Exposure-Resilient One-Round Tripartite Key Exchange without Random Oracles

Koutarou Suzuki and Kazuki Yoneyama

NTT Secure Platform Laboratories
3-9-11 Midori-cho Musashino-shi Tokyo 180-8585, Japan
yoneyama.kazuki@lab.ntt.co.jp

Abstract. This paper studies Tripartite Key Exchange (3KE) which is a special case of Group Key Exchange. Though general one-round GKE satisfying advanced security properties such as forward secrecy and maximal-exposure-resilience (MEX-resilience) is not known, it can be efficiently constructed with the help of pairings in the 3KE case. In this paper, we introduce the first one-round 3KE which is MEX-resilient in the standard model, though existing one-round 3KE schemes are proved in the random oracle model (ROM), or not MEX-resilient. Each party broadcasts 4 group elements, and executes 14 pairing operations. Complexity is only three or four times larger in computation and communication than the existing most efficient MEX-resilient 3KE scheme in the ROM; thus, our protocol is adequately practical.

Keywords: authenticated key exchange, tripartite key exchange, standard model, dual-receiver encryption.

1 Introduction

Authenticated Key Exchange (AKE) is a cryptographic primitive to share a common *session key* among multiple parties through unauthenticated networks such as the Internet. This work considers the PKI-based setting that each party locally keeps his own *static secret key* (SSK) and publish a *static public key* (SPK) corresponding to the SSK. Validity of SPKs is guaranteed by a certificate authority. In a key exchange session, each party generates an *ephemeral secret key* (ESK) and sends an *ephemeral public key* (EPK) corresponding to the ESK. A session key is derived from these keys with a *key derivation function*.

In the two-party AKE (2KE) setting, many practical and provably secure AKE protocols have been introduced since [1]. For example, MQV [2] and its variants [3,4,5,6,7,8,9] achieve one-round schemes from the Diffie-Hellman (DH) assumptions. One-round protocols mean that parties send their messages independently and simultaneously only once. On the other hand, in the general group AKE (GKE) setting, to achieve one-round protocols is not so easy. A known approach [10,11] is that each party broadcasts multiple ciphertexts encrypting a common nonce to all parties, and aggregates nonces with a key derivation function. These schemes do not satisfy some important security properties such as forward secrecy. The other approach is using a multilinear map from ideal

M. Jacobson et al. (Eds.): ACNS 2013, LNCS 7954, pp. 458–474, 2013.

lattices [12]. This multilinear map is still a candidate, and the resultant GKE is impractical. Thus, to construct secure and practical one-round GKE is a challenging problem in the research area of AKE.

Interestingly, in the three-party AKE (3KE) setting (i.e., a special case of GKE), secure one-round protocols can be achieved practically thanks to pairings. Joux [13] firstly proposed one-round (unauthenticated) 3KE by extending the ordinary DH protocol to 3KE with pairings. After that, several one-round authenticated 3KE schemes are studied such as [14,15,16]. The security model for GKE in [15,16] (called the MSU model) captures a very strong security property; that is, even though an adversary can reveal any non-trivial[1] combination of ephemeral secret keys and static secret keys, any information of the session key is not exposed. We call such a property *maximal-exposure-resilience* (MEX-resilience). MEX-resilience implies various important security properties for GKE such as forward secrecy and key compromise impersonation resilience. Unfortunately, the known MEX-resilient one-round 3KE schemes [15,16] are proved in the random oracle model (ROM).

1.1 Our Contribution

We achieve the first MEX-resilient one-round 3KE scheme in the standard model (StdM). Our key idea is to utilize *dual-receiver encryption* (DRE) [17,18]. DRE allows a ciphertext to be decrypted into the same plaintext by two independent receivers. The situation of DRE is very similar to 3KE; a party tries to share common secret information with other two parties. Thus, our basic strategy is that each party broadcasts a ciphertext of DRE and aggregates three plaintexts by a pseudo-random function. However, we must carefully consider several special situations of 3KE, which do not occur in encryption; e.g., the simulator must manage decryption of the challenge ciphertext. We modify the original DRE to be able to simulate such situations correctly.

Also, we reformulate the MSU model [15,16] for GKE (called the G-CK$^+$ model) by combining with one of the 'strongest' models for 2KE, CK$^+$ model [3,19]. The G-CK$^+$ model allows adversaries to reveal intermediate computation results of sessions in addition to the MSU model. Such a reveal capability is also considered in several security models for 2KE [20,3,8,19]. We prove that our scheme is secure in the G-CK$^+$ model in the StdM under the decisional bilinear DH (DBDH) assumption.

2 Preliminaries

In this section, we recall definitions of building blocks.

Throughout this paper we use the following notations. If M is a set, then by $m \in_R \mathsf{M}$ we denote that m is sampled uniformly from M. If $\mathcal{R}$ is an algorithm, then by $y \leftarrow \mathcal{R}(x; r)$ we denote that y is output by $\mathcal{R}$ on input x and randomness r (if $\mathcal{R}$ is deterministic, r is empty).

[1] If both the static key and the ephemeral key of a party in the target session are revealed, the adversary trivially obtains the session key for any protocol.

2.1 Bilinear Group

Let G and G_T be cyclic groups of prime order p where g is a generator of G. We say that $e : G \times G \to G_T$ is a bilinear map if for all $X, Y \in G$ and $a, b \in \mathbb{Z}_p$, $e(X^a, Y^b) = e(X, Y)^{ab}$, and $e(g, g) \neq 1$. We say that G is a bilinear group if e, and group operations in G and G_T can be computed efficiently.

2.2 Decisional Bilinear Diffie-Hellman Assumption

Let κ be the security parameter and p be a κ-bit prime. Let G be a bilinear group of a prime order p with a generator g, and G_T be a cyclic group of the prime order p. Let $e : G \times G \to G_T$ be a bilinear map.

The DBDH assumption is defined by two experiments, $\mathsf{Exp}^{\text{dbdh-real}}(\mathcal{D})$ and $\mathsf{Exp}^{\text{dbdh-rand}}(\mathcal{D})$. For a distinguisher $\mathcal{D}$, inputs $(g, \alpha = g^a, \beta = g^b, \gamma = g^c, \delta)$ are provided, where $(a, b, c) \in_R (\mathbb{Z}_p)^3$. $\delta = g_T^{abc}$ in $\mathsf{Exp}^{\text{dbdh-real}}(\mathcal{D})$ and $\delta \in_R G$ in $\mathsf{Exp}^{\text{dbdh-rand}}(\mathcal{D})$ where $g_T = e(g, g)$. We define advantage

$$\mathbf{Adv}^{\text{dbdh}}(\mathcal{D}) = |\Pr[\mathsf{Exp}^{\text{dbdh-real}}_{g,p}(\mathcal{D}) = 1] - \Pr[\mathsf{Exp}^{\text{dbdh-rand}}_{g,p}(\mathcal{D}) = 1]|,$$

where the probability is taken over the choices of a, b, c, δ and the random tape of $\mathcal{D}$.

Definition 1 (DBDH Assumption). *We say that the DBDH assumption in (G, G_T) holds if for all probabilistic polynomial-time (PPT) distinguisher $\mathcal{D}$ the advantage $\mathbf{Adv}^{\text{dbdh}}(\mathcal{D})$ is negligible in security parameter κ.*

2.3 Pseudo-random Function

Let κ be a security parameter and $\mathsf{F} = \{F_\kappa : Dom_\kappa \times Kspace_\kappa \to Rng_\kappa\}_\kappa$ be a function family with a family of domains $\{Dom_\kappa\}_\kappa$, a family of key spaces $\{Kspace_\kappa\}_\kappa$ and a family of ranges $\{Rng_\kappa\}_\kappa$.

Definition 2 (Pseudo-Random Function). *We say that function family $\mathsf{F} = \{F_\kappa\}_\kappa$ is the PRF family, if for any PPT distinguisher $\mathcal{D}$, $\mathbf{Adv}^{\text{prf}} = |\Pr[\mathcal{D}^{F_\kappa(\cdot)} \to 1] - \Pr[\mathcal{D}^{RF_\kappa(\cdot)} \to 1]| \leq negl$, where $RF_\kappa : Dom_\kappa \to Rng_\kappa$ is a truly random function.*

2.4 Target-Collision Resistant Hash Function

We say a function $TCR : Dom \to Rng$ is a target-collision resistant hash function if the following condition holds for a security parameter κ: For any PPT adversary $\mathcal{A}$, $\Pr[x \in_R Dom; x' \leftarrow \mathcal{A}(x) \text{ s.t. } x \neq x' \wedge TCR(x) = TCR(x')] \leq negl$.

3 G-CK$^+$ Model

In this section, we introduce a new security model, G-CK$^+$ model, for GKE by combining the CK$^+$ model [19] for two-party AKE and the MSU model [15,16] for GKE.

Note that we show a model specified to one-round protocols for simplicity. It can be trivially extended to any round protocol.

