Strongly Secure One Round Authenticated Key Exchange Protocol with Perfect Forward Security

Hai Huang

Zhejiang Sci-Tech University, People's Republic of China
haihuang1005@gamil.com

Abstract. This paper investigates the two-pass authenticated key exchange protocol in the enhanced Canetti-Krawczyk (eCK) with perfect forward security. Currently, there exist no authenticated key exchange protocols which are provably secure in eCK model and meanwhile achieve perfect forward security against active adversary in one round.
We propose a new two-pass authenticated key exchange protocol which enjoys following desirable properties. **First**, our protocol is shown secure in the eCK model under the gap Diffie-Hellman (GDH) assumption. Moreover, our protocol does not use the NAXOS transformation, the drawback of which will be discussed in the introduction. **Second**, under the same assumption, we prove that our protocol achieves perfect forward security against active adversary in one round.
To the best of our knowledge, our proposal is first two-pass (one round) AKE protocol provably secure in the eCK model and achieving perfect forward security against active adversary.

Keywords: Authenticated key exchange, eCK model, Perfect forward security, Provably secure

1 Introduction

Key exchange (KE) protocol enables two parties, Alice (A) and Bob (B), to establish a shared session key over an insecure channel. Later, the session key can be used to ensure data confidentiality and integrity between A and B using efficient symmetric encryptions and message authentication codes.

Since the classic Diffie-Hellman (DH) key exchange protocol is only secure against a passive adversary, much of work has been dedicated to armor the DH protocol against active, man-in-the-middle attacks. This is the goal of authenticated key exchange (AKE) in which both parties are assured that no other parties aside from their intended peers may learn the established session key.

The authenticated key exchange protocols have been established to be surprisingly difficult to design. The traditional trial-and-error design method has led to the situation that the protocols have been broken or the flaws in the protocols have taken many years to discover. In last years, attentions have been focused on the development of rigorous security models for authenticated key exchange.

Recently, LaMacchia, Lauter and Mityagin [9, 10] presented a new security model for authenticated key exchange protocols, the enhanced Canetti-Krawczyk (eCK) model in which the adversary's ability is extended to the extent such that it is allowed to reveal any static private key and ephemeral private key of parties involved except for both static private key and ephemeral private key of one of parties involved. To achieve eCK security, they introduce so called NAXOS transformation which requires that the ephemeral public key X is computed as $X = g^{H(x,a)}$ instead of $X = g^x$, where x, a are ephemeral private key and static private key respectively. However, it seems that NAXOS transformation does not prevent the leakage of the ephemeral DH exponents. In some scenarios, we do not guarantee that leakages on DH exponents cannot occur [14]. On the other hand, constructing the authenticated key exchange protocol secure in eCK model without NAXOS transformation has its advantages. For example, it can reduce the risk of leakage of the static private key and use of the random oracle [7].

An important property not captured by the two-pass AKE protocols secure in eCK model is perfect forward security (PFS) against active adversary. Recall that PFS guarantees that the leakages on the static private keys of both parties involved do not compromise the previously established session keys by these parties. However, as observed in [8], no two-pass AKE protocols with basic DH message can achieve PFS, if the adversary is *actively* involved with the choice of the DH values X, Y at a session. So the best the two-pass AKE protocols with DH message can achieve is the weak form of perfect forward security (wPFS), which guarantees security against the passive adversary.

Based on Okamoto-Tanaka's work [4], Gennaro, Krawczyk and Rabin propose a two-pass AKE protocol called mOT [5]. While preserving the communication complexity of a basic DH (two messages with a single group element per message), they prove that mOT protocol achieves PFS security against active adversary under a non-standard knowledge of exponent assumption (KEA1) [1]. However, mOT protocol does not resist the ephemeral key query attack, i.e, mOT protocol is insecure in the eCK model. In fact, the design of two-pass AKE protocol with PFS secure against the ephemeral key query attack is one of the open problems in [5].

1.1 Our Contributions

In this paper we investigates two-pass authenticated key exchange protocol with perfect forward security. While there have already been some two-pass AKE protocols [9, 15, 6, 11, 7] provably secure in the eCK model, *none* of them achieve perfect forward security against active adversary. Although it is possible to transform a two-pass AKE protocol provably secure in eCK model into a three-pass AKE protocol with perfect forward security against active adversary by adding two messages [2, 8], the resulting protocol have a higher round-complexity.

This paper proposes a new two-pass (one round) authenticated key exchange protocol in the eCK model with PFS property. The key ingredient of our protocol is that instead of a single group element per message, we use two group elements

in each message. With this relaxation, our protocol enjoys following desirable properties. **First**, without the NAXOS transformation our protocol is shown secure in the eCK model under the gap Diffie-Hellman (GDH) assumption. There are very few AKE protocols provably secure in the eCK model which do not use NAXOS transformation. **Second**, under the same assumption, we prove that our two-pass (one round) protocol achieves perfect forward security against active adversary. The proof of mOT protocol for PFS active adversary needs the non-standard KEA1 assumption and is comparatively more complicated. While the mOT protocol achieves optimal communication complexity as the basic Diffie-Hellman, i.e, a single group element per message, from a practical point of view our protocol with two group element per message does not increase communication overhead too much. Comparatively, the merit of our protocol is that the security does not rely on the KEA1 assumption and the proof is straightforward and hence simpler.

