Authenticated Key Exchange Protocol in the Standard Model under Weaker Assumptions

Janaka Alawatugoda[⋆] Taechan Kim[‡]

[⋆]*Department of Computer Engineering, University of Peradeniya, Peradeniya 20400, Sri Lanka.*
alawatugoda@eng.pdn.ac.lk

[‡]*Seoul, South Korea*
tckim1458@gmail.com

Abstract

A two-party authenticated key exchange (AKE) protocol allows each of the two parties to share a common secret key over insecure channels even in the presence of active adversaries who can actively control and modify the exchanged messages. To capture the various kind of malicious behaviors of the adversaries, there have been lots of efforts to define the security models. Amongst them, the extended Canetti-Krawczyk (eCK) security model is considered as one of the strongest ones and widely adopted.

In this paper, we present a pairing-based eCK-secure AKE protocol in the standard model. The underlying assumptions of our construction are the hardness of the decisional bilinear Diffie-Hellman (DBDH) problem and the existence of pseudorandom functions. It is notable that the previous constructions either relied their security on random oracles or used somewhat strong assumptions such as the existence of strong-pseudorandom functions. We believe our construction is well-suited for real-world implementations such as the TLS protocol suite since our construction is simple and based on standard assumptions without random oracles.

Keywords: Authenticated key exchange eCK model Standard Model Pairing Weaker assumptions

1 Introduction

A two-party key exchange protocol has been a fundamental building block of cryptography and network security. It allows any two parties to share a common session key over an insecure channel. Since its early introduction in 1976, the Diffie-Hellman key exchange protocol [DH76] has been the most famous key exchange protocol. However, as is well known, the Diffie-Hellman protocol is insecure against the man-in-the-middle attack, where an adversary impersonates one party to the other to read and modify the exchanged message between two parties. This vulnerability is possible since the parties are not authenticated in the Diffie-Hellman protocol.

To capture such vulnerabilities, including the man-in-the-middle attack, there have been many attempts [BR93, BR95, Can01, LLM07] to define security models for key exchange protocols in the presence of active adversaries who can actively read and modify the exchanged messages. Amongst several security models, the extended Canetti-Krawczyk (eCK) model proposed by LaMacchia, Lauter, and Mityagin [LLM07] is considered as one of the strongest security models, since it captures various possible behaviors of an active adversary. For instance, the properties captured by the eCK model include the following:

- **Implicit Key Authentication:** If a key exchange protocol provides a guarantee that no party apart from the protocol participants can compute the session key, the key exchange protocol is said to provide *implicit key authentication*. If a key exchange protocol provides implicit key authentication it is said to be an *authenticated key exchange (AKE)* protocol.

Janaka Alawatugoda acknowledges the NTT Corporation, Japan for offering him the opportunity to pursue his internship at the NTT Musashino R& D Center in April 2015. Moreover, the authors are grateful to Tatsuaki Okamoto for his valuable comments.

- **Key Confirmation:** If a key exchange protocol provides a guarantee that each party is assured that all other participants possess the same session key, the key exchange protocol is said to provide *key confirmation*.
- **Known Key Security:** The knowledge of a session key should not allow the adversary to learn the session keys in other sessions; all session keys should not depend on the session keys of the other sessions.
- **Security against Unknown Key Share (UKS) Attacks:** Party A should not share a session key with party B, believing that it is sharing the session key with party C. The public keys and identities of the parties should be certified and confirmed or incorporated into protocol execution.
- **Security against Key Compromise Impersonation (KCI) Attacks:** Knowledge of the long-term secret key of party A should not enable the adversary to impersonate the other honest parties to the party A.
- **(weak) Forward Secrecy:** A (passive) adversary who knows the long-term secret keys of any two parties should not be able to compute the past session keys of the two parties.

Since the proposal of the eCK security model, many eCK-secure AKE protocols have been presented [LLM07, KFU09, MO09, Ust08, Yan13, ASB15]. However, some of them [LLM07, KFU09, Ust08, ASB15] are constructed to be secure under the ideal-world assumption of the random oracle model (ROM), and the others are constructed to be secure in the standard model but based on somewhat strong hardness assumptions such as the existence of strong-pseudorandom functions [MO09] or randomness extractor functions [Yan13].

Our Contribution. In this paper, we construct an eCK-secure AKE protocol based on pairings. Our construction is proven to be secure without the ROM assumption, and the only assumptions are the existence of pseudorandom functions and the hardness of the decisional bilinear Diffie-Hellman (DBDH) problem. We remark that we use fewer and more standard assumptions compared to the previous works. As a result, we believe our construction is well-suited for real-world implementations such as the TLS protocol suite.

To help the reader's understanding, we provide a comparison of our protocol with several existing eCK-secure AKE protocols in Table 1. In general, the AKE protocols in the standard model require more computational costs compared to those in the ROM. Nevertheless, it is remarkable that, among the AKE protocols in the standard model, our protocol outperforms Yang's P1 protocol [Yan13] and is comparable to Yang's GC-KKN protocol [Yan13] and MO protocol [MO11]. Also, note that our protocol uses a weaker assumption (the existence of the pseudorandom functions) than those protocols where they rely on the assumption of the existence of either the target collision-resistant function or the strong pseudorandom function.

Protocol	**Proof Model**	**Hardness Assumptions**	**Overall Computational Cost** *at a protocol principal*
NAXOS [LLM07]	ROM	GDH	4E
CMQV [Ust08]	ROM	GDH	3E
MO [MO09]	Standard	DDH, CR, πPRF	3E, 2CR, 1ME, 1πPRF
KFU P1 [KFU09]	ROM	GDH	3E
KFU P2 [KFU09]	ROM	CDH	5E
Yang P1[Yan13]	Standard	DBDH, PRF, TCR	2E, 4ME, 4Pair, 2TCR, 1PRF
Yang GC-KKN[Yan13]	Standard	DDH, TCR, PRF, FAC, EXT	7E, 2ME, 2TCR, 3PRF
ASB [ASB15]	ROM	GDH	6Exp
EC-P1 (this paper)	Standard	DBDH, PRF	3E, 1ME, 2Pair, 1PRF

Key: ROM – random oracle model; GDH – gap DH; CDH – computational DH; DDH – decisional DH; DBDH – decisional bilinear DH; PRF – pseudorandom function; πPRF – strong-pseudorandom function; CR – collision resistant function; TCR – target collision resistant function; FAC – factorization; EXT – randomness extractor function; E – exponentiation operation; ME – multi-exponentiation operation; Pair – pairing operation

Table 1: Basic characteristics of few eCK-secure AKE protocols

2 Preliminaries

Now we recall the preliminaries that we use in our protocol construction.