3.1 Protocol Participants and Initialization

Let $\mathcal{U} := \{U_1, \ldots, U_N\}$ be a set of potential protocol participants. Each party U_i is modeled as a PPT Turing machine w.r.t. security parameter κ. For party U_i, we denote static secret (public) key by SSK_i (SPK_i) and ephemeral secret (public) key by ESK_i (EPK_i). Party U_i generates its own keys, SSK_i and SPK_i, and the static public key SPK_i is linked with U_i's identity in some systems like PKI.

3.2 Session

An invocation of a protocol is called a *session*. We suppose that a session contains n parties $\{U_{j_1}, \ldots, U_{j_n}\}$, where $2 \leq n \leq N$. A session is managed by a tuple $(\Pi, \mathsf{role}_i, U_{j_\ell}, \{U_{j_1}, \ldots, U_{j_n}\})$, where Π is a protocol identifier, role_i is a role identifier, and U_{j_ℓ} is a party identifier. Hereafter, for simplicity, we can suppose that $U_{j_\ell} = U_\ell$ without loss of generality. If U_j is activated with $(\Pi, \mathsf{role}_i, U_j, \{U_1, \ldots, U_n\}, \mathit{Init})$, then U_j is called the *i-th player*. The role of a party in a session is decided by the lexicographic order of party identities, and $\mathsf{role}_i \neq \mathsf{role}_{i'}$ for any i and i' in a session. U_j outputs EPK_j, receives $EPK_{j'}$ from $U_{j'}$ for $j' = 1, \ldots, j-1, j+1, \ldots, n$, and computes the session key SK.

If U_j is the i-th player of a session, the session is identified by $\mathsf{sid} = (\Pi, \mathsf{role}_i, U_j, \{U_1, \ldots, U_n\}, EPK_j)$ or $\mathsf{sid} = (\Pi, \mathsf{role}_i, U_j, \{U_1, \ldots, U_n\}, \{EPK_1, \ldots, EPK_n\})$. We say that U_j is the *owner* of session sid, if the third coordinate of sid is U_j. We say that U_j is a *peer* of session sid, if the third coordinate of sid is not U_j. We say that a session is *completed* if its owner computes the session key. We say $(\Pi, \mathsf{role}_{i'}, U_{j'}, \{U_1, \ldots, U_n\}, \{EPK_1, \ldots, EPK_n\})$ is *matching session* of $(\Pi, \mathsf{role}_i, U_j, \{U_1, \ldots, U_n\}, \{EPK_1, \ldots, EPK_n\})$, where $i' \neq i$ and $j' \neq j$.

3.3 Adversary

The adversary $\mathcal{A}$, which is modeled as a PPT Turing machine, controls all communications between parties including session activation and registrations of parties by performing the following adversary queries.

- $\mathsf{Send}(U_j, \text{message})$: U_j is the receiver. The message has the following form: $(\Pi, \mathsf{role}_i, U_j, \{U_1, \ldots, U_n\}, \mathit{Init})$ for session activation, or $(\Pi, \mathsf{role}_{i'}, U_{j'}, \{U_1, \ldots, U_n\}, EPK_{j'})$. $\mathcal{A}$ obtains the response from U_j.
- $\mathsf{Establish}(U_j, SPK_j)$: This query allows $\mathcal{A}$ to introduce new parties. In response, if $U_j \notin \mathcal{U}$ (due to the uniqueness of identities) then U_j with the static public key SPK_j is added to $\mathcal{U}$. Note that $\mathcal{A}$ is not required to prove the possession of the corresponding secret key SSK_j. If a party is registered by a $\mathsf{Establish}$ query issued by $\mathcal{A}$, then we call the party *dishonest*. If not, we call the party *honest*.

To capture exposure of secret information, the adversary $\mathcal{A}$ is allowed to issue the following queries.

- SessionReveal(sid): The adversary $\mathcal{A}$ obtains the session key SK for the session sid if the session is completed.
- StateReveal(sid): The adversary $\mathcal{A}$ obtains the session state of the owner of session sid if the session is not completed (the session key is not established yet). The session state includes all ephemeral secret keys and intermediate computation results except for immediately erased information but does not include the static secret key. Note that the protocol specifies what the session state contains.
- StaticReveal(U_j): This query allows the adversary $\mathcal{A}$ to obtain all static secret keys of the party U_j.
- EphemeralReveal(sid): This query allows the adversary $\mathcal{A}$ to obtain all ephemeral secret keys of the owner of the session sid if the session is not completed (the session key is not established yet). It is necessary to represent a MEX situation that an adversary can reveal ESKs but is prevented to obtain other session state such that the adversary trivially wins.

3.4 Freshness

For the security definition, we need the notion of freshness.

Definition 3 (Freshness). *Let* $\mathsf{sid}^* = (\Pi, \mathsf{role}_i, U_j, \{U_1, \ldots, U_n\}, \{EPK_1, \ldots, EPK_n\})$ *be a completed session between honest parties* $\{U_1, \ldots, U_n\}$*, which is owned by* U_j*. If a matching session exists, then let* $\overline{\mathsf{sid}^*}$ *be a matching session of* sid^**. We say session* sid^* *is* fresh *if none of the following conditions hold:*

1. *The adversary* $\mathcal{A}$ *issues* $\mathsf{SessionReveal}(\mathsf{sid}^*)$*, or* $\mathsf{SessionReveal}(\overline{\mathsf{sid}^*})$ *for any* $\overline{\mathsf{sid}^*}$ *if* $\overline{\mathsf{sid}^*}$ *exists,*
2. $\overline{\mathsf{sid}^*}$ *exists, and adversary* $\mathcal{A}$ *makes either of* $\mathsf{StateReveal}(\mathsf{sid}^*)$ *or* $\mathsf{StateReveal}(\overline{\mathsf{sid}^*})$*,*
3. $\overline{\mathsf{sid}^*}$ *does not exist, and adversary* $\mathcal{A}$ *makes* $\mathsf{StateReveal}(\mathsf{sid}^*)$*,*
4. *adversary* $\mathcal{A}$ *makes both of* $\mathsf{StaticReveal}(U_j)$ *and* $\mathsf{EphemeralReveal}(\mathsf{sid}^*)$*,*
5. $\overline{\mathsf{sid}^*}$ *exists (the owner of* $\overline{\mathsf{sid}^*}$ *is* $U_{j'}$*), and adversary* $\mathcal{A}$ *makes both of* $\mathsf{StaticReveal}(U_{j'})$ *and* $\mathsf{EphemeralReveal}(\overline{\mathsf{sid}^*})$*,*
6. $\overline{\mathsf{sid}^*}$ *does not exist, and adversary* $\mathcal{A}$ *makes* $\mathsf{StaticReveal}(U_{j'})$ *for any intended peer* $U_{j'}$ *of* U_j *in* sid^**.*

3.5 Security Experiment

For the security definition, we consider the following security experiment. Initially, the adversary $\mathcal{A}$ is given a set of honest users and makes any sequence of the queries described above. During the experiment, the adversary $\mathcal{A}$ makes the following query.

- Test(sid^*): Here, sid^* must be a fresh session. Select random bit $b \in_R \{0, 1\}$, and return the session key held by sid^* if $b = 0$, and return a random key if $b = 1$.

The experiment continues until the adversary $\mathcal{A}$ makes a guess b'. The adversary $\mathcal{A}$ *wins* the game if the test session sid^* is still fresh and if the guess of the adversary $\mathcal{A}$

is correct, i.e., $b' = b$. The advantage of the adversary $\mathcal{A}$ is defined as $\mathbf{Adv}_{\Pi}^{\mathrm{gke}}(\mathcal{A}) = \Pr[\mathcal{A}\ wins] - \frac{1}{2}$. We define the security as follows.

Definition 4 (G-CK$^+$ **Security).** *We say that a GKE protocol* Π *is* secure in the G-CK$^+$ model *if the following conditions hold:*

1. *If two honest parties complete matching sessions, then, except with negligible probability, they both compute the same session key.*
2. *For any PPT adversary* $\mathcal{A}$, $\mathrm{Adv}_{\Pi}^{\mathrm{gke}}(\mathcal{A})$ *is negligible in security parameter* κ *for the test session* sid^*.

4 Maximal-Exposure-Resilient One-Round Tripartite Key Exchange without ROs

In this section, we introduce a new one-round 3KE protocol. We use the technique of DRE with some modification. Our protocol is G-CK$^+$ secure in the StdM under the DBDH assumption.

4.1 What Is Barrier to Remove ROs?

All of known 3KE protocols in the ROM use an RO as the key derivation function. For example, in the simplest variant of the FMSU framework [16] a session key is the output of an RO as follows: Let κ be a security parameter. Let G and G_T be bilinear groups with pairing $e : G \times G \to G_T$ of order κ-bit prime p with generators g and $g_T = e(g, g)$, respectively. Party U_A, U_B and U_C own $a, b, c \in_R \mathbb{Z}_p$ as SSKs and $A = g^a, B = g^b, C = g^c \in G$ as SPKs, and $x, y, z \in_R \mathbb{Z}_p$ as ESKs and $X = g^x, Y = g^y, Z = g^z \in G$ as EPKs, respectively. Then, parties share 8 combinations of their SSKs and ESKs with pairings (i.e., g_T^{abc}, g_T^{xbc}, g_T^{ayc}, g_T^{abz}, g_T^{xyc}, g_T^{xbz}, g_T^{ayz} and g_T^{xyz}). The session key SK is the output of RO inputting these shared information.