To the best of our knowledge, our proposal is first two-pass (one round) AKE protocol which is provably secure in the eCK model and achieves perfect forward security against active adversary.

1.2 Organization

The paper is organized as follows. Section 2 reviews the related building techniques. Section 3 introduces a new two-pass AKE protocol with perfect forward security. Section 4 gives the full security proof of our protocol in the eCK model. Section 5 is dedicated to the proof of the PFS security of our protocol. Section 6 compares our protocol with several popular AKE protocols in term of efficiency, security model and underlying hardness assumptions. Finally, concluding remarks are made in section 7.

2 Preliminaries

In this section, we present several established tools needed in this paper.

2.1 Computational Diffie-Helleman (CDH) Assumption

Let the value κ be the security parameter. Let $\mathbb{G} = \langle g \rangle$ be a cyclic group of prime order q and $g \in \mathbb{G}$ be the generator. Define $\text{CDH}(U, V) := U^v$ where $U = g^u, V = g^v$. For any probabilistic polynomial time (PPT) algorithm A,

$$Pr[A(\mathbb{G}, g, U = g^u, V = g^v) = \text{CDH}(U, V)] \leq \epsilon(\kappa).$$

where $u, v \in \mathbb{Z}_q$ and $\epsilon(\kappa)$ is negligible. The probability is taken over the coin tosses of A, the choice of g and the random choices of u, v in $\mathbb{Z}_q$.

2.2 Gap Diffie-Helleman (GDH) assumption [13]

Let $\mathbb{G} = \langle g \rangle$ be the cyclic group of order q, and DDH(.) be a decisional Diffie-Helleman (DDH) oracle for $\mathbb{G}$. Then, for any probabilistic polynomial time algorithm A,

$$Pr[A^{\mathrm{DDH}(.)}(\mathbb{G}, g, U = g^u, V = g^v) = \mathrm{CDH}(U, V)] \leq \epsilon(k)$$

where $u, v \in \mathbb{Z}_q$, and where $\epsilon(k)$ is negligible. The DDH(.) denotes that A has oracle access to DDH, which given a quadruple $(g, U = g^u, V = g^v, W = g^w)$ of elements in $\mathbb{G}$, outputs 1 if $w = uv \ mod \ q$ and 0 otherwise. The probability is taken over the coin tosses of A, the choice of g and the random choices of u, v in $\mathbb{Z}_q$.

3 Strongly Secure One Round Authenticated Key Exchange Protocol with Perfect Forward Security

In this section, we propose a new one round AKE protocol with perfect forward security.

3.1 Protocol Setup.

Let the value κ be the security parameter. Let $\mathbb{G} = \langle g \rangle$ be a cyclic group of order q in which decisional Diffe-Helleman (DDH) problem can be efficiently solved. Let $g \in \mathbb{G}$ be a generator and $\mathbb{G}^*$ be the non-identity elements set of $\mathbb{G}$. Let $h : \{0,1\}^* \rightarrow \mathbb{G}^*, H : \{0,1\}^* \rightarrow \{0,1\}^\kappa$ be two hash functions. The party Alice($\hat{A}$)'s static private key is a and its static public key is $A = g^a$. Similarly, the party Bob($\hat{B}$)'s static private key is b and its static public key is $B = g^b$.

3.2 Protocol Description.

The protocol runs between Alice and Bob. Its description is given in Figure 1.

1. Alice($\hat{A}$) chooses an ephemeral private key $x \in \mathbb{Z}_q$ at random, computes the ephemeral public key $X = g^x$ and sends $X, c_1 = h(X)^a$ to $\hat{B}$.
2. Bob($\hat{B}$) chooses an ephemeral private key $y \in \mathbb{Z}_q$ at random, computes the ephemeral public key $Y = g^y$ and sends $Y = g^y, c_2 = h(Y)^b$ to $\hat{A}$.
3. Upon receiving X, c_1, $\hat{B}$ verifies $X \in \mathbb{G}^*$ and checks if $(g, h(X), A, c_1)$ is a valid Diffie-Hellman tuple. If so, $\hat{B}$ computes $sk = H((XA)^{y+b}, sid)$, where $sid = (\hat{A}, \hat{B}, X, c_1, Y, c_2)$. Then, $\hat{B}$ keeps sk as the established session key..
4. Upon receiving Y, c_2, $\hat{A}$ verifies $Y \in \mathbb{G}^*$ and checks if $(g, h(Y), B, c_2)$ is a valid Diffie-Hellman tuple. If so, $\hat{A}$ computes $sk = H((YB)^{x+a}, sid)$, where $sid = (\hat{A}, \hat{B}, X, c_1, Y, c_2)$. Then, $\hat{A}$ keeps sk as the established session key..