2.1 Pseudorandom Function

We describe the security definition of pseudorandom functions [KL07].

Definition 2.1 (Pseudorandom Function)**.** *Let $F : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ be an efficient, length preserving, keyed function. We say F is a pseudorandom function, if for all probabilistic polynomial-time adversaries $\mathcal{A}$, there exists a negligible function ϵ_{PRF} in the security parameter k such that,*

$$\left| \Pr[\mathcal{A}^{F(key,\cdot)}(1^k) = 1] - \Pr[\mathcal{A}^{f_{rnd}(\cdot)}(1^k) = 1] \right| \leq \epsilon_{\mathrm{PRF}},$$

where $key \in \{0,1\}^k$ is chosen uniformly at random an f_{rnd} is chosen uniformly at random from the set of functions mapping k-bit strings to k-bit strings.

2.2 Decisional Bilinear Diffie-Hellman Assumption

We describe the decisional bilinear Diffie-Hellman assumption (DBDH) assumption [BF03].

Definition 2.2 (Decisional Bilinear Diffie-Hellman assumption (DBDH))**.** *Let k be the security parameter and $\mathcal{G}$ be a group generation algorithm. Let $(\mathbb{G}, \mathbb{G}_T, q, e) \leftarrow \mathcal{G}(1^k)$, where q is a prime number, the description of two groups $\mathbb{G}, \mathbb{G}_T$ of order q, and the description of an admissible bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Let g, g_1 be two arbitrary generators of $\mathbb{G}$.*

The decisional bilinear Diffie-Hellman (DBDH) problem in $(\mathbb{G}, \mathbb{G}_T, q, e)$ is as follows: Consider two distributions $(g, g_1, g^a, g^b, e(g, g_1)^{ab})$ and (g, g_1, g^a, g^b, T) for some $a, b \in \mathbb{Z}_q$, and random $T \in \mathbb{G}_T$. It is said that decisional BDH assumption holds in $(\mathbb{G}, \mathbb{G}_T, q, e)$, if for all probabilistic polynomial-time algorithms $\mathcal{A}$, the advantage in distinguishing the two distributions is given as,

$$\mathrm{Adv}^{\mathrm{DBDH}}_{\mathbb{G},\mathbb{G}_T,q,e}(\mathcal{A}) = \left| \Pr\left[\mathcal{A}\left(g, g_1, g^a, g^b, e(g, g_1)^{ab}\right) = 1\right] - \Pr\left[\mathcal{A}\left(g, g_1, g^a, g^b, T\right) = 1\right] \right|$$

is negligible for a given security parameter k.

3 Extended Canetti-Krawczyk Model (eCK)

The motivation of LaMacchia et al. [LLM07] in designing the eCK model was that an adversary should have to compromise both the long-term and ephemeral secret keys of a party to recover the session key.

Parties and Long-term Keys. Let $\mathcal{U} = \{U_1, \ldots, U_{N_P}\}$ be a set of N_P parties. Each party U_i where $i \in [1, N_P]$ has a pair of long-term public and secret keys, (pk_{U_i}, sk_{U_i}). Each party U_i owns at most N_S number of protocol sessions.

Sessions. Each party may run multiple instances of the protocol concurrently or sequentially; we use the term *principal* to refer a party involved in a protocol instance, and the term *session* to identify a protocol instance at a principal. The notation $\Pi^s_{U,V}$ represents the s^{th} session at the owner principal U, with intended partner principal V. The principal which sends the first protocol message of a session is the *initiator* of the session, and the principal which responds to the first protocol message is the *responder* of the session. A session $\Pi^s_{U,V}$ enters an *accepted* state when it computes a session key. Note that a session may terminate without ever entering into the accepted state. The information of whether a session has terminated with or without acceptance is public.

Partnering. Legitimate execution of a key exchange protocol between two principals U and V makes two partnering sessions owned by U and V respectively. Two sessions $\Pi^s_{U,V}$ and $\Pi^{s'}_{U',V'}$ are said to be partners if all of the following hold:

1. both $\Pi^s_{U,V}$ and $\Pi^{s'}_{U',V'}$ have computed session keys;
2. messages sent from $\Pi^s_{U,V}$ and messages received by $\Pi^{s'}_{U',V'}$ are identical;

3. messages sent from $\Pi^{s'}_{U',V'}$ and messages received by $\Pi^{s}_{U,V}$ are identical;
4. $U' = V$ and $V' = U$;
5. Exactly one of U and V is the initiator and the other is the responder.

The protocol is said to be *correct* if two partner sessions compute identical session keys.

Adversarial Powers. The adversary $\mathcal{A}$ is a probabilistic polynomial time algorithm in the security parameter k, that has the control over the whole network. $\mathcal{A}$ interacts with set of sessions which represent protocol instances. $\mathcal{A}$ can adaptively ask following queries.

