Journal of Universal Computer Science, vol. 14, no. 3 (2008), 377-396
submitted: 2/6/07, accepted: 1/11/07, appeared: 1/2/08 © J.UCS

Parallel Key Exchange

Ik Rae Jeong
(Graduate School of Information Security CIST, Korea University, Seoul, Korea
irjeong@korea.ac.kr)

Dong Hoon Lee
(Graduate School of Information Security CIST, Korea University, Seoul, Korea
donghlee@korea.ac.kr)

Abstract: In the paper we study parallel key exchange among multiple parties. The status of parallel key exchange can be depicted by a *key graph*. In a key graph, a vertex represents a party and an edge represents a relation of two parties who are to share a key.

We first propose a security model for a key graph, which extends the Bellare-Rogaway model for two-party key exchange. Next, we clarify the relations among the various security notions of key exchange. Finally, we construct an efficient key exchange protocol for a key graph using the *randomness re-use* technique. Our protocol establishes the multiple keys corresponding to all edges of a key graph in a *single* session. The security of our protocol is proven in the standard model.

Key Words: key exchange, key graph, security notions, randomness re-use
Category: C.2.0, E.3

1 Introduction

Key exchange protocols enable two or more parties to establish a common *session key*. The distributed systems like file sharing system, database system, broadcasting radio/TV system, and audio/video conference system require key establishment between communicating parties at the same time. These requirements of key establishment can be conceptually represented by a graph, called a *key graph*, where each vertex represents a party and each edge represents a relation of two parties who are to share a key.

There exist several useful structures of key graphs. A star structure may be used to play an on-line card game, where a dealer and players need to establish the common keys. A complete structure may be used in the on-line conference, where each pair of members in the conference needs to communicate secretly. A tree structure may be used in a company, where the hierarchy of personnel in the company is described by a tree structure.

In this paper, we study a key exchange method for key graphs to simultaneously generate multiple keys corresponding to all edges of the key graph in a single session, more efficiently than by running parallel executions of a two-party protocol.

1.1 Security Notions

We briefly recall various notions of security for key exchange protocols (formal definitions are given in Section 3). This paper concerns protocols for *authenticated* key exchange (AKE) using public-key authentication. The following is adapted from [Jeong et al. 2006].

At the most basic level, an authenticated key-exchange protocol must provide secrecy of a generated session key. To completely define the notions of security, we must consider adversarial behaviors which should be tolerated by a protocol. The *key independence* (KI) considers the case that some session keys are revealed. A bit more formally, key independence protects against "Denning-Sacco" attacks [Denning et al. 1981] involving compromise of multiple session keys (for sessions other than the one whose secrecy must be guaranteed).

Protocols achieving *forward secrecy* (FS) maintain secrecy of session keys even if an adversary is able to obtain long-term secret keys of parties who have previously generated a common session key in an honest execution of the protocol. This type of forward secrecy is known as weak forward secrecy in the literature. However, in the real system, it is highly feasible that the adversary interferes in the process of session key establishment. To incorporate this feasibility, forward secrecy in this paper is defined in a strong sense, additionally requiring secrecy of session keys which have been generated even with interference of the adversary.

The above notions are most widely used in key exchange protocols. Besides above security notions, there are various security notions such as key compromise impersonation and unknown key share [Blake-Wilson et al. 1998, Law et al. 2003, Menezes et al. 1995]. If an adversary obtains a long-term secret key of a party, the adversary can trivially impersonate the party to the other parties. But the adversary may not impersonate other parties to the party. A protocol is secure against *key compromise impersonation* (KCI) attacks, if an adversary can not impersonate other parties (whose long-term secret keys are not revealed) to the parties (whose long-term secret keys are revealed). The security against *session state reveal* (SSR) is formally considered in [Canetti et al. 2001, Krawczyk 2005]. This security is originated from the consideration that the random values of the sessions may be more easily leaked than the secret keys of the public keys. A protocol is secure against unknown key share (UKS) attacks, if the following holds: If two parties *Alice* and *Bob* compute the same session key, *Alice* should consider that she is establishing the session key with *Bob* and *Bob* should consider that he is establishing the session key with *Alice*. In our security model in Section 3, FS implies KCI, KCI implies KI, and KI implies UKS.

1.2 The Related Works and Our Contributions

To simultaneously generate multiple keys in a single session, a naive approach would be to execute a two-party key exchange protocol in parallel. In this case, a party executes a two-party key exchange protocol for each key independently, thus using independent random numbers for each key. But, we can improve the computational efficiency by using the same random number for different keys. This "randomness re-use technique" has been used in multi-recipient encryption schemes to reduce bandwidth and computational cost [Kurosawa 2002, Bellare et al. 2003].

Our main contributions are as follows:

1. We define a security model for a key graph, which extends the Bellare-Rogaway model in [Bellare et al. 1993] for two-party key exchange. Our security model for a key graph incorporates a remarkable list of security properties stated in [Krawczyk 2005] such as FS, KCI, KI, UKS and SSR.
2. We clarify the relations between the security notions. We prove that KI implies UKS and FS implies KCI. From these results, we just need to prove that the protocol provides FS and SSR to show that a key exchange protocol satisfies all security notions.
3. We suggest an efficient key exchange protocol for a key graph using the randomness re-use technique. Our protocol is secure in the standard model.

2 Preliminaries

In this section we review the well-known definitions of primitives which we use to construct a key exchange protocol for key graphs. We use notation $[a, b]$ for a set of integers from a to b. We use notation $c \leftarrow S$ to denote that c is randomly selected from a set S. We denote the concatenation of two strings a and b as $a||b$. If evt is an event, $\mathsf{Pr}[\mathsf{evt}]$ is a probability that evt occurs.

2.1 Pseudorandom Functions [Goldreich et al. 1986]

Let θ be a security parameter. Let $\mathsf{F}_K : \{0,1\}^\theta \rightarrow \{0,1\}^\theta$ be a function selected from a function family F where

$$\mathsf{F} = \{\mathsf{F}_K | K \text{ is in the space of } \theta\text{-bit strings}\}.$$

Let $\mathsf{Rand}^{\{0,1\}^\theta \rightarrow \{0,1\}^\theta}$ be a set of all functions from domain $\{0,1\}^\theta$ to range $\{0,1\}^\theta$. We consider two experiments:

$\mathbf{Exp}^{\text{PRF-1}}_{\mathsf{F},\mathcal{A}}(\theta)$	$\mathbf{Exp}^{\text{PRF-0}}_{\mathsf{F},\mathcal{A}}(\theta)$
$K \leftarrow \{0,1\}^{\theta}$	$h \leftarrow \mathsf{Rand}^{\{0,1\}^{\theta} \rightarrow \{0,1\}^{\theta}}$
$d \leftarrow \mathcal{A}^{\mathsf{F}_K(\cdot)}(1^{\theta})$	$d \leftarrow \mathcal{A}^{h(\cdot)}(1^{\theta})$
return d	return d

The advantage of an adversary $\mathcal{A}$ is defined as follows:

$$\mathsf{Adv}^{\text{PRF}}_{\mathsf{F},\mathcal{A}}(\theta) = \mathsf{Pr}[\mathbf{Exp}^{\text{PRF-1}}_{\mathsf{F},\mathcal{A}}(\theta) = 1] - \mathsf{Pr}[\mathbf{Exp}^{\text{PRF-0}}_{\mathsf{F},\mathcal{A}}(\theta) = 1].$$

The advantage function is defined as follows:

$$\mathsf{Adv}^{\text{PRF}}_{\mathsf{F}}(\theta, t, q, \mu) = \max_{\mathcal{A}}\{\mathsf{Adv}^{\text{PRF}}_{\mathcal{A}}\},$$

where $\mathcal{A}$ is any adversary with time complexity t making at most q oracle queries and the sum of the length of these queries being at most μ bits. The scheme F is a secure pseudorandom function family if the advantage of any adversary $\mathcal{A}$ with time complexity polynomial in θ is negligible.