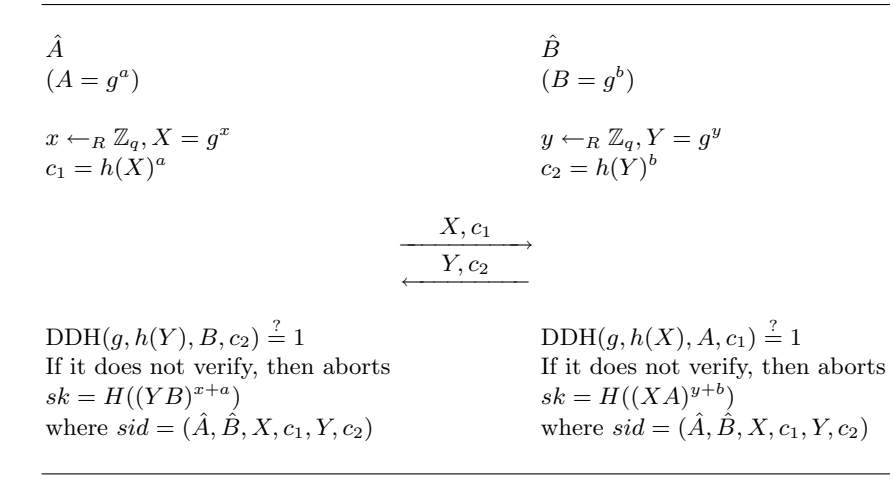

Fig. 1. Strongly Secure One Round Authenticated Key Exchange Protocol with Perfect Forward Security

4 Security Proof

Theorem 1. *Suppose that the GDH assumption for group $\mathbb{G}$ holds, h, H are hash functions modeled as random oracles, then the proposed scheme in Fig. 1 is a secure authenticated key exchange protocol in the eCK model.*

Proof. Assume that the adversary succeeds with non-negligible probability in the environment described in Appendix A. Following the standard approach, we use it to build an algorithm to solve GDH problem. The proof starts with the fact: Since the input to the key derivation function $H(\cdot)$ includes all exchanged information contained in *sid* and H is modeled as random oracle, we know that two different sessions necessarily have two different session keys, and the only way for the adversary to succeed is by computing the value $\mathrm{GDH}(XA, YB)$, which is called forging attack.

The rest of this section is mainly devoted to the analysis of the forging attack. According to freshness definition, we consider separately two complementary subcases below:

CASE 1: No honest party owns a matching session to the Test session.

CASE 2: The Test session has a matching session owned by another honest party.

4.1 The Analysis of CASE 1

In this case, it suffices to discuss the following two subcases:

CASE 1.1: The adversary issues a StaticKeyReveal query on party $\hat{A}$ and EphemeralKeyReveal query on party $\hat{B}$ communicating with party $\hat{A}$ (neither

EphemeralKeyReveal query on the Test session nor StaticKeyReveal query on party $\hat{B}$ is allowed).

CASE 1.2: The adversary issues a EphemeralKeyReveal query on the Test session and EphemeralKeyReveal query on party $\hat{B}$ communicating with party $\hat{A}$ (neither StaticKeyReveal query on party $\hat{A}$ nor StaticKeyReveal query on party $\hat{B}$ is allowed).

CASE 1.1: To show that the success probability of the adversary is negligible, we will construct a GDH problem solver SIM that uses an adversary M who succeeds with non-negligible probability in the attack.

Input to SIM. The input to the SIM is a GDH problem instance $(U = g^u, V = g^v)$, where $u, v \in \mathbb{Z}_q$ and $U, V \in \mathbb{G}$. The goal of SIM is to compute $\mathrm{GDH}(U, V) = g^{uv}$.

Guessed Test session. SIM guesses the adversary M will select one party denoted by $\hat{A}$ as the owner of the Test session and the other party denoted by $\hat{B}$ as the peer. Further, SIM guesses the adversary M will select the session $\Pi^s_{\hat{A},\hat{B}}$ as the Test session. Note that the probability that the Test session is chosen by M is non-negligible. If this is not the case, SIM aborts.

Setup of SIM. SIM assigns static public key V for $\hat{B}$, and random static public/private key pairs for the remaining parties (including $\hat{A}$). This way, SIM knows all static private keys of parties except for $\hat{B}$.