- `Send` (U, V, s, m) query- This query allows $\mathcal{A}$ to run the protocol. It sends the message m to the session $\prod^{s}_{U,V}$ as coming from the session $\prod^{s'}_{V,U}$. $\prod^{s}_{U,V}$ will return to $\mathcal{A}$ the next message according to the protocol conversation so far or decision on whether to accept or reject the session. $\mathcal{A}$ can also use this query to initiate a new protocol instance with blank m. This query captures capabilities of active adversary, who can initiate sessions and modify or delay protocol messages.
- `SessionKeyReveal` (U, V, s) query- If a session $\prod^{s}_{U,V}$ has accepted and holds a session key, $\mathcal{A}$ gets the session key of $\prod^{s}_{U,V}$. A session can only accept a session key once. This query captures the known key attacks.
- `EphemeralKeyReveal` (U, V, s) query- Gives all the ephemeral keys (per session randomness) of the session $\prod^{s}_{U,V}$ to $\mathcal{A}$.
- `Corrupt` (U) query- $\mathcal{A}$ gets all the long-term secrets of the principal U. Then $\mathcal{A}$ may set up long-term secrets at principal U at will. But this query does not reveal any session keys to $\mathcal{A}$. This query captures the KCI attacks, UKS attacks and (weak) forward secrecy
- `Test` (U, s) query- Once a session $\prod^{s}_{U,V}$ has accepted and holds a session key, $\mathcal{A}$ can attempt to distinguish it from a random key. When $\mathcal{A}$ asks the `Test` query, the session $\prod^{s}_{U,V}$ first chooses a random bit $b \in \{0, 1\}$ and if $b = 1$, the actual session key is returned to $\mathcal{A}$, otherwise a random session key is chosen uniformly at random from the same session key distribution, and is returned to $\mathcal{A}$. This query is only allowed to be asked once.

Freshness. A session $\prod^{s}_{U,V}$ is said to be *fresh* if and only if all of the following hold:

1. The session $\prod^{s}_{U,V}$ and its partner (if it exists), $\prod^{s'}_{V,U}$ have not been asked the `Session- Key reveal` query.
2. If partner $\prod^{s'}_{V,U}$ exists none of the following combinations have been asked:
   (a) `Corrupt`(U) and `EphemeralKeyReveal`(U, V, s)
   (b) `Corrupt`(V) and `EphemeralKeyReveal`(V, U, s')
3. If partner $\prod^{s'}_{V,U}$ does not exist none of the following combinations have been asked
   (a) `Corrupt`(V)
   (b) `Corrupt`(U) and `EphemeralKeyReveal`(U, V, s)

Security Game.

- **Stage 0:** The challenger generates the keys by using the security parameter k.
- **Stage 1:** $\mathcal{A}$ is executed and may ask any of `Send`, `SessionKeyReveal`, `EphemeralKeyReveal`, `Corrupt` queries to any session at will.
- **Stage 2:** At some point $\mathcal{A}$ chooses a fresh session and asks the `Test` query.
- **Stage 3:** $\mathcal{A}$ continue asking `Send`, `SessionKeyReveal`, `EphemeralKeyReveal`,`Corrupt` queries. The only condition is that $\mathcal{A}$ cannot violate the freshness of the test session.
- **Stage 4:** At some point $\mathcal{A}$ outputs the bit $b' \in \{0, 1\}$ which is its guess of the value b on the test session. $\mathcal{A}$ wins if $b' = b$.

Definition of Security. Let $\text{Succ}_{\mathcal{A}}$ be the event that the adversary $\mathcal{A}$ wins the eCK game.

Definition 3.1. *A protocol (π) is said to be secure in the eCK model if there is no probabilistic polynomial-time adversary $\mathcal{A}$ who can win the eCK game with non-negligible advantage in the security parameter k. The advantage of an adversary $\mathcal{A}$ is defined as:*

$$\text{Adv}_{\pi}^{eCK}(\mathcal{A}) = |2Pr(Succ_{\mathcal{A}}) - 1| \ .$$

4 Construction of the Pairing-based eCK-secure AKE Protocol

We present a pairing-based construction of an eCK-secure AKE protocol, namely protocol EC-P1. Security of the protocol EC-P1 is proven in the standard model, assuming the hardness of the decisional bilinear Diffie-Hellman (DBDH) problem, and the existence of pseudorandom functions.

4.1 Construction Details

The protocol EC-P1 shown in Table 2 is a Diffie-Hellman-style [DH76] key exchange protocol. Let k be the security parameter and $\mathcal{G}$ be a group generation algorithm. Let $(\mathbb{G}, \mathbb{G}_T, q, e) \leftarrow \mathcal{G}(1^k)$, where q is a prime number, the description of two groups $\mathbb{G}, \mathbb{G}_T$ of order q, and the description of an admissible bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. Let g, g_1 be arbitrary generators of $\mathbb{G}$ such that $g_1 = g^{\alpha}$ for arbitrary $\alpha \in \mathbb{Z}_q$.

Let (a_1, a_2) and (b_1, b_2) be the long-term secret keys of Alice and Bob respectively, whereas A and B be the long-term public keys of Alice and Bob respectively. Let x, X and y, Y be the ephemeral secret and public keys of Alice and Bob respectively for the current session. The execution of the protocol EC-P1 is clearly illustrated in Table 2.

Alice (Initiator)	$\mathbb{G}, \mathbb{G}_T, q, e, g, g_1 \leftarrow \mathcal{G}(1^k)$	*Bob* (Responder)
	Initial Setup	
$a_1, a_2 \leftarrow \mathbb{Z}_q, A \leftarrow g^{a_1} g^{a_2}$		$b_1, b_2 \leftarrow \mathbb{Z}_q, B \leftarrow g^{b_1} g^{b_2}$
	Protocol Execution	
$x \leftarrow \mathbb{Z}_q, X \leftarrow g_1^x$		$y \leftarrow \mathbb{Z}_q, Y \leftarrow g_1^y$
$W_1 \leftarrow e(g^{a_2}, X)$		$W_2 \leftarrow e(g^{b_2}, Y)$
	$\xrightarrow{Alice, W_1, X}$	
	$\xleftarrow{Bob, W_2, Y}$	
$Z_1 \leftarrow \left(\frac{e(Y,B)}{W_2}\right)^{a_1 x}$		$Z_2 \leftarrow \left(\frac{e(X,A)}{W_1}\right)^{b_1 y}$
$K \leftarrow \text{PRF}(Z_1, Alice\|\|W_1\|\|X\|\|Bob\|\|W_2\|\|Y)$		$K \leftarrow \text{PRF}(Z_2, Alice\|\|W_1\|\|X\|\|Bob\|\|W_2\|\|Y)$
	K is the session key	