This structure helps the simulation to keep consistency between SessionReveal and Send queries in the security proof. The simulator must answer correct session keys for the SessionReveal query according to EPKs for the Send query. In the ROM, the simulator can arbitrarily chooses the output of the key derivation function without computing shared information. Thus, if the simulator cannot know the ESK corresponding to the EPK received from the Send query, he can make the session key consistent by the simulation of the RO.

Conversely, in the StdM, the simulator must compute all shared information in order to answer the session key to the SessionReveal query correctly. However, MEX includes exposure of all non-trivial combinations of SSKs and ESKs, and the simulator must embed an instance of a hard problem into unexposed keys to solve the problem. For example, we consider the case that the key derivation function in the above FMSU variant is not RO. If a, y and z are revealed, then the simulator must embed DBDH tuple $(\alpha, \beta, \gamma, \delta)$ into $X = \alpha$, $B = \beta$, $C = \gamma$, and $g_T^{xbc} = \delta$. Then, the simulator must return the correct session key for the SessionReveal and Send query though x, b and c are not known. Such a situation is hard to simulate as it is.

4.2 Modifying Dual-Receiver Encryption

A promising approach is to use techniques to simulate decryption queries in chosen ciphertext (CCA) secure encryption. Especially, for the 3KE setting, we need DRE rather than ordinary public key encryption. We use the KEM version of a CCA secure DRE [18] (CFZ DRE) to construct our basic protocol; that is, each party encapsulates a random nonce with DRE and broadcasts it to other parties. The protocol of the CFZ DRE is as follows:

Public Parameters. Let κ be a security parameter. Let G and G_T be bilinear groups with pairing $e : G \times G \to G_T$ of order κ-bit prime p with generators g and $g_T = e(g, g)$, respectively. Let $TCR : G \to \mathbb{Z}_p$ be a target collision resistance hash function.

Secret and Public Keys. The secret key of receiver U_i is $sk_i := (x_i, y_i) \in_R \mathbb{Z}_p^2$. The public key of receiver U_i is $pk_i := (X_i = g^{x_i}, Y_i = g^{y_i})$.

Encapsulation. Given pk_1 and pk_2, the sender chooses $r \in_R \mathbb{Z}_p$, and computes $R = g^r$, $\mathsf{tag} = TCR(R)$, $\pi_1 = (X_1^{\mathsf{tag}} Y_1)^r$, and $\pi_2 = (X_2^{\mathsf{tag}} Y_2)^r$. The KEM session key is $K = e(X_1, X_2)^r$, and the ciphertext is $CT = (R, \pi_1, \pi_2)$.

Decapsulation. Given pk_1, pk_2, sk_1, and CT, the receiver computes $\mathsf{tag} = TCR(R)$, and checks $e(g, \pi_1) = e(R, x_1^{\mathsf{tag}} y_1)$ and $e(g, \pi_2) = e(R, x_2^{\mathsf{tag}} y_2)$. If not, return $\perp$. Otherwise, return the KEM session key $K = e(R, X_2)^{x_1}$.

In the security proof, the simulator can handle decryption queries by utilizing the fact that tag^* corresponding to the challenge ciphertext CT^* is different from tags corresponding to ciphertexts of decryption queries. Specifically, the simulator embeds a DBDH tuple $(g, \alpha, \beta, \gamma, \delta)$ into $R^* = \alpha$, $\pi_1^* = \alpha^{d_1}$, $\pi_2^* = \alpha^{d_2}$, $X_1^* = \beta$, $Y_1^* = \beta^{-\mathsf{tag}^*} g^{d_1}$, $X_2^* = \gamma$, $Y_2^* = \gamma^{-\mathsf{tag}^*} g^{d_2}$ and $K^* = \delta$, where $d_1, d_2 \in_R \mathbb{Z}_p$ and $\mathsf{tag}^* = TCR(R^*)$. When a ciphertext $CT = (R, \pi_1, \pi_2)$ is posed, the simulator can return $K = e((\pi_1 R^{-d_1})^{(\mathsf{tag}-\mathsf{tag}^*)^{-1}}, X_2^*)$ or $K = e((\pi_2 R^{-d_2})^{(\mathsf{tag}-\mathsf{tag}^*)^{-1}}, X_1^*)$, where $\mathsf{tag} = TCR(R)$. Owing to this simulation, we can simulate the SessionReveal and Send query without knowing secret keys. It is likely that 3KE could be constructed by setting the ciphertext of the CFZ DRE as the EPK. However, a simple application of DRE does not correctly work in the 3KE setting.

First, though the adversary is prevented to pose the challenge ciphertext to the decryption oracle in the CCA game of DRE, an adversary can be forward the message, corresponding to the challenge ciphertext, in the test session for other sessions by Send query in the G-CK$^+$ model. For example, an adversary specifies U_A as the owner of the test session, and reveals the SSK of U_A. Then, the simulator embeds an element in the DBDH tuple into R^* of a part of the EPK of U_A as the simulation of the CFZ DRE. The adversary can reuse R^* as a part of the EPK of U_B in another session. In this case, the simulator must manage the decryption of it, but the original decryption simulation technique of DRE does not help him because $\mathsf{tag} = \mathsf{tag}^* = TCR(R^*)$. Thus, our first modification is that the way to generate tags is changed to be different in distinct two sessions even if the challenge ciphertext is reused. Specifically, we make tags dependent on identities of the sender and receivers. That is, if the sender is U_A, and receivers

are U_B and U_C, then $\mathsf{tag} = TCR(R, U_A, U_B, U_C)$. Even if the same R is reused, the tag is different from tag because the sender is not U_A. Note that when R is reused in another session such that the sender is U_A, and receivers are U_B and U_C, the decryption simulation is not necessary because we can return δ in the DBDH tuple.

Next, we must consider the other problem. To resist exposure of all SSKs, parties must share a secret state only with their ESKs to be independent with SSKs as g_T^{xyz} in the simplest variant of the FMSU framework. If each party knows own ESK, such a secret state can be computed. Unfortunately, in the case that the simulator must embed an element in the DBDH tuple into R^* of a part of the EPK of U_A in the test session, the secret state cannot be simulated. If the test session has no matching session, and an adversary reuses R^* as EPKs of U_B and U_C, the simulator must computes the secret state from R^*. However, the simulator cannot generate it because no ESK is known. On the other hand, to resist exposure or adversarial generation of all ESKs, parties must share a secret state only with their SSKs to be independent with ESKs as g_T^{abc} in the simplest variant of the FMSU framework. A similar case as above occurs; that is, the simulator cannot generate such a secret state because no SSK is known when an element in the DBDH tuple is embedded into the SPK of a party and other parties are established by the adversary. To resolve this problem, our second modification is that each party generates an additional group element as a part of SPK, and broadcasts an additional group element as a part of EPK. Secret states corresponding to g_T^{abc} and g_T^{xyz} are generated with them. This modification allows the simulator to know ESKs and SSKs to generate secret states even if the simulator embeds an element in the DBDH tuple into EPK or SPK. In our construction (Section 4.3), EPKs R'_A, R'_B, R'_C and SPKs Z_A, Z_B, Z_C correspond to the modification.

4.3 Our Construction

Public Parameters. Let κ be a security parameter. Let G and G_T be bilinear groups with pairing $e : G \times G \to G_T$ of order κ-bit prime p with generators g and $g_T = e(g, g)$, respectively. Let $F : \{0,1\}^* \times G_T \to \{0,1\}^\kappa$ be a pseudo-random function where the key space for F is G_T. Let $TCR : G \to \mathbb{Z}_p$ be a target collision resistance hash function.

Secret and Public Keys. Party U_I chooses $x_I, y_I, z_I \in_R \mathbb{Z}_p$ as the static secret key. Then, U_I computes $X_I = g^{x_I}$, $Y_I = g^{y_I}$ and $Z_I = g^{z_I}$ as the static public key.

Key Exchange. We suppose a session executed by U_A, U_B and U_C.