Simulating the non-Test sessions. The adversary M can activate sessions between any two parties and insert its own messages into these sessions by either generating or scheduling the messages. The simulator SIM needs to respond the sessions on behalf of honest parties. Simulating the actions of any honest party other than $\hat{B}$ is simple as SIM knows their static private keys. Assume that $\hat{B}$ is a responder and $\hat{C}$ is the peer, and the messages it receives is of the form $\tilde{X}, \tilde{c_1}$ allegedly from $\hat{C}$. Whenever $\hat{B}$ is activated in a session, SIM first verifies that $\tilde{X} \in \mathbb{G}^*$ and calls its DDH oracle to check if $\mathrm{DDH}(g, h(\tilde{X}), C, \tilde{c_1}) \stackrel{?}{=} 1$. If so, SIM chooses an ephemeral private key $\tilde{y} \in \mathbb{Z}_q$ at random, computes ephemeral public key $\tilde{Y} = g^{\tilde{y}}$, and sets $h(\tilde{Y})$ to be $g^{\tilde{r}}$, where $\tilde{r} \in \mathbb{Z}_q$. Then SIM sets the values $\tilde{Y}, \tilde{c_2} = V^{\tilde{r}}$ as the outgoing messages.

Response to the static private key and session key queries (non-Test session). SIM can respond the static private key queries on any party except for $\hat{B}$. Likewise, session key queries for these sessions owned by any party other than $\hat{B}$ can be easily responded by SIM as it knows the corresponding static private keys and generates the ephemeral private keys itself. However, sessions in which $\hat{B}$ is a participant are problematic since SIM does not know $\hat{B}$'s static private key.

Again, assume that $\hat{B}$ is a responder and the peer is $\hat{C}$. Since SIM does not knows $\hat{B}$'s static private key , it can not generate the session key itself. To respond the session key queries and keep the consistency of the random oracles H, SIM calls DDH oracles to check if $\mathrm{DDH}(g, \tilde{X}C, \tilde{Y}V, \sigma) \stackrel{?}{=} 1$ where σ is the first element int H.

Response to the ephemeral private key queries. SIM can respond the ephemeral private key queries on any party including $\hat{B}$ as in the simulation SIM chooses the values for all the parties itself.
Simulating the Test session. When the adversary activates the Test session at $\hat{A}$, SIM acts as follows. Without loss of generality, assume that $\hat{A}$ is an initiator. SIM computes $c_1 = h(U)^a$ and sets the outgoing message to be U, c_1. Upon receiving the message Y, c_2 allegedly from $\hat{B}$, SIM first verifies that $Y \in \mathbb{G}^*$ and calls its DDH oracle to check if $\mathrm{DDH}(g, h(Y), B, c_2) \stackrel{?}{=} 1$. If so, SIM waits for the adversary's next query, else it aborts.
Computing the forgery GDH$(U, V) = g^{uv}$. The goal of SIM is to compute $\mathrm{GDH}(U, V) = g^{uv}$. Below we show that whenever the adversary M succeeds in the forging attack SIM can compute $\mathrm{GDH}(U, V) = g^{uv}$. Assume that the outgoing message of the Test session is $X = U, c_1$ and the incoming message is Y, c_2 allegedly from $\hat{B}$. Indeed, to succeed in the forging attack it must be that the adversary M queries the first element of the form $(YB)^{x+a} = (YV)^{u+a}$ in H. In order to compute U^v, the value Y must be eliminated (SIM knows the value a). However, without knowing y, this elimination seems difficult. Fortunately, it can be shown that the message Y cannot be generated by the adversary itself except with negligible probability. In other words, if there is an adversary who correctly generates a message Y, c_2 itself with non-negligible probability, we can construct a GDH problem solver $\overline{SIM}$ that uses the adversary. The action of $\overline{SIM}$ is as follows: With the input U, V, setting the static private key of party $\hat{B}$ to be U, $\overline{SIM}$ responds the adversary's queries in the same way as SIM. Finally, if the adversary generates a message Y, c_2 itself, then $\overline{SIM}$ call its DDH oracle to checks if $\mathrm{DDH}(g, h(Y), B, c_2) \stackrel{?}{=} 1$, where $h(Y) = V$. If so, $\overline{SIM}$ outputs c_2 which equals $\mathrm{GDH}(U, V) = g^{uv}$.

Now we learn that Y must have been generated by SIM on behalf of party $\hat{B}$. Then Y can be easily eliminated from $(YB)^{x+a} = (YV)^{u+a}$ as SIM knows y. Denote the first element in H by σ. SIM proceeds as follows.

$$(\sigma/(YV)^a)/U^y = ((YV)^{u+a}/(YV)^a)/U^y = (YV)^u/U^y = U^v$$

This contradicts the GDH assumption.

CASE 1.2:

In this case, since the adversary can issue neither StaticKeyReveal query on party $\hat{A}$ nor staticKeyReveal query on party $\hat{B}$, SIM sets the static public keys of party $\hat{A}$ and $\hat{B}$ to be U and V respectively. Simulating the actions of any honest parties other than $\hat{A}$ and $\hat{B}$ is simple as SIM knows their static private keys. Whenever $\hat{B}$ (or $\hat{A}$) is activated in a session, SIM acts like that of CASE 1.1 dealing with the queries on party $\hat{B}$.
Computing the forgery GDH$(U, V) = g^{uv}$. If the adversary succeeds in the forging attack, i.e., the adversary M queries the first element of the form $(YB)^{x+a} = (YV)^{x+u}$ in H. Note that the value X, Y is generated by the SIM itself as shown by $\overline{SIM}$ in CASE 1.1. Knowing x, y, the value $\mathrm{GDH}(U, V)$ can be easily determined as follows (denote the first element in H by σ).

$$(\sigma/(YB)^x)/U^y = ((YV)^{x+u}/(YV)^x)/U^y = (YV)^u/U^y = U^v$$

This contradicts the GDH assumption.

