Towards Modelling Perfect Forward Secrecy for One-round Group Key Exchange

Zheng Yang and Daigu Zhang
(Corresponding author: Daigu Zhang)

School of Computer Science and Engineering, Chongqing University of Technology
No. 69 Hongguang Avenue, Banan District, Chongqing 400054, China
(Email: daigu@cqut.edu.cn)

(Received May 10, 2014; revised and accepted Mar. 23 & July 4, 2015)

Abstract

We propose two security models for one-round group key exchange (ORGKE), which are called as g-eCKw and g-eCK-PFS. The g-eCK-PFS is a stronger variant of g-eCKw, which particularly formulates perfect forward secrecy for ORGKE. A new tripartite ORGKE is proposed to provide g-eCKw security without random oracles under standard assumptions, that is also more efficient than its predecessor by Li and Yang on CANS'13. We also show how to transform (compile) a g-eCKw secure protocol to achieve g-eCK-PFS security. In particular, our result enables us to prove the security of the first ORGKE protocol that achieves perfect forward secrecy without random oracles in a strong security model allowing adversary to compromise critical information of session participants such as long-term or ephemeral private key.

Keywords: Authenticated key exchange, group key exchange, perfect forward secrecy standard model, programmable hash function

1 Introduction

A group authenticated key exchange protocol (GAKE) enables multiple parties to share a common secret key over a public network, where the generated key can be only known by those intended communication partners can compute that shared key. As a fundamental building block, GAKE plays an important role in protecting various kinds of group applications such as digital conferences and file sharing etc. It is well-known that GAKE is normally generalized from two party authenticated key exchange (2AKE) protocols, as well as the security notions for GAKE. The first formal GAKE security model was proposed by Bresson et al. [7] that follows the seminal work of the indistinguishability based 2AKE model by Bellare and Rogawary [2]. Since then many modifications and improvements on GAKE models appeared thereafter. The traditional GAKE-security notion defined in models [4, 5, 9, 20], is made popular in different flavors depending on whether the protocol provides forward secrecy against outsider adversary, i.e. assuming that adversary is not part of the group. Such security notion does not take into account any protection against *insider attacks*, and in fact it is not hard to see that GAKE secure protocols may be completely insecure against attacks by malicious insiders.

The insider security was first studied by Katz and Shin [19], e.g. preventing honest users from computing different keys and from having distinct views on the identities of other participants. Several models [8, 15, 16, 19] have been proposed to augment GAKE security against insider threats.

Besides consideration of outsider and insider security, formulating stronger security notions for GAKE recently has gained much attention of the researchers. Quite a few attempts have been made in order to enlarge the class of attacks that a GAKE protocol can resist. Gorantla et al. (GBG) [16] inspired by two party approaches [21] reformulated the key compromise impersonation (KCI) resilience attribute for GAKE by considering outsider and insider KCI attacks respectively. In a successful KCI attack, an adversary with the long-term private key of Alice can impersonate Bob to Alice without knowledge of Bob's long-term private key. Thus resistance to KCI attacks is important in situations where an adversary wishes to obtain some information possessed by Alice, who is only willing to divulge this information to Bob (and where the adversary is not able to obtain Bob's long-term private key). The GBG model [16] also consider the leakage of secret states as in [8, 10] that allows the adversary to obtain long-term secret keys and secret states independently, with a restriction that the leakage of secret states from sessions for which the adversary is not required to distinguish the key. Such restriction is quite necessary since many GAKE protocols are insecure if secret session states used to compute session key are exposed.

In this paper, we study the security and construction for one-round group key exchange (ORGKE) that

only one message was sent (and received) by each session participants during key exchange procedure. Due to the attractive advantage of bandwidth-efficient, ORGKE has drawn great attention of research community. The seminal work of ORGKE is the pairing based tripartite protocol introduced by Joux [18]. However it is unauthenticated and subject to well known man-in-the-middle attacks. During recent years, some solutions, e.g. [1, 14, 23, 24, 27], have been made to improve the original protocol. This has also led the development of security model for GAKE.

Meantime, the FMSU protocol [14] was proven secure in a very strong model called as g-eCK (see also in [28]) which is extended from two party eCK model [22] to group case that captures the security properties concerning resilience of KCI attacks, chosen identity and public key (CIDPK) attacks and leakage of session states, and provision of wPFS in a single model. The g-eCK model is considered as one of the strongest GAKE security model in [14, 24], and it was used to prove the recent GAKE protocol [24]. However, it fails to model the full perfect forward secrecy (PFS) for ORGKE. Hence there is an interesting open question whether it is possible to achieve full PFS within one communication round. As well it is an open question on how to model PFS for ORGKE. In contrast to PFS security notion, the wPFS has one more restriction that an adversary does not modify messages received by the test session and the session is executed before the long-term private keys are compromised. So that wPFS is much weaker than PFS. On the other hand, the GAKE protocol with PFS may be more appealing. To our best of knowledge, there is no ORGKE protocol that can provide PFS without random oracles.

%paragraphContributions We solve the above open questions by first introducing two security models (called as g-eCK and g-eCK-PFS) for ORGKE which follow the idea of modelling approach [12] for two party key exchange. The g-eCKw model is a variant of g-eCK model [14]. Whereas the g-eCK-PFS model is developed from g-eCKw model with particularly modelling the perfect forward secrecy. We also show that it is possible to compile any g-eCKw secure protocols to be g-eCK-PFS secure by introducing a signature-based protocol transformation (compiler). In order to illustrate that our models are reasonable and practical for our analysis, we focus on three-party one-round key exchange which is a special class of ORGKE protocols. As a concrete example we come up with a new provably secure solution without random oracles in the g-eCKw model. To be of independent interesting, the new proposed protocol is more efficient than previous g-eCK secure protocol without random oracles. The security proof for our scheme is also much simpler which is mainly based on a strong Cube Bilinear Decisional Diffie-Hellman assumption (that is derived from the Cube Bilinear Decisional Diffie-Hellman assumption used in [24]).

1n 2012, Cremers and Feltz [12] proposed a stronger security model (referred to as eCKw) to reformulate the wPFS notion based on a new concept so called *origin-session*. The resultant model is claimed to provide a slightly stronger form of wPFS than eCK model's. On the second, they further develop eCKw to model PFS that yields another new model (which is referred to as eCK-PFS). More interestingly, it is possible to transform any eCKw secure protocol to be eCK-PFS secure using the signature based compiler in [12]. The implication relationship between eCK and eCKw models was studied in literature [13, 34]. Their results somehow inspire us to deal with the issue on modelling PFS for ORGAKE. But unlike their works, our protocol instance is analyzed in the new proposed model without random oracles.

In the recent work, Li et al. [24] introduced a new construction for pairing-based one-round 3AKE protocol. This protocol is the first one that is provably secure in the g-eCK model without random oracles. Security of proposed protocol is reduced to the hardness of Cube Bilinear Decisional Diffie-Hellman (CBDDH) problem for symmetric pairing. However it only satisfies wPFS rather than PFS. On the other side, this protocol requires many pairing operations for key exchange which might lose practical interesting. One of the motivations of our works is to improve the efficiency of that protocol.

Some other group key exchange protocols, for instance. [11, 17, 26, 31, 32] have been recently proposed from different motivations. But none of them can be proven secure in our new proposed models.

2 Preliminaries

In this section, we recall the required definitions for our result on proposed protocols.

Notations. We let $\kappa \in \mathbb{N}$ denote the security parameter and 1^κ the string that consists of κ ones. Let a capital letter with a 'hat' denote an identity; without the hat the letter denotes the public key of that party. Let $[n] = \{1, \ldots, n\} \subset \mathbb{N}$ be the set of integers between 1 and n. If S is a set, then $a \xleftarrow{\$} S$ denotes the action of sampling a uniformly random element from S. Let '||' denote the operation concatenating two binary strings.

Digital Signature Schemes. A digital signature scheme Σ is defined by three PPT algorithms $\mathsf{SIG} = (\mathsf{SIG.Gen}, \mathsf{SIG.Sign}, \mathsf{SIG.Vfy})$ with associated public/private key spaces $\{\mathcal{PK}, \mathcal{SK}\}$, message space $\mathcal{M}_{\mathsf{SIG}}$ and signature space $\mathcal{S}_{\mathsf{SIG}}$ in the security parameter κ:

- $(sk, pk) \xleftarrow{\$} \mathsf{SIG.Gen}(1^\kappa)$: this algorithm takes as input the security parameter κ and outputs a (public) verification key $pk \in \mathcal{PK}$ and a secret signing key $sk \in \mathcal{SK}$;
- $\sigma \xleftarrow{\$} \mathsf{SIG.Sign}(sk, m)$: the signing algorithm generates a signature $\sigma \in \mathcal{S}_{\mathsf{SIG}}$ for message $m \in \mathcal{M}_{\mathsf{SIG}}$ using signing key sk;

- $\{0,1\} \leftarrow \mathsf{SIG.Vfy}(pk, m, \sigma)$: the verification algorithm outputs – on input verification key pk, a message m and corresponding signature σ – 1 if σ is a valid signature for m under key pk, and 0 otherwise.

