An eCK-secure Authenticated Key Exchange Protocol without Random Oracles

Daisuke Moriyama[1] and Tatsuaki Okamoto[2]
[1] Institute of Information Security
2-14-1, Tsuruya-cho, Kanagawa-ku, Yokohama-shi, Kanagawa - JAPAN
[e-mail: dgs082101@iisec.ac.jp]
[2] NTT, 3-9-11, Midori-cho, Musashino-shi, Tokyo - JAPAN
[e-mail: okamoto.tatsuaki@lab.ntt.co.jp]
*Corresponding author: Tatsuaki Okamoto

Received September 15, 2010; revised November 5, 2010; accepted February 22, 2011; published March 31, 2011

Abstract

Two-party key exchange protocol is a mechanism in which two parties communicate with each other over an insecure channel and output the same session key. A key exchange protocol that is secure against an active adversary who can control and modify the exchanged messages is called authenticated key exchange (AKE) protocol. LaMacchia, Lauter and Mityagin presented a strong security definition for public key infrastructure (PKI) based two-pass protocol, which we call the extended Canetti-Krawczyk (eCK) security model, and some researchers have provided eCK-secure AKE protocols in recent years. However, almost all protocols are provably secure in the random oracle model or rely on a special implementation technique so-called the NAXOS trick. In this paper, we present a PKI-based two-pass AKE protocol that is secure in the eCK security model. The security of the proposed protocol is proven without random oracles (under three assumptions), and does not rely on implementation techniques such as the NAXOS trick.

Keywords: Key exchange protocol, PKI-based, eCK security model, provable security, without random oracles

A preliminary version of this paper appeared in Provsec 2009, November 11-13, Guangzhou, China. This version provides more efficient protocol than the preliminary version and includes a concrete analysis of efficiency with the previous results. We would like to thank Berkant Ustaoglu for invaluable comments on the preliminary version of this paper.

DOI: 10.3837/tiis.2011.03.009

1. Introduction

A two-party key exchange protocol is a cryptographic protocol between Alice and Bob in which each party communicates with each other over an insecure channel and shares a common session key. The most famous key exchange protocol is the Diffie-Hellman key exchange, which is secure against a benign adversary who passively eavesdrops on the party's communication. However, it is well-known that the Diffie-Hellman protocol is insecure against an active adversary who can control and modify the exchanged messages (i.e., man-in-the-middle attacks). A protocol proven secure against an active adversary is called an authenticated key exchange (AKE) protocol. In a public key infrastructure (PKI) based AKE protocol, each party possesses static secret key used for static public key, in addition to the ephemeral secret key used for ephemeral public key (ephemeral secret/public key is randomly selected per session), and computes the session key.

The security definitions for two-party key exchange were developed by several researchers [1][2][3][4]. In their literature, an active adversary can obtain several types of secret information. The main difference among the security definitions in these papers is that an adversary can obtain what kind of internal secret of the party (static secret key, ephemeral secret key and session key). Among these papers, LaMacchia, Lauter and Mityagin [4] presented a strong security definition for two-pass key exchange protocol, which we call the extended Canetti-Krawczyk (eCK) security model. This security model allows an active adversary to obtain various private information regarding a target session and it captures many known desirable security properties for AKE including the resistance to key-compromise impersonation (KCI) attacks, weak perfect forward secrecy (wPFS), and resilience to the leakage of ephemeral secret keys (RLE).

There are some protocols proven in the eCK security model [4][5][6][7][8][9][10][11][12][13]. The first eCK-secure AKE protocol, called NAXOS, was proposed by LaMacchia, Lauter and Mityagin [4]. NAXOS is provably secure in the random oracle model under the gap Diffie-Hellman (GDH) assumption. In their paper, they used an implementation technique that we call the NAXOS trick. Ustaoglu employed the NAXOS trick to the HMQV protocol [14] and proposed an eCK-secure AKE protocol called CMQV [5]. HMQV is secure in a slight variant of the Canetti-Krawczyk security model and satisfies the KCI-security, wPFS and RLE, but has not been proven to be eCK-secure. CMQV is eCK-secure in the random oracle model under the GDH assumption. Lee *et al.* proposed two eCK-secure key exchange protocols to improve the efficiency of the security reduction or complexity assumption [7][8], but these protocols also use the NAXOS trick and are provably secure in the random oracle model. Okamoto [6] presented an eCK-secure key exchange protocol without random oracles and under the NAXOS trick. Therefore, many of the eCK-secure AKE protocols rely on the NAXOS trick.

The NAXOS trick is an implementation technique that hides the exponent of an ephemeral public key from an adversary even if the adversary obtains the ephemeral secret key. In a typical Diffie-Hellman based key exchange protocol, the ephemeral secret key, x, is used as the exponent of the ephemeral public key such as $X := g^x$. However, in a key exchange protocol that utilizes the NAXOS trick, the exponent of the ephemeral public key is generated by hashing of the ephemeral secret key, x, and the static secret key, a, e.g., $H(x, a)$. Therefore, even if the secrity model allows an adversary to obtain the ephemeral secret key,

the exponent of the ephemeral public key is still unknown to the adversary. Recently, Ustaoglu [10], Kim et al. [11], Sarr et al. [12], and Moriyama-Okamoto [13] proposed eCK-secure key exchange protocols without the NAXOS trick. The main motivation to avoid the NAXOS trick is to consider the leakage of the exponent of the ephemeral public key since it may be revealed via side-channel attacks or power analysis in a realistic setting. Therefore, even though the NAXOS trick hides the exponent of the ephemeral public key, it may be leaked via such side-channel attacks and then the eCK security is no longer ensured for AKE protocols that employs the NAXOS trick.

We present a new eCK-secure AKE protocol that does not use the NAXOS trick, which is provably secure without random oracles, and is as efficient as Okamoto protocol [6]. The exponent, i.e., the ephemeral secret key of an ephemeral public key, is purely independent from the static secret key and the risk of leaking the static secret key is reduced. We note that all AKE protocols provided in [10], [11] and [12] are provably secure in the random oracle model and that this does not imply the security in the real world (see [15]). In comparison to [13], the size of the static secret/public key is reduced and the session key computation is further improved. Nonetheless, we can provide a security proof under the eCK security model. The proposed protocol is an extended version of the Okamoto protocol [6] and is eCK-secure without random oracles under the Decision Diffie-Hellman (DDH) assumption, pair-wise independent pseudo-random function family, and collision resistant hash function family.

2. Preliminaries

2.1 Notation

When B is a probabilistic machine or algorithm, $A(x)$ denotes the random variable of the output of A on input x. $y \xleftarrow{R} A(x)$ denotes that y is randomly selected from $A(x)$ according to its distribution. Then, $A(x) \to a$ indicates the event that A outputs a on input x if a is a value. When A is a set, $y \xleftarrow{U} A(x)$ means that y is uniformly selected from A. When A is a value, $y := x$ denotes that A is set as y.

2.2 The DDH assumption

Let G be a group of prime order q and $\{G_k\}_k$ be set of group G with security parameter k. For all $k \in N$, we define sets $D(k) := \{(G, g_1, g_2, g_1^x, g_2^x) \mid G \in \{G\}_k, g_1, g_2 \in G^2, x \in Z_q\}$ and $R(k) := (G, g_1, g_2, y_1, y_2) \mid G \in \{G\}_k, (g_1, g_2, y_1, y_2 \in G^4\}$.
The DDH advantage of algorithm M is defined as

$$\mathrm{Adv}_M^{\mathrm{DDH}}(k) := \left| \begin{array}{l} \Pr\left[M(1^k, \rho) \to 1 \mid \rho \xleftarrow{U} D(k)\right] \\ -\Pr\left[M(1^k, \rho) \to 1 \mid \rho \xleftarrow{U} R(k)\right] \end{array} \right| \quad (1)$$

for all $k \in N$. We say that the DDH assumption holds in $\{G_k\}_{k \in N}$ if for any probabilistic polynomial-time adversary M, $\mathrm{Adv}_M^{\mathrm{DDH}}(k)$ is negligible in k.