4.2 The Analysis of CASE 2

Compared to that of CASE 1 the proof for this case is simpler as there is a session matching to the Test session (i.e., the adversary neither generates the message itself nor delivers the message from other sessions towards the Test session). The simulations of party $\hat{A}$ and $\hat{B}$ are similar to that of CASE 1. Due to space limitations, the details are left to the readers.

5 Further Security Properties

5.1 Resistance to reflection attacks.

In the security proof of section 4 we assume that party $\hat{A}$ and $\hat{B}$ are different. In some scenarios, however, party $\hat{A}$ wants to establish a session key with itself. For example, Alice with mobile device wants to establishes a secure channel with her office desktop computer where two devices uses the same certificate. An attack that exploits the fact the two parties use the same identity is called *reflection attack* in which the adversary simply copies party $\hat{A}$'s outgoing message and sends back to $\hat{A}$. We now prove that our protocol is secure against such attacks as follows.

Input to SIM. The input to the SIM is a GDH problem instance $(U = g^u, V = g^v)$, where $u, v \in \mathbb{Z}_q$ and $U, V \in \mathbb{G}$. The goal of SIM is to compute $\mathrm{GDH}(U, V) = g^{uv}$.

SIM sets the static public key of party $\hat{A}$ to be U. The simulation of party $\hat{A}$ (initiator or responder) is similar to that of CASE 1.2 where SIM knows neither of the static private keys of two parties. Further, we assume that the outgoing message of the Test session is of the form X, c_1 and incoming message is of the form $\tilde{Y}, \tilde{c_2}$. [1] As shown by $\overline{SIM}$ in CASE 1.1, however, the message $\tilde{Y}, \tilde{c_2}$ can not be generated by the adversary itself except with negligible probability. In other words, it must be that SIM generates the value $\tilde{Y}, \tilde{c_2}$. By applying the similar argument in CASE 1.2 where SIM knows neither of the static private keys of $\hat{A}$ and $\hat{B}$, knowing the values $x, \tilde{y}$ we can transform an adversary into an algorithm which given U computes U^u. Further, such algorithm can be used to solve the general GDH problem as observed by Maurer and Wolf [12] as said in [8].

[1] If the adversary simply copies X, c_1 and send back to $\hat{A}$, i.e., $X = \tilde{Y}$ and $c_1 = \tilde{c_2}$, this kind of attack is called reflection attack.

5.2 Proof of PFS Property

In the section, under the same GDH assumption, we prove that our protocol enjoys perfect forward security (PFS) against the active adversary. Our proof does not use any additional assumption, e.g. KEA1 assumption and is thus comparatively straightforward. To show that the success probability of the adversary M is negligible, we will construct a GDH problem solver SIM that uses an adversary M who succeeds with non-negligible probability in the attack.

Input to SIM. The input to the SIM is a GDH problem instance $(U = g^u, V = g^v)$, where $u, v \in \mathbb{Z}_q$ and $U, V \in \mathbb{G}$. The goal of SIM is to compute $\mathrm{GDH}(U, V) = g^{uv}$.

Guessed Test session. SIM guesses the adversary M will select one party denoted by $\hat{A}$ as the owner of the Test session and the another party denoted by $\hat{B}$ as the peer. Further, SIM guesses the adversary M will select the session $\Pi^s_{\hat{A},\hat{B}}$ as Test session. Note that the probability that the Test session is chosen by M is non-negligible. If this is not the case, SIM aborts.

Setup of SIM. According to the definition of PFS game, the adversary M can issue StaticKeyReveal query on neither party $\hat{A}$ nor party $\hat{B}$ before the Test session is complete. However, M is allowed to reveal the static private keys of party $\hat{A}$ and $\hat{B}$ after the Test session is complete. To deal with StaticKeyReveal query, SIM assigns random static public/private key pairs for *all* the parties (including $\hat{A}$ and $\hat{B}$) itself. This way, SIM knows all the static private keys of parties.