2.2 Decisional Diffie-Hellman Problem [Diffie et al. 1976]

Let $\theta \in N$ be a security parameter. Let $\mathcal{GG}$ be a group generator which generates $(\mathbb{G}, q, g)$. $\mathbb{G}$ is a group with prime order q and generator g. Consider the following experiment:

$\mathbf{Exp}^{\text{DDH-1}}_{\mathcal{A}_{\text{DDH}}}(\theta)$	$\mathbf{Exp}^{\text{DDH-0}}_{\mathcal{A}_{\text{DDH}}}(\theta)$
$(\mathbb{G}, q, g) \leftarrow \mathcal{GG}(1^{\theta})$	$(\mathbb{G}, q, g) \leftarrow \mathcal{GG}(1^{\theta})$
$u_1, u_2 \leftarrow [1, q]$	$u_1, u_2, w \leftarrow [1, q]$
$U_1 \leftarrow g^{u_1}; U_2 \leftarrow g^{u_2}$	$U_1 \leftarrow g^{u_1}; U_2 \leftarrow g^{u_2}$
$W \leftarrow g^{u_1 u_2}$	$W \leftarrow g^{w}$
$d \leftarrow \mathcal{A}_{\text{DDH}}(\mathbb{G}, q, g, U_1, U_2, W)$	$d \leftarrow \mathcal{A}_{\text{DDH}}(\mathbb{G}, q, g, U_1, U_2, W)$
return d	return d

The advantage of an adversary $\mathcal{A}_{\text{DDH}}$ is defined as follows:

$$\mathsf{Adv}^{\text{DDH}}_{\mathcal{A}_{\text{DDH}}}(\theta) = \mathsf{Pr}[\mathbf{Exp}^{\text{DDH-1}}_{\mathcal{A}_{\text{DDH}}}(\theta) = 1] - \mathsf{Pr}[\mathbf{Exp}^{\text{DDH-0}}_{\mathcal{A}_{\text{DDH}}}(\theta) = 1].$$

The advantage function is defined as follows:

$$\mathsf{Adv}^{\text{DDH}}_{\mathcal{GG}}(\theta, t) = \max_{\mathcal{A}}\{\mathsf{Adv}^{\text{DDH}}_{\mathcal{A}_{\text{DDH}}}(\theta)\},$$

where $\mathcal{A}_{\text{DDH}}$ is any adversary with time complexity t. The DDH assumption is that the advantage of any adversary $\mathcal{A}_{\text{DDH}}$ with time complexity polynomial in θ is negligible.

2.3 Strong Unforgeability (SUF) of Signature Scheme [An et al. 2002]

A signature scheme S consists of three algorihms ($\mathsf{S.key}, \mathsf{S.sign}$, and $\mathsf{S.ver}$). $\mathsf{S.key}$ generates a pair of private-/public-keys for a signer. $\mathsf{S.sign}$ generates a signature for a message with the private key. $\mathsf{S.ver}$ verifies the message-signature pair with the public key and returns 1 if valid or 0 otherwise.

Let $\theta \in N$ be a security parameter and S be a signature scheme. Consider the following experiment:

$\mathbf{Exp}^{\mathrm{SUF}}_{\mathsf{S},\mathcal{A}_{\mathrm{SUF}}}(\theta)$
$(sk, vk) \leftarrow \mathsf{S.key}(1^\theta)$
$\omega \leftarrow \mathcal{A}^{\mathsf{S.sign}_{sk}(\cdot)}(vk)$
if $\omega = \perp$ then return 0
else parse ω as (M, σ)
if $\mathsf{S.ver}_{vk}(M, \sigma) = 1$ and signing oracle $\mathsf{S.sign}_{sk}(\cdot)$ has
never returned σ on input M then return 1
else return 0

The advantage of an adversary $\mathcal{A}_{\mathrm{SUF}}(\theta)$ is defined as follows:

$$\mathsf{Adv}^{\mathrm{SUF}}_{\mathsf{S},\mathcal{A}_{\mathrm{SUF}}}(\theta) = \mathsf{Pr}[\mathbf{Exp}^{\mathrm{SUF}}_{\mathsf{S},\mathcal{A}_{\mathrm{SUF}}}(\theta) = 1].$$

The advantage function of the scheme is defined as follows:

$$\mathsf{Adv}^{\mathrm{SUF}}_{\mathsf{S}}(\theta, t, q_s) = \max_{\mathcal{A}}\{\mathsf{Adv}^{\mathrm{SUF}}_{\mathsf{S},\mathcal{A}_{\mathrm{SUF}}}(\theta)\},$$

where $\mathcal{A}_{\mathrm{SUF}}$ is any adversary with time complexity t making at most q_s signing queries. The scheme S is SUF–secure if the advantage of any adversary $\mathcal{A}_{\mathrm{SUF}}$ with time complexity polynomial in θ is negligible.

3 A Key Exchange Model

The security model in [Bellare et al. 1993], called Bellare-Rogaway (BR) model, considers KI for two-party key exchange. The security models in [Krawczyk 2005, Jeong et al. 2006] consider FS, KCI, UKS and SSR for two-party key exchange. We extend the security model of [Bellare et al. 1993, Jeong et al. 2006] and make the security model for a key graph.

We assume that each party's identity is denoted as P_i, and each party holds a private-/public key pair. Π_i^k represents the k-th instance of player P_i. If a key exchange protocol terminates, then Π_i^k generates the multiple keys for the edges.

Session Identifier of an instance, sid_i^k, is a string different from those of all other sessions in the system (with high probability), and simply the concatenation of all messages sent and received by a particular instance Π_i^k, where the order of these messages is determined *by the lexicographic ordering of the parties' identities*. (Note that ordering messages according to the time they were sent cannot be used when a protocol runs over a broadcasting network since multiple parties may send their messages simultaneously.)

Consider instance Π_i^k of player P_i. An *e-partner* of P_i, denoted as $\mathsf{pid}_i^{k,e}$, is a party with whom P_i believes it is interacting to make a key for an edge e. We define pid_i^k as a set of all partners of P_i in Π_i^k. We say that two instances Π_i^k and $\Pi_j^{k'}$ are *e-partnered*, if $\mathsf{pid}_i^{k,e} = P_j$, $\mathsf{pid}_j^{k',e} = P_i$, and $\mathsf{sid}_i^k = \mathsf{sid}_j^{k'}$ where $e = (i, j)$.

An undirected key graph G_i^k of Π_i^k consists of $\mathsf{V}_i^k = \{P_i\} \cup \mathsf{pid}_i^k$ and a set of edges E_i^k which is defined as follows:

$$\mathsf{E}_i^k = \{(i, j) | P_j \in \mathsf{pid}_i^k \wedge P_i \text{ and } P_j (\neq P_i) \text{ establish a key between them}\}.$$

Note that $\mathsf{pid}_i^k = \bigcup_{e \in \mathsf{E}_i^k} \mathsf{pid}_i^{k,e}$ and $(i, j) = (j, i)$ in an undirected graph. $\mathsf{sk}_i^{k,e}$ denotes a key computed for an edge $e = (i, j)$ of G_i^k. Any protocol should satisfy the following *correctness* condition: if Π_i^k and $\Pi_j^{k'}$ are e-partnered, then $\mathsf{sk}_i^{k,e}$ and $\mathsf{sk}_j^{k',e}$ are equal.