Table 2: Protocol EC-P1

4.2 Security Analysis of the Protocol EC-P1

Theorem 4.1. *Let k be the security parameter and $\mathcal{G}$ be a group generation algorithm. Let $(\mathbb{G}, \mathbb{G}_T, q, e) \leftarrow \mathcal{G}(1^k)$, where q is a prime number, the description of two groups $\mathbb{G}, \mathbb{G}_T$ of order q, and the description of an admissible bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. Let g, g_1 be arbitrary generators of $\mathbb{G}$ such that $g_1 = g^{\alpha}$, where $\alpha \in \mathbb{Z}_q$. If the* DBDH *assumption holds in $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ and the function* PRF *is a pseudorandom function, then the protocol* EC-P1 *is secure in the* eCK *model.*

Let $\mathcal{U} = \{U_1, \ldots, U_{N_P}\}$ be a set of N_P parties. Each party U_i owns at most N_s number of protocol sessions. Let $\mathcal{A}$ be any adversary against the eck challenger of the protocol EC-P1*. Then, the advantage of $\mathcal{A}$ against the* eCK *security challenge of the protocol* EC-P1*, $\text{Adv}_{\text{EC-P1}}^{\text{eCK}}$ is:*

$$\text{Adv}_{\text{EC-P1}}^{\text{eCK}}(\mathcal{A}) \leq N_P^2 {N_s}^2 \left(\text{Adv}_{\mathbb{G}, \mathbb{G}_T, q, e}^{\text{DBDH}}(\mathcal{C}) + \epsilon_{\text{PRF}} \right) \ .$$

where $\mathcal{C}$ is the algorithm against a DBDH *challenger.*

Proof. We split the proof of Theorem 4.1 into two main cases: when the partner to the test session exists, and when it does not.

1. A partner to the test session exists.
   (a) Adversary corrupts both the owner and the partner principals to the test session - Case **1a**
   (b) Adversary corrupts neither the owner nor the partner principal to the test session - Case **1b**
   (c) Adversary corrupts the owner to the test session, but does not corrupt the partner to the test session - Case **1c**
   (d) Adversary corrupts the partner to the test session, but does not corrupt the owner to the test session - Case **1d**
2. A partner to the test session does not exist: the adversary is not allowed to corrupt the peer to the target session.
   (a) Adversary corrupts the owner to the test session - Case **2a**
   (b) Adversary does not corrupt the owner to the test session - Case **2b**

We show that the advantage of the adversary $\mathcal{A}$ in each of the above cases is negligible.

Case 1a: Adversary corrupts both the owner and partner principals to the test session.

Game 1: This is the original game. When `Test` query is asked the game 1 challenger will choose a random bit $b \leftarrow \{0,1\}$. If $b = 1$, the real session key is given to $\mathcal{A}$, otherwise a random value chosen from the same session-key space is given. Hence,

$$\mathrm{Adv}_{\text{Game 1}}(\mathcal{A}) = \mathrm{Adv}^{\text{eCK}}_{\text{EC-P1,Case 1a}}(\mathcal{A}) \ . \tag{1}$$

Game 2: Same as game 1 with the following exception: Before $\mathcal{A}$ begins, two distinct random principals $U^*, V^* \leftarrow \{U_1, ..., U_{N_P}\}$ are chosen and two random numbers $s^*, t^* \leftarrow \{1, ...N_s\}$ are chosen, where N_P is the number of protocol principals and N_s is the number of sessions on a principal. The session $\Pi^{s^*}_{U^*,V^*}$ is chosen as the target session and the session $\Pi^{t^*}_{V^*,U^*}$ is chosen as the partner to the target session. If the test session is not the session $\Pi^{s^*}_{U^*,V^*}$ or partner to the session is not $\Pi^{t^*}_{V^*,U^*}$, the game 2 challenger aborts the game. Unless the incorrect choice happens, the game 2 is identical to the game 1. Hence,

$$\mathrm{Adv}_{\text{Game 2}}(\mathcal{A}) = \frac{1}{{N_P}^2 N_s^2}\mathrm{Adv}_{\text{Game 1}}(\mathcal{A}) \ . \tag{2}$$

Game 3: Same as game 2 with the following exception: The game 3 challenger randomly chooses $\delta \leftarrow \mathbb{Z}_q$ and computes K according to the protocol description, using $Z_1 = \left(e(g, g_1)^{\delta}\right)^{a_1 b_1}$. When the adversary asks the $\texttt{Test}(U^*, V^*, s^*)$ query, game 3 challenger will answer with K.

We construct an algorithm $\mathcal{C}$ against a DBDH challenger, using the adversary $\mathcal{A}$ as a subroutine. The game 3 challenger sets all the long-term secret/public key pairs of the protocol principals. The algorithm $\mathcal{C}$ runs a copy of $\mathcal{A}$ and interacts with $\mathcal{A}$, such that $\mathcal{A}$ is interacting with either game 2 or game 3. The DBDH challenger sends values $(g, g_1, g_1^{\beta}, g_1^{\gamma}, e(g, g_1)^{\delta})$ such that either $\delta = \beta\gamma$ or $\delta \leftarrow \mathbb{Z}_q$, as the inputs to the algorithm $\mathcal{C}$. The game 3 challenger uses g and g_1 as the generators for the protocol setup. Moreover, the game 3 challenger sets the value X of the target session $(\Pi^{s^*}_{U^*,V^*})$ as g_1^{β}, the value Y of the target session $(\Pi^{s^*}_{U^*,V^*})$ as g_1^{γ}, and computes $W_1 = e(g^{\beta}, g)^{a_2}$ and $W_2 = e(g^{\gamma}, g)^{b_2}$. Upon receiving the $\texttt{Test}(U^*, V^*, s^*)$ query, the game 3 challenger computes the K using $\left(e(g, g_1)^{\delta}\right)^{a_1 b_1}$ and answers. The game 3 challenger can answer all the other queries normally.