Simulating the non-Test sessions. Simulating the actions of any honest party other than $\hat{B}$ is simple as SIM knows their static private keys. On the other hand, to solve the GDH problem, the GDH instance U, V must be embedded into the outgoing and incoming messages of Test session. As the adversary is an active attacker in the PFS game, i.e, the incoming message of the Test session may be generated or scheduled from other sessions of party $\hat{B}$, the simulation of party $\hat{B}$ is slightly different. Assume that $\hat{B}$ is a responder and $\hat{C}$ is the peer, and the messages $\hat{B}$ receives is of the form $\tilde{X}, \tilde{c_1}$ allegedly from $\hat{C}$. Whenever $\hat{B}$ is activated in a session, SIM first verifies that $\tilde{X} \in \mathbb{G}^*$ and calls its DDH oracle to check if $\mathrm{DDH}(g, h(\tilde{X}), C, \tilde{c_1}) \stackrel{?}{=} 1$. If so, SIM chooses $\tilde{t_i} \in \mathbb{Z}_q$ at random, computes ephemeral public key $\tilde{Y} = V^{\tilde{t_i}}$, and sets the values $\tilde{Y}, \tilde{c_2} = h(\tilde{Y})^b$ as the outgoing messages [2].

Response to the static private key and session key queries (non-Test session). SIM can respond these queries since it knows the static private keys of all the parties.

Response to the ephemeral private key queries. Since the definition of PFS stipulates that the adversary is not allowed to make any EphemeralKeyReveal query, if these happen, SIM aborts.

Simulating the Test session. When the adversary activates the Test session at $\hat{A}$, SIM acts as follows. Without loss of generality, assume that $\hat{A}$ is an initiator. SIM computes $c_1 = h(U)^a$ and sets the outgoing message to be U, c_1. Upon

[2] The value V must be embedded into each session of party $\hat{B}$.

receiving the message Y, c_2 allegedly from $\hat{B}$, SIM first verifies that $Y \in \mathbb{G}^*$ and calls its DDH oracle to check if $\text{DDH}(g, h(Y), B, c_2) \stackrel{?}{=} 1$. If so, SIM waits for the adversary's next query, else it aborts.

Computing the forgery GDH$(U, V) = g^{uv}$. The goal of SIM is to compute $\text{GDH}(U, V) = g^{uv}$. Below we show that whenever the adversary M succeeds in the forging attack SIM can compute $\text{GDH}(U, V) = g^{uv}$. Assume that the outgoing message of the Test session is U, c_1 and the incoming message is Y, c_2 allegedly from $\hat{B}$. As shown by $\overline{SIM}$ in CASE 1.1, the message Y, c_2 can not be generated by the adversary itself except with negligible probability. Thus, it must be that the message Y, c_2 have been scheduled by the adversary from other sessions of party $\hat{B}$. That is to say, the value Y, c_2 has been generated by SIM itself with the form $Y = V^{t_i}$ and $c_2 = h(Y)^b$. Denote the first element in H by σ. With value t_i, SIM proceeds as follows.

$$\overline{\sigma} = (\sigma/(YB)^a)/U^b = ((YB)^{u+a}/(YB)^a)/U^b = (YB)^u/U^b = Y^u$$

Then,

$$(\overline{\sigma})^{{t_i}^{-1}} = (Y^u)^{{t_i}^{-1}} = (V^{t_i u})^{{t_i}^{-1}} = U^v$$

This contradicts the GDH assumption.

6 Comparison of Protocols

In Table 1 we compare our protocol with several popular AKE protocols in term of efficiency, security model and underlying hardness assumptions. For simplicity, we do not take into account subgroup validation and speedup trick that may be applicable. The "E" denote the exponentiation in $\mathbb{G}$ and "E_N" denote the exponentiation in the RSA group. The CK_{HMQV} denotes modified Canetti-Krawczyk security [3] which captures CK model, KCI, wPFS and ephemeral key query. $\text{CK}_{\text{HMQV-C}}$ captures CK_{HMQV} model, PFS. The KEA1 stands for Knowledge of Exponent Assumption [1]. RSA and GDH stand respectively for RSA and gap Diffie-Hellman assumptions.

Protocol	*Efficiency*		*Security*	
	Computation	Round	Model	Assumption
NAXOS [9, 10]	4E	1	eCK	GDH
CMQV [15]	3E	1	eCK	GDH
HMQV [8]	3E	1	CK_{HMQV}	GDH,KEA1
HMQV-C [8]	3E	3	$\text{CK}_{\text{HMQV-C}}$	GDH
mOT[5]	2E_N	1	CK,PFS	RSA,KEA1
Our scheme	3E+1DDH	1	eCK,PFS	GDH

Table 1. Comparison of Protocols

Compared with the NAXOS, CMQV and HMQV protocols, all of which only achieve weak perfect forward security (wPFS), the main advantage of our scheme is that it achieves perfect forward security (PFS). On the other hand, to be secure in the eCK model the former two protocols use the NAXOS transformation while our scheme does not. Compared with HMQV-C protocol which achieves perfect forward security (PFS), our scheme has lower round complexity (within one round). While mOT protocol achieves perfect forward security (PFS) within one round, it does not resist the ephemeral key query, i.e, mOT is insecure in eCK model. Compared to it, our scheme is provably secure in eCK model and meanwhile achieves perfect forward security (PFS).