To define a notion of security, we define the capabilities of an adversary. We allow the adversary to potentially control all communication in the network via access to a set of oracles (instances) as defined below. We consider an *experiment* in which the adversary asks queries to oracles, and the oracles answer back to the adversary. Oracle queries model attacks which an adversary may use in the real system. We consider the following types of queries in this paper.

- The query $\mathsf{Initiate}(i, \mathsf{G})$ is used to "prompt" party P_i to initiate an execution of the protocol for given key graph G. P_i sends a protocol message to the adversary.

- A query $\mathsf{Send}(i, k, e, M)$ is used to send a message M to instance Π_i^k as a message from e-partner. When Π_i^k receives M, it responds according to the key-exchange protocol.

- A query $\mathsf{Reveal}(i, k, e)$ models *known key* attacks (or Denning-Sacco attacks) in the real system. The adversary is given the key $\mathsf{sk}_i^{k,e}$ for the specified instance.

- A query $\mathsf{Corrupt}(i)$ models exposure of the secret key corresponding to the public key held by player P_i. The adversary is assumed to be able to obtain secret keys of players, but cannot control the behavior of these players directly (of course, once the adversary has asked a query $\mathsf{Corrupt}(i)$, the adversary may impersonate P_i in subsequent Send queries.)

– A query $\mathsf{State}(i, k, e)$ models exposure of the random values used in making $\mathsf{sk}_i^{k,e}$ in Π_i^k.

– A query $\mathsf{Test}(i, k, e)$ is used to define the advantage of an adversary. When an adversary $\mathcal{A}$ asks a Test query to an *e-fresh* instance (defined below) Π_i^k, a coin b is flipped. If b is 1, then the key $\mathsf{sk}_i^{k,e}$ is returned. Otherwise, a random string chosen uniformly from the space of θ-bit strings is returned, where θ is a security parameter.

To define a meaningful notion of security, we need to define *e-freshness*:

Definition. An instance Π_i^k is *e-fresh* if the following conditions are true at the conclusion of the experiment described above:

(a) The adversary has not queried $\mathsf{Reveal}(i, k, *)$.

(b) Π_i^k is e-partnered with instance $\Pi_j^{k'}$ and the adversary has not queried $\mathsf{Reveal}(j, k', *)$.

(c) The adversary does not control P_i or P_j for $e = (i, j)$. That is, neither P_i nor P_j is an *insider* attacker controlled by the adversary. An insider attacker and its public keys are created by the adversary. And all of the information known to an insider attacker are also known to the adversary, and its behaviors are completely controlled by the adversary.

The following notions of security may then be considered, depending on the types of queries the adversary is allowed to ask:

- KI (Key Independence): An adversary $\mathcal{A}$ can ask Reveal queries, but can not ask $\mathsf{Corrupt}$ or State queries.

- FS (Forward Secrecy): An adversary $\mathcal{A}$ can ask $\mathsf{Corrupt}$ and Reveal queries, but can not ask State queries. It is possible that after corrupting P_j, $\mathcal{A}$ itself may impersonate P_j at a specific session. In this case, $\mathcal{A}$ can trivially find out a session key of this session. To eliminate this trivial case, the e-freshness of Π_i^k requires the following additional condition:
  (d) If the adversary has queried $\mathsf{Corrupt}(j)$ and $\mathsf{Send}(i, k, e, *)$, $\mathsf{Corrupt}(j)$ should have been queried *after* all $\mathsf{Send}(i, k, e, *)$ queries, where $e = (i, j)$.

- KCI (Key Compromise Impersonation): An adversary $\mathcal{A}$ can ask $\mathsf{Corrupt}$ and Reveal queries, but can not ask State queries. Suppose that $\mathcal{A}$ has queried $\mathsf{Corrupt}(i)$ and wants to impersonate P_j to P_i. This is trivial if the adversary has also corrupted P_j. To eliminate this case, the e-freshness of Π_i^k requires the following additional condition:

(d) P_j has not been corrupted, where $e = (i, j)$. Even though an adversary can not corrupt P_j, the adversary can corrupt any party $P_k(\neq P_j)$ including P_i for e-fresh Π_i^k.

- SSR (Session State Reveal): An adversary $\mathcal{A}$ can ask State queries, but can not ask Reveal and $\mathsf{Corrupt}$ queries. Suppose that $\mathcal{A}$ has queried $\mathsf{State}(i, k, e)$ and wants to know session key $\mathsf{sk}_i^{k,e}$. This is trivial if $\mathcal{A}$ has also corrupted P_i. To eliminate this case, the e-freshness of Π_i^k requires that $\mathcal{A}$ can not ask Reveal and $\mathsf{Corrupt}$ queries.

- UKS (Unknown Key Share): An adversary $\mathcal{A}$ can not ask any Reveal, $\mathsf{Corrupt}$, or State queries.

For an adversary $\mathcal{A}$ attacking a scheme in the sense of UKS, an adversary $\mathcal{A}$ outputs (i, k, e) and (j, k', e') at the end of the experiment above, where P_i and P_j are not insider attackers. The advantage of $\mathcal{A}$, denoted $\mathsf{Adv}_{\mathcal{A}}^{\mathrm{UKS}}(\theta)$, is defined as $\mathsf{Pr}[\mathsf{sk}_i^{k,e} = \mathsf{sk}_j^{k',e'} \wedge e \neq e']$.

In all the security notions considered above except UKS, an adversary $\mathcal{A}$ outputs a bit b' at the end of the experiment. The advantage of $\mathcal{A}$, denoted $\mathsf{Adv}_{\mathcal{A}}(\theta)$, is defined as $2 \cdot \mathsf{Pr}[b' = b] - 1$.

For an adversary $\mathcal{A}$ attacking a scheme in the sense of XX (where XX is either KI, FS, KCI, SSR, or UKS), we denote the advantage of this adversary by $\mathsf{Adv}_{\mathcal{A}}^{XX}(\theta)$. For a protocol P, we define its security as:

$$\mathsf{Adv}_P^{XX}(\theta, t) = \max_{\mathcal{A}}\{\mathsf{Adv}_{\mathcal{A}}^{XX}(\theta)\},$$

where the maximum is taken over all adversaries running in time t. A scheme P is said to be XX–secure if $\mathsf{Adv}_P^{XX}(\theta, t)$ is negligible (in θ) for any $t = \mathrm{poly}(\theta)$.

We show that if a key exchange protocol provides FS and SSR, it also provides KCI, KI, and UKS by the following theorems.

Theorem 1. KI (Key Independence) implies UKS (Unknown Key Share) for any key exchange protocol $\mathcal{P}$.

Proof of Theorem 1. To prove the theorem, we construct adversary $\mathcal{B}$ attacking KI using adversary $\mathcal{A}$ attacking UKS. The description of $\mathcal{B}$ is as follows:

1. Using its own oracle query, $\mathcal{B}$ can answer every query $\mathcal{A}$ asks.
2. If $\mathcal{A}$ outputs (i, k, e) and (j, k', e'), $\mathcal{B}$ first makes $\mathsf{Reveal}(j, k', e')$ query and gets a key $\mathsf{sk}_j^{k',e'}$.