2.3 Pseudo-Random Function (PRF)

Let $k \in N$ be a security parameter. A pseudo-random function (PRF) family F associated

with $\{\text{seed}_k\}_{k\in N}$, $\{\text{Dom}_k\}_{k\in N}$ and $\{\text{Rng}_k\}_{k\in N}$ is indexed by k. When σ is randomly chosen from random seeds $\Sigma \xleftarrow{R} \text{seed}_k$, $\text{F}^{k,\Sigma,D,R}$ maps an element of D to an element of R where $D \xleftarrow{R} \text{Dom}_k$ and $R \xleftarrow{R} \text{Rng}_k$. Let A^O be a probabilistic polynomial-time machine with oracle access to O. The advantage of algorithm A^O breaking the PRF function is defined by

$$\text{Adv}^{\text{PRF}}_{\text{F},M}(k) := \left| \begin{array}{c} \Pr\left[M^{F}(1^k, D, R) \to 1\right] \\ -\Pr\left[M^{RF}(1^k, D, R) \to 1\right] \end{array} \right| \tag{2}$$

where $\Sigma \xleftarrow{R} \text{seed}_k$, $\sigma \xleftarrow{U} \Sigma$, $D \xleftarrow{R} \text{Dom}_k$, $R \xleftarrow{R} \text{Rng}_k$, and $F := \text{F}^{k,\Sigma,D,R}$, and $RF : D \to R$ is a truly random function. F is a PRF family if for any probabilistic polynomial-time adversary M, $\text{Adv}^{\text{PRF}}_{\text{F},M}(k)$ is negligible in k.

2.4 Pseudo-Random Function with Pairwise Independent Random Sources (π PRF)

Okamoto [6] introduced a specific class of PRF, π PRF. As described in Section 2.3, the traditional PRF takes as input a (truly random) seed and it is reused many times. However, if this is determined by the output of a cryptographic primitive in each invocation, two or more seeds may be correlated. Note that the PRF family is useless in this setting. Nonetheless, π PRF states that if a specific variable σ_{i_0} (associated with 'seed') is pairwise-independent from the other variables, then the output of the function with σ_{i_0} is indistinguishable from random.

Suppose that $f_\Sigma : I_\Sigma \to X_\Sigma$ is a deterministic polynomial-time algorithm, where X_Σ is a set of random variables and I_Σ is a set of indices regarding Σ, then this algorithm outputs $\sigma_i \in X_\Sigma$ from $i \in I_\Sigma$. Let $(\sigma_{i_0}, \sigma_{i_1}, \ldots, \sigma_{i_{t(k)}})$ ($i_j \in I_\Sigma$) be pairwise independent random variables indexed by (I_Σ, f_Σ), and each variable be uniformly distributed over Σ. That is, for any pair of $(\sigma_{i_0}, \sigma_{i_j})$ ($j = 1, \ldots, t(k)$), for any $(x, y) \in \Sigma^2$, we have $\Pr[\sigma_{i_0} \to x \wedge \sigma_{i_j} \to y] = 1/|\Sigma|^2$. Consider a probabilistic polynomial-time machine M^{F,I_Σ} that can issue oracle queries. When M sends $q_j \in D$ and $i_j \in I_\Sigma$ to the query, the oracle replies with $F^{k,\Sigma,D,R}_{\bar{\sigma}_j}(q_j)$ for each $j = 0,1,\ldots,t(k)$, where $(\bar{\sigma}_{i_0}, \bar{\sigma}_{i_1}, \ldots, \bar{\sigma}_{i_{t(k)}}) \xleftarrow{R} (\sigma_{i_0}, \sigma_{i_1}, \ldots, \sigma_{i_{t(k)}})$. M^{RF,I_Σ} is the same as M^{F,I_Σ} except $F^{k,\Sigma,D,R}_{\bar{\sigma}_0}(q_0)$ is replaced by a truly random function $RF(q_0)$. The advantage of algorithm M breaking the π PRF function is defined by

$$\text{Adv}^{\text{PRF}}_{\text{F},I_\Sigma,M}(k) := \left| \begin{array}{c} \Pr\left[M^{F,I_\Sigma}(1^k, D, R) \to 1\right] \\ -\Pr\left[M^{RF,I_\Sigma}(1^k, D, R) \to 1\right] \end{array} \right| \tag{3}$$

We say that F is a π PRF family if for any probabilistic polynomial-time adversary M, $\text{Adv}^{\text{PRF}}_{\text{F},I_\Sigma,M}(k)$ is negligible in k.

Okamoto presented an example of index (I_Σ, f_Σ) for the π PRF function. For group G of prime order q,

$$I_G := \{(U,V,d) \mid (U,V,d) \in G^2 \times Z_q\}, \tag{4}$$

$$X_G := \left\{ \sigma_{(U,V,d)} \mid (U,V,d) \in G^2 \times Z_q, (r_1,r_2) \overset{U}{\leftarrow} Z_q^2, \sigma_{(U,V,d)} := U^{r_1+dr_2}V \right\}, \tag{5}$$

$$f_G : I_G \rightarrow X_G \tag{6}$$

is a pair-wise independent pseudo-random function (See [7]).

2.5 Collision Resistant (CR) Hash Function

Let $k \in N$ be a security parameter. Collision resistant (CR) hash function family H associated with $\{\mathrm{Dom}_k\}_{k\in N}$ and $\{\mathrm{Rng}_k\}_{k\in N}$ specifies two items:

1. A family of key spaces indexed by k. Each such key space is a probability space on bit strings denoted by KH_k. There must be a probabilistic polynomial-time algorithm whose output distribution on input 1^k is equal to KH_k.
2. A family of hash functions indexed by k, $h \overset{R}{\leftarrow} \mathrm{KH}_k$, $\{\mathrm{Dom}_k\}_{k\in N}$, and $\{\mathrm{Rng}_k\}_{k\in N}$, where each such function $\mathrm{H}_h^{k,D,R}$ maps an element of D to an element of R. There must exist a deterministic polynomial-time algorithm that on input 1^k, h and $\rho \in D$, output $\mathrm{H}_h^{k,D,R}(\rho)$.

We define the advantage of algorithm M breaking the CR hash function as

$$\mathrm{Adv}_{\mathrm{H},M}^{\mathrm{CR}}(k) := \left| \begin{array}{l} (\rho_1,\rho_2) \in D^2 \wedge \rho_1 \neq \rho_2 \wedge \mathrm{H}_h^{k,D,R}(\rho_1) = \\ \mathrm{H}_h^{k,D,R}(\rho_2) \mid M(1^k,h,D,R) \rightarrow (\rho_1,\rho_2) \end{array} \right|, \tag{7}$$

where $D \overset{R}{\leftarrow} \mathrm{Dom}_k$, $R \overset{R}{\leftarrow} \mathrm{Rng}_k$ and $h \overset{R}{\leftarrow} \mathrm{KH}_k$. H is a CR hash function family if for any probabilistic polynomialtime adversary M, $\mathrm{Adv}_{\mathrm{H},M}^{\mathrm{CR}}(k)$ is negligible in k.

3. The Extended Canetti-Krawczyk (eCK) Security Model

The eCK security model was originally introduced by LaMacchia, Lauter and Mityagin [4]. We refer the reader to [4][5] and [13] for more background.

Suppose that there are n parties that are modeled as probabilistic polynomial-time Turing machines. Each party generates a static public/secret key pair and the static public key is certificated by a certification authority. When we consider two parties Alice and Bob, Alice has static public key A and Bob has static public key B. The certified public key, $\hat{A}$ ($\hat{B}$), binds the identity of the party, the static public key and its certificate. The precise certification procedure is dependent on its implementation, but we assume that if the key exchange protocol explicitly specifies the proof of knowledge for a part of the static public key, each party is required to prove the knowledge of a part of the corresponding static secret key by some method, e.g., by a zero-knowledge proof of knowledge or in an administrative manner, to obtain a certificate of the static public key. Otherwise, the certification authority only checks whether or not the static public key is included in the certain static public key space. As denoted in [13], here we only assume two conditions for the proof of knowledge (we do not assume any specific method for the proof):

1. There exists an efficient (or polynomial-time) measure using the corresponding party as a black-box to extract the secret key (with overwhelming probability).
2. Any adversary obtains (information theoretically) no additional information via the process of the proof of knowledge if the corresponding party is honest (see the definition of `honest' later).