1. U_A chooses $r_A, r'_A \in_R \mathbb{Z}_p$ as the ephemeral secret key, and computes $R_A = g^{r_A}$, $R'_A = g^{r'_A}$, $\pi_{AB} = (X_B^{\mathsf{tag}_A} Y_B)^{r_A}$, and $\pi_{AC} = (X_C^{\mathsf{tag}_A} Y_C)^{r_A}$ as the ephemeral public key, where $\mathsf{tag}_A = TCR(R_A, R'_A, U_A, U_B, U_C)$. Then, U_A broadcasts $(\Pi, \mathsf{role}_1, U_A, \{U_A, U_B, U_C\}, (R_A, R'_A, \pi_{AB}, \pi_{AC}))$ to U_B and U_C.
2. U_B chooses $r_B, r'_B \in_R \mathbb{Z}_p$ as the ephemeral secret key, and computes $R_B = g^{r_B}$, $R'_B = g^{r'_B}$, $\pi_{BA} = (X_A^{\mathsf{tag}_B} Y_A)^{r_B}$, and $\pi_{BC} = (X_C^{\mathsf{tag}_B} Y_C)^{r_B}$ as the ephemeral public key, where $\mathsf{tag}_B = TCR(R_B, R'_B, U_B, U_C, U_A)$. Then, U_A broadcasts $(\Pi, \mathsf{role}_2, U_B, \{U_A, U_B, U_C\}, (R_B, R'_B, \pi_{BA}, \pi_{BC}))$ to U_A and U_C.

3. U_C chooses $r_C, r'_C \in_R \mathbb{Z}_p$ as the ephemeral secret key, and computes $R_C = g^{r_C}$, $R'_C = g^{r'_C}$, $\pi_{CA} = (X_A^{\mathsf{tag}_C} Y_A)^{r_C}$, and $\pi_{CB} = (X_B^{\mathsf{tag}_C} Y_B)^{r_C}$ as the ephemeral public key, where $\mathsf{tag}_C = TCR(R_C, R'_C, U_C, U_A, U_B)$. Then, U_C broadcasts $(\Pi, \mathsf{role}_3, U_C, \{U_A, U_B, U_C\}, (R_C, R'_C, \pi_{CA}, \pi_{CB}))$ to U_A and U_B.
4. On receiving $(\Pi, \mathsf{role}_2, U_B, \{U_A, U_B, U_C\}, (R_B, R'_B, \pi_{BA}, \pi_{BC}))$ and $(\Pi, \mathsf{role}_3, U_C, \{U_A, U_B, U_C\}, (R_C, R'_C, \pi_{CA}, \pi_{CB}))$, U_A computes $\mathsf{tag}_B = TCR(R_B, R'_B, U_B, U_C, U_A)$ and $\mathsf{tag}_C = TCR(R_C, R'_C, U_C, U_A, U_B)$, and verify the following equations.

$$e(g, \pi_{BA}) = e(R_B, X_A^{\mathsf{tag}_B} Y_A); e(g, \pi_{BC}) = e(R_B, X_C^{\mathsf{tag}_B} Y_C);$$
$$e(g, \pi_{CA}) = e(R_C, X_A^{\mathsf{tag}_C} Y_A); e(g, \pi_{CB}) = e(R_C, X_B^{\mathsf{tag}_C} Y_B)$$

If the verification does not hold, U_A aborts. Otherwise, U_A computes the following shared information.

$$\sigma_1 = e(Z_B, Z_C)^{z_A};\ \sigma_2 = e(X_B, X_C)^{r_A}; \sigma_3 = e(R_B, X_C)^{x_A};\ \sigma_4 = e(X_B, R_C)^{x_A};$$
$$\sigma_5 = e(R_B, X_C)^{r_A};\ \sigma_6 = e(X_B, R_C)^{r_A}; \sigma_7 = e(R_B, R_C)^{x_A};\ \sigma_8 = e(R'_B, R'_C)^{r'_A}$$

Then, U_A sets the session transcript $\mathsf{ST} = (U_A, (R_A, R'_A, \pi_{AB}, \pi_{AC}), U_B, (R_B, R'_B, \pi_{BA}, \pi_{BC}), U_C, (R_C, R'_C, \pi_{CA}, \pi_{CB}))$. Finally, U_A generates the session key $SK = F_{\sigma_1}(\mathsf{ST}) \oplus \cdots \oplus F_{\sigma_8}(\mathsf{ST})$, and completes the session.
5. On receiving $(\Pi, \mathsf{role}_1, U_A, \{U_A, U_B, U_C\}, (R_A, R'_A, \pi_{AB}, \pi_{AC}))$ and $(\Pi, \mathsf{role}_3, U_C, \{U_A, U_B, U_C\}, (R_C, R'_C, \pi_{CA}, \pi_{CB}))$, U_B computes $\mathsf{tag}_A = TCR(R_A, R'_A, U_A, U_B, U_C)$ and $\mathsf{tag}_C = TCR(R_C, R'_C, U_C, U_A, U_B)$, and verify the following equations.

$$e(g, \pi_{AB}) = e(R_A, X_B^{\mathsf{tag}_A} Y_B); e(g, \pi_{AC}) = e(R_A, X_C^{\mathsf{tag}_A} Y_C);$$
$$e(g, \pi_{CA}) = e(R_C, X_A^{\mathsf{tag}_C} Y_A); e(g, \pi_{CB}) = e(R_C, X_B^{\mathsf{tag}_C} Y_B)$$

If the verification does not hold, U_B aborts. Otherwise, U_B computes the following shared information.

$$\sigma_1 = e(Z_A, Z_C)^{z_B};\ \sigma_2 = e(R_A, X_C)^{x_B}; \sigma_3 = e(X_A, X_C)^{r_B};\ \sigma_4 = e(X_A, R_C)^{x_B};$$
$$\sigma_5 = e(R_A, X_C)^{r_B};\ \sigma_6 = e(R_A, R_C)^{x_B}; \sigma_7 = e(X_A, R_C)^{r_B};\ \sigma_8 = e(R'_A, R'_C)^{r'_B}$$

Then, U_B sets the session transcript $\mathsf{ST} = (U_A, (R_A, R'_A, \pi_{AB}, \pi_{AC}), U_B, (R_B, R'_B, \pi_{BA}, \pi_{BC}), U_C, (R_C, R'_C, \pi_{CA}, \pi_{CB}))$. Finally, U_B generates the session key $SK = F_{\sigma_1}(\mathsf{ST}) \oplus \cdots \oplus F_{\sigma_8}(\mathsf{ST})$, and completes the session.
6. On receiving $(\Pi, \mathsf{role}_1, U_A, \{U_A, U_B, U_C\}, (R_A, R'_A, \pi_{AB}, \pi_{AC}))$ and $(\Pi, \mathsf{role}_2, U_B, \{U_A, U_B, U_C\}, (R_B, R'_B, \pi_{BA}, \pi_{BC}))$, U_C computes $\mathsf{tag}_A = TCR(R_A, R'_A, U_A, U_B, U_C)$ and $\mathsf{tag}_B = TCR(R_B, R'_B, U_B, U_C, U_A)$, and verify the following equations.

$$e(g, \pi_{AB}) = e(R_A, X_B^{\mathsf{tag}_A} Y_B); e(g, \pi_{AC}) = e(R_A, X_C^{\mathsf{tag}_A} Y_C);$$
$$e(g, \pi_{BA}) = e(R_B, X_A^{\mathsf{tag}_B} Y_A); e(g, \pi_{BC}) = e(R_B, X_C^{\mathsf{tag}_B} Y_C)$$

If the verification does not hold, U_C aborts. Otherwise, U_C computes the following shared information.

$$\sigma_1 = e(Z_A, Z_B)^{z_C};\ \sigma_2 = e(R_A, X_B)^{x_C}; \sigma_3 = e(X_A, R_B)^{x_C};\ \sigma_4 = e(X_A, X_B)^{r_C};$$
$$\sigma_5 = e(R_A, R_B)^{x_C};\ \sigma_6 = e(R_A, X_B)^{r_C}; \sigma_7 = e(X_A, R_B)^{r_C};\ \sigma_8 = e(R'_A, R'_B)^{r'_C}$$

Table 1. Comparison of previous one-round 3KE schemes and our scheme

	MEX-resilient?	Resource	Assumption	Computation (#parings+ #[multi,regular]-exp)	Communication complexity	
[10]	no	ROM	DDH	$0 + [0, 7]^{\dagger}$ or $0 + [2, 1]$	$20\kappa^{\dagger}$ or 2κ	(2560 or 256)
[11]	no	StdM	DDH	$0 + [3, 7]$	16κ	(2048)
[14]	no	ROM	BDH	$1 + [0, 3]$	4κ	(512)
[15,16]	yes	ROM	GBDH	$4 + [0, 7]$	4κ	(512)
Ours	yes	StdM	DBDH	$14 + [2, 15]$	16κ	(2048)

† Since the protocol is asymmetric, the cost for a party is higher than the others.

DDH means the Decisional Diffie-Hellman assumption. BDH means the Bilinear Diffie-Hellman assumption. DBDH means the Decisional Bilinear Diffie-Hellman assumption. GBDH means the gap Bilinear Diffie-Hellman assumption. For concreteness the expected communication complexity for a 128-bit security implementation is also given. Note that computational costs are estimated without any pre-computation technique.