3. $\mathcal{B}$ makes $\mathsf{Test}(i,k,e)$ and gets τ. If $\tau = \mathsf{sk}_j^{k',e'}$, $\mathcal{B}$ returns 1. Otherwise $\mathcal{B}$ returns 0.

Since Π_i^k is not e-partnered with $\Pi_j^{k'}$, the strategy of $\mathcal{B}$ is correct. So if $\mathcal{A}$ succeeds to break UKS, $\mathcal{B}$ succeeds to break KI. Thus,

$$\begin{aligned}\frac{\mathsf{Adv}^{\mathrm{KI}}_{\mathcal{P},\mathcal{B}}+1}{2} &= \mathsf{Pr}_{\mathcal{B}}[b=b'] \\ &\geq \mathsf{Pr}_{\mathcal{A}}[\mathsf{sk}_i^{k,e} = \mathsf{sk}_j^{k',e'} \wedge e \neq e'] - \frac{1}{2^\theta} \\ &= \mathsf{Adv}^{\mathrm{UKS}}_{\mathcal{P},\mathcal{A}} - \frac{1}{2^\theta}.\end{aligned}$$

From the above equation, we get

$$\mathsf{Adv}^{\mathrm{UKS}}_{\mathcal{P}}(\theta,t) \leq \frac{\mathsf{Adv}^{\mathrm{KI}}_{\mathcal{P}}(\theta,t)+1}{2} + \frac{1}{2^\theta}.$$

Note that if a random string τ is returned to $\mathcal{B}$ for $\mathsf{Test}(i,k,e)$ query such that $\tau = \mathsf{sk}_j^{k',e'}$, $\mathcal{B}$ outputs 1. In this case $\mathcal{B}$ fails to guess correctly, but this case only occurs with a probability $\frac{1}{2^\theta}$. □

Theorem 2. FS (Forward Secrecy) implies KCI (Key Compromise Impersonation) for any key exchange protocol.

Proof of Theorem 2. As noted in the definition of FS, if an adversary is allowed to ask a sequence of queries, "$\mathsf{Send}(i,k,e,*)$ *after* $\mathsf{Corrupt}(j)$", then the adversary always can impersonate P_j without interaction with P_j. To exclude this sequence of queries, an experiment of FS does not allow this sequence of queries if Π_i^k is e-fresh. In an experiment of KCI, since even $\mathsf{Corrupt}(j)$ is not allowed for e-fresh Π_i^k, a sequence of queries "$\mathsf{Corrupt}(j)$ *after* $\mathsf{Send}(i,k,e,*)$" is not allowed for e-fresh Π_i^k, whereas such a sequence is allowed in an experiment of FS. So FS implies KCI. □

By the definitions, it is clear that KCI implies KI. So FS implies KCI, KCI implies KI, and KI implies UKS.

4 A Key Exchange Protocol

A description of the proposed key exchange protocol PKA is given in Figure 1. We assume that parties can be ordered by their names (e.g., lexicographically) and write $P_i < P_j$ to denote this ordering. Let θ be a security parameter, and let $\mathbb{G}$ be a group of prime order q (where $|q| = \theta$) with generator g. (We assume that $\mathbb{G}, q$ and g are fixed in advance and known to the entire network.) Let H be a hash function such that $H : \{0,1\}^* \rightarrow \{0,1\}^\theta$, and S be an unforgeable and *deterministic* signature scheme. A deterministic signature scheme does not use any random number [Bellare et al. 2001]. We assume that each party P_i has a pair of public-/private-keys $(y_i = g^{x_i}, x_i)$ and another pair of public-/private-keys (vk_i, sk_i) for a signature scheme S. We also assume that the group membership test of y_i is done by the trusted center which issues a certificate of a public key. We do not describe procedures related to certificates such as validity check of certificates before using public keys.

PKA

Setup: F is a pseudorandom function and S is a signature scheme. A key graph for P_i is $\mathsf{G}_i^k = \{\mathsf{V}_i^k, \mathsf{E}_i^k\}$.

Round 1: P_i selects a random number $r_i \leftarrow \{0,1\}^\theta$, and broadcasts r_i.

Round 2: P_i selects a random number $\alpha_i \leftarrow \mathbb{Z}_q$ and makes $Z_i = g^{\alpha_i}$. P_i makes sid' by concatenating the first round messages by lexicographic ordering of the owners. P_i calculates $\sigma_i \leftarrow \mathsf{S.sign}_{sk_i}(\mathsf{sid}'||Z_i)$ and broadcasts $Z_i||\sigma_i$.

Computation of keys: P_i verifies that the received signatures. If the verification is successful, P_i makes sid_i^k by concatenating all messages by lexicographic ordering of the owners. For each $e = (i,j) \in \mathsf{E}_i^k$, P_i calculates $k_{i,j} = H((y_j g^{\alpha_j})^{x_i+\alpha_i}) = H(g^{(x_i+\alpha_i)(x_j+\alpha_j)})$, and computes a key $\mathsf{sk}_i^{k,e} = \mathsf{F}_{k_{i,j}}(H(\mathsf{sid}_i^k))$. Note that the hash function H is used to adjust the space of inputs to the key and domain spaces of F.

Figure 1: Description of PKA

The round messages of PKA depend only on the vertices in the key graph. The information of edges of the key graph is required when the parties calculate the session keys. Of course, it is possible that a party, without information of edges, calculates all of the session keys for parties in the vertices, and uses some of them whenever necessary. But this approach is not efficient, since a party has to compute some redundant session keys which are never going to be used.

An example of an execution of PKA is shown in Figure 2. In the following theorem, we provide a formal proof of security for PKA in the model of Section

$\mathsf{G} = (\mathsf{V}, \mathsf{E}), \mathsf{V} = \{P_1, P_2, P_3, P_4\}, \mathsf{E} = \{(1,2), (1,3), (1,4), (2,3)\}$

	P_1	P_2	P_3	P_4
Round 1	r_1	r_2	r_3	r_4
Round 2	$g^{\alpha_1}\|\|\sigma_1$	$g^{\alpha_2}\|\|\sigma_2$	$g^{\alpha_3}\|\|\sigma_3$	$g^{\alpha_4}\|\|\sigma_4$

$$\mathsf{sid}' = r_1||r_2||r_3||r_4$$
$$\sigma_i \leftarrow \mathsf{S.sign}_{sk_i}(\mathsf{sid}'||g^{\alpha_i})$$
$$\mathsf{sid} = \mathsf{sid}'||g^{\alpha_1}||\sigma_1||g^{\alpha_2}||\sigma_2||g^{\alpha_3}||\sigma_3||g^{\alpha_4}||\sigma_4$$
$$\mathsf{sk}^{(1,2)} = \mathsf{F}_{H(g^{(x_1+\alpha_1)(x_2+\alpha_2)})}(H(\mathsf{sid}))$$
$$\mathsf{sk}^{(1,3)} = \mathsf{F}_{H(g^{(x_1+\alpha_1)(x_3+\alpha_3)})}(H(\mathsf{sid}))$$
$$\mathsf{sk}^{(1,4)} = \mathsf{F}_{H(g^{(x_1+\alpha_1)(x_4+\alpha_4)})}(H(\mathsf{sid}))$$
$$\mathsf{sk}^{(2,3)} = \mathsf{F}_{H(g^{(x_2+\alpha_2)(x_3+\alpha_3)})}(H(\mathsf{sid}))$$

Figure 2: An example of an execution of PKA where $P_1 < P_2 < P_3 < P_4$

3.