If $\mathcal{C}$'s input satisfies $\delta = \beta\gamma$, simulation constructed by the game 3 challenger is identical to game 2, otherwise it is identical to game 3. If $\mathcal{A}$ can distinguish the difference between games, then $\mathcal{C}$ can answer the DBDH challenge. Hence,

$$|\mathrm{Adv}_{\text{Game 2}}(\mathcal{A}) - \mathrm{Adv}_{\text{Game 3}}(\mathcal{A})| \leq \mathrm{Adv}^{\text{DBDH}}_{\mathbb{G},\mathbb{G}_T,q,e}(\mathcal{C}) \ . \tag{3}$$

Game 4: Same as game 3 with the following exception: the game 4 challenger randomly chooses $K \leftarrow \{0,1\}^k$ and sends it to the adversary $\mathcal{A}$ as the answer to the $\texttt{Test}(U^*, V^*, s^*)$ query.

The game 4 challenger sets all the long-term secret/public key pairs and all the encryption key pairs of the protocol principals. Therefore, the challenger can answer all the queries normally.

If K is computed using the real pseudorandom function with a hidden key, the simulation is identical to game 3, whereas if K is chosen randomly from the session key space, the simulation constructed is identical to game 4. Hence,

$$|\mathrm{Adv}_{\text{Game 3}}(\mathcal{A}) - \mathrm{Adv}_{\text{Game 4}}(\mathcal{A})| \leq \epsilon_{\mathrm{PRF}} \ . \tag{4}$$

Semantic security of the session key in Game 4: Since the session key K of $\Pi^{s^*}_{U^*,V^*}$ is chosen randomly and independently from all other values, $\mathcal{A}$ does not have any advantage in game 4. Hence,

$$\mathrm{Adv}_{\text{Game 4}}(\mathcal{A}) = 0 \ . \tag{5}$$

Using equations (1)–(5) we find,

$$\mathrm{Adv}^{\mathrm{eCK}}_{\text{EC-P1,Case 1a}}(\mathcal{A}) \leq {N_P}^2 {N_s}^2 \Big(\mathrm{Adv}^{\mathrm{DBDH}}_{\mathbb{G},\mathbb{G}_T,q,e}(\mathcal{C}) + \epsilon_{\mathrm{PRF}} \Big).$$

Case 1b: Adversary corrupts neither the owner nor the partner principals to the test session.

Game 1: This is the original game. When $\texttt{Test}$ query is asked the game 1 challenger will choose a random bit $b \leftarrow \{0,1\}$. If $b = 1$, the real session key is given to $\mathcal{A}$, otherwise a random value chosen from the same session-key space is given. Hence,

$$\mathrm{Adv}_{\text{Game 1}}(\mathcal{A}) = \mathrm{Adv}^{\mathrm{eCK}}_{\text{EC-P1,Case 1b}}(\mathcal{A}). \tag{6}$$

Game 2: Same as game 1 with the following exception: Before $\mathcal{A}$ begins, two distinct random principals $U^*, V^* \leftarrow \{U_1, ..., U_{N_P}\}$ are chosen and two random numbers $s^*, t^* \leftarrow \{1, ...N_s\}$ are chosen, where N_P is the number of protocol principals and N_s is the number of sessions on a principal. The session $\Pi^{s^*}_{U^*,V^*}$ is chosen as the target session and the session $\Pi^{t^*}_{V^*,U^*}$ is chosen as the partner to the target session. If the test session is not the session $\Pi^{s^*}_{U^*,V^*}$ or partner to the session is not $\Pi^{t^*}_{V^*,U^*}$, the game 2 challenger aborts the game. Unless the incorrect choice happens, the game 2 is identical to the game 1. Hence,

$$\mathrm{Adv}_{\text{Game 2}}(\mathcal{A}) = \frac{1}{{N_P}^2 N_s^2} \mathrm{Adv}_{\text{Game 1}}(\mathcal{A}). \tag{7}$$

Game 3: Same as game 2 with the following exception: The game 3 challenger randomly chooses $\delta \leftarrow \mathbb{Z}_q$ and computes K according to the protocol description, using $Z_1 = \big(e(g, g_1)^{\delta}\big)^{xy}$. When the adversary asks the $\texttt{Test}(U^*, V^*, s^*)$ query, game 3 challenger will answer with K.

We construct an algorithm $\mathcal{C}$ against a DBDH challenger, using the adversary $\mathcal{A}$ as a subroutine. The game 3 challenger sets all the long-term secret/public key pairs of the protocol principals except for the principals U^* and V^*. The algorithm $\mathcal{C}$ runs a copy of $\mathcal{A}$ and interacts with $\mathcal{A}$, such that $\mathcal{A}$ is interacting with either game 2 or game 3. The DBDH challenger sends values $(g, g_1, g^{\beta}, g^{\gamma}, e(g, g_1)^{\delta})$ such that either $\delta = \beta\gamma$ or $\delta \leftarrow \mathbb{Z}_q$, as the inputs to the algorithm $\mathcal{C}$. The game 3 challenger uses g and g_1 as the generators for the protocol setup. For the principal U^*, the long-term public key is computed as $g^{\beta} g^{a_2}$, and for the principal V^*, the long-term public key is computed as $g^{\gamma} g^{b_2}$. The game 3 challenger computes $W_1 = e(g, g_1)^{a_2 x}$ and $W_2 = e(g, g_1)^{b_2 y}$. Upon receiving the $\texttt{Test}(U^*, V^*, s^*)$ query, the game 3 challenger computes the K using $\big(e(g, g_1)^{\delta}\big)^{xy}$ and answers. The game 3 challenger can answer all the other queries normally.

If $\mathcal{C}$'s input satisfies $\delta = \beta\gamma$, simulation constructed by the game 3 challenger is identical to game 2, otherwise it is identical to game 3. If $\mathcal{A}$ can distinguish the difference between games, then $\mathcal{C}$ can answer the DBDH challenge. Hence,

$$|\mathrm{Adv}_{\text{Game 2}}(\mathcal{A}) - \mathrm{Adv}_{\text{Game 3}}(\mathcal{A})| \leq \mathrm{Adv}^{\mathrm{DBDH}}_{\mathbb{G},\mathbb{G}_T,q,e}(\mathcal{C}) \ . \tag{8}$$

Game 4: Same as game 3 with the following exception: the game 4 challenger randomly chooses $K \leftarrow \{0,1\}^k$ and sends it to the adversary $\mathcal{A}$ as the answer to the $\texttt{Test}(U^*, V^*, s^*)$ query.