When a key exchange protocol is executed, each party starts an instance of the protocol called a session. When Alice executes a key exchange protocol with Bob, she is activated by an incoming message: $(\hat{A}, \hat{B})$ or $(\hat{A}, \hat{B}, Y)$ where Y denotes the ephemeral public key of Bob. If Alice receives $(\hat{A}, \hat{B})$, she is called the initiator. Alice generates ephemeral public/secret key pair (X, x) and sends $(\hat{B}, \hat{A}, X)$ to Bob. When Bob receives $(\hat{B}, \hat{A}, X)$, he is called the responder. Bob generates ephemeral public/static key pair (Y, y) and responds with $(\hat{A}, \hat{B}, X, Y)$ to Alice. When an execution of the party is finished and outputs the session key, the session is said to be completed. Even if a party concurrently executes many sessions with many parties, each session is uniquely determined by the session identifier. The session identifier is in the form of $(\text{role}, \hat{A}, \hat{B}, X, Y)$, which indicates that owner Alice who is activated as $\text{role} \in \{\text{initiator}, \text{responder}\}$ executes a session with Bob. The ephemeral public key X is included in the first message of the protocol (outgoing or incoming) and Y is included in the second message. If there exists a session identifier in the form of $(\text{role}', \hat{B}, \hat{A}, X, Y)$ against $(\text{role}, \hat{A}, \hat{B}, X, Y)$ where $\text{role} \neq \text{role}'$, these sessions are said to be matching.

The adversary, M, is modeled as a probabilistic polynomial-time Turing machine and controls all communications. The adversary activates parties with incoming messages via the Send(message), thereby controlling the activation of sessions. Furthermore, M can issue the following queries.

EphemeralKeyReveal(SID) – The adversary obtains the ephemeral secret key for the session SID.
SessionKeyReveal(SID) – The adversary obtains the session key for the completed session SID.
StaticKeyReveal(PID) – The adversary obtains the static secret key of party PID.
EstablishParty(PID, Z) – The adversary registers a static public key Z on behalf of party PID. The party is controlled by the adversary.

If a party PID is established by EstablishParty(PID, Z) then we call the party dishonest. If a party is not dishonest, we call the party honest.

When adversary M issues test query Test(SID^*), bit $b \xleftarrow{U} \{0,1\}$ is chosen. If $b = 1$, M receives the actual session key SK^* of the test session SID^*. Otherwise, M is given random key R^* where $R^* \xleftarrow{U} \{0,1\}^{|SK^*|}$. Finally, M outputs bit $b' \in \{0,1\}$ for the test query. To define the advantage of M, we need the notation of a fresh session given hereafter.

Definition 1: (fresh session of eCK security) Let $\mathrm{SID} := (\mathrm{role}, \hat{A}, \hat{B}, X, Y)$ be the session identifier executed by honest parties Alice and Bob. We define the matching session of SID as $\overline{\mathrm{SID}}$, if it exists.
We define session SID to be fresh if none of the following conditions hold.

1. M issues
   - SessionKeyReveal(SID), or
   - SessionKeyReveal($\overline{\mathrm{SID}}$) (if $\overline{\mathrm{SID}}$ exists).
2. If $\overline{\mathrm{SID}}$ exists, then M issues
   - Both StaticKeyReveal($\hat{A}$) and EphemeralKeyReveal(SID), or
   - Both StaticKeyReveal($\hat{B}$) and EphemeralKeyReveal($\overline{\mathrm{SID}}$).
3. If $\overline{\mathrm{SID}}$ does not exists, then M issues
   - Both StaticKeyReveal($\hat{A}$) and EphemeralKeyReveal(SID), or
   - StaticKeyReveal($\hat{B}$).

Definition 2: (eCK security) Let test session SID^*, where adversary M issues Test(SID^*), be fresh. Then, we define the advantage of M by

$$\mathrm{Adv}_M^{\mathrm{eCK}}(k) := \left| 2 \cdot \Pr[b' = b] - 1 \right| \tag{8}$$

A key exchange protocol is eCK-secure if the following conditions hold.

1. If two honest parties complete matching sessions, they compute the same session key.
2. For any probabilistic polynomial-time adversary M, $\mathrm{Adv}_M^{\mathrm{eCK}}(k)$ is negligible in k.

4. The NAXOS Trick

The NAXOS trick is one of the implementation techniques proposed by LaMacchia, Lauter, and Mityagin [4] to construct an eCK-secure key exchange protocol. In a typical key exchange protocol, ephemeral public key X is computed by $X := g^x$ and x is the ephemeral secret key. If adversary M issues an ephemeral key reveal query to the session, M can directly obtain x, which is the exponent of X.

On the other hand, the NAXOS trick specifies that ephemeral public key X is computed by $X := g^{\tilde{x}}$, where $\tilde{x}$ is the hashing of ephemeral secret key x and static secret key a, i.e., $\tilde{x} := H(x, a)$. The hashed value, $\tilde{x}$, is not stored as the ephemeral secret key but is computed each time when required. So, adversary M cannot directly obtain the exponent of X, even if M issues an ephemeral key reveal query.
This trick is adopted by many of the existing eCK-secure key exchange protocols [4][5][6][7][8]. However, this technique does not work in some realistic settings such as side-channel attacks. For example, $\tilde{x}$ with $X := g^{\tilde{x}}$ may be revealed to an adversary through up-to-date power-analysis. Therefore, an eCK-secure protocol based on the NAXOS trick may be vulnerable (in the sense of the eCK security) to some realistic side-channel attacks.

5. The Proposed Protocol

We propose an eCK-secure key exchange protocol that does not use the ``NAXOS trick."

5.1 Protocol

Let $k \in N$ be a security parameter and $G \xleftarrow{U} \{G\}_k$ be a group that has prime order q with $|q| = k$. Set $(g_1, g_2) \xleftarrow{U} G^2$. Let $\{\mathrm{H}\}_k$ be a CR hash function family and $h_1, h_2 \xleftarrow{R} \{\mathrm{H}\}_k$ be an index of CR hash function $H_i := \mathrm{H}_{h_i}^{k,D_H,R_H}$ (for $i = \{1,2\}$), where $D_H := (\Pi_k)^2 \times G^6$, $R_H := Z_q$ and Π_k denotes the space of certified static public keys. Let F be a π PRF family and $F := \mathrm{F}^{k,\Sigma_F,D_F,R_F}$ where $\Sigma_F := G$, $D_F := (\Pi_k)^2 \times G^6$ and $R_F := \{0,1\}^k$. The system parameter of the proposed AKE protocol is $(G, g_1, g_2, H_1, H_2, F)$.

Party Alice selects static secret key $(a_1, \ldots, a_4, a) \xleftarrow{U} (Z_q)^5$ and computes static public key $(A_1, A_2, A_3, A_4) := (g_1^{a_1} g_2^{a_2}, g_1^{a_3} g_2^{a_4}, g_1^{a}, g_2^{a})$. Similarly, Bob selects static secret key $(b_1, \ldots, b_4, b) \xleftarrow{U} (Z_q)^5$ and computes static public key $(B_1, B_2, B_3, B_4) := (g_1^{b_1} g_2^{b_2}, g_1^{b_3} g_2^{b_4}, g_1^{b}, g_2^{b})$.

In this protocol, a party is required to prove the knowledge of a part of the static secret key, from the first component to the fourth component (e.g., $(a_1, \ldots, a_4)$ (see Section 2 for the proof of knowledge).

When the proposed key exchange protocol between initiator Alice and responder Bob is executed,
Alice performs the following procedure to establish a session key with party Bob.

1. Select ephemeral secret key $(x, x_3) \xleftarrow{U} Z_q$ and compute the ephemeral public key
2. Compute ephemeral public key $(X_1, X_2, X_3) := (g_1^{x}, g_2^{x}, g_1^{x_3})$.
3. Send $(\hat{B}, \hat{A}, X_1, X_2, X_3)$ to Bob.

Upon receiving $(\hat{B}, \hat{A}, X_1, X_2, X_3)$, Bob checks that $(X_1, X_2, X_3) \in G^3$. If the condition holds, Bob executes the following procedure.

1. Select ephemeral secret key $(y, y_3) \xleftarrow{U} Z_q$.
2. Compute ephemeral public key $(Y_1, Y_2, Y_3) := (g_1^{y}, g_2^{y}, g_1^{y_3})$.
3. Send $(\hat{A}, \hat{B}, X_1, X_2, X_3, Y_1, Y_2, Y_3)$ to Alice.

Upon receiving $(\hat{A}, \hat{B}, X_1, X_2, X_3, Y_1, Y_2, Y_3)$, Alice checks if she sent $(\hat{B}, \hat{A}, X_1, X_2, X_3)$ to Bob and verifies that $(Y_1, Y_2, Y_3) \in G^3$.