Theorem 3. If F is a secure pseudorandom function and S is an unforgeable signature scheme, PKA is an FS/SSR–secure key exchange scheme under the decisional Diffie-Hellman assumption.

Proof of Theorem 3. Theorem 3 is proved by proving two lemmas. We prove FS security in Lemma 1 and SSR security in Lemma 2. Throughout the proof, an adversary $\mathcal{A}$ attacking PKA is involved in an experiment, where a simulator runs PKA for $\mathcal{A}$. Whenever $\mathcal{A}$ queries one of oracles described in Section 3, the simulator answers to the query by either using its oracle query or running a proper algorithm, which may depend on the purpose of the simulator exploiting $\mathcal{A}$. □

Lemma 1. If F is a secure pseudorandom function and S is an unforgeable signature scheme, PKA is an FS–secure key exchange scheme under the decisional Diffie-Hellman assumption. More formally,

$$\begin{aligned}\mathsf{Adv}^{\mathrm{FS}}_{\mathsf{PKA}}(\theta, t) \leq \frac{2q_s^2}{2^\theta} &+ 2N \cdot \mathsf{Adv}^{\mathrm{SUF}}_{\mathsf{S}}(\theta, t, q_s) \\ &+ 2(Nq_s)^2 \cdot \mathsf{Adv}^{\mathrm{DDH}}_{\mathcal{GG}}(\theta, t) + 2\mathsf{Adv}^{\mathrm{PRF}}_{\mathsf{F}}(\theta, t, 1, \theta),\end{aligned}$$

where t is the maximum total experiment time including an adversary's execution time. Here, N is an upper bound on the number of *honest* parties, and q_s is an upper bound on the number of the sessions an adversary makes. We address the parties which are not insider attackers as honest parties.

Proof of Lemma 1. Let $\mathcal{A}$ be an adversary attacking FS-security of PKA. Let col be the event that r (the message of Round 1 in PKA) repeats at some point during the experiment and forge be the event that $\mathcal{A}$ forges at least one signature. $\mathcal{A}$ may get information concerning the particular keys when col or forge occurs, or do so even when neither col nor forge occurs. The advantage of $\mathcal{A}$ then is

$$\mathsf{Pr}_{\mathcal{A}}[b = b'] \leq \mathsf{Pr}_{\mathcal{A}}[\mathsf{col}] + \mathsf{Pr}_{\mathcal{A}}[\mathsf{forge}] + \mathsf{Pr}_{\mathcal{A}}[b = b' \wedge \overline{\mathsf{col}} \wedge \overline{\mathsf{forge}}].$$

We bound the probability of the terms of the above equation in the following claims.

Claim 1. $\mathsf{Pr}_{\mathcal{A}}[\mathsf{col}] \leq \frac{q_s^2}{2^\theta}$.

Claim 2. $\mathsf{Pr}_{\mathcal{A}}[\mathsf{forge}] \leq N \cdot \mathsf{Adv}_{\mathsf{S}}^{\mathrm{SUF}}(\theta, t, q_s)$.

Claim 3. $\mathsf{Pr}_{\mathcal{A}}[b = b' \wedge \overline{\mathsf{col}} \wedge \overline{\mathsf{forge}}] \leq (Nq_s)^2 \cdot \mathsf{Adv}^{\mathrm{DDH}} + \mathsf{Adv}^{\mathrm{PRF}} + \frac{1}{2}$.

From Claim 1, Claim 2 and Claim 3, we have

$$\begin{aligned}\mathsf{Adv}_{\mathsf{PKA}}^{\mathrm{FS}}(\theta, t) \leq \frac{2q_s^2}{2^\theta} &+ 2N \cdot \mathsf{Adv}_{\mathsf{S}}^{\mathrm{SUF}}(\theta, t, q_s) \\ &+ 2(Nq_s)^2 \cdot \mathsf{Adv}_{\mathcal{GG}}^{\mathrm{DDH}}(\theta, t) + 2\mathsf{Adv}_{\mathsf{F}}^{\mathrm{PRF}}(\theta, t, 1, \theta).\end{aligned}$$ □

Proof of Claim 1. We can easily see that due to "birthday paradox", the collision probability is bounded as $\mathsf{Pr}_{\mathcal{A}}[\mathsf{col}] \leq \frac{q_s^2}{2^\theta}$ because at least one honest party has to select the same random number at lease twice from q_s different sessions. †

Proof of Claim 2. We only consider the advantage of the adversary from the forgery of signatures. We construct a simulator $\mathcal{F}$ which tries to break the underlying signature scheme by exploiting $\mathcal{A}$. Given a public key vk and a signing oracle $\mathsf{S.sign}_{sk}(\cdot)$ in the experiment of unforgeability, $\mathcal{F}$ randomly selects a party and sets vk as a public key of the party. $\mathcal{F}$ uses $\mathsf{S.sign}_{sk}(\cdot)$ to generate a signature of the party. That is, a signature of the party for a message m is $\mathsf{S.sign}_{sk}(m)$. A formal description of $\mathcal{F}$ is as follows:

1. $\mathcal{F}$ selects a random value i^* from $[1, N]$, and sets vk as P_{i^*}'s verification key for the signature scheme. Public keys of other players are chosen in the specified way.

2. As defined in Section 3, $\mathcal{A}$ may ask all types of queries except State query. For each oracle query of $\mathcal{A}$, $\mathcal{F}$ can answers it according to the protocol except the following cases :

   - Send query for which $\mathcal{F}$ needs to generate P_{i^*}'s signature as a part of a message in Round 2 of PKA : $\mathcal{F}$ uses its signing oracle $\mathsf{S.sign}_{sk}(\cdot)$. That is, a signature of P_{i^*} for a message m is $\mathsf{S.sign}_{sk}(m)$.

- $\mathsf{Corrupt}(i)$ query with $i = i^*$: $\mathcal{F}$ fails and stops since $\mathcal{F}$ does not know the secret key sk of its signing oracle.

3. If $\mathcal{F}$ finds a forged signature σ during simulation such that σ is a valid signature of the party P_{i^*}, then $\mathcal{F}$ outputs σ and stops.

The probability that $\mathcal{F}$ succeeds depends on the probabilities that $\mathcal{A}$ forges a signature of P_{i^*} and $\mathcal{F}$ correctly guesses i^*. Only when $\mathcal{F}$ makes a correct guess, $\mathcal{F}$ can provides $\mathcal{A}$ with exactly the same view as in the experiment until $\mathcal{F}$ ends. Hence, the advantage of $\mathcal{F}$ is bounded as follows:

$$\begin{aligned}\mathsf{Adv}^{\mathrm{SUF}}_{\mathsf{S},\mathcal{F}} &\geq \mathsf{Pr}_{\mathcal{A}}[\mathsf{forge}] \cdot \mathsf{Pr}_{\mathcal{F}}[\text{Guess correctly the party whose signature is forged}] \\ &\geq \mathsf{Pr}_{\mathcal{A}}[\mathsf{forge}] \cdot \frac{1}{N}.\end{aligned}$$

So the claim follows. †

Proof of Claim 3. Assume that an adversary $\mathcal{A}$ breaks FS-security of PKA with a non-negligible probability without having events col or forge. This allows us to solve the Decisional Diffie-Hellman problem or the pseudorandomness (with probability related to that of the adversary's success probability). We now proceed with a more formal proof.

We define a series of games in the following. Game_0 represents the real execution of the experiment, while Game_1 and Game_2 only differ from Game_0 in the method constructing a session key.