The game 4 challenger sets all the long-term secret/public key pairs and all the encryption key pairs of the protocol principals, as in the previous game. Therefore, the challenger can answer all the queries normally.

If K is computed using the real pseudorandom function with a hidden key, the simulation is identical to game 3, whereas if K is chosen randomly from the session key space, the simulation constructed is identical to game 4. Hence,

$$|\mathrm{Adv}_{\text{Game 3}}(\mathcal{A}) - \mathrm{Adv}_{\text{Game 4}}(\mathcal{A})| \leq \epsilon_{\text{PRF}} \ . \tag{9}$$

Semantic security of the session key in Game 4: Since the session key K of $\Pi^{s^*}_{U^*,V^*}$ is chosen randomly and independently from all other values, $\mathcal{A}$ does not have any advantage in game 4. Hence,

$$\mathrm{Adv}_{\text{Game 4}}(\mathcal{A}) = 0. \tag{10}$$

Using equations (6)–(10) we find,

$$\mathrm{Adv}^{\text{eCK}}_{\text{EC-P1,Case 1b}}(\mathcal{A}) \leq {N_P}^2 {N_s}^2 \Big(\mathrm{Adv}^{\text{DBDH}}_{\mathbb{G},\mathbb{G}_T,q,e}(\mathcal{C}) + \epsilon_{\text{PRF}} \Big).$$

Case 1c: Adversary corrupts the owner to the test session, but does not corrupt the partner.

Game 1: This is the original game. When $\texttt{Test}$ query is asked the game 1 challenger will choose a random bit $b \leftarrow \{0,1\}$. If $b = 1$, the real session key is given to $\mathcal{A}$, otherwise a random value chosen from the same session-key space is given. Hence,

$$\mathrm{Adv}_{\text{Game 1}}(\mathcal{A}) = \mathrm{Adv}^{\text{eCK}}_{\text{EC-P1,Case 1c}}(\mathcal{A}). \tag{11}$$

Game 2: Same as game 1 with the following exception: before $\mathcal{A}$ begins, two distinct random principals $U^*, V^* \leftarrow \{U_1, ..., U_{N_P}\}$ are chosen and two random numbers $s^*, t^* \leftarrow \{1, ...N_s\}$ are chosen, where N_P is the number of protocol principals and N_s is the number of sessions on a principal. The session $\Pi^{s^*}_{U^*,V^*}$ is chosen as the target session and the session $\Pi^{t^*}_{V^*,U^*}$ is chosen as the partner to the target session. If the test session is not the session $\Pi^{s^*}_{U^*,V^*}$ or partner to the session is not $\Pi^{t^*}_{V^*,U^*}$, the game 2 challenger aborts the game. Unless the incorrect choice happens, game 2 is identical to game 1. Hence,

$$\mathrm{Adv}_{\text{Game 2}}(\mathcal{A}) = \frac{1}{{N_P}^2 N_s^2} \mathrm{Adv}_{\text{Game 1}}(\mathcal{A}). \tag{12}$$

Game 3: Same as game 2 with the following exception: the game 3 challenger randomly chooses $\delta \leftarrow \mathbb{Z}_q$ and computes K according to the protocol description, using $Z_1 = \big(e(g, g_1)^{\delta}\big)^{a_1 y}$. When the adversary asks the $\texttt{Test}(U^*, V^*, s^*)$ query, game 3 challenger will answer with K.

We construct an algorithm $\mathcal{C}$ against a DBDH challenger, using the adversary $\mathcal{A}$ as a subroutine. The game 3 challenger sets all the long-term secret/public key pairs of the protocol principals except for the principals V^*. The algorithm $\mathcal{C}$ runs a copy of $\mathcal{A}$ and interacts with $\mathcal{A}$, such that $\mathcal{A}$ is interacting with either game 2 or game 3. The DBDH challenger sends values $(g, g_1, g^{\beta}, g^{\gamma}, e(g, g_1)^{\delta})$ such that either $\delta = \beta\gamma$ or $\delta \leftarrow \mathbb{Z}_q$, as the inputs to the algorithm $\mathcal{C}$. The game 3 challenger uses g and g_1 as the generators for the protocol setup. For the principal V^*, the long-term public key is computed as $g^{\beta} g^{b_2}$. Moreover, the game 3 challenger sets the value X of the target session $(\Pi^{s^*}_{U^*,V^*})$ as g^{γ}, and computes $W_1 = e(g^{\gamma}, g)^{a_2}$ and $W_2 = e(g, g_1)^{b_2 y}$. Upon receiving the $\texttt{Test}(U^*, V^*, s^*)$ query, the game 3 challenger computes the K using $\big(e(g, g_1)^{\delta}\big)^{a_1 y}$ and answers. The game 3 challenger can answer all the other queries normally.

If $\mathcal{C}$'s input satisfies $\delta = \beta\gamma$, simulation constructed by the game 3 challenger is identical to game 2, otherwise it is identical to game 3. If $\mathcal{A}$ can distinguish the difference between games, then $\mathcal{C}$ can answer the DBDH challenge. Hence,

$$|\mathrm{Adv}_{\text{Game 2}}(\mathcal{A}) - \mathrm{Adv}_{\text{Game 3}}(\mathcal{A})| \mathrm{Adv}^{\text{DBDH}}_{\mathbb{G},\mathbb{G}_T,q,e}(\mathcal{C}) \ . \tag{13}$$

Game 4: Same as game 3 with the following exception: the game 4 challenger randomly chooses $K \leftarrow \{0,1\}^k$ and sends it to the adversary $\mathcal{A}$ as the answer to the $\texttt{Test}(U^*, V^*, s^*)$ query.

The game 4 challenger sets all the long-term secret/public key pairs and all the encryption key pairs of the protocol principals, as in the previous game. Therefore, the challenger can answer all the queries normally.