Then, U_C sets the session transcript $\mathsf{ST} = (U_A, (R_A, R'_A, \pi_{AB}, \pi_{AC}), U_B, (R_B, R'_B, \pi_{BA}, \pi_{BC}), U_C, (R_C, R'_C, \pi_{CA}, \pi_{CB}))$. Finally, U_C generates the session key $SK = F_{\sigma_1}(\mathsf{ST}) \oplus \cdots \oplus F_{\sigma_8}(\mathsf{ST})$, and completes the session.

The session state of a session owned by U_I contains ephemeral secret keys (r_I, r'_I), shared information $(\sigma_1, \ldots, \sigma_8)$, and outputs of PRFs $(F_{\sigma_1}(\mathsf{ST}), \ldots, F_{\sigma_8}(\mathsf{ST}))$.

4.4 Efficiency

Our construction needs 2 regular exponentiations and 2 multi exponentiations to generate a message, 4 regular exponentiations and 8 pairings to verify received messages, and 8 regular exponentiations and 6 pairings to compute shared information for each party. The total computational cost for each party is 2 multi exponentiations, 15 regular exponentiations, and 14 pairings. A message contains 4 group elements in G, and each party broadcasts the message to two other parties. The total communication complexity (the message size sent by a party) is 16κ with an elliptic curve.

Table 1 summarizes the efficiency comparison of previous one-round 3KE schemes and our scheme. Schemes in [10] and [11] are designed for GKE, and we describe 3KE versions of them. The instantiation of [10] in Table 1 is with the ElGamal KEM as semantically secure public key encryption and the Chevallier-Mames signature [21] as existentially unforgeable signature. The instantiation of [11] in Table 1 is with the multiple Cramer-Shoup encryption [22] as CCA secure multiple KEM according to the generic construction [23].

This table hints that communication complexity grows to achieve security in the standard model, and computational cost grows to achieve MEX-resilient. Hence, our scheme is less efficient than existing schemes but still practical because complexity is only three or four times larger both in computation and communication than [15,16].

Table 2. Classification of events, when A, B and C are distinct

	SSK_A	ESK_A	SSK_B	ESK_B	SSK_C	ESK_C
E_1	r	ok	ok	r/n	ok	r/n
E_2	ok	r	ok	r/n	ok	r/n
E_3	r	ok	r	ok	r	ok
E_4	ok	r	r	ok	r	ok
E_5	r	ok	ok	r/n	r	ok
E_6	ok	r	ok	r/n	r	ok
E_7	r	ok	r	ok	ok	r/n
E_8	ok	r	r	ok	ok	r/n

"ok" means the static secret key is not revealed, or a partnered instance exists and its ephemeral secret key is not revealed. "r" means the static or ephemeral secret key may be revealed. "r/n" means the ephemeral secret key may be revealed if the corresponding partnered instance exists, or no corresponding partnered instance exists.

5 Security

We show the following theorem.

Theorem 1. *If the DBDH assumption holds, and F is a PRF, then our 3KE protocol is* G-CK$^+$*-secure.*

Proof. In the experiment of G-CK$^+$ security, we suppose that sid^* is the session identity for the test session, and that there are N parties and at most ℓ sessions are activated per a party. Let κ be the security parameter, and let $\mathcal{A}$ be a PPT (in κ) adversary. Suc denotes the event that $\mathcal{A}$ wins. We consider eight events in Table 2, that cover all cases of the behavior of $\mathcal{A}$.
To finish the proof, we investigate events $E_i \wedge Suc$ $(i = 1, \ldots, 8)$ that cover all cases of event Suc. Due to the space limitation, we only show the full proof of $E_1 \wedge Suc$ which is the most difficult event. Other events can be proved in a similar way.

5.1 Event $E_1 \wedge Suc$

We change the interface of oracle queries and the computation of the session key. These instances are gradually changed over hybrid experiments, depending on specific subcases. In the last hybrid experiment, the session key in the test session does not contain information of the bit b. Thus, the adversary clearly only output a random guess. We denote these hybrid experiments by $\mathbf{H}_0, \ldots, \mathbf{H}_4$, and the advantage of the adversary $\mathcal{A}$ when participating in experiment $\mathbf{H}_i$ by $\mathbf{Adv}(\mathcal{A}, \mathbf{H}_i)$.

Hybrid Experiment $\mathbf{H}_0$. This experiment denotes the real experiment for G-CK$^+$ security and in this experiment the environment for $\mathcal{A}$ is as defined in the protocol. Thus, $\mathbf{Adv}(\mathcal{A}, \mathbf{H}_0)$ is the same as the advantage of the real experiment.

Hybrid Experiment $\mathbf{H}_1$. In this experiment, if session identities in two sessions are identical, the experiment halts.

When randomness in generating EPKs are identical, session identities in two sessions are also identical. However, such an event occurs with negligible probability. Thus, $|\mathbf{Adv}(\mathcal{A}, \mathbf{H}_1) - \mathbf{Adv}(\mathcal{A}, \mathbf{H}_0)| \leq negl$.

Hybrid Experiment $\mathbf{H}_2$. In this experiment, the experiment selects parties U_A, U_B and U_C, and integer $k \in [1, \ell]$ randomly in advance. If $\mathcal{A}$ poses Test query to a session except k-th session of U_A whose the intended peers U_B and U_C, the experiment halts.

Since guess of the test session matches with $\mathcal{A}$'s choice with probability $1/N^3\ell$, $\mathbf{Adv}(\mathcal{A}, \mathbf{H}_2) \geq (1/N^3\ell) \cdot \mathbf{Adv}(\mathcal{A}, \mathbf{H}_1)$. After this experiment, without loss of generality, we can suppose that the intended peers of the k-th session of U_A are U_B and U_C.

Hybrid Experiment $\mathbf{H}_3$. In this experiment, the computation of σ_2 in the test session is changed. Instead of computing $\sigma_2 = g_T^{r_A x_B x_C}$, it is changed as choosing $\sigma_2 \leftarrow G_T$ randomly, where we suppose that U_B and U_C are intended peers of U_A in the test session.

We construct a DBDH distinguisher $\mathcal{D}$ from $\mathcal{A}$ in $\mathbf{H}_2$ or $\mathbf{H}_3$. $\mathcal{D}$ performs the following steps.

Init. $\mathcal{D}$ receives a DBDH tuple $(g, \alpha, \beta, \gamma, \delta)$ as a challenge.

Setup. $\mathcal{D}$ chooses pseudo-random function $F : \{0,1\}^* \times G_T \rightarrow \{0,1\}^\kappa$, and provides it as a part of the public parameters.

First, $\mathcal{D}$ sets the ephemeral public key $(R_A^*, R'^*_A, \pi_{AB}^*, \pi_{AC}^*)$ of k-th session of U_A. $\mathcal{D}$ randomly chooses r'_A, d_1 and d_2, and sets $R_A^* := \alpha$, $R'^*_A := g^{r'_A}$, $\pi_{AB}^* := \alpha^{d_1}$ and $\pi_{AC}^* := \alpha^{d_2}$.

Next, $\mathcal{D}$ implicitly sets all N parties' static secret and public keys. Keys of parties except U_B and U_C are generated as the protocol. If $\mathcal{A}$ poses Establish query with a party identifier and a SPK, then $\mathcal{D}$ replaces the preset SPK of the party with the given SPK. Static public keys of U_B and U_C ((X_B^*, Y_B^*, Z_B^*) and (X_C^*, Y_C^*, Z_C^*)) are set as $X_B^* := \beta$, $Y_B^* := \beta^{-\mathsf{tag}_A^*} g^{d_1}$, $Z_B^* := g^{z_B^*}$, $X_C^* := \gamma$, $Y_C^* := \gamma^{-\mathsf{tag}_A^*} g^{d_2}$, and $Z_C^* := g^{z_C^*}$, where $\mathsf{tag}_A^* = TCR(R_A^*, R'^*_A, U_A, U_B, U_C)$ and $z_B^*, z_C^* \in_R \mathbb{Z}_p$.

Simulation. $\mathcal{D}$ maintains the list $\mathcal{L}_{SK}$ that contains queries and answers of SessionReveal. $\mathcal{D}$ simulates oracle queries by $\mathcal{A}$ as follows.

1. Send$(U_{j_1}, \Pi, \mathsf{role}_i, U_{j_1}, \{U_{j_1}, U_{j_2}, U_{j_3}\}, Init)$:
   (a) If $j_1 = A$, $j_2 = B$, $j_3 = C$, $i = 1$, the session is k-th session of U_A, then $\mathcal{D}$ returns $(\Pi, \mathsf{role}_1, U_A, \{U_A, U_B, U_C\}, (R_A^*, R'^*_A, \pi_{AB}^*, \pi_{AC}^*))$ and records it.
   (b) Otherwise, $\mathcal{D}$ computes the ephemeral public key as the protocol, returns it and records $(\Pi, \mathsf{role}_i, U_{j_1}, \{U_{j_1}, U_{j_2}, U_{j_3}\}, (R_{j_1}, R'_{j_1}, \pi_{j_1 j_2}, \pi_{j_1 j_3}))$.
2. Send$(U_{j_2}, \Pi, \mathsf{role}_i, U_{j_1}, \{U_{j_1}, U_{j_2}, U_{j_3}\}, EPK_{j_1})$ or Send$(U_{j_3}, \Pi, \mathsf{role}_i, U_{j_1}, \{U_{j_1}, U_{j_2}, U_{j_3}\}, EPK_{j_1})$:
   (a) If both $(\Pi, \mathsf{role}_{i'}, U_{j_2}, \{U_{j_1}, U_{j_2}, U_{j_3}\}, EPK_{j_2})$ and $(\Pi, \mathsf{role}_{i''}, U_{j_3}, \{U_{j_1}, U_{j_2}, U_{j_3}\}, EPK_{j_3})$ are not recorded, $\mathcal{D}$ only records $(\Pi, \mathsf{role}_i, U_{j_1}, \{U_{j_1}, U_{j_2}, U_{j_3}\}, EPK_{j_1})$.