– In Game_0, a session key for the test query is calculated and returned to the adversary as follows: If $b = 1$, $\mathsf{sk} = \mathsf{F}_{H(g^{(x_i+\alpha_i)(x_j+\alpha_j)})}(H(\mathsf{sid}_i^k))$, where $e = (i, j)$. If $b = 0$, $\mathsf{sk} \leftarrow \{0, 1\}^\theta$.

– In Game_1, a session key for the test query is calculated and returned to the adversary as follows: If $b = 1$, $\mathsf{sk} = \mathsf{F}_{H(g^w)}(H(\mathsf{sid}_i^k))$, where $w \leftarrow [1, q]$ and $e = (i, j)$. If $b = 0$, $\mathsf{sk} \leftarrow \{0, 1\}^\theta$.

– In Game_2, a session key for the test query is calculated and returned to the adversary as follows: If $b = 1$, $\mathsf{sk} = h(H(\mathsf{sid}_i^k))$, where $h \leftarrow \mathsf{Rand}^{\{0,1\}^\theta \rightarrow \{0,1\}^\theta}$ and $e = (i, j)$. If $b = 0$, $\mathsf{sk} \leftarrow \{0, 1\}^\theta$.

The difference of the advantage of an adversary in each game is as follows:

Claim 3.1. $\mathsf{Pr}_{\mathcal{A}}[b = b' \wedge \overline{\mathsf{col}} \wedge \overline{\mathsf{forge}} \text{ in } \mathsf{Game}_0] - \mathsf{Pr}_{\mathcal{A}}[b = b' \wedge \overline{\mathsf{col}} \wedge \overline{\mathsf{forge}} \text{ in } \mathsf{Game}_1] \leq (Nq_s)^2 \cdot \mathsf{Adv}^{\mathrm{DDH}}$.

Claim 3.2. $\mathsf{Pr}_{\mathcal{A}}[b = b' \wedge \overline{\mathsf{col}} \wedge \overline{\mathsf{forge}} \text{ in } \mathsf{Game}_1] - \mathsf{Pr}_{\mathcal{A}}[b = b' \wedge \overline{\mathsf{col}} \wedge \overline{\mathsf{forge}} \text{ in } \mathsf{Game}_2] \leq \mathsf{Adv}^{\mathrm{PRF}}$.

It is obvious that $\mathsf{Pr}_{\mathcal{A}}[b = b' \wedge \overline{\mathsf{col}} \wedge \overline{\mathsf{forge}} \text{ in } \mathsf{Game}_2]$ is $\frac{1}{2}$. Thus, from Claim 3.1 and Claim 3.2, Claim 3 immediately follows. †

Now we proceed to prove Claim 3.1 and Claim 3.2.

Proof of Claim 3.1. We remind that $\mathcal{A}$ breaks PKA with a non-negligible probability without having events col or forge. This implies the following facts:

1. Without event col, $\mathcal{A}$ can not replay any message because each party P_i checks if Round 2 messages (received from the other parties) contain Round 1 message r_i which is randomly selected by P_i in each session.
2. Without event forge, $\mathcal{A}$ has to send to a tested oracle one of Round 2 messages made by the honest parties because $\mathcal{A}$ can not forge a signature.

We construct a distinguisher $\mathcal{D}_1$ which tries to break the decisional Diffie-Hellman assumption using $\mathcal{A}$. That is, $\mathcal{D}_1$ tries to decide whether or not a given $(\mathbb{G}, q, g, U_1, U_2, W)$ is an instance of the decisional Diffie-Hellman problem. The more concrete description of $\mathcal{D}_1$ is as follows:

1. Given $(\mathbb{G}, q, g, U_1, U_2, W)$, $\mathcal{D}_1$ begins by choosing public keys for all parties normally (i.e., choosing a random x_i and letting $y_i = g^{x_i}$, and making secret/public keys for a signature scheme S). $\mathcal{D}_1$ randomly selects i^*, j^* from $[1, N]$, and t_1, t_2 from $[1, q_s]$.
2. For each oracle query of $\mathcal{A}$, $\mathcal{D}_1$ answers it according to the protocol except the following cases:
   - Send query for which $\mathcal{D}_1$ needs to generate a message to be sent by $\Pi_{i^*}^{t_1}$ ($\Pi_{j^*}^{t_2}$) in Round 2 : $\mathcal{D}_1$ uses U_1 (U_2) as an ephemeral Diffie-Hellman message (i.e., the first component of a message in Round 2 of PKA).
   - $\mathsf{Reveal}(i, k, e)$ for $(i = i^*$ and $k = t_1)$ or $(i = j^*$ and $k = t_2)$: $\mathcal{D}_1$ fails and stops since $\mathcal{D}_1$ has used U_1 or U_2 as the ephemeral Diffie-Hellman message of Π_i^k and the Discrete Logarithm problem is hard. We note that if $i \notin \{i^*, j^*\}$ and Π_i^k has received U_1 from $\Pi_{i^*}^{t_1}$ as an ephemeral Diffie-Hellman message, $\mathcal{D}_1$ still can calculate the correct key $k_{i,i^*} = H(g^{x_i x_{i^*}} U_1^{x_i} (g^{\alpha_i})^{x_{i^*}} U_1^{\alpha_i})$, where g^{α_i} is Π_i^k's ephemeral Diffie-Hellman message. For the case Π_i^k has received U_2 from $\Pi_{j^*}^{t_2}$, $\mathcal{D}_1$ can also calculate the correct key $k_{i,j^*} = H(g^{x_i x_{j^*}} U_2^{x_i} (g^{\alpha_i})^{x_{j^*}} U_2^{\alpha_i})$.
   - $\mathsf{Test}(i, k, e)$ query such that $\Pi_i^k \in \{\Pi_{i^*}^{t_1}, \Pi_{j^*}^{t_2}\}$, and $\Pi_{i^*}^{t_1}$ and $\Pi_{j^*}^{t_2}$ are e-partnered: $\mathcal{D}_1$ flips a coin b as usual. If b is equal to 1, $\mathcal{D}_1$ calculates $k_{i^*,j^*} = H(g^{x_{i^*} x_{j^*}} U_2^{x_{i^*}} U_1^{x_{j^*}} W)$ and returns $\mathsf{F}_{k_{i^*,j^*}}(H(\mathsf{sid}_i^k))$ to $\mathcal{A}$. If b is equal to 0, $\mathcal{D}_1$ returns a random value selected from the space $\{0, 1\}^\theta$. For $\mathsf{Test}(i, k, e)$ query such that $\Pi_i^k \notin \{\Pi_{i^*}^{t_1}, \Pi_{j^*}^{t_2}\}$, or $\Pi_{i^*}^{t_1}$ and $\Pi_{j^*}^{t_2}$ are not e-partnered, $\mathcal{D}_1$ fails and stops.

3. When $\mathcal{A}$ outputs b', $\mathcal{D}_1$ checks if $b = b'$. If so, $\mathcal{D}_1$ outputs 1 and stops. Otherwise, $\mathcal{D}_1$ outputs 0 and stops.