Definition 1. *We say that* SIG *is* $(t, \epsilon_{\mathsf{SIG}})$*-secure against* existential forgeries under adaptive chosen-message attacks, *if* $\Pr[\mathsf{EXP}^{seuf-cma}_{\Sigma,\mathcal{A}}(\kappa) = 1] \leq \epsilon_{\mathsf{SIG}}$ *for all adversaries* $\mathcal{A}$ *running in time at most* t *in the following experiment:*

$\mathsf{EXP}^{seuf-cma}_{\mathsf{SIG},\mathcal{A}}(\kappa)$
$(sk, pk) \xleftarrow{\$} \mathsf{SIG.Gen}(1^\kappa)$;
$(\sigma^*, m^*) \leftarrow \mathcal{A}^{\mathcal{SIG}(sk,\cdot)}$, *which can make up to* q *queries to the signing oracle* $\mathcal{SIG}(sk, \cdot)$ *with arbitrary messages* m;
return 1, if the following conditions are held:

1) $\mathsf{SIG.Vfy}(pk, m^*, \sigma^*) = 1$, *and*
2) (m^*, σ^*) *is not among the previously submitted to* $\mathcal{SIG}(sk, \cdot)$ *oracle;*

output 0, otherwise;

where ϵ_{SIG} *is a negligible function in* κ, *on input message* m *the oracle* $\mathcal{SIG}(sk, \cdot)$ *returns signature* $\sigma \leftarrow SIGsign(sk, m)$ *and the number of queries* q *is bound by time* t.

Strong Cube Bilinear Decisional Diffie-Hellman Assumption. We first recall the notion of bilinear groups. Our pairing based scheme will be parameterized by a symmetric pairing parameter generator, denoted by $\mathsf{PG.Gen}$. This is a polynomial time algorithm that on input a security parameter 1^κ, returns the description of two multiplicative cyclic groups $\mathbb{G}$ and $\mathbb{G}_T$ of the same prime order p, generator g for $\mathbb{G}$, and a bilinear computable pairing $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. We call $\mathcal{PG} = (\mathbb{G}, g, \mathbb{G}_T, p, e) \xleftarrow{\$} \mathsf{PG.Gen}(1^\kappa)$ be a set of symmetric bilinear groups, if the function e is an (admissible) bilinear map and it holds that:

- **Bilinear:** $\forall(a, b) \in \mathbb{G}$ and $\forall(x, y) \in \mathbb{Z}_p$, we have $e(a^x, b^y) = e(a, b)^{xy}$.
- **Non-degenerate:** $e(g, g) \neq 1_{\mathbb{G}_T}$, is a generator of group $\mathbb{G}_T$.
- **Efficiency:** $\forall(a, b) \in \mathbb{G}$, e is efficiently computable.

The strong Cube Bilinear Decisional Diffie-Hellman (sCBDDH) problem that is formally defined as follows.

Definition 2. *We say that the* sCBDDH *problem relative to generator* $\mathsf{PG.Gen}$ *is* $(t, \epsilon_{\mathsf{sCBDDH}})$*-hard, if the probability bound* $|\Pr[\mathsf{EXP}^{scbddh}_{\mathsf{PG.Gen},\mathcal{A}}(\kappa, n) = 1] - 1/2| \leq \epsilon_{\mathsf{sCBDDH}}$ *holds for all adversaries* $\mathcal{A}$ *running in probabilistic polynomial time* t *in the following experiment:*

$\mathsf{EXP}^{scbddh}_{\mathsf{PG.Gen},\mathcal{A}}(\kappa, n)$
$\mathcal{PG} = (\mathbb{G}, g, \mathbb{G}_T, p, e) \xleftarrow{\$} \mathsf{PG.Gen}(1^\kappa)$;
$a, \gamma \xleftarrow{\$} \mathbb{Z}^*_p$;
$b \xleftarrow{\$} \{0,1\}$, *if* $b = 1$ $\Gamma \leftarrow e(g, g)^{a^3}$, *otherwise* $\Gamma \leftarrow e(g, g)^\gamma$;
$b' \leftarrow \mathcal{A}(1^\kappa, \mathcal{PG}, g^a, g^{1/a}, \Gamma)$;
if $b = b'$ *then return 1, otherwise return 0;*

where $\epsilon_{\mathsf{sCBDDH}} = \epsilon_{\mathsf{sCBDDH}}(\kappa)$ *is a negligible function in the security parameter* κ.

General One-round Group Key Exchange. We present a generic definition of one-round group key exchange (ORGKE) to allow us to describe our generic result for this class of protocols. In a ORGKE protocol, each party may send a single 'message' and this message is always assumed to be independent of the message sent by the other party without loss of generality. The independence property of sent messages is required since the session participants can't achieve mutual authentication in one-round and it enables parties to run protocol instances simultaneously (which is a key feature of one-round protocol). The key exchange procedure is done within two pass and a common shared session key is generated to be known only by session participants.

Let $\mathsf{GD} := ((\mathsf{ID}_1, pk^{ke}_{\mathsf{ID}_1}), \ldots, (\mathsf{ID}_n, pk^{ke}_{\mathsf{ID}_n}))$ be a list which is used to store the public information of a group of parties formed as tuple $(\mathsf{ID}_i, pk^{ke}_{\mathsf{ID}_i})$, where n is the size of the group members which intend to share a key and $pk^{ke}_{\mathsf{ID}_i}$ is the public key of party $\mathsf{ID}_i \in \mathcal{IDS}$ $(i \in [n])$. Let T denote the transcript storing the messages sent and received by a protocol instance at a party which are sorted orderly. A general PKI-based ORGKE protocol may consist of four polynomial time algorithms $(\mathsf{ORGKE.Setup}, \mathsf{ORGKE.KGen}, \mathsf{ORGKE.MF}, \mathsf{ORGKE.SKG})$ with following semantics:

- $pms \leftarrow \mathsf{Setup}(1^\kappa)$: This algorithm takes as input a security parameter κ and outputs a set of system parameters storing in a variable pms.
- $(sk^{ke}_{\mathsf{ID}}, pk^{ke}_{\mathsf{ID}}) \xleftarrow{\$} \mathsf{ORGKE.KGen}(pms, \mathsf{ID})$: This algorithm takes as input system parameters pms and a party's identity ID, and outputs a pair of long-term private/public key $(sk^{ke}_{\mathsf{ID}}, pk^{ke}_{\mathsf{ID}}) \in (\mathcal{PK}, \mathcal{SK})$ for party ID.
- $m_{\mathsf{ID}_1} \xleftarrow{\$} \mathsf{ORGKE.MF}(pms, sk^{ke}_{\mathsf{ID}_1}, r_{\mathsf{ID}_1}, \mathsf{GD})$: This algorithm takes as input system parameters pms and the sender ID_1's secret key $sk^{ke}_{\mathsf{ID}_1}$, a randomness $r_{\mathsf{ID}_1} \xleftarrow{\$} \mathcal{R}_{\mathsf{ORGKE}}$ and the group information variable GD, and outputs a message to be sent in a protocol pass, where $\mathcal{R}_{\mathsf{ORGKE}}$ is the randomness space.[1]

[1] We remark that the parameter GD of algorithm $\mathsf{ORGKE.MF}$ is only optional, which can be any empty string if specific protocol compute the message without knowing any information about its indented partners.

- $K \leftarrow \mathsf{ORGKE.SKG}(pms, sk^{ke}_{\mathsf{ID}_1}, r_{\mathsf{ID}_1}, \mathsf{GD}, \mathsf{T})$: This algorithm take as the input system parameters pms and ID_1's secret key $sk^{ke}_{\mathsf{ID}_1}$, a randomness $r_{\mathsf{ID}_1} \xleftarrow{\$} \mathcal{R}_{\mathsf{ORGKE}}$ and the group information GD and a transcript T orderly recorded all protocol messages exchanged[2], and outputs session key $K \in \mathcal{K}_{\mathsf{ORGKE}}$.

  For correctness, we require that, on input the same group description $\mathsf{GD} = ((\mathsf{ID}_1, pk^{ke}_1), \ldots, (\mathsf{ID}_n, pk^{ke}_n))$ and transcript T, algorithm $\mathsf{ORGKE.SKG}$ satisfies the constraint:

  - $\mathsf{ORGKE.SKG}(pms, sk^{ke}_{\mathsf{ID}_1}, r_{\mathsf{ID}_1}, \mathsf{GD}, \mathsf{T}) = \mathsf{ORGKE.SKG}(pms, sk^{ke}_{\mathsf{ID}_i}, r_{\mathsf{ID}_i}, \mathsf{GD}, \mathsf{T})$,

  where $sk^{ke}_{\mathsf{ID}_i}$ is the secret key of a party $\mathsf{ID}_i \in \mathsf{GD}$ who generates randomness $r_{\mathsf{ID}_i} \in \mathcal{R}_{\mathsf{ORGKE}}$ for $i \in [n]$.

Besides these algorithms, each protocol might consist of other steps such as long-term key registration and message exchange, which should be described by each protocol independently. The key exchange procedure among n parties is informally depicted in Figure 1.