(b) Else if $\mathcal{D}$ checks EPK_{j_1} with pairing equations as the protocol, and the verification is not valid, then $\mathcal{D}$ rejects the session.

(c) Else if ($j_2 = A$, $j_1 = C$) or ($j_3 = A$, $j_1 = B$), and the session is the k-th session of U_A, then $\mathcal{D}$ sets all σ_i as follows: σ_2 is set as δ. Since $\mathcal{D}$ knows x_A, z_A, z_B^*, z_C^* and r'^*_A, σ_1, σ_3, σ_4, σ_7 and σ_8 are computed with these values. $\sigma_5 = e((\pi_{BC}R_B^{-d_2})^{(\mathsf{tag}_B-\mathsf{tag}_{A^*})^{-1}}, R_A^*)$ where $\mathsf{tag}_B = TCR(R_B, R'_B, U_B, U_C, U_A)$[2]. $\sigma_6 = e((\pi_{CB}R_C^{-d_1})^{(\mathsf{tag}_C-\mathsf{tag}_{A^*})^{-1}}, R_A^*)$ where $\mathsf{tag}_C = TCR(R_C, R'_C, U_C, U_A, U_B)$[3]. $\mathcal{D}$ computes the session key SK^* as the protocol, and records $(\Pi, \mathsf{role}_1, U_A, \{U_A, U_B, U_C\}, \{EPK_A, EPK_B, EPK_C\})$, $(\Pi, \mathsf{role}_2, U_B, \{U_A, U_B, U_C\}, \{EPK_A, EPK_B, EPK_C\})$ and $(\Pi, \mathsf{role}_3, U_C, \{U_A, U_B, U_C\}, \{EPK_A, EPK_B, EPK_C\})$ as the completed session and SK^* in the list $\mathcal{L}_{SK}$.

(d) Else if $j_1 = A$, $j_2 = B$, $j_3 = C$, the first content of EPK_{j_1} is R_A^*, then $\mathcal{D}$ sets all σ_i as follows: σ_2 is set as δ. Since $\mathcal{D}$ knows x_A, z_A, z_B^*, z_C^* and r'^*_A, σ_1, σ_3, σ_4, σ_7 and σ_8 are computed with these values. $\sigma_5 = e((\pi_{j_2j_3}R_{j_2}^{-d_2})^{(\mathsf{tag}_{j_2}-\mathsf{tag}_{A^*})^{-1}}, R_A^*)$ where $\mathsf{tag}_{j_2} = TCR(R_{j_2}, R'_{j_2}, U_B, U_C, U_A)$[4]. $\sigma_6 = e((\pi_{j_3j_2}R_{j_3}^{-d_1})^{(\mathsf{tag}_{j_3}-\mathsf{tag}_{A^*})^{-1}}, R_A^*)$ where $\mathsf{tag}_{j_3} = TCR(R_{j_3}, R'_{j_3}, U_C, U_A, U_B)$[5]. $\mathcal{D}$ computes the session key SK as the protocol, and records $(\Pi, \mathsf{role}_1, U_A, \{U_A, U_B, U_C\}, \{EPK_A, EPK_B, EPK_C\})$, $(\Pi, \mathsf{role}_2, U_B, \{U_A, U_B, U_C\}, \{EPK_A, EPK_B, EPK_C\})$ and $(\Pi, \mathsf{role}_3, U_C, \{U_A, U_B, U_C\}, \{EPK_A, EPK_B, EPK_C\})$ as the completed session and SK in the list $\mathcal{L}_{SK}$.

(e) Else if $j_2 = B$ (resp. $j_3 = B$), then $\mathcal{D}$ parses EPK_{j_1} into $(R_{j_1}, R'_{j_1}, \pi_{j_1j_2}, \pi_{j_1j_3})$, and sets σ_i as follows: Since $\mathcal{D}$ knows z_B^*, r_B and r'_B, σ_1, σ_3, σ_4, σ_7 and σ_8 are computed with these values. $\sigma_2 = e((\pi_{j_1j_2}R_{j_1}^{-d_1})^{(\mathsf{tag}_{j_1}-\mathsf{tag}_{A^*})^{-1}}, X_{j_3})$ (resp. $\sigma_2 = e((\pi_{j_1j_3}R_{j_1}^{-d_1})^{(\mathsf{tag}_{j_1}-\mathsf{tag}_{A^*})^{-1}}, X_{j_2})$) where $\mathsf{tag}_{j_1} = TCR(R_{j_1}, R'_{j_1}, U_{j_1}, U_B, U_{j_3})$ (resp. $\mathsf{tag}_{j_1} = TCR(R_{j_1}, R'_{j_1}, U_{j_1}, U_{j_3}, U_B)$). $\sigma_5 = e(R_{j_1}, X_{j_3})^{r_B}$ (resp. $\sigma_5 = e((\pi_{j_1j_3}\cdot R_{j_1}^{-d_1})^{(\mathsf{tag}_{j_1}-\mathsf{tag}_{A^*})^{-1}}, R_{j_3})$) where $\mathsf{tag}_{j_1} = TCR(R_{j_1}, R'_{j_1}, U_{j_1}, U_{j_3}, U_B)$. $\sigma_6 = e((\pi_{j_1j_2}\cdot R_{j_1}^{-d_1})^{(\mathsf{tag}_{j_1}-\mathsf{tag}_{A^*})^{-1}}, R_{j_3})$ (resp. $\sigma_6 = e(R_{j_1}, X_{j_3})^{r_B}$) where $\mathsf{tag}_{j_1} = TCR(R_{j_1}, R'_{j_1}, U_{j_1}, U_B, U_{j_3})$. $\mathcal{D}$ computes the session key SK as the protocol, and records $(\Pi, \mathsf{role}_i, U_{j_1}, \{U_{j_1}, U_B, U_{j_3}\}, \{EPK_{j_1}, EPK_B, EPK_{j_3}\})$, $(\Pi, \mathsf{role}_{i'}, U_B, \{U_{j_1}, U_B, U_{j_3}\}, \{EPK_{j_1}, EPK_B, EPK_{j_3}\})$ and $(\Pi, \mathsf{role}_{i''}, U_{j_3}, \{U_{j_1}, U_B, U_{j_3}\}, \{EPK_{j_1}, EPK_B, EPK_{j_3}\})$ (resp. $(\Pi, \mathsf{role}_i, U_{j_1}, \{U_{j_1}, U_{j_2}, U_B\}, \{EPK_{j_1}, EPK_{j_2}, EPK_B\})$, $(\Pi, \mathsf{role}_{i'}, U_{j_2}, \{U_{j_1}, U_{j_2}, U_B\}, \{EPK_{j_1}, EPK_{j_2}, EPK_B\})$ and $(\Pi, \mathsf{role}_{i''}, U_B, \{U_{j_1}, U_{j_2}, U_B\}, \{EPK_{j_1}, EPK_{j_2}, EPK_B\})$) as the completed session and SK in the list $\mathcal{L}_{SK}$

(f) Else if $j_2 = C$ (resp. $j_3 = C$), then $\mathcal{D}$ parses EPK_{j_1} into $(R_{j_1}, R'_{j_1}, \pi_{j_1j_2}, \pi_{j_1j_3})$, and sets σ_i as follows: Since $\mathcal{D}$ knows z_C^*, r_C and r'_C, σ_1, σ_3, σ_4, σ_7 and σ_8 are computed with these values. $\sigma_2 = e((\pi_{j_1j_2}R_{j_1}^{-d_2})^{(\mathsf{tag}_{j_1}-\mathsf{tag}_{A^*})^{-1}}, X_{j_3})$ (resp. $\sigma_2 = e((\pi_{j_1j_3}R_{j_1}^{-d_2})^{(\mathsf{tag}_{j_1}-\mathsf{tag}_{A^*})^{-1}}, X_{j_2})$) where $\mathsf{tag}_{j_1} = TCR(R_{j_1}, R'_{j_1}, U_{j_1}, U_C, U_{j_3})$ (resp. $\mathsf{tag}_{j_1} = TCR(R_{j_1}, R'_{j_1}, U_{j_1}, U_{j_3}, U_C)$). $\sigma_5 =$

[2] Even if $R_B = R_A^*$, the simulation validly works because $\mathsf{tag}_B \neq \mathsf{tag}_{A^*}$ always holds.
[3] Even if $R_C = R_A^*$, the simulation validly works because $\mathsf{tag}_C \neq \mathsf{tag}_{A^*}$ always holds.
[4] Even if $R_{j_2} = R_A^*$, the simulation validly works because $\mathsf{tag}_{j_2} \neq \mathsf{tag}_{A^*}$ always holds.
[5] Even if $R_{j_3} = R_A^*$, the simulation validly works because $\mathsf{tag}_{j_3} \neq \mathsf{tag}_{A^*}$ always holds.