After the verification step, Alice computes

$$K_A := (Y_1 B_3)^{a_1 + \alpha a_3} (Y_2 B_4)^{a_2 + \alpha a_4} B_1^{x+a} B_2^{\beta(x+a)} Y_3^{x_3} \tag{9}$$

and outputs session key $SK_A := F_{K_A}(s)$, where $s := (\hat{A}, \hat{B}, X_1, X_2, X_3, Y_1, Y_2, Y_3)$, $\alpha := H_1(s)$ and $\beta := H_2(s)$. Here, s is specified as the remaining part of the initiator's session identifier except the role part `initiator'. In a similar way, Bob computes

$$K_B := (X_1 A_3)^{b_1 + \beta b_3} (X_2 A_4)^{b_2 + \beta b_4} A_1^{y+b} A_2^{\alpha(y+b)} X_3^{y_3} \tag{10}$$

and outputs session key $SK_B := F_{K_B}(s)$.

This construction is a variant of the Okamoto protocol [7], but the proposed protocol does not

require the NAXOS trick. Instead, we add (A_3, A_4, B_3, B_4) to the static public key and use them in the computation for the session key to satisfy the eCK security model. In [13], the authors add more extra information $(A_5, A_6, B_5, B_6) := (g_1^{a_5} g_2^{a_6}, g_1^{a_7} g_2^{a_8}, g_1^{b_5} g_2^{b_6}, g_1^{b_7} g_2^{b_8})$ to the static public key. The exchanged messages between the two parties are the same as those above but the computation of the session key is more complicated; that is, Alice computes K_A as $K_A := Y_1^{a_1+\alpha a_3} Y_2^{a_{2+}\alpha a_4} B_1^{x} B_2^{\beta x} B_3^{a} B_4^{\chi a} B_5^{a_5+\chi a_7} B_6^{a_6+\chi a_8} Y_3^{x_3}$. This computation ensures that Alice's ephemeral secret key x and static secret key a are used for the *independent* static secret key of Bob ((B_1, B_2) and (B_3, B_4)). Instead, the cost for computing the session key is less efficient than the original Okamoto protocol (see Section 6). In contrast, the proposed protocol in this paper does not require such independence and the computational cost for the session key is equivlent to that in the Okamoto protocol. Nonetheless, we show that the proposed protocol is provably secure in the eCK security model.

Alice		Bob
$(a_1,\ldots,a_4,a) \overset{U}{\leftarrow} (Z_q)^5$		$(b_1,\ldots,b_4,b) \overset{U}{\leftarrow} (Z_q)^5$
$(A_1, A_2, A_3, A_4) := (g_1^{a_1} g_2^{a_2}, g_1^{a_3} g_2^{a_4}, g_1^{a}, g_2^{a})$		$(B_1, B_2, B_3, B_4) := (g_1^{b_1} g_2^{b_2}, g_1^{b_3} g_2^{b_4}, g_1^{b}, g_2^{b})$
$(x, x_3) \overset{U}{\leftarrow} Z_q$		
$(X_1, X_2, X_3) := (g_1^{x}, g_2^{x}, g_1^{x_3})$		
	$(\hat{B}, \hat{A}, X_1, X_2, X_3) \longrightarrow$	$(X_1, X_2, X_3) \in G^3$?
		$(y, y_3) \overset{U}{\leftarrow} Z_q$
	$\longleftarrow$	$(Y_1, Y_2, Y_3) := (g_1^{y}, g_2^{y}, g_1^{y_3})$
$(Y_1, Y_2, Y_3) \in G^3$?	$(\hat{A}, \hat{B}, X_1, X_2, X_3, Y_1, Y_2, Y_3)$	
$\alpha := H_1(s),\ \beta := H_2(s)$		$\alpha := H_1(s),\ \beta := H_2(s)$
$s := (\hat{A}, \hat{B}, X_1, X_2, X_3, Y_1, Y_2, Y_3)$		$s := (\hat{A}, \hat{B}, X_1, X_2, X_3, Y_1, Y_2, Y_3)$
$K_A := (Y_1 B_3)^{a_1+\alpha a_3} (Y_2 B_4)^{a_{2+}\alpha a_4} B_1^{x+a} B_2^{\beta(x+a)} Y_3^{x_3}$		$K_B := (X_1 A_3)^{b_1+\beta b_3} (X_2 A_4)^{b_{2+}\beta b_4} A_1^{y+b} A_2^{\alpha(y+b)} X_3^{y_3}$
$SK_A := F_{K_A}(s)$		$SK_B := F_{K_B}(s)$

Fig. 1. The Proposed Protocol.

5.2 Security

Theorem 1: Suppose that the DDH assumption holds for $\{G_k\}_{k\in N}$, F is the π PRF family with index $\{\mathrm{I}_G, f_G\}_{G\in\{G\}_k, k\in N}$ where $I_G := \{(U,V,d) \mid (U,V,d) \in G^2 \times Z_q\}$ and $f_G : (U,V,d) \rightarrow U^{r_1+dr_2} V$ with $(r_1, r_2) \overset{U}{\leftarrow} Z_q^2$, and H is the CR hash function family. Then the

proposed AKE protocol is eCK-secure (in the sense of Definition 2).

It is clear that the first condition of Definition 2 holds since we have

$$\left(Y_1 B_3\right)^{a_1+\alpha a_3}\left(Y_2 B_4\right)^{a_{2+}\alpha a_4} = A_1^{y+b} A_2^{\alpha(y+b)} = g_1^{(y+b)(a_1+\alpha a_3)} g_2^{(y+b)(a_{2+}\alpha a_4)}, \tag{11}$$

$$B_1^{x+a} B_2^{\beta(x+a)} = \left(X_1 A_3\right)^{b_1+\beta b_3}\left(X_2 A_4\right)^{b_{2+}\beta b_4} = g_1^{(x+a)(b_1+\beta b_3)} g_2^{(x+a)(b_{2+}\beta b_4)}, \text{ and} \tag{12}$$

$$Y_3^{x_3} = X_3^{y_3} = g_1^{x_3 y_3}. \tag{13}$$

Thus, we will prove that the second condition of Definition 2 holds under the assumptions.

We consider that the adversary chooses the test session between owner Alice and Bob. Without loss of generality, we assume the role of Alice at the test session is the initiator and we use the following notation:

- $(a_1^*, a_2^*, a_3^*, a_4^*)$: the static secret key of Alice
- $(A_1^*, A_2^*, A_3^*, A_4^*) := (g_1^{a_1^*} g_2^{a_2^*}, g_1^{a_3^*} g_2^{a_4^*}, g_1^{a^*}, g_2^{a^*})$: the static puclic key of Alice
- $(B_1^*, B_2^*, B_3^*, B_4^*) := (g_1^{b_1^*} g_2^{b_2^*}, g_1^{b_3^*} g_2^{b_4^*}, g_1^{b^*}, g_2^{b^*})$: the static public key of Bob
- (x^*, x_3^*): the ephemeral secret key of Alice in the test session
- $(X_1^*, X_2^*, X_3^*) := (g_1^{x^*}, g_2^{x^*}, g_1^{x_3^*})$: the ephemeral public key of Alice in the test session
- (Y_1^*, Y_2^*, Y_3^*): the ephemeral public key Alice received during the session
- $\mathrm{SID}^* := (\text{initiator}, \hat{A}, \hat{B}, X_1^*, X_2^*, X_3^*, Y_1^*, Y_2^*, Y_3^*)$: the session identifier in the test session

The session key of test session sid^* is computed by

1. Set $s^* := (\hat{A}, \hat{B}, X_1^*, X_2^*, X_3^*, Y_1^*, Y_2^*, Y_3^*)$.
2. Compute $\alpha^* := H_1(s^*)$ and $\beta^* := H_2(s^*)$.
3. Compute $K_A^* := (Y_1^* B_3^*)^{a_1^*+\alpha^* a_3^*}(Y_2^* B_4^*)^{a_2^*+\alpha^* a_4^*}(B_1^*)^{x^*+a^*}(B_2^*)^{\beta^*(x^*+a^*)}(Y_3^*)^{x_3^*}$.
4. Compute $SK_A^* := F_{K_A^*}(s^*)$.

To estimate the advantage of the adversary, we consider the following cases in which the adversary can obtain the secret information through oracle queries. Note that these cases are sufficient to prove the eCK security.

When there exists a matching session $\overline{\mathrm{SID}}^*$ of test session SID^*,

- **C1.** M issues both EphemeralKeyReveal(SID^*) and EphemeralKeyReveal($\overline{\mathrm{SID}}^*$).
- **C2.** M issues both EphemeralKeyReveal(SID^*) and StaticKeyReveal($\hat{B}$).
- **C3.** M issues both StaticKeyReveal($\hat{A}$) and EphemeralKeyReveal($\overline{\mathrm{SID}}^*$).
- **C4.** M issues both StaticKeyReveal($\hat{A}$) and StaticKeyReveal($\hat{B}$).