Pseudo-Random Functions. Let $\mathsf{PRF} : \mathcal{K}_{\mathsf{PRF}} \times \mathcal{D}_{\mathsf{PRF}} \rightarrow \mathcal{R}_{\mathsf{PRF}}$ denote a family of deterministic functions, where $\mathcal{K}_{\mathsf{PRF}}$ is the key space, $\mathcal{D}_{\mathsf{PRF}}$ is the domain and $\mathcal{R}_{\mathsf{PRF}}$ is the range of PRF for security parameter κ. Let $\mathsf{RL} = \{(x_1, y_1), \ldots, (x_q, y_q)\}$ be a list which is used to record bit strings formed as tuple $(x_i, y_i) \in (\mathcal{D}_{\mathsf{PRF}}, \mathcal{R}_{\mathsf{PRF}})$ where $1 \leq i \leq q$ and $q \in \mathbb{N}$. So that in RL each x is associated with a y. Let $\mathsf{RF} : \mathcal{D}_{\mathsf{PRF}} \rightarrow \mathcal{R}_{\mathsf{PRF}}$ be a stateful uniform random function, which can be executed at most a polynomial number of q times and keeps a list RL for recording each invocation. On input a message $x \in \mathcal{D}_{\mathsf{PRF}}$, the function $\mathsf{RF}(x)$ is executed as follows:

- If $x \in \mathsf{RL}$, then return corresponding $y \in \mathsf{RL}$,
- Otherwise return $y \xleftarrow{\$} \mathcal{R}_{\mathsf{PRF}}$ and record (x, y) into RL.

Definition 3. *We say that* PRF *is a* $(q, t, \epsilon_{\mathsf{PRF}})$*-secure pseudo-random function family, if it holds that* $|\Pr[\mathsf{EXP}^{prf}_{\mathsf{PRF},\mathcal{A}}(\kappa) = 1] - 1/2| \leq \epsilon_{\mathsf{PRF}}$ *for all adversaries* $\mathcal{A}$ *running in probabilistic polynomial time* t *and making at most* q *oracle queries in the following experiment:*

$\mathsf{EXP}^{prf}_{\mathsf{PRF},\mathcal{A}}(\kappa)$	$\mathcal{F}(b, x)$
$b \xleftarrow{\$} \{0,1\}, k \xleftarrow{\$} \mathcal{K}_{\mathsf{PRF}}$	*If* $x \notin \mathcal{D}_{\mathsf{PRF}}$ *then return* $\perp$
$b' \leftarrow \mathcal{A}^{\mathcal{F}(b,\cdot)}(\kappa)$	*If* $b = 1$ *then return* $\mathsf{PRF}(k, x)$
if $b = b'$ *then return 1, otherwise return 0.*	*Otherwise return* $\mathsf{RF}(x)$

where $\epsilon_{\mathsf{PRF}} = \epsilon_{\mathsf{PRF}}(\kappa)$ *is a negligible function in the security parameter* κ*, and the number of allowed queries* q *is bound by* t*.*

[2]The detail order needs to be specified by each protocol.

Target Collision-Resistant Hash Functions. Let $\mathsf{TCRHF} : \mathcal{K}_{\mathsf{TCRHF}} \times \mathcal{M}_{\mathsf{TCRHF}} \rightarrow \mathcal{Y}_{\mathsf{TCRHF}}$ be a family of keyed-hash functions associated with key space $\mathcal{K}_{\mathsf{TCRHF}}$, message space $\mathcal{M}_{\mathsf{TCRHF}}$ and hash value space $\mathcal{Y}_{\mathsf{TCRHF}}$. The public key $hk_{\mathsf{TCRHF}} \in \mathcal{K}_{\mathsf{TCRHF}}$ of a hash function $\mathsf{TCRHF}(hk_{\mathsf{TCRHF}}, \cdot)$ is generated by a PPT algorithm $\mathsf{TCRHF.KG}(1^\kappa)$ on input security parameter κ. If the hash key hk_{TCRHF} is obvious from the context, we write $\mathsf{TCRHF}(m)$ for $\mathsf{TCRHF}(hk_{\mathsf{TCRHF}}, m)$.

Definition 4. TCRHF *is a* $(t, \epsilon_{\mathsf{TCRHF}})$*-secure target collision resistant hash function family if for all* t*-time adversaries* $\mathcal{A}$ *it holds that*

$$\Pr\left[\begin{array}{c} hk_{\mathsf{TCRHF}} \xleftarrow{\$} \mathsf{TCRHF.KG}(1^\kappa), \\ m \xleftarrow{\$} \mathcal{M}_{\mathsf{TCRHF}}, \\ m' \leftarrow \mathcal{A}(1^\kappa, hk_{\mathsf{TCRHF}}, m), \\ m \neq m',\ m' \in \mathcal{M}_{\mathsf{TCRHF}}, \\ \mathsf{TCRHF}(m) = \mathsf{TCRHF}(m') \end{array}\right] \leq \epsilon_{\mathsf{TCRHF}},$$

where the probability is over the random bits of $\mathcal{A}$*.*

3 New Security Models

In this section we present two new strong security model for one-round group key exchange that are generalized from the models [12] for two party case. In these models, the active adversary is provided with an uniform 'execution environment' that follows an important research line research [10, 14, 20, 22, 27] which is initiated by Bellare and Rogaway [2].

Execution Environment. In the execution environment, we fix a set of honest parties $\{\mathsf{ID}_1, \ldots, \mathsf{ID}_\ell\}$ for $\ell \in \mathbb{N}$, where ID_i ($i \in [\ell]$) is the identity of a party which is chosen uniquely from space $\mathcal{IDS}$. Each identity is associated with a long-term key pair $(sk_{\mathsf{ID}_i}, pk_{\mathsf{ID}_i}) \in (\mathcal{SK}, \mathcal{PK})$ for authentication.

Each honest party ID_i can sequentially and concurrently execute the protocol multiple times with different intended partners, this is characterized by a collection of oracles $\{\pi^s_i : i \in [\ell], s \in [d]\}$ for $d \in \mathbb{N}$.[3] Oracle π^s_i behaves as party ID_i carrying out a process to execute the s-th protocol instance (session), which has access to the long-term key pair $(sk_{\mathsf{ID}_i}, pk_{\mathsf{ID}_i})$ and to all other public keys. Moreover, we assume each oracle π^s_i maintains a list of independent internal state variables with semantics listed as follows.

- Ψ^s_i – a variable storing the identities and public keys of session participants which are sorted lexicographically in terms of identity, including ID_i itself.
- Φ^s_i – a variable denoting the decision $\Phi^s_i \in \{\texttt{accept}, \texttt{reject}\}$.

[3]An oracle in this paper might be alternatively written as $\pi^s_{\mathsf{ID}_i}$ which is conceptually equivalent to π^s_i.

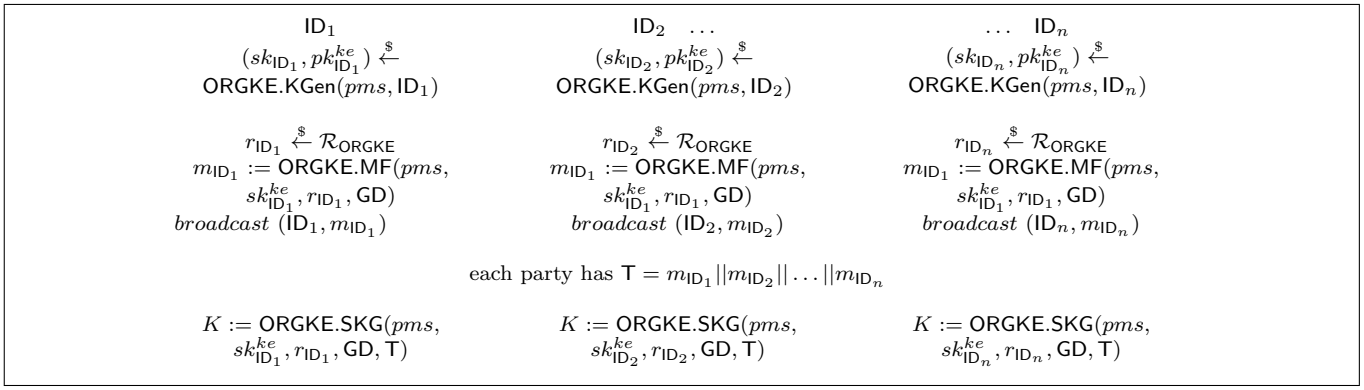

Figure 1: General one-round group key exchange

- K_i^s – a variable recording the session key $K_i^s \in \mathcal{K}_{\mathsf{GAKE}}$.
- st_i^s – a variable storing the ephemeral keys that allows to be revealed, e.g. the randomness used to generate ephemeral public key.
- sT_i^s – a variable recording the transcript of messages sent by oracle π_i^s.
- $\{rT_j^t\}$ – a set of variables each of which records the transcript of messages received by oracle π_i^s from party $\mathsf{ID}_j \in \Psi_i^s$ such that $j \neq i$.
- T_i^s – a variable storing the transcript of all messages sent and received by π_i^s during its execution, where the messages are ordered by round and within each round lexicographically by the identities of the purported senders.

All those variables of each oracle are initialized with empty string which is denoted by the symbol $\emptyset$ in the following. At some point, each oracle π_i^s may complete the execution always with a decision state $\Phi_i^s \in \{\texttt{accept}, \texttt{reject}\}$. Furthermore, we assume that the session key is assigned to the variable K_i^s (such that $K_i^s \neq \emptyset$) iff oracle π_i^s has reached an internal state $\Phi_i^s = \texttt{accept}$.

Adversarial Model. An adversary $\mathcal{A}$ in our model is a PPT Turing Machine taking as input the security parameter 1^κ and the public information (e.g. generic description of above environment), which may interact with these oracles by issuing the following queries.