7 Conclusions and Open Problem

Although there have already been some two-pass AKE protocols provably secure in the eCK model, *none* of them achieve perfect forward security against active adversary. On the other hand, while mOT protocol achieves PFS security against active adversary within one round, it is not secure against the ephemeral key query attack, i.e, insecure in the eCK model.

This paper proposes a new two-pass (one round) authenticated key exchange protocol in eCK model with PFS property. Our protocol provably enjoys following desirable properties. First , without the NAXOS transformation our protocol is shown secure in the eCK model under the gap Diffie-Hellman (GDH) assumption. Second, under the same assumption, we prove that our two-pass (one round) protocol achieves perfect forward security against active adversary.

To the best of our knowledge, our proposal is the first two-pass (one round) AKE protocol which is provably secure in the eCK model and achieves perfect forward security against active adversary. Finally, while our work takes provably secure AKE protocol further, our protocol needs two group element per message. The design of the AKE protocol provably secure in the eCK model without NAXOS transformation and achieving PFS in one round (a single group element per message) remains an open problem.

References

1. M. Bellare and A. Palacio. The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols. In M. K. Franklin, editor, *CRYPTO*, volume 3152 of *Lecture Notes in Computer Science*, pages 273–289. Springer, 2004.
2. M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated key exchange secure against dictionary attacks. In *EUROCRYPT*, pages 139–155, 2000.
3. R. Canetti and H. Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In B. Pfitzmann, editor, *EUROCRYPT*, volume 2045 of *Lecture Notes in Computer Science*, pages 453–474. Springer, 2001.
4. E.Okamoto and K.Tanaka. Key distribution systems based on identification information. In *IEEE Journal on Selected Areas in Communications*, volume 7(4), pages 481–485, 1989.

5. R. Gennaro, H. Krawczyk, and T. Rabin. Okamoto-tanaka revisited: Fully authenticated diffie-hellman with minimal overhead. In J. Zhou and M. Yung, editors, *ACNS*, volume 6123 of *Lecture Notes in Computer Science*, pages 309–328, 2010.
6. H. Huang and Z. Cao. Strongly secure authenticated key exchange protocol based on computational diffie-hellman problem. In *Cryptology ePrint Archive, Report 2008/500*, 2008.
7. M. Kim, A. Fujioka, and B. Ustaoglu. Strongly secure authenticated key exchange without naxos' approach. In T. Takagi and M. Mambo, editors, *IWSEC*, volume 5824 of *Lecture Notes in Computer Science*, pages 174–191. Springer, 2009.
8. H. Krawczyk. HMQV: A high-performance secure Diffie-Hellman protocol. In V. Shoup, editor, *CRYPTO*, volume 3621 of *Lecture Notes in Computer Science*, pages 546–566. Springer, 2005.
9. B. LaMacchia, K. Lauter, and A. Mityagin. Stronger security of authenticated key exchange. Cryptology ePrint Archive, Report 2006/073, 2006. http://eprint.iacr.org.
10. B. A. LaMacchia, K. Lauter, and A. Mityagin. Stronger security of authenticated key exchange. In W. Susilo, J. K. Liu, and Y. Mu, editors, *ProvSec*, volume 4784 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2007.
11. J. Lee and C. S. Park. An efficient authenticated key exchange protocol with a tight security reduction. In *Cryptology ePrint Archive, Report 2008/345, 2008.*, 2008.
12. U. M. Maurer and S. Wolf. Diffie-hellman oracles. volume 1109 of *Lecture Notes in Computer Science*, pages 268–282. Springer, 1996.
13. T. Okamoto and D. Pointcheval. The gap-problems: A new class of problems for the security of cryptographic schemes. In K. Kim, editor, *Public Key Cryptography*, volume 1992 of *Lecture Notes in Computer Science*, pages 104–118. Springer, 2001.
14. A. P. Sarr, P. Elbaz-Vincent, and J.-C. Bajard. A new security model for authenticated key agreement. In J. A. Garay and R. D. Prisco, editors, *SCN*, volume 6280 of *Lecture Notes in Computer Science*, pages 219–234. Springer, 2010.
15. B. Ustaoglu. Obtaining a secure and efficient key agreement protocol from (H)MQV and NAXOS. *Des. Codes Cryptography*, 46(3):329–342, 2008.

A. Security Model

In this section, we review the eCK security model for authenticated key exchange protocols. For the details of the original eCK model, see [9, 10].

Participants. We model the protocol participants as a finite set $\mathcal{U}$ of fixed size with each ID_i being a probabilistic polynomial time (PPT) Turing machine. Each protocol participant $ID_i \in \mathcal{U}$ may execute a polynomial number of protocol instances in parallel. We will refer to s-th instance of participant ID_i communicating with peer ID_j as $\Pi^s_{ID_i,ID_j}(i, j \in N)$ (*a session* or *an instance*).