When the experiment terminates without failure, $\mathcal{D}_1$ successfully simulates Game_0 or Game_1 to $\mathcal{A}$, depending on the value of W. That is, for $U_1 = g^{u_1}$ and $U_2 = g^{u_2}$, if $W = g^{u_1 u_2}$, $\mathcal{D}_1$ simulates Game_0. Otherwise $\mathcal{D}_1$ simulates Game_1 since if W is random, then $g^{x_{i^*} x_{j^*}} U_2^{x_{i^*}} U_1^{x_{j^*}} W$ is also random. The probability of success of $\mathcal{D}_1$ depends on whether or not $\mathcal{D}_1$ correctly guesses i^*, j^*, t_1 and t_2. If these guesses are correct, $\mathcal{D}_1$ provides exactly the same view as in Game_0 or Game_1 to $\mathcal{A}$. So the following inequality holds:

$$\begin{aligned}
\mathsf{Adv}_{\mathcal{D}_1}^{\mathrm{DDH}} &= \mathsf{Pr}[\mathcal{D}_1(U_1, U_2, W) = 1 | U_1 = g^{u_1}, U_2 = g^{u_2}, W = g^{u_1 u_2}] \\
&\quad - \mathsf{Pr}[\mathcal{D}_1(U_1, U_2, W) = 1 | U_1 = g^{u_1}, U_2 = g^{u_2}, W = g^{w}] \\
&\geq \frac{1}{(Nq_s)^2} \cdot (\mathsf{Pr}_{\mathcal{A}}[b = b' \wedge \overline{\mathsf{col}} \wedge \overline{\mathsf{forge}} \text{ in } \mathsf{Game}_0] \\
&\quad - \mathsf{Pr}_{\mathcal{A}}[b = b' \wedge \overline{\mathsf{col}} \wedge \overline{\mathsf{forge}} \text{ in } \mathsf{Game}_1]).
\end{aligned}$$

The claim immediately follows from the above. †

Proof of Claim 3.2. Consider a distinguisher $\mathcal{D}_2$ to break pseudorandomness of a pseudorandom function family F. Given an oracle function $f(\cdot)$ in the experiment of pseudorandomness of the function family F, $\mathcal{D}_2$ uses $f(\cdot)$ to make a session key for the test oracle. The more concrete description of $\mathcal{D}_2$ is as follows:

1. Given an oracle function $f(\cdot)$, $\mathcal{D}_2$ begins by choosing public keys for all parties normally (i.e., choosing a random x_i and letting $y_i = g^{x_i}$, and making a pair of private-/public-keys for a signature scheme S).

2. For each oracle query of $\mathcal{A}$, $\mathcal{D}_2$ handles it as in Game_1 except a Test query. For $\mathsf{Test}(i, k, e)$ query, $\mathcal{D}_2$ flips a coin b. If b is equal to 1, $\mathcal{D}_2$ returns $f(H(\mathsf{sid}_i^k))$. Otherwise, $\mathcal{D}_2$ returns a random value selected from the space $\{0,1\}^\theta$.

3. When $\mathcal{A}$ outputs b', $\mathcal{D}_2$ checks if $b = b'$. If so, $\mathcal{D}_2$ outputs 1 and stops. Otherwise, $\mathcal{D}_2$ outputs 0 and stops.

$\mathcal{D}_2$ simulates Game_1 or Game_2 depending on whether $f(\cdot)$ is a function from F or not. So the following inequality holds:

$$\begin{aligned}
\mathsf{Adv}_{\mathcal{D}_2}^{\mathrm{PRF}} &= \mathsf{Pr}[\mathcal{D}_2{}^{f(\cdot)} = 1 | K \leftarrow \{0,1\}^\theta; f = \mathsf{F}_K] \\
&\quad - \mathsf{Pr}[\mathcal{D}_2{}^{f(\cdot)} = 1 | h \leftarrow \mathsf{Rand}^{\{0,1\}^\theta \rightarrow \{0,1\}^\theta}; f = h] \\
&\geq \mathsf{Pr}_{\mathcal{A}}[b = b' \wedge \overline{\mathsf{col}} \wedge \overline{\mathsf{forge}} \text{ in } \mathsf{Game}_1] - \mathsf{Pr}_{\mathcal{A}}[b = b' \wedge \overline{\mathsf{col}} \wedge \overline{\mathsf{forge}} \text{ in } \mathsf{Game}_2].
\end{aligned}$$

The claim immediately follows from the above. †

Lemma 2. If F is a secure pseudorandom function and S is an unforgeable signature scheme, PKA is an SSR–secure key exchange scheme under the decisional

Diffie-Hellman assumption. More formally,

$$\mathsf{Adv}^{\mathrm{SSR}}_{\mathsf{PKA}}(\theta, t) \leq 2N^2 \cdot \mathsf{Adv}^{\mathrm{DDH}}_{\mathcal{GG}}(\theta, t) + 2\mathsf{Adv}^{\mathrm{PRF}}_{\mathsf{F}}(\theta, t, 1, \theta),$$

where t is the maximum total experiment time including an adversary's execution time. Here, N is an upper bound on the number of honest parties, and q_s is an upper bound on the number of the sessions an adversary makes.

Proof of Lemma 2. Assume that an adversary $\mathcal{A}$ breaks PKA with a non-negligible probability. This allows us to solve the Decisional Diffie-Hellman problem or the pseudorandomness (with probability related to that of the adversary's success probability). We now proceed with a more formal proof.

We define a series of games in the following. Game_0 represents the real execution of the experiment, while Game_1 and Game_2 only differ from Game_0 in the method constructing a session key.

- In Game_0, a session key for the test query is calculated and returned to the adversary as follows: If $b = 1$, $\mathsf{sk} = \mathsf{F}_{H(g^{(x_i+\alpha_i)(x_j+\alpha_j)})}(H(\mathsf{sid}_i^k))$, where $e = (i, j)$. If $b = 0$, $\mathsf{sk} \leftarrow \{0,1\}^\theta$.
- In Game_1, a session key for the test query is calculated and returned to the adversary as follows: If $b = 1$, $\mathsf{sk} = \mathsf{F}_{H(g^w)}(H(\mathsf{sid}_i^k))$, where $w \leftarrow [1, q]$ and $e = (i, j)$. If $b = 0$, $\mathsf{sk} \leftarrow \{0,1\}^\theta$.
- In Game_2, a session key for the test query is calculated and returned to the adversary as follows: If $b = 1$, $\mathsf{sk} = h(H(\mathsf{sid}_i^k))$, where $h \leftarrow \mathsf{Rand}^{\{0,1\}^\theta \rightarrow \{0,1\}^\theta}$ and $e = (i, j)$. If $b = 0$, $\mathsf{sk} \leftarrow \{0,1\}^\theta$.

The difference of the advantage of an adversary in each game is as follows:

Claim 4. $\mathsf{Adv}^{\mathrm{SSR},\mathsf{Game}_0}_{\mathsf{PKA},\mathcal{A}} - \mathsf{Adv}^{\mathrm{SSR},\mathsf{Game}_1}_{\mathsf{PKA},\mathcal{A}} \leq 2N^2 \cdot \mathsf{Adv}^{\mathrm{DDH}}$.

Claim 5. $\mathsf{Adv}^{\mathrm{SSR},\mathsf{Game}_1}_{\mathsf{PKA},\mathcal{A}} - \mathsf{Adv}^{\mathrm{SSR},\mathsf{Game}_2}_{\mathsf{PKA},\mathcal{A}} \leq 2\mathsf{Adv}^{\mathrm{PRF}}$.

It is obvious that the advantage of any adversary is 0 in Game_2. Thus, from Claim 4 and Claim 5, Lemma 2 immediately follows. $\square$

Now we proceed to prove Claim 4 and Claim 5.