If K is computed using the real pseudorandom function with a hidden key, the simulation is identical to game 3, whereas if K is chosen randomly from the session key space, the simulation constructed is identical to game 4. Hence,

$$|\mathrm{Adv}_{\text{Game 3}}(\mathcal{A}) - \mathrm{Adv}_{\text{Game 4}}(\mathcal{A})| \leq \epsilon_{\mathrm{PRF}} \ . \tag{14}$$

Semantic security of the session key in Game 4: Since the session key K of $\Pi^{s^*}_{U^*,V^*}$ is chosen randomly and independently from all other values, $\mathcal{A}$ does not have any advantage in game 4. Hence,

$$\mathrm{Adv}_{\text{Game 4}}(\mathcal{A}) = 0. \tag{15}$$

Using equations (11)–(15) we find,

$$\mathrm{Adv}^{\text{eCK}}_{\text{EC-P1,Case 1c}}(\mathcal{A}) \leq N_P^2 {N_s}^2 \Big(\mathrm{Adv}^{\text{DBDH}}_{\mathbb{G},\mathbb{G}_T,q,e}(\mathcal{C}) + \epsilon_{\mathrm{PRF}} \Big).$$

Case 1d: Adversary corrupts the partner to the test session, but does not corrupt the owner.

The analysis of this case is similar the analysis of case 1c. The only difference at the game 3. We briefly explain the simulation of game 3 as follows:

We construct an algorithm $\mathcal{C}$ against a DBDH challenger, using the adversary $\mathcal{A}$ as a subroutine. The game 3 challenger sets all the long-term secret/public key pairs of the protocol principals except for the principals U^*. The algorithm $\mathcal{C}$ runs a copy of $\mathcal{A}$ and interacts with $\mathcal{A}$, such that $\mathcal{A}$ is interacting with either game 2 or game 3. The DBDH challenger sends values $(g, g_1, g^\beta, g^\gamma, e(g,g_1)^\delta)$ such that either $\delta = \beta\gamma$ or $\delta \leftarrow \mathbb{Z}_q$, as the inputs to the algorithm $\mathcal{C}$. The game 3 challenger uses g and g_1 as the generators for the protocol setup. For the principal U^*, the long-term public key is computed as $g^\beta g^{a_2}$. Moreover, the game 3 challenger sets the value Y of the target session $(\Pi^{s^*}_{U^*,V^*})$ as g^γ, and computes $W_1 = e(g,g_1)^{a_2 x}$ and $W_2 = e(g^\gamma, g)^{b_2}$. Upon receiving the $\texttt{Test}(U^*, V^*, s^*)$ query, the game 3 challenger computes the K using $\big(e(g,g_1)^\delta\big)^{b_1 x}$ and answers. The game 3 challenger can answer all the other queries normally.

Apart from the foregoing changes in game 3 simulation, the rest of the simulation of case 1d is the same as case 1c. Therefore, we get,

$$\mathrm{Adv}^{\text{eCK}}_{\text{EC-P1,Case 1d}}(\mathcal{A}) \leq N_P^2 {N_s}^2 \Big(\mathrm{Adv}^{\text{DBDH}}_{\mathbb{G},\mathbb{G}_T,q,e}(\mathcal{C}) + \epsilon_{\mathrm{PRF}} \Big).$$

Case 2a: Adversary corrupts the owner to the test session.

There is no partner existing to the target session. Note that the owner of the target session is U^*. We can further classify this case into two subcases as follows:

- (2a.1) There is no peer session existing to the target session, the adversary computes the protocol message itself as the peer principal.
- (2a.2) There is a peer session existing to the target session, the adversary tricks the peer principal to compute the protocol message and delivers it to the owner principal.

2a.1: There is no peer session existing to the target session, the adversary computes the protocol message itself as the peer principal.

In this case, the peer session is supposed to be at the principal V^*, but the peer session does not exist at V^*. If there is no peer session existing, the adversary $\mathcal{A}$ needs to compute the protocol message as the partner of the target session by itself. In order to compute this message the adversary needs the long-term secret key b_2 of the principal V^*. Even for an unbounded adversary b_2 value is information theoretically hidden, as the corresponding long-term public key B is computed as $g^{b_1} g^{b_2}$. Therefore, the advantage of the adversary in this case is zero. Therefore, we get,

$$\mathrm{Adv}^{\text{eCK}}_{\text{EC-P1,Case 2a.1}}(\mathcal{A}) = 0.$$

2a.2: There is a peer session existing to the target session, the adversary tricks the peer principal to compute the protocol message and delivers it to the owner principal.

In this case, the adversary $\mathcal{A}$ corrupts the owner principal U^*. Then, the adversary picks an ephemeral secret key, computes a protocol message as coming from the owner U^* (or the adversary may also use a previous message computed by the the principal U^*), and sends it to the peer principal V^*. That way the adversary can trick the peer principal to compute a protocol message. Once the peer computes a protocol message as a response to the message sent by the adversary (as came from U^*), the adversary use this message to send to the owner principal U^*, as the message from the peer principal V^*. This message can be used as a responding message, if the principal U^* is the initiator of the target session. Otherwise,it can be used as an initial message if the principal U^* is the responder of the target session. Note that at this case, the adversary does not know the ephemeral secret key, that is picked at the intended peer principal V^*.

We construct an algorithm $\mathcal{C}$ against a DBDH challenger, using the adversary $\mathcal{A}$ as a subroutine. Game hopping simulation of this case is the same as the cases 1c. Thus,

$$\mathrm{Adv}^{\mathrm{eCK}}_{\text{EC-P1,Case 2a.2}}(\mathcal{A}) \leq {N_P^2}{N_s}^2\Big(\mathrm{Adv}^{\mathrm{DBDH}}_{\mathbb{G},\mathbb{G}_T,q,e}(\mathcal{C}) + \epsilon_{\mathrm{PRF}}\Big).$$

Case 2b: Adversary does not corrupt the owner to the test session.

We can further classify this case into two subcases as follows:

- (2b.1) There is no peer session existing to the target session, the adversary computes the protocol message itself as the peer principal.
- (2b.2) There is a peer session existing to the target session, the adversary tricks the peer principal to compute the protocol message and delivers it to the owner principal.

2b.1: There is no peer session existing to the target session, the adversary computes the protocol message itself as the peer principal.