Adversary Model. The adversary M is modeled as a PPT Turing machine and has full control of the communication network and may eavesdrop, delay, replay, alter and insert messages at will. We model the adversary's capability by providing it with oracle queries.

– **EphemeralKeyReveal($\Pi^s_{ID_i,ID_j}$)** The adversary obtains the ephemeral private key of $\Pi^s_{ID_i,ID_j}$. These queries are motivated by practical scenarios,

such as if session-specific secret information is stored in insecure memory on device or if the random number generator of the party is corrupted.
- **SessionKeyReveal($\Pi^s_{ID_i,ID_j}$)** The adversary obtains the session key for a session s of ID_i, provided that the session holds a session key.
- **StaticKeyReveal(ID_i)** The adversary obtains the static private key of ID_i.
- **EstablishParty(ID_i)** The query models that the adversary can arbitrarily register a legal user on behalf of the party ID_i. In this way the adversary gets the party ID_i's static private key and totally controls the party ID_i. Parties against whom the adversary does not issue this query are called *honest*.
- **Send($\Pi^s_{ID_i,ID_j}, m$)** The adversary sends the message m to the session s executed by ID_i communicating with ID_j and gets a response according to the protocol specification.
- **Test($\Pi^s_{ID_i,ID_j}$)** Only one query of this form is allowed for the adversary. Provided that the session key is defined, the adversary M can execute this query at any time. Then depending on a randomly chosen bit b, with probability $1/2$ the session key and with probability $1/2$ a uniformly chosen random value $\zeta \in \{0,1\}^\kappa$ is returned.

Definition 1 (Matching Session). *Let $\Pi^s_{ID_i,ID_j}$ be a completed session with identifier $(ID_i, ID_j, out, in, role)$, where ID_i is the owner of the session, ID_j is the peer, and out is ID_i's outgoing message, in is ID_j's outgoing message, and role is the ID_i's role in the session (initiator or responder). The session $\Pi^t_{ID_j,ID_i}$ is called the* matching session *of $\Pi^s_{ID_i,ID_j}$, if the identifier of $\Pi^t_{ID_j,ID_i}$ is $(ID_j, ID_i, \overline{out}, \overline{in}, \overline{role})$, where $out = \overline{in}, in = \overline{out}, role \neq \overline{role}$.*

Definition 2 (Freshness for AKE Protocols). *Let instance $\Pi^s_{ID_i,ID_j}$ be a completed session, which was executed by an honest party ID_i with another honest party ID_j. We define $\Pi^s_{ID_i,ID_j}$ to be* fresh *if none of the following three conditions hold:*

- *The adversary M reveals the session key of $\Pi^s_{ID_i,ID_j}$ or of its matching session (if latter exists).*
- *ID_j is engaged in session $\Pi^t_{ID_j,ID_i}$ matching to $\Pi^s_{ID_i,ID_j}$ and M issues either:*
  - *both **StaticKeyReveal**(ID_i) and **EphemeralKeyReveal**($\Pi^s_{ID_i,ID_j}$) queries; or*
  - *both **StaticKeyReveal**(ID_j) and **EphemeralKeyReveal**($\Pi^t_{ID_j,ID_i}$) queries.*
- *No sessions matching to $\Pi^s_{ID_i,ID_j}$ exist and M issues either:*
  - *both **StaticKeyReveal**(ID_i) and **EphemeralKeyReveal**($\Pi^s_{ID_i,ID_j}$) queries; or*
  - ***StaticKeyReveal**(ID_j) queries.*

Definition 3 (AKE Security). *As a function of the security parameter k, we define the advantage $Adv^{AKE}_{M,\Sigma}(k)$ of the PPT adversary M in attacking protocol Σ as*

$$Adv^{AKE}_{M,\Sigma}(k) \stackrel{def}{=} |Succ^{AKE}_{M,\Sigma}(k) - \tfrac{1}{2}|$$

Here $Succ^{AKE}_{M,\Sigma}$ *is the probability that the adversary queries* ***Test*** *oracle to a* fresh *instance* $\Pi^{s}_{ID_i,ID_j}$*, outputs a bit* $\hat{b}$ *such that* $\hat{b} = b$*, where the bit* b *is used by the* ***Test*** *oracle.*

We call the authenticated key exchange protocol Σ *to be* AKE *secure if for any* PPT *adversary* M *the function is negligible.*

An important property not captured in the eCK model is perfect forward security (PFS) which guarantees that the session key can not be learned by the adversary even if the static private keys of the parties are subsequently revealed. The freshness definition 2 above only captures weak perfect forward security (wPFS), which assume that the adversary is not *actively* involved with the choice of the messages at a session. Below, we give the definition of perfect forwards security against active adversary.

Perfect Forward Security. An authenticated key exchange protocol is said to be secure with PFS if Definition 3 holds even when the adversary is allowed to reveal the static private keys of two parties after the Test session is complete. Note that in this case the adversary is allowed to make all oracle queries above except for EphemeralKeyReveal query.