- $\mathsf{Send}(\pi_i^s, m)$: The adversary can use this query to send any message m of his own choice to oracle π_i^s. The oracle will respond the next message m^* (if any) to be sent according to the protocol specification and its internal states. Oracle π_i^s would be initiated via sending the oracle the first message $m = (\top, \Psi_i^s)$ consisting of a special initialization symbol $\top$ and a variable storing partner identities.
- $\mathsf{RevealKey}(\pi_i^s)$: Oracle π_i^s responds with the contents of variable K_i^s.
- $\mathsf{StateReveal}(\pi_i^s)$: Oracle π_i^s responds with the secret state stored in variable st_i^s. We assume that the st_i^s only include all ephemeral randomness generated on host machine (such as personal computer).
  Namely, the ephemeral states on secure device (like the smart card) where the long-term key is stored are excluded from st_i^s. This modelling approach is widely used in literatures [6, 29, 33].
- $\mathsf{Corrupt}(\mathsf{ID}_i)$: Oracle π_i^1 responds with the long-term secret key sk_{ID_i} of party ID_i if $i \in [\ell]$; otherwise a failure symbol $\perp$ is returned.
- $\mathsf{EstablishParty}(\mathsf{ID}_\tau, pk_{\mathsf{ID}_\tau})$: This query allows the adversary to register an identity ID_τ ($\ell < \tau$ and $\tau \in \mathbb{N}$) and a static public key pk_{ID_τ} on behalf of a party ID_τ. Parties established by this query are called dishonest.
- $\mathsf{Test}(\pi_i^s)$: If the oracle has state $\Phi_i^s = \texttt{reject}$ or $K_i^s = \emptyset$, then the oracle π_i^s returns some failure symbol $\perp$. Otherwise it flips a fair coin b, samples a random element K_0 from key space $\mathcal{K}_{\mathsf{GAKE}}$, and sets $K_1 = K_i^s$. Finally the key K_b is returned.

We highlight that the exact meaning of the StateReveal must be defined by each protocol separately, i.e., the content stored in the variable st during protocol execution. Our goal is to define the maximum states that can be leaked from each session.

Secure GAKE Protocols. In order to denote the situation that two oracles are engaged in an on-line communication, we first define the partnership via *matching sessions.*

Definition 5 (Matching sessions)**.** *We say that an oracle π_i^s has a* matching session *to oracle π_j^t, if π_i^s has sent all protocol messages and $\Psi_i^s = \Psi_j^t$ and $T_j^t = T_i^s$. The oracle π_j^t is said to be the partner-oracle of π_i^s.*

We also recall the notion of *origin session* defined in [12].

Definition 6 (Origin Session)**.** *We say that an oracle π_j^t has an* origin session *to oracle π_i^s, if π_j^t has sent all*

protocol messages and $sT_j^t = rT_{i,j}^s$. *The oracle* π_j^t *is said to be the origin-oracle of* π_i^s.

Please note that if the protocol message does not include any information about its owner, then the origin-oracle of an oracle may not come from its intended communication partner.

Correctness. We say a group authenticated key exchange (GAKE) protocol Σ is correct, if two oracles π_i^s and π_j^t accept with matching sessions, then both oracles hold the same session key, i.e. $K_i^s = K_j^t$.

For the security definition, we need the notion of *freshness* of an oracle. in the sequel, we give two freshness definitions. The difference between those two notions is that the first one formulates only weak perfect forward secrecy and the second one formulates the perfect forward secrecy. Let π_i^s be an accepted oracle. Meanwhile, let π_j^t be an oracle (if it exists) with intended partner ID_i, such that π_i^s has a matching session to π_j^t. Let π_v^l be an oracle (if it exists), such that π_v^l has a origin session to π_i^s. Let π_{MS} be a variable storing all partner-oracles of π_i^s, and π_{RO} be a variable storing all origin-oracles of π_i^s.

Definition 7 (g-eCKw Freshness)**.** *The oracle* π_i^s *is said to be g-eCKw fresh if none of the following conditions holds:*

1) $\mathcal{A}$ *queried* $\mathsf{EstablishParty}(\mathsf{ID}_j, pk_{\mathsf{ID}_j})$ *to some party* $\mathsf{ID}_j \in \Psi_i^s$.
2) $\mathcal{A}$ *queried* $\mathsf{RevealKey}(\pi_i^s)$.
3) *if* $\pi_{MS} \neq \emptyset$, $\mathcal{A}$ *queried* $\mathsf{RevealKey}(\pi_j^t)$ *to some oracle* $\pi_j^t \in \pi_{MS}$.
4) $\mathcal{A}$ *queried both* $\mathsf{Corrupt}(\mathsf{ID}_i)$ *and* $\mathsf{StateReveal}(\pi_i^s)$.
5) *For some oracle* $\pi_l^v \in \pi_{RO}$ *and some* $\mathsf{ID}_j \in \mathsf{pid}_i^s$ $(j \neq i)$, *if* $sT_l^v = rT_{i,j}^s \neq \emptyset$, $\mathcal{A}$ *queried both* $\mathsf{Corrupt}(\mathsf{ID}_j)$ *and* $\mathsf{StateReveal}(\pi_l^v)$.
6) *For some* $\mathsf{ID}_j \in \mathsf{pid}_i^s$ $(j \neq i)$, *if there is no oracle* π_j^t *such that* π_j^t *has an origin session to* π_i^s, $\mathcal{A}$ *queried* $\mathsf{Corrupt}(\mathsf{ID}_j)$.

Definition 8 (g-eCK-PFS Freshness)**.** *The oracle* π_i^s *is said to be g-eCK-PFS fresh if none of the following conditions holds:*

1) $\mathcal{A}$ *queried* $\mathsf{EstablishParty}(\mathsf{ID}_j, pk_{\mathsf{ID}_j})$ *to some party* $\mathsf{ID}_j \in \Psi_i^s$.
2) $\mathcal{A}$ *queried* $\mathsf{RevealKey}(\pi_i^s)$.
3) *If* $\pi_{MS} \neq \emptyset$, $\mathcal{A}$ *queried* $\mathsf{RevealKey}(\pi_j^t)$ *to some oracle* $\pi_j^t \in \pi_{MS}$.
4) $\mathcal{A}$ *queried both* $\mathsf{Corrupt}(\mathsf{ID}_i)$ *and* $\mathsf{StateReveal}(\pi_i^s)$.
5) *For some oracle* $\pi_l^v \in \pi_{RO}$ *and some* $\mathsf{ID}_j \in \mathsf{pid}_i^s$ $(j \neq i)$, *if* $sT_l^v = rT_{i,j}^s \neq \emptyset$, $\mathcal{A}$ *queried both* $\mathsf{Corrupt}(\mathsf{ID}_j)$ *and* $\mathsf{StateReveal}(\pi_l^v)$.
6) *For some* $\mathsf{ID}_j \in \mathsf{pid}_i^s$ $(j \neq i)$, *if there is no oracle* π_j^t *such that* π_j^t *has an origin session to* π_i^s, $\mathcal{A}$ *queried* $\mathsf{Corrupt}(\mathsf{ID}_j)$ *priori to the acceptance of oracle* π_i^s.

Security Experiment $\mathsf{EXP}_{\Sigma,\mathcal{A}}^{\mathsf{GAKE}}(\kappa)$: On input security parameter 1^κ, the security experiment is proceeded as a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ based on an AKE protocol Σ, where the following steps are performed:

1) At the beginning of the game, the challenger $\mathcal{C}$ implements the collection of oracles $\{\pi_i^s : i \in [\ell], s \in [\rho]\}$, and generates ℓ long-term key pairs $(pk_{\mathsf{ID}_i}, sk_{\mathsf{ID}_i})$ for all honest parties ID_i for $i \in [\ell]$ where the identity $\mathsf{ID}_i \in \mathcal{IDS}$ of each party is chosen uniquely. $\mathcal{C}$ gives adversary $\mathcal{A}$ all identities and public keys $\{(\mathsf{ID}_1, pk_{\mathsf{ID}_1}), \ldots, (\mathsf{ID}_\ell, pk_{\mathsf{ID}_\ell})\}$ as input.
2) $\mathcal{A}$ may issue polynomial number of queries as aforementioned, namely $\mathcal{A}$ makes queries: Send, StateReveal, Corrupt, EstablishParty and RevealKey.
3) At some point, $\mathcal{A}$ may issue a $\mathsf{Test}(\pi_i^s)$ query on an oracle π_i^s during the game with only once.
4) At the end of the game, the $\mathcal{A}$ may terminate with returning a bit b' as its guess for b of Test query.
5) Finally, 1 is returned if all following conditions hold:
   - $\mathcal{A}$ has issued a Test query to a fresh oracle π_i^s without failure,
   - $\mathcal{A}$ returned a bit b' which equals to b of Test-query;

   Otherwise 0 is returned.

Let variable $M \in \{\text{g-eCKw, g-eCK-PFS}\}$ denote specific model.

Definition 9 (GAKE Security)**.** *We say that an adversary* $\mathcal{A}$ (M, t, ϵ)*-breaks the* M *security of a correct GAKE protocol* Σ, *if* $\mathcal{A}$ *runs the AKE security game within time* t, *and the following condition holds:*

- *If a* Test *query has been issued to a* M *fresh oracle* π_i^s, *then the probability holds that* $|\Pr[\mathsf{EXP}_{\Sigma,\mathcal{A}}^{\mathsf{GAKE}}(\kappa) = 1] - 1/2| > \epsilon$.