When there exists no matching session $\overline{\mathrm{SID}}^*$ of test session SID^*,

- **C5.** M issues EphemeralKeyReveal(SID^*).
- **C6.** M issues StaticKeyReveal($\hat{A}$).

We provide the two propositions to facilitate the security proof of the proposed protocol.

Proposition 1: If adversary M breaks the security of the proposed protocol in Case **C3**, then there exists adversary M' who can achive a successful attack in Case **C2**.

Proof. We briefly sketch the security reduction as follows. M' runs M and responds all the oracle queries except the test session. When M issues the test query to SID^*, M' selects the matching session $\overline{\mathrm{SID}}^*$ as the test session. When M' receives the real session key or random key, M' sends it to M. If M outputs a bit, M' outputs the same bit. Note that M' can issue EphemeralKeyReveal(SID^*) since SID^* is the matching session to the test session from the view point of M'. So M' can correctly respond to all the queries issued by M. Therefore, if M breaks the security of the proposed protocol in Case **C3**, M' wins the game with non-negligible probability.

Proposition 2: If an adversary M breaks the security of our protocol in Case **C1** (**C2**), then there exists an adversary M' that can successfully attack in Case **C5** (**C6**).

Proof. This proof is similar to the previous one. When M issues the test query to SID^*, M' selects the session in the form of $\mathrm{SID}' := (\text{initiator}, \hat{A}, \hat{B}, X_1^*, X_2^*, X_3^*, Y^*)$ where Y^* is generated by M'. When M' receives the real session key or random key, M' sends it to M. If M outputs a bit, M' outputs the same bit. M' can respond to any oracle query issued by M since the restricted oracle queries between M and M' are equivalent. Therefore, M' breaks the security of the proposed protocol in Case **C5** (**C6**) if wins the game in Case **C1** (**C2**).

To complete the proof of Theorem 1, we must provide the security proofs for six cases. However, from Propositions 1 and 2, the security proofs for Cases C1, C2 and C3 can be omitted if we can describe the security proofs for Cases C5 and C6. Furthermore, one can easily obtain the security proof for Case 6 if (A_3^*, A_4^*) in the proof for Case 5 is replaced by (X_1^*, X_2^*) in the proof for Case 6, where $(X_1^*, X_2^*, B_1^*, B_2^*)$ corresponds to $(A_3^*, A_4^*, B_1^*, B_2^*)$ in our proof technique. The other gap between them is that the proof of knowledge used in the proof for Case 5 is not necessary in Case 6, i.e., a game corresponding to Game 2-3 can be omitted in Case 6. The reduction efficiency for Case 6 is almost equivalent to that for Case 5.
Therefore, we prove the advantage of the adversary is negligible in k even if Cases C4 and C5 occur, respectively.

Case C4:

We proceed in games, starting with Game 1-0 which is the original eCK game between a challenger and adversary M_1, the challenger simulates all honest parties and the answer of the test query. In each Game i, we define Adv_i as the advantage in which the adversary wins the game.

Game 1-0. This is the original eCK game with adversary M_1 in Case **C4**. Hence we have $\mathrm{Adv}_{1-0} = \mathrm{Adv}_{M_1}^{\mathrm{eCK}}(k)$.

Game 1-1. The challenger proceeds as Game 1-0 but aborts the game if it does not correctly guess the test session.

Game 1-2. We modify Game 1-1 by changing the value of $(Y_3^*)^{x_3^*}$ to a random element $\delta \overset{\mathrm{U}}{\leftarrow} G$.

Game 1-3. We change Game 1-2 to Game 1-3 by changing PRF F to a random function RF for test query Test(SID^*).

We evaluate the relations between the game transformations using the following claims.

Claim 1. We have

$$\mathrm{Adv}_{1-0} \le n(k)^2 q_a(k) \cdot \mathrm{Adv}_{1-1}. \tag{14}$$

Proof. Suppose that M_1 activates at most $q_a(k)$ sessions for each $n(k)$ party. The challenger uniformly selects the owner of test session $\hat{A}$ and peer $\hat{B}$ in $n(k)$ parties, and guesses that $\hat{A}$'s i-th session will be chosen as the test session in advance. Then the probability that the challenger correctly guesses the test session is at least $1/n(k)^2 q_a(k)$. Therefore, $\mathrm{Adv}_{1-0} \le n(k)^2 q_a(k) \cdot \mathrm{Adv}_{1-1}$.

Claim 2. There exists a probabilistic algorithm S_1 such that

$$\left|\mathrm{Adv}_{1-1} - \mathrm{Adv}_{1-2}\right| \le \mathrm{Adv}_{S_1}^{\mathrm{DDH}}(k). \tag{15}$$

Proof. If the adversary distinguishes Game 1-2 from Game 1-1 with non-negligible probability, we can construct an algorithm S_1 that solves the DDH problem.

For a given DDH instance $\rho := (G, u, v, w, z)$, where $\rho \xleftarrow{U} D(k)$ or $\rho \xleftarrow{U} R(k)$, S_1 sets $g_1 := u$ and chooses all parameters as Game 1-1 except X_3^* and Y_3^*. S_1 sets $X_3^* := v$ and $Y_3^* := w$ as the ephemeral public key at the test session and the matching session, respectively. If M_1 issues the test session, the challenger responds with the session key using

$$K_A^* := \left(Y_1^* B_3^*\right)^{a_1^* + \alpha^* a_3^*} \left(Y_2^* B_4^*\right)^{a_2^* + \alpha^* a_4^*} \left(B_1^*\right)^{x^* + a^*} \left(B_2^*\right)^{\beta^*(x^* + a^*)} \cdot z. \tag{16}$$

When M_1 outputs a guess b', S_1 outputs 1 iff $b' = b$ holds.

If $\rho \xleftarrow{U} D(k)$, the advantage of S_1 in this simulation is equivalent to that in Game 1-1, since $z = (Y_3^*)^{x_3^*} = (X_3^*)^{y_3^*}$ where $X_3^* := g^{x_3^*}$ and $Y_3^* := g^{y_3^*}$. Otherwise, the advantage of S_1 is the same as that in Game 1-2 because $\rho \xleftarrow{U} R(k)$ and $z \xleftarrow{U} G$ hold.
Therefore, we obtain $\left|\mathrm{Adv}_{1-1} - \mathrm{Adv}_{1-2}\right| \le \mathrm{Adv}_{S_1}^{\mathrm{DDH}}(k)$.

Claim 3. There exists a probabilistic algorithm S_2 such that

$$\left|\mathrm{Adv}_{1-2} - \mathrm{Adv}_{1-3}\right| \le \mathrm{Adv}_{\mathrm{F},S_2}^{\mathrm{PRF}}(k). \tag{17}$$

Proof. If the adversary distinguishes Game 1-3 from Game 1-2 with non-negligible probability, we can construct algorithm S_2 that breaks the PRF function (in this case, π PRF property is not needed).

Given oracle access to F or truly random function RF, S_2 selects all parameters as Game 1-2 and proceeds with the game except for the computation of test query Test(SID^*). When

M_1 issues a test query to SID^*, S_2 issues SID^* to the oracle and responds with the value to the adversary. When M_1 outputs guess b', S_2 outputs 1 iff $b' = b$ holds.

If the oracle is F, the advantage of S_2 in this simulation is equivalent to that in Game 1-2 because K_A^* is uniformly random and the session key of the test session is indistinguishable from that of $F_{K_A^*}(\mathrm{SID}^*)$. Otherwise, the advantage of S_2 is the same as that in Game 1-3 because RF is a random function. Therefore, we obtain $\left|\mathrm{Adv}_{1-2} - \mathrm{Adv}_{1-3}\right| \le \mathrm{Adv}_{\mathrm{F},S_2}^{\mathrm{PRF}}(k)$.

It is obvious that $\mathrm{Adv}_{1-3} = 0$, and we obtain

$$\mathrm{Adv}_{M_1}^{\mathrm{eCK}}(k) \le n(k)^2 q_a(k)\left(\mathrm{Adv}_{S_1}^{\mathrm{DDH}}(k) + \mathrm{Adv}_{S_2}^{\mathrm{PRF}}(k)\right). \quad (18)$$

Case C5:

Before describing the security proof, we consider sessions between $\hat{A}$ and $\hat{C}^{(i)}$ ($i = 1, \ldots, q_a(k)$) and we use the following notations.

