ACM STOC 1998, pp. 419–428 (1998)
5. Bellare, M., Pointcheval, D., Rogaway, P.: Authenticated key exchange secure against dictionary attacks. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 139–155. Springer, Heidelberg (2000)
6. Bellare, M., Rogaway, P.: Entity authentication and key distribution. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 232–249. Springer, Heidelberg (1994)
7. Bellare, M., Rogaway, P.: Provably secure session key distribution — the three party case. In: ACM STOC 1995, pp. 57–66 (1995)
8. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: ACM CCS 1993, pp. 62–73 (1993)
9. Biham, E., Shamir, A.: Differential fault analysis of secret key cryptosystems. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 513–525. Springer, Heidelberg (1997)
10. Boyd, C., Cliff, Y., Gonzalez Nieto, J.M., Paterson, K.G.: Efficient one-round key exchange in the standard model. In: Mu, Y., Susilo, W., Seberry, J. (eds.) ACISP 2008. LNCS, vol. 5107, pp. 69–83. Springer, Heidelberg (2008), http://eprint.iacr.org/2008/007
11. Canetti, R., Krawczyk, H.: Analysis of key-exchange protocols and their use for building secure channels. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 453–474. Springer, Heidelberg (2001), http://eprint.iacr.org/2001/040/
12. Canetti, R., Krawczyk, H.: Security analysis of IKE's signature-based key-exchange protocol. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 143–161. Springer, Heidelberg (2002), http://eprint.iacr.org/2002/120/
13. Di Crescenzo, G., Lipton, R.J., Walfish, S.: Perfectly secure password protocols in the bounded retrieval model. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 225–244. Springer, Heidelberg (2006)
14. Dodis, Y., Goldwasser, S., Tauman Kalai, Y., Peikert, C., Vaikuntanathan, V.: Public-key encryption schemes with auxiliary inputs. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 361–381. Springer, Heidelberg (2010)
15. Dodis, Y., Haralambiev, K., López-Alt, A., Wichs, D.: Efficient public-key cryptography in the presence of key leakage. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 613–631. Springer, Heidelberg (2010)
16. Dodis, Y., Tauman Kalai, Y., Lovett, S.: On cryptography with auxiliary input. In: ACM STOC 2009, pp. 621–630 (2009)
17. Dziembowski, S.: Intrusion-resilience via the bounded-storage model. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 207–224. Springer, Heidelberg (2006)
18. Faust, S., Hazay, C., Nielsen, J.B., Nordholt, P.S., Zottarel, A.: Signature schemes secure against hard-to-invert leakage. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 98–115. Springer, Heidelberg (2012)
19. Gandolfi, K., Mourtel, C., Olivier, F.: Electromagnetic analysis: Concrete results. In: Koç, Ç.K., Naccache, D., Paar, C. (eds.) CHES 2001. LNCS, vol. 2162, pp. 251–261. Springer, Heidelberg (2001)
20. Entity authentication mechanisms - Part 3: Entity authentication using asymmetric techniques. ISO/IEC IS 9798-3 (1993)
21. Kocher, P.C.: Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996)

22. Kocher, P.C., Jaffe, J., Jun, B.: Differential power analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999)
23. Krawczyk, H.: SIGMA: The 'SIGn-and-MAc' Approach to Authenticated Diffie-Hellman and Its Use in the IKE Protocols. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 400–425. Springer, Heidelberg (2003)
24. LaMacchia, B.A., Lauter, K., Mityagin, A.: Stronger security of authenticated key exchange. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) ProvSec 2007. LNCS, vol. 4784, pp. 1–16. Springer, Heidelberg (2007)
25. Moriyama, D., Okamoto, T.: Leakage resilient eCK-secure key exchange protocol without random oracles. In: ACM ASIACCS 2011, pp. 441–447 (2011)
26. Naor, M., Segev, G.: Public-key cryptosystems resilient to key leakage. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 18–35. Springer, Heidelberg (2009)
27. Okamoto, T.: Authenticated key exchange and key encapsulation in the standard model. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 474–484. Springer, Heidelberg (2007), Full paper available at `http://eprint.iacr.org/2007/473`
28. Yang, G., Duan, S., Wong, D.S., Tan, C.H., Wang, H.: Authenticated key exchange under bad randomness. In: Danezis, G. (ed.) FC 2011. LNCS, vol. 7035, pp. 113–126. Springer, Heidelberg (2012)
29. Yang, G., Wong, D.S., Wang, H., Deng, X.: Two-factor mutual authentication based on smart cards and passwords. J. Comput. Syst. Sci. 74(7), 1160–1172 (2008)