We say that a correct GAKE protocol Σ *is* (M, t, ϵ)*-secure, if there exists no adversary that* (M, t, ϵ)*-breaks the* M *security of* Σ.

Remark 1. Please note that the freshness of g-eCK model [28] is defined based on only the notion of *matching sessions* (MS). Whereas our new proposed models also make use of the notion of origin session (OS) which has less restriction than matching sessions. Namely OS only compares transcript of messages from one protocol move (e.g. sent or received by an oracle) other than all transcript of messages required by MS. Informally speaking, less restriction in freshness definition would provide more power to adversary.

4 A Tripartite AKE Protocol from Bilinear Maps

In this section we present a three party one-round AKE protocol based on symmetric bilinear groups, a target collision resistant hash function and a pseudo-random function family. The new proposed protocol is more efficient and than its predecessor.

Protocol Description. We describe the protocol in terms of the following three parts: Setup, long-term key generation and registration, protocol execution. Please note that the general algorithms (defined in Section 2) are implied in specific part.

Setup: The proposed protocol takes as input the following building blocks which are initialized respectively in terms of the security parameter $\kappa \in \mathbb{N}$: (i) Symmetric bilinear groups $\mathcal{PG} = (\mathbb{G}, g, \mathbb{G}_T, p, e) \xleftarrow{\$} \mathsf{PG.Gen}(1^\kappa)$ and a set of random values $(u_0, u_1, u_2, u_3) \xleftarrow{\$} \mathbb{G}$ and $(U_0, U_1, U_2, U_3) = (e(u_0 g, g), e(u_1, g), e(u_2, g), e(u_3, g)$, (ii) a target collision resistant hash function $\mathsf{TCRHF}(hk_{\mathsf{TCRHF}}, \cdot) : \mathcal{K}_{\mathsf{TCRHF}} \times \mathbb{G} \to \mathbb{Z}_p$, where $hk_{\mathsf{TCRHF}} \xleftarrow{\$} \mathsf{TCRHF.KG}(1^\kappa)$, and (iii) a pseudo-random function family $\mathsf{PRF}(\cdot,\cdot) : \mathbb{G}_T \times \{0,1\}^* \to \mathcal{K}_{\mathsf{GAKE}}$. The system parameter variables encompass $pms := (\mathcal{PG}, \{u_i\}_{0\le i\le 3}, \{U_i\}_{0\le i\le 3}, hk_{\mathsf{TCRHF}})$.

Construction Idea. Our new protocol is motivated to improve the efficiency of the LY [24] protocol. We notice that the consistency check on long-term and ephemeral public keys in LY scheme requires eight pairing operations which is quite inefficiency. Hence we try to reduce the computation cost of the consistency check. Our main idea is to make use the pre-computation value in the target group $\mathbb{G}_T$, and the inversion of Diffie-Hellman key to facilitate the validation of a consistency proof. Namely, we utilize the inversion of a Diffie-Hellman key e.g. $g^{1/a}$ provided together with the proof of g^a, to remove corresponding exponent a in the target group $\mathbb{G}_T$ during verifying process. So that we could use pre-computed values in target group to build the verification equation.

Long-term Key Generation and Registration: On input pms, a party $\hat{A}$ may run an efficient algorithm $(sk_{\hat{A}}, pk_{\hat{A}}) \xleftarrow{\$} \mathsf{ORGKE.KGen}(pms, \hat{A})$ to generate the long-term key pair as: $sk_{\hat{A}} = a \xleftarrow{\$} \mathbb{Z}_p^*, pk_{\hat{A}} = (A, A', t_A)$ where $A = g^a$, $A' = g^{1/a}$, $t_A := (u_0 u_1^{h_A} u_2^{h_A^2} u_3^{h_A^3})^a = (\sum_{i=0}^{3} u_i^{h_A^i})^a$ and $h_A = \mathsf{TCRHF}(A)$.

Protocol Execution: On input pms, the protocol among parties $\hat{A}$, $\hat{B}$ and $\hat{C}$ is executed as Figure 2.

Comparisons. We summarize the comparisons among some existing well-known concrete g-eCK secure one-round AKE protocol (i.e. [24]) in the Table 1 from the following perspectives: (i) the security model; (ii) the security assumptions; (iii) number of long-term (LL) and ephemeral (Eph) keys; (iv) and overall computation cost of considered protocol. In the table, 'Exp' denotes the exponentiation and 'ME' denotes multi-exponentiations, 'Pair' denotes pairing evaluation, 'CBDDH' denotes the Cubic Bilinear Decisional Diffie-Hellman assumption, 'GBDH' denotes the gap Bilinear Diffie-Hellman assumption and 'sCBDDH' denotes the strong Cubic Bilinear Decisional Diffie-Hellman assumption. Let 'Rom' denote the random oracle model and 'Std.' denote the standard model.

It is noticeable that our scheme reduce four expensive pairing operations comparing to the construction [33]. It is remarkable that our new scheme is even more efficient than the one [27] secure in the random oracle model. Hence our proposal can provide much more practical interesting.

Security Result of Proposed Protocol. We show the security result of our proposed protocol in the g-eCKw model via the following theorem.

Theorem 1. *Suppose that the pseudo-random function family* PRF *is* $(t, \epsilon_{\mathsf{PRF}})$*-secure, the* TCRHF *is* $(t, \epsilon_{\mathsf{TCRHF}})$*-secure and the strong Cub Bilinear Decisional Diffie-Hellman assumption is* $(t, \epsilon_{\mathsf{sCBDDH}})$ *hard. Then the proposed protocol is* $(g\text{-}eCKw, t, \epsilon)$*-secure in the sense of Definition 9, such that* $t \approx t'$ *and* $\epsilon \le \frac{(\rho\ell)^2}{2^\lambda} + \epsilon_{\mathsf{TCRHF}} + 14(\rho\ell)^3(\epsilon_{\mathsf{sCBDDH}} + \epsilon_{\mathsf{PRF}})$.

The proof of this theorem is presented in Appendix A.

5 Protocol Transformation from g-eCKw to g-eCK-PFS

Next, we proposed a compiler called as $\mathsf{SIG}(\Sigma)$ which make use of a deterministic SEUF-CMA secure signature scheme $\mathsf{SIG} = (\mathsf{SIG.Gen}, \mathsf{SIG.Sign}, \mathsf{SIG.Vfy})$ to transform any g-eCKw secure one-round group key exchange protocols Σ to provide g-eCK-PFS security. Our compiler is generalized from the CF compiler [12]. Namely we digital sign each out-going ephemeral key. Moreover, we assume that the signature scheme is executed on secure device (where the long-term private key is stored). So that no state can be revealed from the signature scheme for simplicity (see more discussion about modelling session states in [33]). The compiler is depicted in Figure 3. In contrast to the original ORGKE protocol, the transformation only adds signature to each outgoing message generated by $\mathsf{ORGKE.MF}$ (without changing the session key generation algorithm).

By applying the $\mathsf{SIG}(\Sigma)$ compiler, we show the resultant protocol is secure in the g-eCK-PFS model as long as the original protocol Σ is g-eCKw secure.