-- $(x^{(i)}, x_3^{(i)})$: $\hat{A}$'s ephemeral secret key in the session
-- $(X_1^{(i)}, X_2^{(i)}, X_3^{(i)}) := (g_1^{x^{(i)}}, g_2^{x^{(i)}}, g_1^{x_3^{(i)}})$: the ephemeral public key of $\hat{A}$ in the session
-- $(Z_1^{(i)}, Z_2^{(i)}, Z_3^{(i)})$: the ephemeral public key of $\hat{A}$ received during the session
-- $(C_1^{(i)}, C_2^{(i)}, C_3^{(i)}, C_4^{(i)})$: $\hat{C}^{(i)}$'s static public key
-- $\mathrm{SID}_A^{(i)}$: $\hat{A}$'s session identifier in the session

The session key at $\mathrm{SID}_A^{(i)}$ is computed by using the variables below:

- $s_A^{(i)} := (\hat{A}, \hat{C}^{(i)}, X_1^{(i)}, X_2^{(i)}, X_3^{(i)}, Z_1^{(i)}, Z_2^{(i)}, Z_3^{(i)})$, $\alpha_A^{(i)} := H_1(s_A^{(i)})$ and $\beta_A^{(i)} := H_2(s_A^{(i)})$ (if $\hat{A}$ is the initiator).
- $s_A^{(i)} := (\hat{C}^{(i)}, \hat{A}, Z_1^{(i)}, Z_2^{(i)}, Z_3^{(i)}, X_1^{(i)}, X_2^{(i)}, X_3^{(i)})$, $\alpha_A^{(i)} := H_1(s_A^{(i)})$ and $\beta_A^{(i)} := H_2(s_A^{(i)})$ (if $\hat{A}$ is the responder).
- $K_A^{(i)} := (Z_1^{(i)} C_3^{(i)})^{a_1^* + \alpha_A^{(i)} a_3^*} (Z_2^{(i)} C_4^{(i)})^{a_2^* + \alpha_A^{(i)} a_4^*} (C_1^{(i)})^{x^{(i)} + a^*} (C_2^{(i)})^{\beta_A^{(i)}\left(x^{(i)} + a^*\right)} (Z_3^{(i)})^{x_3^{(i)}}$.

These variables are used in the game sequence in Game 2-3.

Similarly, we consider sessions between $\hat{B}$ and $\hat{D}^{(i)}$ ($i = 1, \ldots, q_a(k)$) and we use the notations below.

-- $(y^{(i)}, y_3^{(i)})$: $\hat{B}$'s ephemeral secret key in the session
-- $(Y_1^{(i)}, Y_2^{(i)}, Y_3^{(i)}) := (g_1^{y^{(i)}}, g_2^{y^{(i)}}, g_1^{y_3^{(i)}})$: the ephemeral public key of $\hat{B}$ in the session
-- $(W_1^{(i)}, W_2^{(i)}, W_3^{(i)})$: the ephemeral public key $\hat{B}$ received during the session
-- $(D_1^{(i)}, D_2^{(i)}, D_3^{(i)}, D_4^{(i)})$: $\hat{D}^{(i)}$'s static public key
-- $\mathrm{SID}_B^{(i)}$: $\hat{B}$'s session identifier in the session

The session key at $\mathrm{SID}_B^{(i)}$ is computed using the variables below.

- $s_B^{(i)} := (\hat{B}, \hat{D}^{(i)}, Y_1^{(i)}, Y_2^{(i)}, Y_3^{(i)}, W_1^{(i)}, W_2^{(i)}, W_3^{(i)})$, $\alpha_B^{(i)} := H_1(s_B^{(i)})$ and $\beta_B^{(i)} := H_2(s_B^{(i)})$ (if $\hat{B}$

is the initiator).

- $s_B^{(i)} := (\hat{D}^{(i)}, \hat{B}, W_1^{(i)}, W_2^{(i)}, W_3^{(i)}, Y_1^{(i)}, Y_2^{(i)}, Y_3^{(i)})$, $\alpha_B^{(i)} := H_1(s_B^{(i)})$ and $\beta_B^{(i)} := H_2(s_B^{(i)})$ (if $\hat{B}$ is the responder).
- $K_B^{(i)} := (W_1^{(i)} D_3^{(i)})^{b_1^* + \alpha_B^{(i)} b_3^*} (W_2^{(i)} D_4^{(i)})^{b_2^* + \alpha_B^{(i)} b_4^*} (D_1^{(i)})^{y^{(i)} + b^*} (D_2^{(i)})^{\beta_B^{(i)} (y^{(i)} + b^*)} (W_3^{(i)})^{y_3^{(i)}}$.

Now, we proceed in games with adversary M_2 as follows.
In each Game i, we define Adv_i as the advantage in which the adversary wins the game.

Game 2-0. This is the original eCK game with adversary M_2 in Case **C5**. Hence we have $\mathrm{Adv}_{2-0} = \mathrm{Adv}_{M_2}^{\mathrm{eCK}}(k)$.

Game 2-1. The challenger proceeds as Game 2-0 but aborts the game if it does not correctly guess the test session in Game 2-1.

Game 2-2. We modify Game 2-1 to Game 2-2 by changing the value of $(B_1^*)^{x^* + a^*} (B_2^*)^{\beta^* (x^* + a^*)}$ to $(X_1^* A_3^*)^{b_1^* + \beta^* b_3^*} (X_2^* A_4^*)^{b_2^* + \beta^* b_4^*}$ in the computation process of K_A^*.

Game 2-3. We modify Game 2-2 to Game 2-3 by changing the value of $(C_1^{(i)})^{x^{(i)} + a^*} (C_2^{(i)})^{\beta_A^{(i)} (x^{(i)} + a^*)}$ to $(X_1^{(i)} A_3^*)^{c_1^{(i)} + \beta_A^{(i)} c_3^{(i)}} (X_2^{(i)} A_4^*)^{c_2^{(i)} + \beta_A^{(i)} c_4^{(i)}}$ in the computation process of $K_A^{(i)}$, where $(c_1^{(i)}, c_2^{(i)}, c_3^{(i)}, c_4^{(i)})$ is the static secret key of $(C_1^{(i)}, C_2^{(i)})$.

Game 2-4. We modify Game 2-3 to Game 2-4 by changing DH tuple $(G, g_1, g_2, A_3^*, A_4^*) \xleftarrow{\mathrm{U}} D(k)$ to random tuple $(G, g_1, g_2, A_3^*, A_4^*) \xleftarrow{\mathrm{U}} R(k)$.

Game 2-5. We proceed as Game 2-4 but abort the game if M_2 establishes session $\mathrm{SID}_B^{(i)}$ such that $H_1(s^*) = H_1(s_B^{(i)})$ or $H_2(s^*) = H_2(s_B^{(i)})$.

Game 2-5. We modify Game 2-5 to Game 2-6 by changing the π PRF function F to a random function RF for test query Test(SID^*).

We evaluate the relations between pairs of advantages.

Claim 4. We have

$$\mathrm{Adv}_{2-0} \le n(k)^2 q_a(k) \cdot \mathrm{Adv}_{2-1}. \tag{19}$$

Proof. This transformation is the same as that in Game 1-1 in the security proof for Case C4. Then we have $\mathrm{Adv}_{2-0} \le n(k)^2 q_a(k) \cdot \mathrm{Adv}_{2-1}$.

Claim 5. We have

$$\mathrm{Adv}_{2-1} = \mathrm{Adv}_{2-2}. \tag{20}$$

Proof. It is clear that this change is purely conceptual, hence $\mathrm{Adv}_{2-1} = \mathrm{Adv}_{2-2}$.

Claim 6. We have

$$\mathrm{Adv}_{2-2} = \mathrm{Adv}_{2-3} + \varepsilon(k) \tag{21}$$

where $\varepsilon(k)$ is negligible in k.

Proof. When M_2 establishes party $\hat{C}^{(i)}$, M_2 is required to prove the knowledge of $(c_1^{(i)}, c_2^{(i)}, c_3^{(i)}, c_4^{(i)})$ in the public key certification process. Due to the conditions of the proof of knowledge (Section 3), the simulator (challenger) can obtain $(c_1^{(i)}, c_2^{(i)}, c_3^{(i)}, c_4^{(i)})$ with overwhelming probability.

Since this change is purely conceptual, we obtain $\mathrm{Adv}_{2-2} = \mathrm{Adv}_{2-3} + \varepsilon(k)$ where $\varepsilon(k)$ is negligible in k.