In this case, the peer session is supposed to be at the principal V^*, but the peer session does not exist at V^*. If there is no peer session existing, the adversary $\mathcal{A}$ needs to compute the protocol message as the partner of the target session by itself. In order to compute this message the adversary needs the long-term secret key b_2 of the principal V^*. Even for an unbounded adversary b_2 value is information theoretically hidden, as the corresponding long-term public key B is computed as $g^{b_1}g^{b_2}$. Therefore, the advantage of the adversary in this case is zero. Therefore, we get,

$$\mathrm{Adv}^{\mathrm{eCK}}_{\text{EC-P1,Case 2b.1}}(\mathcal{A}) = 0.$$

2b.2: There is a peer session existing to the target session, the adversary tricks the peer principal to compute the protocol message and delivers it to the owner principal.

In this case, the adversary $\mathcal{A}$ does not corrupt the owner principal U^*. The adversary may use a previous message computed by the the principal U^*, and sends it to the peer principal V^*. That way the adversary can trick the peer principal to compute a protocol message. Once the peer computes a protocol message as a response to the message sent by the adversary (as came from U^*), the adversary use this message to send to the owner principal U^*, as the message from the peer principal V^*. This message can be used as a responding message, if the principal U^* is the initiator of the target session. Otherwise,it can be used as an initial message if the principal U^* is the responder of the target session. Note that at this case, the adversary does not know the ephemeral secret key, that is picked at the intended peer principal V^*.

We construct an algorithm $\mathcal{C}$ against a DBDH challenger, using the adversary $\mathcal{A}$ as a subroutine. Game hopping simulation of this case is the same as the cases 1d (but without allowing to corrupt the peer to the target session). Thus,

$$\mathrm{Adv}^{\mathrm{eCK}}_{\text{EC-P1,Case 2b.2}}(\mathcal{A}) \leq {N_P^2}{N_s}^2\Big(\mathrm{Adv}^{\mathrm{DBDH}}_{\mathbb{G},\mathbb{G}_T,q,e}(\mathcal{C}) + \epsilon_{\mathrm{PRF}}\Big).$$

Combining all the above cases: According to the analysis we can see the adversary $\mathcal{A}$'s advantage of winning against the eCK challenger of the protocol EC-P1 is:

$$\mathrm{Adv}^{\mathrm{eCK}}_{\text{EC-P1}}(\mathcal{A}) \leq N_P^2 {N_s}^2 \Big(\mathrm{Adv}^{\mathrm{DBDH}}_{\mathbb{G},\mathbb{G}_T,q,e}(\mathcal{C}) + \epsilon_{\mathrm{PRF}} \Big) \ .$$

□

4.3 Computational Costs

We provide the overall computational cost of our protocol at a protocol principal (either the initiator or the responder) in Table 3. Note that the costs for light computations such as the multiplication/division are ignored. In total, our protocol costs three exponentiations, one multi-exponentiation, two pairings and one execution of the underlying pseudorandom function.

	Computation *at the initiator or the responder*	**Cost**
Initial setup	A or B	1ME
Protocol execution	X or Y	1E
	W_1 or W_2	1E, 1Pair
	Z_1 or Z_2	1E, 1Pair
	K	1PRF

Key: PRF – pseudorandom function; E – exponentiation operation; ME – multi-exponentiation operation; Pair – pairing operation

Table 3: Overall computational cost at a protocol principal

5 Conclusions and Future Works

Usually, the AKE protocols that are proven to be secure in the standard model require strong hardness assumptions to achieve the eCK security. We construct a standard model eCK-secure AKE protocol based on pairings, only assuming the existence of pseudorandom functions and the hardness of the DBDH problem. We emphasize that we use fewer and more standard assumptions compared to the previous works. Thus, our contribution is a significant improvement in the context of key exchange protocols. As a future work, we can implement this protocol to be used with the TLS protocol suite. Moreover, it is worthwhile to research on quantum-safe and leakage-resilient AKE protocols.

References

[ASB15] Janaka Alawatugoda, Douglas Stebila, and Colin Boyd. Continuous after-the-fact leakage-resilient eck-secure key exchange. In *Cryptography and Coding - 15th IMA International Conference, IMACC 2015, Oxford, UK, December 15-17, 2015. Proceedings*, pages 277–294, 2015.

[BF03] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. *SIAM J. Comput.*, 32(3):586–615, 2003.

[BR93] Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In *CRYPTO*, pages 232–249, 1993.

[BR95] Mihir Bellare and Phillip Rogaway. Provably secure session key distribution - the three party case. pages 57–66. ACM Press, 1995.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136–145, 2001.

[DH76] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, pages 644 – 654, 1976.

[KFU09] Minkyu Kim, Atsushi Fujioka, and Berkant Ustaoglu. Strongly secure authenticated key exchange without naxos' approach. In *Advances in Information and Computer Security, 4th International Workshop on Security, IWSEC 2009, Toyama, Japan, October 28-30, 2009, Proceedings*, pages 174–191, 2009.

[KL07] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography.* Chapman and Hall/CRC Press, 2007.

[LLM07] Brian LaMacchia, Kristin Lauter, and Anton Mityagin. Stronger security of authenticated key exchange. In *ProvSec*, pages 1–16, 2007.

[MO09] Daisuke Moriyama and Tatsuaki Okamoto. An eck-secure authenticated key exchange protocol without random oracles. In *Provable Security, Third International Conference, ProvSec 2009, Guangzhou, China, November 11-13, 2009. Proceedings*, pages 154–167, 2009.

[MO11] Daisuke Moriyama and Tatsuaki Okamoto. Leakage resilient eCK-secure key exchange protocol without random oracles. In *ASIACCS*, pages 441–447, 2011.

[Ust08] Berkant Ustaoglu. Obtaining a secure and efficient key agreement protocol from (H)MQV and NAXOS. *Des. Codes Cryptography*, 46(3):329–342, 2008.

[Yan13] Zheng Yang. Efficient eck-secure authenticated key exchange protocols in the standard model. In *Information and Communications Security - 15th International Conference, ICICS 2013, Beijing, China, November 20-22, 2013. Proceedings*, pages 185–193, 2013.