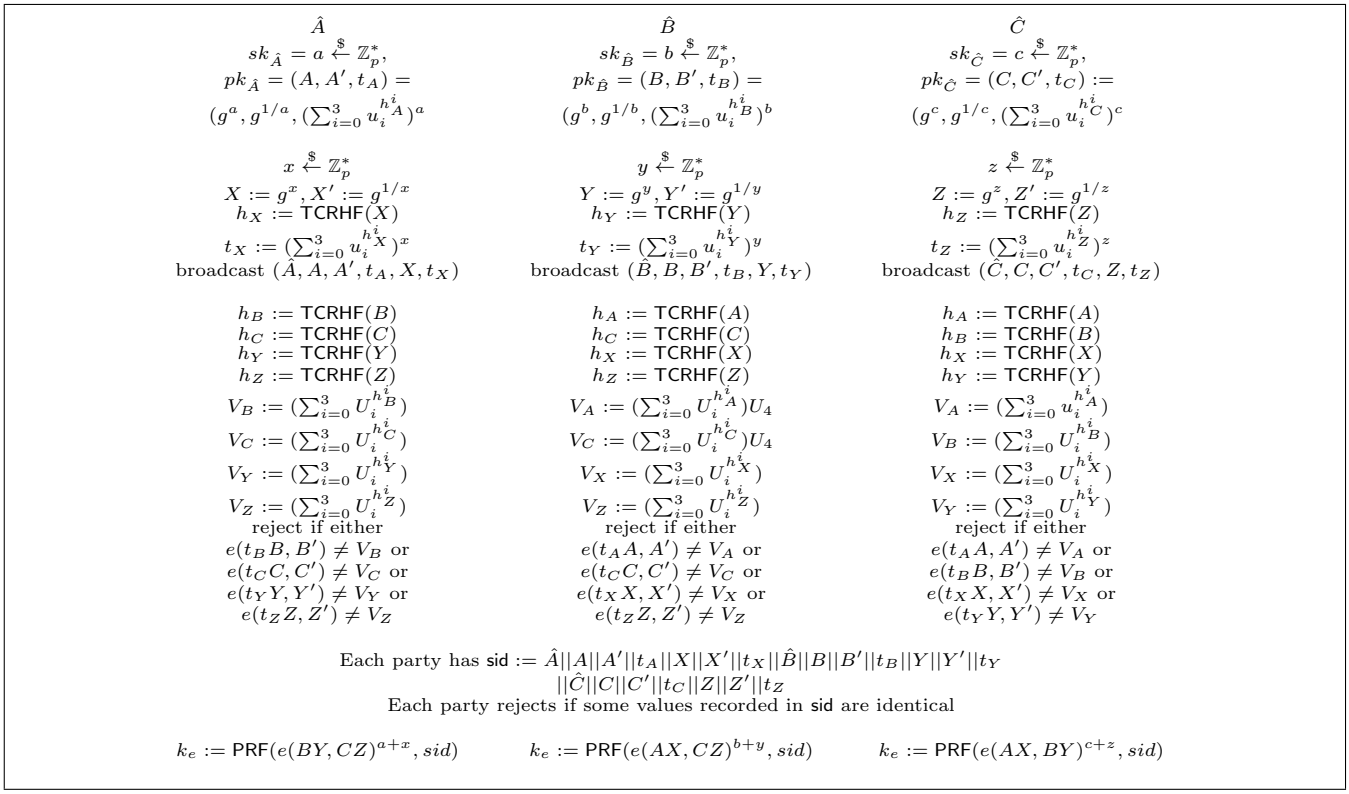

Figure 2: One-round tripartite AKE protocol

Table 1: Comparisons

	Security model	Security assumptions	LL (pk,sk)	Eph (pk,sk)	Overall cost
[27]	g-eCK	ROM, GBDH	(1,1)	(1,1)	9 Exp, 4.Pair
[24]	g-eCK	Std, TCR, PRF, CBDDH	(1,2)	(1,2)	2 Exp, 5 ME, 9 Pair
Proposed	g-eCKw	TCR, PRF, sCBDDH	(1,3)	(1,3)	3 Exp, 5 ME, 5 Pair

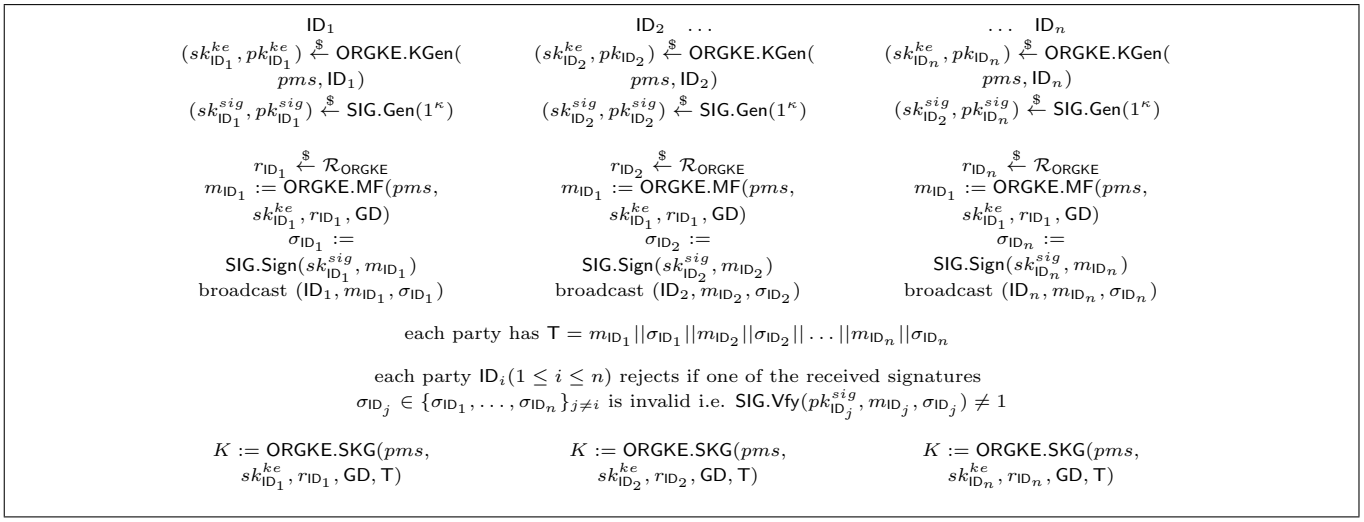

Figure 3: Signature-based generic compiler

Theorem 2. *For any* ORGKE *protocol* Σ*, if* Σ *is* $t \approx t'$ *and* $\epsilon' \leq \ell \cdot \epsilon_{\mathsf{SIG}} + 2 \cdot \epsilon_{g\text{-}eCKw}$*.* $(g\text{-}eCKw, t, \epsilon_{g\text{-}eCKw})$*-secure and the signature scheme* SIG *is deterministic and* $(t, \epsilon_{\mathsf{SIG}})$*-secure, then the protocol* $\mathsf{SIG}(\Sigma)$ *is* $(g\text{-}eCK\text{-}PFS, t', \epsilon_{g\text{-}eCK\text{-}PFS})$*-secure, such that*

The proof of this theorem is shown in Appendix B.

6 Conclusions

We have shown how to model perfect forward secrecy for on one-round group key exchange by introducing two new security models called as g-eCKw and g-eCK-PFS. We also showed a new practical construction for one-round group key exchange protocol which is the first one which can be proven g-eCKw secure in the standard model. Our proposal is more efficient than previous g-eCK secure protocol without random oracles. Our construction idea (in particular for the new consistency check) can be applied to other pairing based protocols with weak programmable hash function that may yield more efficient schemes. Furthermore, it is possible to transform our proposal or any other g-eCKw secure protocols to satisfy g-eCK-PFS security following the new compiler. It is an interesting open problem to formally consider generic constructions for g-eCKw secure AKE in the standard model.

Acknowledgments

This study was supported by Scientific and Technological Research Program of Chongqing Municipal Education Commission (Grant No. KJ1500918).

References

[1] S. S. Al-Riyami and K. G. Paterson, "Tripartite authenticated key agreement protocols from pairings," in *9th IMA International Conference on Cryptography and Coding*, LNCS 2898, pp. 332–359, Springer, Dec. 2003.

[2] M. Bellare and P. Rogaway, "Entity authentication and key distribution," in *Advances in Cryptology (CRYPTO'93)*, LNCS 773, pp. 232–249, Springer, Aug. 1994.

[3] M. Bellare and P. Rogaway, "The security of triple encryption and a framework for code-based game-playing proofs," in *Advances in Cryptology (EUROCRYPT'06)*, LNCS 4004, pp. 409–426, Springer, 2006.

[4] E. Bresson, O. Chevassut, and D. Pointcheval, "Provably authenticated group Diffie-Hellman key exchange – the dynamic case," in *Advances in Cryptology (ASIACRYPT'01)*, LNCS 2248, pp. 290–309, Springer, Dec. 2001.

[5] E. Bresson, O. Chevassut, and D. Pointcheval, "Dynamic group Diffie-Hellman key exchange under standard assumptions," in *Advances in Cryptology (EUROCRYPT'02)*, LNCS 2332, pp. 321–336, Springer, 2002.

[6] E. Bresson, O. Chevassut, and D. Pointcheval, "Dynamic group diffie-hellman key exchange under standard assumptions," in *Advances in Cryptology (EUROCRYPT'02)*, LNCS 2332, pp. 321–336, Springer, 2002.

[7] E. Bresson, O. Chevassut, D. Pointcheval, and J. J. Quisquater, "Provably authenticated group Diffie-Hellman key exchange," in *ACM 8th Conference on Computer and Communications Security (CCS'01)*, pp. 255–264, Nov. 2001.

[8] E. Bresson and M. Manulis, "Securing group key exchange against strong corruptions," in *ACM 3rd Conference on Computer and Communications Security (ASIACCS'08)*, pp. 249–260, Mar. 2008.

[9] E. Bresson, M. Manulis, and J. Schwenk, "On security models and compilers for group key exchange protocols," in *2nd International Workshop on Security, Advances in Information and Computer Security (IWSEC'07)*, LNCS 4752, pp. 292–307, Springer, Oct. 2007.

[10] R. Canetti and H. Krawczyk, "Analysis of key-exchange protocols and their use for building secure channels," in *Advances in Cryptology (EUROCRYPT'01)*, LNCS 2045, pp. 453–474, Springer, May 2001.

[11] T. Yi Chang, M. S. Hwang, and W. P. Yang, "A communication-efficient three-party password authenticated key exchange protocol," *Information Sciences*, vol. 181, no. 1, pp. 217–226, 2011.

[12] C. J. F. Cremers and M. Feltz, "Beyond eCK: Perfect forward secrecy under actor compromise and ephemeral-key reveal," in *17th European Symposium on Research in Computer Security (ESORICS'12)*, LNCS 7459, pp. 734–751, Springer, Sep. 2012.

[13] C. J. F. Cremers and M. Feltz, "Beyond eCK: perfect forward secrecy under actor compromise and ephemeral-key reveal," *Designs, Codes and Cryptography*, vol. 74, no. 1, pp. 183–218, 2015.

[14] A. Fujioka, M. Manulis, K. Suzuki, and B. Ustaoglu, "Sufficient condition for ephemeral key-leakage resilient tripartite key exchange," in *17th Australasian Conference on Information Security and Privacy (ACISP'12)*, LNCS 7372, pp. 15–28, Springer, July 2012.

[15] M. C. Gorantla, C. Boyd, and J. M. González Nieto, "Universally composable contributory group key exchange," in *4th ACM Conference on Computer and Communications Security (ASIACCS'09)*, pp. 146–156, Mar. 2009.