$e(R_{j_1}, X_{j_3})^{r_C}$ (resp. $\sigma_5 = e((\pi_{j_1 j_3} \cdot R_{j_1}^{-d_2})^{(\mathsf{tag}_{j_1} - \mathsf{tag}_{A^*})^{-1}}, R_{j_3})$) where $\mathsf{tag}_{j_1} = TCR(R_{j_1}, R'_{j_1}, U_{j_1}, U_{j_3}, U_C)$. $\sigma_6 = e((\pi_{j_1 j_2} \cdot R_{j_1}^{-d_2})^{(\mathsf{tag}_{j_1} - \mathsf{tag}_{A^*})^{-1}}, R_{j_3})$ (resp. $\sigma_6 = e(R_{j_1}, X_{j_3})^{r_C}$) where $\mathsf{tag}_{j_1} = TCR(R_{j_1}, R'_{j_1}, U_{j_1}, U_C, U_{j_3})$. $\mathcal{D}$ computes the session key SK as the protocol, and records $(\Pi, \mathsf{role}_i, U_{j_1}, \{U_{j_1}, U_C, U_{j_3}\}, \{EPK_{j_1}, EPK_C, EPK_{j_3}\})$, $(\Pi, \mathsf{role}_{i'}, U_C, \{U_{j_1}, U_C, U_{j_3}\}, \{EPK_{j_1}, EPK_C, EPK_{j_3}\})$ and $(\Pi, \mathsf{role}_{i''}, U_{j_3}, \{U_{j_1}, U_C, U_{j_3}\}, \{EPK_{j_1}, EPK_C, EPK_{j_3}\})$ (resp. $(\Pi, \mathsf{role}_i, U_{j_1}, \{U_{j_1}, U_{j_2}, U_C\}, \{EPK_{j_1}, EPK_{j_2}, EPK_C\})$, $(\Pi, \mathsf{role}_{i'}, U_{j_2}, \{U_{j_1}, U_{j_2}, U_C\}, \{EPK_{j_1}, EPK_{j_2}, EPK_C\})$ and $(\Pi, \mathsf{role}_{i''}, U_C, \{U_{j_1}, U_{j_2}, U_C\}, \{EPK_{j_1}, EPK_{j_2}, EPK_C\})$) as the completed session and SK in the list $\mathcal{L}_{SK}$

(g) Otherwise, $\mathcal{D}$ computes the session key SK as the protocol, and records $(\Pi, \mathsf{role}_i, U_{j_1}, \{U_{j_1}, U_{j_2}, U_{j_3}\}, \{EPK_{j_1}, EPK_{j_2}, EPK_{j_3}\})$ as the completed session and SK in the list $\mathcal{L}_{SK}$.

3. $\mathsf{Establish}(U_j, SPK_j)$: $\mathcal{D}$ sets U_j as a new party and SPK_j as the SPK of U_j as the definition. Note that U_A, U_B and U_C are not posed due to the freshness definition.
4. $\mathsf{SessionReveal}(\mathsf{sid})$:
   (a) If the session sid is not completed, $\mathcal{D}$ returns an error message.
   (b) Otherwise, $\mathcal{D}$ returns the recorded value SK.
5. $\mathsf{StateReveal}(\mathsf{sid})$: $\mathcal{D}$ answers the ephemeral secret key and intermediate computation results of sid as the definition. Note that the $\mathsf{StateReveal}$ query is not posed to sid^* from the freshness definition. Thus, $\mathcal{D}$ can avoid to return the ephemeral secret key corresponding to α.
6. $\mathsf{StaticReveal}(U_j)$: $\mathcal{D}$ answers the static secret key of U_j as the definition. Note that the $\mathsf{StaticReveal}$ query is not posed to U_B and U_C in the event E_1. Thus, $\mathcal{D}$ can avoid to return static secret key corresponding to β and γ.
7. $\mathsf{EphemeralReveal}(\mathsf{sid})$: $\mathcal{D}$ answers the ephemeral secret key of sid as the definition. Note that the $\mathsf{EphemeralReveal}$ query is not posed to sid^* in the event E_1. Thus, $\mathcal{D}$ can avoid to return the ephemeral secret key corresponding to α.
8. $\mathsf{Test}(\mathsf{sid})$: $\mathcal{D}$ responds to the query as the definition.
9. If $\mathcal{A}$ outputs a guess b', $\mathcal{D}$ outputs b'.

Analysis. The simulation is perfect except that the following event occurs: In Send query, $\mathsf{tag}_{j_2} = \mathsf{tag}_A^*$ or $\mathsf{tag}_{j_3} = \mathsf{tag}_A^*$ in case 2.(c), and $\mathsf{tag}_{j_1} = \mathsf{tag}_A^*$ in case 2.(d) and 2.(e). If these events occur, since $\mathsf{tag}_{j_1} - \mathsf{tag}_{A^*} = 0$, $\mathsf{tag}_{j_2} - \mathsf{tag}_{A^*} = 0$ or $\mathsf{tag}_{j_3} - \mathsf{tag}_{A^*} = 0$, σ_2, σ_5 or σ_6 cannot be computed correctly. This event means that $\mathcal{A}$ finds a collision in TCR. Thus, the probability that the event occurs is negligible.

It is easy to see that static public keys of U_B and U_C are distributed as in $\mathbf{H}_2$. Also, the ephemeral public key of sid^* is distributed as in $\mathbf{H}_2$.

For $\mathcal{A}$, the simulation is same as the experiment $\mathbf{H}_2$ if the challenge δ is g_T^{abc}. Otherwise, the simulation is same as the experiment $\mathbf{H}_3$. Thus, if the advantage of $\mathcal{D}$ is negligible, then $|\mathbf{Adv}(\mathcal{A}, \mathbf{H}_3) - \mathbf{Adv}(\mathcal{A}, \mathbf{H}_2)| \leq negl$.

Hybrid Experiment $\mathbf{H}_4$. In this experiment, the computation of SK in the test session is changed. Instead of computing $SK = F_{\sigma_1}(\mathsf{ST}) \oplus \cdots \oplus F_{\sigma_8}(\mathsf{ST})$, it is changed as $SK = F_{\sigma_1}(\mathsf{ST}) \oplus K \oplus F_{\sigma_3}(\mathsf{ST}) \cdots \oplus F_{\sigma_8}(\mathsf{ST})$ where $K \in_R \{0,1\}^\kappa$.

We construct a distinguisher $\mathcal{D}'$ between PRF $F^* : \{0,1\}^* \times G_T \rightarrow \{0,1\}^k$ and a random function RF from $\mathcal{A}$ in $\mathbf{H}_3$ or $\mathbf{H}_4$. $\mathcal{D}'$ performs the following steps.

Setup. $\mathcal{D}'$ sets PRF $F = F^*$, and provides it as a part of the public parameters. Also, $\mathcal{D}'$ sets all N parties' static secret and public keys.

Simulation. $\mathcal{D}'$ maintains the list $\mathcal{L}_{SK}$ that contains queries and answers of SessionReveal. $\mathcal{D}'$ simulates oracle queries by $\mathcal{A}$ as follows.