Claim 7. There exists a probabilistic polynomial-time algorithm S_1 such that

$$\left|\mathrm{Adv}_{2-3} - \mathrm{Adv}_{2-4}\right| \le \mathrm{Adv}_{S_1}^{\mathrm{DDH}}(k). \tag{22}$$

Proof. If the adversary distinguishes Game 2-4 from Game 2-3 with non-negligible probability, we can construct algorithm S_1 that solves the DDH problem.

For a given DDH instance $\rho := (G, u, v, w, z)$, where $\rho \xleftarrow{\mathrm{U}} D(k)$ or $\rho \xleftarrow{\mathrm{U}} R(k)$, S_1 sets $g_1 := u$, $g_2 := v$ and chooses all parameters as Game 2-2 except A_3^* and A_4^*. S_1 sets $A_3^* := w$ and $A_4^* := z$. S_1 proceeds with the game and outputs 1 iff M_2 correctly guesses b'.

If $\rho \xleftarrow{\mathrm{U}} D(k)$, the advantage of S_1 in this simulation is equivalent to that in Game 2-3. If $\rho \xleftarrow{\mathrm{U}} R(k)$, the advantage of S_1 in this simulation is equivalent to that in Game 2-4. Therefore, we have $\left|\mathrm{Adv}_{2-3} - \mathrm{Adv}_{2-4}\right| \le \mathrm{Adv}_{S_1}^{\mathrm{DDH}}(k)$.

Claim 8. There exists probabilistic polynomial-time algorithm S_2 such that

$$\left|\mathrm{Adv}_{2-4} - \mathrm{Adv}_{2-5}\right| \le \mathrm{Adv}_{\mathrm{H},S_2}^{\mathrm{CR}}(k). \tag{23}$$

Proof. Here we can assume that the matching session of test session SID^* does not exists. Hence we have $s^* \neq s_B^{(i)}$ for any i. Then, if the collision event does not occur, Game 2-5 is equivalent to Game 2-4. When the event does occur, we can easily construct algorithm S_2 that breaks the CR hash function by outputting $(s^*, s_B^{(i)})$. Hence we obtain $\left|\mathrm{Adv}_{2-4} - \mathrm{Adv}_{2-5}\right| \le \mathrm{Adv}_{\mathrm{H},S_2}^{\mathrm{CR}}(k)$.

Claim 9. There exists a probabilistic polynomial-time algorithm S_3 such that

$$\left|\mathrm{Adv}_{2-5} - \mathrm{Adv}_{2-6}\right| \le \mathrm{Adv}_{\mathrm{F},S_3}^{\pi\mathrm{PRF}}(k) + 4/q. \tag{24}$$

Proof. To prove Claim 9, we consider two cases for each session $\mathrm{SID}_B^{(i)}$.

Case (i). $(G, g_1, g_2, W_1^{(i)} D_3^{(i)}, W_2^{(i)} D_4^{(i)}) \in D(k)$. That is, there exists $w^{(i)} \in Z_q$ such that $W_1^{(i)} D_3^{(i)} = g_1^{w^{(i)}}$ and $W_2^{(i)} D_4^{(i)} = g_2^{w^{(i)}}$.

Case (ii). $(G, g_1, g_2, W_1^{(i)} D_3^{(i)}, W_2^{(i)} D_4^{(i)}) \notin D(k)$.

The probability that $(G, g_1, g_2, W_1^{(i)}D_3^{(i)}, W_2^{(i)}D_4^{(i)}) \notin D(k)$ and $g_1 \neq 1$, $g_2 \neq 1$, $g_1 \neq g_2$ is at least $1-4/q$ because these are uniformly selected from $R(k)$.

Then, $(B_1^*, B_2^*, K_A^*, K_B^{(i)})$ are denoted by the following equations:

$$\log_{g_1} B_1^* \equiv b_1^* + \eta b_2^* \pmod{q} \tag{25}$$

$$\log_{g_1} B_2^* \equiv b_3^* + \eta b_4^* \pmod{q} \tag{26}$$

$$\log_{g_1} K_A^* \equiv (x^* + a^*)(b_1^* + \beta^* b_3^*) + \eta(x^* + a_0^*)(b_2^* + \beta^* b_4^*) + \delta \pmod{q} \tag{27}$$

$$\log_{g_1} K_B^{(i)} \equiv w^{(i)}(b_1^* + \beta_B^{(i)} b_3^*) + \eta w_0^{(i)}(b_2^* + \beta_B^{(i)} b_4^*) + \gamma \pmod{q} \tag{28}$$

where $g_2 = g_1^{\eta}$, $A_3^* = g_1^{a^*}$, $A_4^* = g_2^{a_0^*}$, $W_1^{(i)}D_3^{(i)} = g_1^{w^{(i)}}$, $W_2^{(i)}D_4^{(i)} = g_2^{w_0^{(i)}}$, $g_1^{\delta} = (Y_1^* B_3^*)^{a_1^* + \alpha^* a_3^*}(Y_2^* B_4^*)^{a_2^* + \alpha^* a_4^*}(Y_3^*)^{x_3^*}$ and $g_1^{\gamma} = (D_1^{(i)})^{y^{(i)}+b^*}(D_2^{(i)})^{a_B^{(i)}(y^{(i)}+b^*)}(W_3^{(i)})^{y_3^{(i)}}$.

If Case (i) occurs, $K_B^{(i)}$ is independent from K_A^* for any $i = 1, \ldots, q_a(k)$ since

$$\log_{g_1} K_B^{(i)} - \gamma \equiv w^{(i)}(b_1^* + \beta_B^{(i)} b_3^*) + \eta w^{(i)}(b_2^* + \beta_B^{(i)} b_4^*) \pmod{q} \tag{29}$$

is linearly dependent on $\log_{g_1} B_1^*$ and $\log_{g_1} B_2^*$, while $\log_{g_1} K_A^*$ is linearly independent from $\log_{g_1} B_1^*$ and $\log_{g_1} B_2^*$. On the other hand, if Case (ii) occurs, we can obtain 4×4 matrix

$$\begin{pmatrix} \log_{g_1} B_1^* \\ \log_{g_1} B_2^* \\ \log_{g_1} K_A^* - \delta \\ \log_{g_1} K_B^{(i)} - \gamma \end{pmatrix} \equiv \begin{pmatrix} 1 & \eta & 0 & 0 \\ 0 & 0 & 1 & \eta \\ x^* + a^* & \eta(x^* + a_0^*) & \beta^*(x^* + a^*) & \eta\beta^*(x^* + a_0^*) \\ w^{(i)} & \eta w_0^{(i)} & \beta_B^{(i)} w^{(i)} & \eta\beta_B^{(i)} w_0^{(i)} \end{pmatrix} \begin{pmatrix} b_1^* \\ b_2^* \\ b_3^* \\ b_4^* \end{pmatrix} \pmod{q} \tag{30}$$

from Eqs. (17)-(20). This 4×4 matrix is regular if

$$\eta^2(a_0^* - a^*)(w_0^{(i)} - w^{(i)})(\beta^* - \beta_B^{(i)}) \equiv 0 \pmod{q} \tag{31}$$

does not hold, so K_A^* is independent from $K_B^{(i)}$ since $a_0^* \neq a^*$, $w_0^* \neq w^{(i)}$ and $\beta^* \neq \beta_B^{(i)}$ hold with probability at least $1-4/q$.

Now, we construct algorithm S_3 that breaks π PRF function F with index $\{\mathrm{I}_G, f_G\}_{G \in \{G\}_k k \in N}$, where $I_G := \{(U,V,d) \mid (U,V,d) \in G^2 \times Z_q\}$ and $f_G : (U,V,d) \to U^{r_1 + dr_2}V$ with $(r_1, r_2) \xleftarrow{\mathrm{U}} Z_q^2$ if the adversary distinguishes Games 2-5 and 2-6 with non-negligible probability. S_3 selects all parameters including $\eta \xleftarrow{\mathrm{U}} Z_q$ such that $g_2 = g_1^{\eta}$, and sets

$$\theta_1 := b_1^* + \eta b_2^*, \quad \theta_2 := b_3^* + \eta b_4^*,$$

$$U^* := A_4^* / (A_3^*)^{\eta}, \quad U_i := W_2^{(i)} D_4^{(i)} / (W_1^{(i)} D_3^{(i)})^{\eta},$$

$$V^* := (Y_1^* B_3^*)^{a_1^* + \alpha^* a_3^*}(Y_2^* B_4^*)^{a_2^* + \alpha^* a_4^*}(g_1^{x^*} A_3^*)^{\theta_1 + \beta^* \theta_2}(Y_3^*)^{x_3^*}, \text{ and}$$

$$V_i := (D_1^{(i)})^{y^{(i)}+b^*}(D_2^{(i)})^{a_B^{(i)}(y^{(i)}+b^*)}(W_1^{(i)} D_3^{(i)})^{\theta_1 + \beta_B^{(i)}\theta_2}(W_3^{(i)})^{y_3^{(i)}}.$$

Algorithm S_3 sets $(r_1, r_2) := (b_2^*, b_4^*)$, and applies the index, $I_G := \{(U,V,d) \mid (U,V,d) \in G^2 \times Z_q\}$ and $f_G : (U,V,d) \to U^{r_1 + dr_2}V$. Then, $\sigma_{(U^*,V^*,\beta^*)} = K_A^*$ and $\sigma_{(U_i,V_i,\beta_B^{(i)})} = K_B^{(i)}$ for $i = 1, \ldots, q_a(k)$. Then S_3 sends (U^*, V^*, β^*) and $(U_i, V_i, \beta_B^{(i)})$ $(i = 1, \ldots, q_a(k))$ to the oracle (F, I_G) or RF. When M_2 outputs guess b', S_3 outputs 1 iff $b' = b$ holds.