[16] M. C. Gorantla, C. Boyd, and J. M. González Nieto, "Modeling key compromise impersonation attacks on group key exchange protocols," in *12th International Conference on Theory and Practice of Public Key Cryptography (PKC'09)*, LNCS 5443, pp. 105–123, Springer, Mar. 2009.

[17] D. He, C. Chen, M. Ma, S. Chan, and J. Bu, "A secure and efficient password-authenticated group key exchange protocol for mobile ad hoc networks," *International Journal Communication Systems*, vol. 26, no. 4, pp. 495–504, 2013.

[18] A. Joux, "A one round protocol for tripartite Diffie-Hellman," *Journal of Cryptology*, vol. 17, pp. 263–276, Sept. 2004.

[19] J. Katz and J. S. Shin, "Modeling insider attacks on group key-exchange protocols," in *ACM 12th Conference on Computer and Communications Security (CCS'05)*, pp. 180–189, Nov. 2005.

[20] J. Katz and M. Yung, "Scalable protocols for authenticated group key exchange," in *Advances in Cryptology (CRYPTO'03)*, LNCS 2729, pp. 110–125, Springer, Aug. 2003.

[21] H. Krawczyk, "HMQV: A high-performance secure Diffie-Hellman protocol," in *Advances in Cryptology (CRYPTO'05)*, LNCS 3621, pp. 546–566, Springer, Aug. 2005.

[22] B. A. LaMacchia, K. Lauter, and A. Mityagin, "Stronger security of authenticated key exchange," in *1st International Conference on Provable Security (ProvSec'07)*, LNCS 4784, pp. 1–16, Springer, Nov. 2007.

[23] T. Fu Lee, I. P. Chang, and C. C. Wang, "Efficient three-party encrypted key exchange using trapdoor functions," *Security and Communication Networks*, vol. 6, no. 11, pp. 1353–1358, 2013.

[24] Y. Li and Z. Yang, "Strongly secure one-round group authenticated key exchange in the standard model," in *CANS*, LNCS 8257, pp. 122–138, Springer, 2013.

[25] Y. Li and Z. Yang, "Strongly secure one-round group authenticated key exchange in the standard model," Cryptology ePrint Archive, Report 2013/393, 2013. (`http://eprint.iacr.org/`)

[26] Y. Li, D. Chen, W. Li, G. Wang, and S. Paul, "A hybrid authenticated group key agreement protocol in wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2013, 2013. (`http://www.hindawi.com/journals/ijdsn/2013/716265/`)

[27] M. Manulis, K. Suzuki, and B. Ustaoglu, "Modeling leakage of ephemeral secrets in tripartite/group key exchange," in *12th International Conference on Information Security and Cryptology (ICISC'09)*, LNCS 5984, pp. 16–33, Springer, Dec. 2010.

[28] M. Manulis, K. Suzuki, and B. Ustaoglu, "Modeling leakage of ephemeral secrets in tripartite/group key exchange," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 96-A, no. 1, pp. 101–110, 2013.

[29] A. P. Sarr, P. Elbaz-Vincent, and J. C. Bajard, "A new security model for authenticated key agreement," in *7th International Conference on Security in Communication Networks (SCN'10)*, LNCS 6280, pp. 219–234, Springer, Sept. 2010.

[30] V. Shoup, "Sequences of games: A tool for taming complexity in security proofs," Cryptology ePrint Archive, Report 2004/332, 2004. (`http://eprint.iacr.org/`)

[31] T. Y. Wu, Y. M. Tseng, and T. T. Tsai, "A revocable id-based authenticated group key exchange protocol with resistant to malicious participants," *Computer Networks*, vol. 56, no. 12, pp. 2994–3006, 2012.

[32] C. Xu, H. Guo, Z. Li, and Yi Mu, "New construction of affiliation-hiding authenticated group key agreement," *Security and Communication Networks*, vol. 6, no. 6, pp. 723–734, 2013.

[33] Z. Yang, "Efficient eCK-secure authenticated key exchange protocols in the standard model," in *15th International Conference on Information and Computer Security (ICICS'13)*, LNCS 8233, pp. 185–193, Springer, 2013.

[34] Z. Yang, Wu Yang, L. Zhu, and D. Zhang, "Towards modelling perfect forward secrecy in two-message authenticated key exchange under ephemeral-key revelation," *Security and Communication Networks*, vol. 8, no. 12, 2015.

A Proof of Theorem 1

The proof of this Theorem 1 is quite similar to the [24, 25, Theorem 1]. We only show a proof idea here to avoid repetition.

Let oracle $\pi_{\mathsf{ID}_1}^{s^*}$ be the test oracle with intended communication partners ID_2 and ID_3. To prove the security of a protocol in the g-eCK model, it is necessary to show the proof under all possible freshness cases (relevant to $\mathsf{StateReveal}$ and $\mathsf{Corrupt}$ queries) which are formulated by Definition 7. Following the similar approach in [24], we may obtain 14 detailed freshness cases at all. In each freshness case, there are three distinct (long-term or ephemeral) secrets from test oracle or its partner oracle or origin-oracle are not compromised by adversary. The proof proceeds in a sequence of games, following [30, 3]. Let S_δ be the event that the adversary wins the security experiment in Game δ. Let $\mathsf{ADV}_\delta := \Pr[\mathsf{S}_\delta] - 1/2$ denote the advantage of $\mathcal{A}$ in Game δ.

Game 0 This is the original game with adversary $\mathcal{A}$. The system parameters are chosen honestly by challenger as protocol specification. Thus we have that $\Pr[\mathsf{S}_0] = 1/2 + \epsilon = 1/2 + \mathsf{ADV}_0$.

Game 1 In this game we want to make sure that the received ephemeral keys are correctly formed. Technically, we add an abort condition, namely the challenger proceeds exactly as before, but it aborts if there exist two distinct (either ephemeral or long-term) public keys W and N such that $\mathsf{TCRHF}(W) = \mathsf{TCRHF}(N)$. Meanwhile the probability that two oracles output the same ephemeral key is bound by birthday paradox. According to the security property of underlying hash function. Thus we have $\mathsf{ADV}_0 \leq \mathsf{ADV}_1 + \frac{(\rho\ell)^2}{2^\lambda} + \epsilon_{\mathsf{TCRHF}}$.

Game 2 This game proceeds as previous game, but $\mathcal{C}$ aborts if one of the following guesses fails: (i) the freshness case occurred to test oracle from all 14 possibilities, (ii) the test oracle, (iii) the intended communication partners of test oracle, and (iv) every oracles (if they exist in terms of specific guessed freshness case) which

have origin-session to test oracle. The probability that all above guesses of $\mathcal{C}$ are correct is at least $\frac{1}{14\rho^3\ell^3}$. Thus we have that $\mathsf{ADV}_1 \leq 14(\rho\ell)^3 \cdot \mathsf{ADV}_2$.

Game 3 Please note that the g-eCKw freshness definition guarantees that for our protocol there are at least 3 Diffie-Hellman (DH) keys from all session participants of fresh test oracle are not compromised by adversary. We call such guessed 3 uncompromised DH keys as *target DH keys*. This game is proceeded as previous game, but the challenger $\mathcal{C}$ replaces the key material k_i^s with random value $\widetilde{k_i^s}$ for oracles $\{\pi_i^s : i \in [\ell], s \in [\rho]\}$ which satisfy the following conditions:

- The k_i^s is computed involving the 3 *target DH keys* which are guessed by $\mathcal{C}$ for test oracle, and
- Those *target DH keys* used by π_i^s are from 3 distinct parties.

The above two conditions ensure that the changed key materials of oracles can not be trivially generated by adversary. This also enables us to embed sCBDDH challenge instance into the simulation of all oracles satisfying above conditions. The second condition is used to exclude the situation that the DH keys from some party are all compromised in which case the adversary can simply compute the session key.

The proof in this game is quite similar to the proof of [25, Theorem 1] but the sCBDDH challenge instance is involved instead of CBDDH. By applying the security of sCBDDH assumption, we therefore obtain that $\mathsf{ADV}_2 \leq \mathsf{ADV}_3 + \epsilon_{\mathsf{sCBDDH}}$.

Game 4 In this game, we change function $\mathsf{PRF}(\widetilde{k_i^*}, \cdot)$ to a truly random function for test oracle and its partner oracles (if they exist). Exploiting the security of PRF, we have that $\mathsf{ADV}_3 \leq \mathsf{ADV}_4 + \epsilon_{\mathsf{PRF}}$.

Note that in this game the session key returned by Test-query is totally a truly random value which is independent to the bit b and any messages. Thus the advantage that the adversary wins this game is $\mathsf{ADV}_4 = 0$.

Sum up the probabilities from Game 0 to Game 4, we proved this theorem.

B Proof of Theorem 2

Since a correct g-eCKw protocol must also be g-eCK-PFS protocol. In the sequel, we wish to show that the adversary is unable to distinguish random value from the session key of any g-eCK-PFS oracle. Please first note that the g-eCKw freshness and g-eCK-PFS freshness only differ in the last condition, i.e. when there is no origin-oracle to test oracle. In other freshness cases, those two freshness notions are the same. Hence, if we can show that the test oracle always has origin-oracle before its intended partner is corrupt, then the proof would go through.