1. Send($U_{j_1}, \Pi, \mathsf{role}_i, U_{j_1}, \{U_{j_1}, U_{j_2}, U_{j_3}\}, Init$): $\mathcal{D}'$ computes the ephemeral public key as the protocol, returns it and records $(\Pi, \mathsf{role}_i, U_{j_1}, \{U_{j_1}, U_{j_2}, U_{j_3}\}, (R_{j_1}, R'_{j_1}, \pi_{j_1 j_2}, \pi_{j_1 j_3}))$.
2. Send($U_{j_2}, \Pi, \mathsf{role}_i, U_{j_1}, \{U_{j_1}, U_{j_2}, U_{j_3}\}, EPK_{j_1}$) or Send($U_{j_3}$, $\Pi, \mathsf{role}_i, U_{j_1}, \{U_{j_1}, U_{j_2}, U_{j_3}\}, EPK_{j_1}$):
   (a) If both $(\Pi, \mathsf{role}_{i'}, U_{j_2}, \{U_{j_1}, U_{j_2}, U_{j_3}\}, EPK_{j_2})$ and $(\Pi, \mathsf{role}_{i''}, U_{j_3}, \{U_{j_1}, U_{j_2}, U_{j_3}\}, EPK_{j_3})$ are not recorded, $\mathcal{D}'$ only records $(\Pi, \mathsf{role}_i, U_{j_1}, \{U_{j_1}, U_{j_2}, U_{j_3}\}, EPK_{j_1})$.
   (b) Else if $\mathcal{D}'$ checks EPK_{j_1} with pairing equations as the protocol, and the verification is not valid, then $\mathcal{D}'$ rejects the session.
   (c) Else if $(j_2 = A, j_1 = C)$ or $(j_3 = A, j_1 = B)$, and the session is the k-th session of U_A, then $\mathcal{D}$ computes all σ_i as the protocol. $\mathcal{D}'$ poses ST to his oracle (i.e., F^* or a random function RF), obtains $K \in \{0,1\}^\kappa$, computes the session key $SK^* = F_{\sigma_1}(\mathsf{ST}) \oplus K \oplus F_{\sigma_3}(\mathsf{ST}) \cdots \oplus F_{\sigma_8}(\mathsf{ST})$, and records $(\Pi, \mathsf{role}_1, U_A, \{U_A, U_B, U_C\}, \{EPK_A, EPK_B, EPK_C\})$, $(\Pi, \mathsf{role}_2, U_B, \{U_A, U_B, U_C\}, \{EPK_A, EPK_B, EPK_C\})$ and $(\Pi, \mathsf{role}_3, U_C, \{U_A, U_B, U_C\}$ $\{EPK_A, EPK_B, EPK_C\})$ as the completed session and SK^* in the list $\mathcal{L}_{SK}$.
   (d) Otherwise, $\mathcal{D}'$ computes the session key SK as the protocol, and records $(\Pi, \mathsf{role}_i, U_{j_1}, \{U_{j_1}, U_{j_2}, U_{j_3}\}, \{EPK_{j_1}, EPK_{j_2}, EPK_{j_3}\})$ as the completed session and SK in the list $\mathcal{L}_{SK}$.
3. Establish(U_j, SPK_j): $\mathcal{D}$ sets U_j as a new party and SPK_j as the SPK of U_j as the definition. Note that U_A, U_B and U_C are not posed due to the freshness definition.
4. SessionReveal(sid):
   (a) If the session sid is not completed, $\mathcal{D}'$ returns an error message.
   (b) Otherwise, $\mathcal{D}'$ returns the recorded value SK.
5. StateReveal(sid): $\mathcal{D}'$ answers the ephemeral secret key and intermediate computation results of sid as the definition.
6. StaticReveal(U_j): $\mathcal{D}'$ answers the static secret key of U_j as the definition.
7. EphemeralReveal(sid): $\mathcal{D}'$ answers the ephemeral secret key of sid as the definition.
8. Test(sid): $\mathcal{D}'$ responds to the query as the definition.
9. If $\mathcal{A}$ outputs a guess $b' = 0$, $\mathcal{D}'$ outputs that the oracle is the PRF F^*. Otherwise, $\mathcal{D}'$ outputs that the oracle is a random function RF.

Analysis. For $\mathcal{A}$, the simulation by $\mathcal{D}'$ is same as the experiment $\mathbf{H}_3$ if the oracle is the PRF F^*. Otherwise, the simulation by $\mathcal{D}'$ is same as the experiment $\mathbf{H}_4$. Thus, if the advantage of $\mathcal{D}'$ is negligible, then $|\mathbf{Adv}(\mathcal{A}, \mathbf{H}_4) - \mathbf{Adv}(\mathcal{A}, \mathbf{H}_3)| \leq negl$.

In $\mathbf{H}_4$, the session key in the test session is perfectly randomized. Thus, $\mathcal{A}$ cannot obtain any advantage from Test query.

Therefore, $\mathbf{Adv}(\mathcal{A}, \mathbf{H}_4) = 0$, and $\Pr[E_1 \wedge Suc]$ is negligible.

References

1. Bellare, M., Rogaway, P.: Entity Authentication and Key Distribution. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 232–249. Springer, Heidelberg (1994)
2. Law, L., Menezes, A., Qu, M., Solinas, J.A., Vanstone, S.A.: An Efficient Protocol for Authenticated Key Agreement. Des. Codes Cryptography 28(2), 119–134 (2003)
3. Krawczyk, H.: HMQV: A High-Performance Secure Diffie-Hellman Protocol. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 546–566. Springer, Heidelberg (2005)
4. LaMacchia, B.A., Lauter, K., Mityagin, A.: Stronger Security of Authenticated Key Exchange. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) ProvSec 2007. LNCS, vol. 4784, pp. 1–16. Springer, Heidelberg (2007)
5. Ustaoglu, B.: Obtaining a secure and efficient key agreement protocol from (H)MQV and NAXOS. Des. Codes Cryptography 46(3), 329–342 (2008)
6. Ustaoglu, B.: Comparing *SessionStateReveal* and *EphemeralKeyReveal* for Diffie-Hellman Protocols. In: Pieprzyk, J., Zhang, F. (eds.) ProvSec 2009. LNCS, vol. 5848, pp. 183–197. Springer, Heidelberg (2009)
7. Sarr, A.P., Elbaz-Vincent, P., Bajard, J.-C.: A Secure and Efficient Authenticated Diffie–Hellman Protocol. In: Martinelli, F., Preneel, B. (eds.) EuroPKI 2009. LNCS, vol. 6391, pp. 83–98. Springer, Heidelberg (2010)
8. Sarr, A.P., Elbaz-Vincent, P., Bajard, J.-C.: A New Security Model for Authenticated Key Agreement. In: Garay, J.A., De Prisco, R. (eds.) SCN 2010. LNCS, vol. 6280, pp. 219–234. Springer, Heidelberg (2010)
9. Fujioka, A., Suzuki, K.: Designing Efficient Authenticated Key Exchange Resilient to Leakage of Ephemeral Secret Keys. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 121–141. Springer, Heidelberg (2011)
10. Boyd, C., González Nieto, J.M.: Round-Optimal Contributory Conference Key Agreement. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 161–174. Springer, Heidelberg (2002)
11. Gorantla, M.C., Boyd, C., González Nieto, J.M., Manulis, M.: Generic One Round Group Key Exchange in the Standard Model. In: Lee, D., Hong, S. (eds.) ICISC 2009. LNCS, vol. 5984, pp. 1–15. Springer, Heidelberg (2010)
12. Garg, S., Gentry, C., Halevi, S.: Candidate Multilinear Maps from Ideal Lattices. In: Johansson, T. (ed.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 1–17. Springer, Heidelberg (2013), `http://eprint.iacr.org/2012/610`
13. Joux, A.: A One Round Protocol for Tripartite Diffie-Hellman. In: Bosma, W. (ed.) ANTS-IV. LNCS, vol. 1838, pp. 385–393. Springer, Heidelberg (2000)
14. Al-Riyami, S.S., Paterson, K.G.: Tripartite authenticated key agreement protocols from pairings. In: Paterson, K.G. (ed.) Cryptography and Coding 2003. LNCS, vol. 2898, pp. 332–359. Springer, Heidelberg (2003)
15. Manulis, M., Suzuki, K., Ustaoglu, B.: Modeling Leakage of Ephemeral Secrets in Tripartite/Group Key Exchange. In: Lee, D., Hong, S. (eds.) ICISC 2009. LNCS, vol. 5984, pp. 16–33. Springer, Heidelberg (2010)
16. Fujioka, A., Manulis, M., Suzuki, K., Ustaoğlu, B.: Sufficient Condition for Ephemeral Key-Leakage Resilient Tripartite Key Exchange. In: Susilo, W., Mu, Y., Seberry, J. (eds.) ACISP 2012. LNCS, vol. 7372, pp. 15–28. Springer, Heidelberg (2012)
17. Diament, T., Lee, H.K., Keromytis, A.D., Yung, M.: The dual receiver cryptosystem and its applications. In: ACM CCS 2004, pp. 330–343 (2004)
18. Chow, S.S.M., Franklin, M., Zhang, H.: Practical Dual-Receiver Encryption: Soundness, Complete Non-Malleability, and Applications. Technical Report, UC Davis (2012), `http://csiflabs.cs.ucdavis.edu/~hbzhang/dual.pdf`

19. Fujioka, A., Suzuki, K., Xagawa, K., Yoneyama, K.: Strongly Secure Authenticated Key Exchange from Factoring, Codes, and Lattices. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 467–484. Springer, Heidelberg (2012)
20. Canetti, R., Krawczyk, H.: Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 453–474. Springer, Heidelberg (2001)
21. Chevallier-Mames, B.: An Efficient CDH-Based Signature Scheme with a Tight Security Reduction. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 511–526. Springer, Heidelberg (2005)
22. Cramer, R., Shoup, V.: Design and Analysis of Practical Public-Key Encryption Schemes Secure against Adaptive Chosen Ciphertext Attack. SIAM Journal on Computing 33, 167–226 (2004)
23. Smart, N.P.: Efficient Key Encapsulation to Multiple Parties. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 208–219. Springer, Heidelberg (2005)