If the oracle is (F, I_G), the simulated game is equivalent to Game 2-5. If the oracle is RF, the simulated game is equivalent to Game 2-6. Therefore, we obtain $\left|\mathrm{Adv}_{2-5} - \mathrm{Adv}_{2-6}\right| \le \mathrm{Adv}^{\pi\mathrm{PRF}}_{\mathrm{F},S_3}(k) + 4/q$.

It is clear that $\mathrm{Adv}_{2-6} = 0$, and we obtain

$$\mathrm{Adv}^{\mathrm{eCK}}_{M_2}(k) \le n^2(k) q_a(k) \left(\mathrm{Adv}^{\mathrm{DDH}}_{S_1}(k) + \mathrm{Adv}^{\mathrm{CR}}_{\mathrm{H},S_2}(k) + \mathrm{Adv}^{\pi\mathrm{PRF}}_{\mathrm{F},S_3}(k) + 4/q + \varepsilon(k)\right). \quad (32)$$

6. Performance

In **Table 1**, we compare the efficiency and security of the proposed protocol with several existing eCK-secure AKE protocols, CMQV [5], Sarr et. al. [12], Okamoto [6], and Moriyama-Okamoto [13]. In the table, `exps.' denotes exponentiations in $G$, `SK' denotes secret key, `PK' denotes public key, and `ROM' denotes the random oracle model.

Table 1. Comparison with Existing eCK-secure AKE Protocols

	[5]	[12]	[6]	[13]	Proposed
Static PK	1 element	1 element	2 elements	6 elements	4 elements
Static SK	1 element	1 element	4 elements	9 elements	5 elements
Ephemeral PK	1 element	1 element	3 elements	3 elements	3 elements
Ephemeral SK	1 element	1 element	2 elements	2 elements	2 elements
Computational complexity	2.0 exps.	2.0 exps.	3.7 exps.	5.0 exps.	3.7 exps.
Implementation trick	NAXOS	-	NAXOS	-	-
Assumptions	GDH	GDH	DDH,CR, π PRF	DDH,CR, π PRF	DDH,CR, π PRF
Random orcle	Yes	Yes	No	No	No

In evaluating of the computational complexity, we take into account the standard binary method and simultaneous multiple exponentiation algorithm to compute an exponentiation with multiple bases. An exponentiation with ℓ bases costs about 2^{ℓ} multiplications for precomputation and $q(2-1/2^{\ell})$ multiplications on average. When we compute $X_1 := g_1^x$ and $X_3 := g_1^{x_3}$, it requires less than two exponentiations. Since they can share base g_1, k square operations $g_1^2, (g_1^2)^2, \ldots, g^{2^k}$ are also shared in the standard binary method where $k = |x| = |x_3|$. Therefore, the computational costs for X_1 and X_3 are equivalent to that of around 1.3 ($\approx 2k/1.5k$) exponentiations in G.

Here we ignore the cost for the test of whether or not an element is in G, since an elliptic curve (ECC) implementation for the underlying group is the most efficient in practice and the cofactor in the ECC case is usually very small (or nothing).

7. Conclusion

This paper presented an efficient eCK-secure key exchange protocol without random oracles that does not rely on the NAXOS trick. The proposed protocol is faster than that in [13] and

each party can compute the common session key as efficient as that in [6], although the protocol in [6] relies on the NAXOS trick.

References

[1] Mihir Bellare and Phillip Rogaway, "Entity authentication and key distribution," in *Proc. of Advances in Cryptology – CRYPTO*, pp.232-249, 1993. Article (CrossRefLink).

[2] Ran Canetti and Hugo Krawczyk, "Analysis of key-exchange protocols and their use for building secure channels," in *Proc. of Advances in Cryptology – EUROCRYPTO*, pp.453-474, 2001. Article (CrossRefLink).

[3] Mihir Bellare, David. Pointcheval and Phillip Rogaway, "Authenticated key exchange secure against dictionary attacks, " in *Proc. of Advances in Cryptology – EUROCRYPTO*, pp.139-155, 2000. Article (CrossRefLink)

[4] Brian LaMacchia, Kristin Lauter and Anton. Mityagin, "Stronger security of authenticated key exchange," in *Proc. of 1st Int. Conference on Provable Security*, pp.1-16, 2007. Article (CrossRefLink).

[5] Berkant Ustaoglu, "Obtaining a secure and efficient key agreement protocol from HMQV and NAXOS," in *Designs, Codes and Cryptography*, vol. 46, no. 3, pp. 329-342, 2008. Article (CrossRefLink).

[6] Tatsuaki Okamoto, "Authenticated key exchange and key encapsulation without random oracles," in *Cryptology ePrint Archive*, Report 2007/473, 2007. Article (CrossRefLink).

[7] Jooyoung Lee and Je Hong Park, "Authenticated key exchange secure under the computational Diffie-Hellman assumption," in *Cryptology ePrint Archive*, Report 2008/344, 2008. Article (CrossRefLink).

[8] Jooyoung Lee and Choon Sik Park, "An efficient authenticated key exchange protocol with a tight security reduction," in *Cryptology ePrint Archive*, Report 2008/345, 2008. Article (CrossRefLink).

[9] Jiang Wu and Berkant Ustaoglu, "Efficient key exchange with tight security reduction," in *Cryptology ePrint Archive*, Report 2009/288, 2009. Article (CrossRefLink).

[10] Berkant Ustaoglu, "Comparing SessionStateReveal and EphemeralKeyReveal for Diffie-Hellman protocols," in *Proc. of 3rd Int. Conference on Provable Security*, pp. 183-197, Springer, Heidelberg, 2009. Article (CrossRefLink).

[11] Minkyu Kim, Atsushi Fujioka and Berkant Ustaoglu, "Strongly secure authenticated key exchange without NAXOS approach," in Proc. of International Workshop on Security, pp. 174-191, 2009. Article (CrossRefLink).

[12] Augustin P. Sarr, Philippe Elbaz-Vincent and Jean-Claude Bajard, "A secure and efficient authenticated Diffie-Hellman protocol," in *Proc. of EUROPKI 2009*, pp. 83-998, 2009. Article (CrossRefLink).

[13] Daisuke Moriyama and Tatsuaki Okamoto, "An eCK-secure authenticated key exchange protocol without random oracles," in *Proc. of 3rd Int. Conference on Provable Security*, pp.154-167, 2009. Article (CrossRefLink).

[14] Hugo Krawczyk, "HMQV: A high-performance secure Diffie-Hellman protocol," in *Proc. of Advances in Cryptology – CRYPTO*, pp. 546-566, 2005. Article (CrossRefLink).

[15] Ran Canetti, Oded Goldreich and Shai Halevi, "The random oracle model revisited," in *Proc. of the 13th Annual ACM Symposium on the Theory of Computing*, pp. 209-218, 1998. Article (CrossRefLink).

Daisuke Moriyama was born in 1983. He received the B.E. and M.E. degrees in information technology from Chuo University, in 2006 and 2008, respectively. He is a doctorial candidate at the Graduate School of Information Security, Institute of Information Security. His main research interests include information security and cryptography.

Tatsuaki Okamoto was born in 1952. He received the B.E., M.E., and Dr. E. degrees from the University of Tokyo, Tokyo, Japan, in 1976, 1978, and 1988, respectively. He is a Fellow of NTT, Nippon Telegraph and Telephone Corporation. He is presently engaged in research on cryptography and information security. Dr. Okamoto is a guest professor of Kyoto University. His main research interests include information security and cryptography.