In the following, we use the superscript '*' to highlight corresponding values processed in test oracle $\pi_i^{s^*}$ which has intended communication partner ID_u. Let π_i^s be an accepted oracle with intended partner ID_j. Let π_j^t be an oracle (if it exists) with intended partner ID_i, such that π_i^s has a matching session to π_j^t. Let π_v^l be an oracle (if it exists), such that π_v^l has a origin session to π_i^s. Let S_δ be the event that the adversary wins the security experiment under the Game δ and one of the above freshness cases. Let $\mathsf{ADV}_\delta := \Pr[\mathsf{S}_\delta] - 1/2$ denote the advantage of $\mathcal{A}$ in Game δ. We consider the following sequence of games.

Game 0 This is the original g-eCK-PFS security game with adversary $\mathcal{A}$. Thus we have that $\Pr[\mathsf{S}_0] = 1/2 + \epsilon = 1/2 + \mathsf{ADV}_0$.

Game 1 In this game, the challenger proceeds exactly like previous game, except that we add a abortion rule. The challenger raises event $\mathsf{abort}_{\mathsf{trans}}$ and aborts, if during the simulation either the message m_{ID_i} replied by an oracle π_i^s but it has been sample by another oracle π_u^w or sent by adversary before. Since there are $\rho\ell$ such values would be sampled randomly. We claim that the event $\mathsf{abort}_{\mathsf{trans}}$ occurs with probability $\Pr[\mathsf{abort}_{\mathsf{trans}}] \leq \epsilon_{\text{g-eCKw}}$. We elaborate the proof as follows. Please first recall that if the test oracle $\pi_i^{s^*}$ (generating message $m_{\mathsf{ID}_i}^{s^*}$) is fresh then the adversary is not allowed to issue both $\mathsf{Corrupt}(\mathsf{ID}_i)$ and $\mathsf{StateReveal}(\pi_i^{s^*})$, as otherwise the security is trivially broken. However, consider the case that the adversary issued $\mathsf{Corrupt}(\mathsf{ID}_i)$, and at the same time there is another oracle π_j^t which outputs the same messages as the one generated by test oracle. Then the adversary can issue $\mathsf{StateReveal}(\pi_j^t)$ to learn the ephemeral secret relative to $m_{\mathsf{ID}_i}^{s^*}$ without violating the g-eCK-PFS of test oracle. Furthermore, the probability that the collisions among the messages generated by $\mathsf{ORGKE.MF}$ in either protocol Σ or $\mathsf{SIG}(\Sigma)$ is the same. The security of Σ in the g-eCKw model, implies the collision probability among outgoing messages is negligible. We therefore have that $\mathsf{ADV}_0 \leq \mathsf{ADV}_1 + \epsilon_{\text{g-eCKw}}$.

Game 2 This game proceeds exactly as before, but the challenger raises event $\mathsf{abort}_{\mathsf{sig}}$ and aborts if a fresh oracle π_i^s with intended communication partner ID_j received a message m_{ID_j} which is not sent by any oracle of ID_j but the signature computed over m_{ID_j} subjecting to $\mathsf{SIG.Vfy}(pk_{\mathsf{ID}_j}^{sig}, m_{\mathsf{ID}_j}, \sigma_{\mathsf{ID}_j}) = 1$. We have $\mathsf{ADV}_1 \leq \mathsf{ADV}_2 + \Pr[\mathsf{abort}_{\mathsf{sig}}]$.

If the event $\mathsf{abort}_{\mathsf{sig}}$ happens with non-negligible probability, then we could construct a signature forger $\mathcal{F}$ as follows. The forger $\mathcal{F}$ receives as input a public key pk^*, and runs the adversary $\mathcal{A}$ as a subroutine and simulates the challenger for $\mathcal{A}$. It first guesses an index $\theta \xleftarrow{\$} [\ell]$ pointing to the public key for which the adversary is able to forge, and sets $pk_{\mathsf{ID}_\theta} = pk^*$. Next $\mathcal{F}$ generates all other long-term public/secret keys honestly as the challenger in the previous game. The $\mathcal{F}$ guesses the party ID_j (such

that $\theta = j$) correctly with probability $1/\ell$. Then the $\mathcal{F}$ proceeds as the challenger in Game 2, except that it uses its chosen-message oracle to generate a signature under pk_{ID_θ} for the oracles of party ID_θ.

The $\mathcal{F}$ can use the signature received by π_i^s to break the SEUF-CMA security of the underlying signature scheme with success probability ϵ_{SIG}. So the event $\mathsf{abort}_{\mathsf{sig}}$ happens with the probability $\frac{\Pr[\mathsf{abort}_{\mathsf{sig}}]}{\ell} \leq \epsilon_{\mathsf{SIG}}$. Therefore we have $\mathsf{ADV}_1 \leq \mathsf{ADV}_2 + \ell \cdot \epsilon_{\mathsf{SIG}}$.

So in Game 2 each accepting g-eCKw fresh oracle π_i^s with intended communication partner ID_j, there always exists an oracle π_j^t which has origin session to π_i^s. That means the last condition of both g-eCK-PFS freshness and g-eCKw freshness would never occurred in this game.

Game 3 This game is proceeded as previous game, but the challenger $\mathcal{C}$ replaces the session key of test oracle and its partner oracle (if it exists) with a uniform random value. If there exists an adversary $\mathcal{A}$ can distinguish the Game 3 from Game 2 then we can use it to construct an adversary $\mathcal{B}$ to break the g-eCKw-security of Σ.

Intuitively, the security reduction from g-eCK-PFS to g-eCKw is possible in this game, since both g-eCK-PFS and g-eCKw encompass the same freshness cases (related to $\mathsf{StateReveal}$ and $\mathsf{Corrupt}$ queries) when the test oracle has origin-oracle. We elaborate the simulation as follows. Let $\mathcal{B}$ be an adversary which interacts with an g-eCKw challenger $\mathcal{C}$ and tries to breaks the g-eCKw security of Σ in the g-eCKw security game. $\mathcal{B}$ runs $\mathcal{A}$ (who is a successful g-eCK-PFS attacker) as subroutine and simulates the challenger for $\mathcal{A}$ as previous game. For each party ID_i ($i \in [\ell]$), $\mathcal{B}$ generate an extra pair of long-term keys $(sk_{\mathsf{ID}_i}^{sig}, pk_{\mathsf{ID}}^{sig}) \xleftarrow{\$} \mathsf{SIG.Gen}(1^\kappa)$ and gives all public keys to $\mathcal{A}$ at beginning of the game. For every oracle $\{\pi_i^s : i \in [\ell], s \in [d]\}$ simulated by $\mathcal{C}$, $\mathcal{B}$ keeps corresponding a dummy oracle $\pi_i^{s'}$ and the adversary $\mathcal{A}$ is able to interacts with those dummy oracles simulated by $\mathcal{B}$. Specifically, a dummy oracle proceeds as follows:

- For any $\mathsf{Send}(\pi_i^{s'}, m)$ query from $\mathcal{A}$, if $m \neq (\top, \widetilde{\mathsf{ID}_j})$ and the signature in m is valid then $\mathcal{B}$ peels off the signature value from m to obtain a truncated message m' and issues $m^* \leftarrow \mathsf{Send}(\pi_i^s, m)$. Meanwhile if $m(\top, \widetilde{\mathsf{ID}_j})$ then $\mathcal{B}$ just issues $m^* \leftarrow \mathsf{Send}(\pi_i^s, m)$. To this end, $\mathcal{B}$ does $\sigma_{m^*} \xleftarrow{\$} \mathsf{SIG.Sign}(sk_{\mathsf{ID}_i}, m^*)$ and returns (m^*, σ_{m^*}) to $\mathcal{A}$.

- For any $\mathsf{Corrupt}(\mathsf{ID}_i)$ ($i \in [\ell]$) query, $\mathcal{B}$ asks $\mathsf{Corrupt}(\mathsf{ID}_i)$ to $\mathcal{C}$ to obtain $sk_{\mathsf{ID}_i}^{ke}$ and returns $(sk_{\mathsf{ID}_i}^{ke}, sk_{\mathsf{ID}_i}^{sig})$ to $\mathcal{A}$.

- For any other oracles queries on $\pi_i^{s'}$ (including Test query), $\mathcal{B}$ just asks corresponding oracles queries on π_i^s to $\mathcal{C}$ and returns the results to $\mathcal{A}$.

So that $\mathcal{B}$ is able to perfectly simulate the environment for $\mathcal{A}$. If the session key returned by Test query is a true key, then the simulation is exactly the same as previous game, otherwise it is equivalent to this game. Finally, $\mathcal{B}$ returns what $\mathcal{A}$ returns to $\mathcal{C}$. If $\mathcal{A}$ wins the game with non-negligible probability, so does $\mathcal{B}$. Thus we have that $\mathsf{ADV}_2 \leq \mathsf{ADV}_3 + \epsilon_{\text{g-eCKw}}$.

In this game, the session key given to adversary is independent of the bit b of Test query, thus $\Pr[\mathsf{S}_3^1] = 0$. Sum up the probabilities from Game 0 to Game 3, we proved this theorem.

Zheng Yang was born in 1982. He received the Master degree in Computer Science from Chongqing University in 2009. He got the doctor degree in IT-security from Ruhr-University Bochum, Germany in 2013. He is a researcher at Chongqing University of Technology, China. His main research interests include information security and cryptography.

Daigu Zhang was born in 1981. He received the Master degree in Computer Science from Chongqing University in 2008. He got the doctor degree from Chongqing University in 2013. He is a Lecturer at Chongqing University of Technology, China. His main research interests include information security and cryptography.