Proof of Claim 4. We construct a simulator $\mathcal{D}_3$ which tries to break the decisional Diffie-Hellman assumption using $\mathcal{A}$. That is, $\mathcal{D}_3$ tries to decide whether or not a given $(\mathbb{G}, q, g, U_1, U_2, W)$ is an instance of the decisional Diffie-Hellman problem. The more concrete description of $\mathcal{D}_3$ is as follows:

1. Given $(\mathbb{G}, q, g, U_1, U_2, W)$, $\mathcal{D}_3$ randomly selects i^*, j^* from $[1, N]$, and uses U_1 as public key y_{i^*} of P_{i^*} and U_2 as y_{j^*} of P_{j^*}. $\mathcal{D}_3$ chooses all other public keys normally.

2. For each oracle query of $\mathcal{A}$, $\mathcal{D}_3$ answers it according to the protocol except the following cases:
   - $\mathsf{Test}(i,k,e)$ for $e \neq (i^*, j^*)$: $\mathcal{D}_3$ fails and stops. For $\mathsf{Test}(i,k,e)$ for $e = (i^*, j^*)$, $\mathcal{D}_3$ flips a coin b. If b is equal to 1, $\mathcal{D}_3$ computes $k_{i^*,j^*} = H(WU_2^{\alpha_{i^*}} U_1^{\alpha_{j^*}} g^{\alpha_{i^*}\alpha_{j^*}})$, and returns $\mathsf{F}_{k_{i^*,j^*}}(H(\mathsf{sid}_i^k))$. If b is equal to 0, $\mathcal{D}_3$ returns a random value selected from the space $\{0,1\}^\theta$.
3. When $\mathcal{A}$ outputs b', $\mathcal{D}_3$ checks if $b = b'$. If so, $\mathcal{D}_3$ outputs 1 and stops. Otherwise, $\mathcal{D}_3$ outputs 0 and stops.

When the experiment terminates without failure, $\mathcal{D}_3$ successfully simulates Game_0 or Game_1 to $\mathcal{A}$ depending on the value W. That is, for $U_1 = g^{u_1}$ and $U_2 = g^{u_2}$, if $W = g^{u_1 u_2}$, $\mathcal{D}_3$ simulates Game_0. Otherwise, $\mathcal{D}_3$ simulates Game_1 since if W is random, then $WU_2^{\alpha_{i^*}} U_1^{\alpha_{j^*}} g^{\alpha_{i^*}\alpha_{j^*}}$ is also random. The probability of success of $\mathcal{D}_3$ depends on whether or not $\mathcal{D}_3$ guesses correctly i^* and j^*. If these guesses are correct, $\mathcal{D}_3$ provides exactly the same view as in Game_0 or Game_1 to $\mathcal{A}$. So the following inequality holds:

$$\begin{aligned}\mathsf{Adv}_{\mathcal{D}_3}^{\mathrm{DDH}} &= \mathsf{Pr}[\mathcal{D}_3(U_1,U_2,W)=1 | U_1 = g^{u_1}, U_2 = g^{u_2}, W = g^{u_1u_2}] \\ &\quad - \mathsf{Pr}[\mathcal{D}_3(U_1,U_2,W)=1 | U_1 = g^{u_1}, U_2 = g^{u_2}, W = g^{w}] \\ &\geq \frac{1}{N^2} \cdot (\mathsf{Pr}_{\mathcal{A}}[b=b' \text{ in } \mathsf{Game}_0] - \mathsf{Pr}_{\mathcal{A}}[b=b' \text{ in } \mathsf{Game}_1]) \\ &= \frac{1}{N^2} \cdot (\frac{\mathsf{Adv}_{\mathcal{A}}^{\mathrm{SSR},\mathsf{Game}_0}+1}{2} - \frac{\mathsf{Adv}_{\mathcal{A}}^{\mathrm{SSR},\mathsf{Game}_1}+1}{2}).\end{aligned}$$

The claim immediately follows from the above. †

Proof of Claim 5. Consider a distinguisher $\mathcal{D}_4$ to break pseudorandomness of a pseudorandom function family F. Given an oracle function $f(\cdot)$ in the experiment of pseudorandomness of the function family F, $\mathcal{D}_4$ uses $f(\cdot)$ to make a session key for the test oracle. The more concrete description of $\mathcal{D}_4$ is as follows:

1. Given an oracle function $f(\cdot)$, $\mathcal{D}_4$ begins by choosing public keys for all parties normally (i.e., choosing a random x_i and letting $y_i = g^{x_i}$, and making a pair of private-/public-keys for a signature scheme S).
2. For each oracle query of $\mathcal{A}$, $\mathcal{D}_4$ handles it as in Game_1 except a Test query. For $\mathsf{Test}(i,k,e)$, $\mathcal{D}_4$ flips a coin b. If b is equal to 1, $\mathcal{D}_4$ returns $f(H(\mathsf{sid}_i^k))$. Otherwise, $\mathcal{D}_4$ returns a random value selected from the space $\{0,1\}^\theta$.
3. When $\mathcal{A}$ outputs b', $\mathcal{D}_4$ checks if $b = b'$. If so, $\mathcal{D}_4$ outputs 1 and stops. Otherwise, $\mathcal{D}_4$ outputs 0 and stops.

$\mathcal{D}_4$ simulates Game_1 or Game_2 depending on whether $f(\cdot)$ is a function from F

or not. So the following inequality holds:

$$\begin{aligned}\mathsf{Adv}^{\mathrm{PRF}}_{\mathcal{D}_4} &= \mathsf{Pr}[{\mathcal{D}_4}^{f(\cdot)} = 1 | K \leftarrow \mathsf{F.key}(1^\theta); f = \mathsf{F}_K] \\ &\quad - \mathsf{Pr}[{\mathcal{D}_4}^{f(\cdot)} = 1 | h \leftarrow \mathsf{Rand}^{\{0,1\}^\theta \rightarrow \{0,1\}^\theta}; f = h] \\ &\geq \mathsf{Pr}_{\mathcal{A}}[b = b' \text{ in } \mathsf{Game}_1] - \mathsf{Pr}_{\mathcal{A}}[b = b' \text{ in } \mathsf{Game}_2] \\ &= \frac{\mathsf{Adv}^{\mathrm{SSR},\mathsf{Game}_1}_{\mathcal{A}} + 1}{2} - \frac{\mathsf{Adv}^{\mathrm{SSR},\mathsf{Game}_2}_{\mathcal{A}} + 1}{2}.\end{aligned}$$

So the claim follows. †

5 Conclusions

In the paper we have studied the parallel key exchange. First, we have defined a security model for a key graph, which extends the security models in [Bellare et al. 1993, Jeong et al. 2006]. Second, we have shown the relation between the various security notions of key exchange. Finally, we have suggested an efficient key exchange protocol for a key graph using the randomness re-use technique. Our protocol is secure in the standard model.

Acknowledgements

D. H. Lee was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Advancement) (IITA-2008-(C1090-0801-0025)).

References

[An et al. 2002] An, J., Dodis, Y., and Rabin, T.: "On the Security of Joint Signature and Encryption"; EUROCRYPT 2002, Lect. Notes Comp. Sci. 2332, Springer, Berlin (2002) 83-107.

[Ateniese et al. 2000] Ateniese, G., Steiner, M., and Tsudik, G.: "New Multi-Party Authentication Services and Key Agreement Protocols"; IEEE Journal of Selected Areas in Communications, 18, 4, (2000) 628-639.

[Bellare et al. 1993] Bellare, M., and Rogaway, P.: "Entity Authentication and Key Distribution"; CRYPTO'93, Lect. Notes Comp. Sci. 773, Springer, Berlin (1993) 232-249.

[Bellare et al. 1998] Bellare, M., Canetti, R., and Krawczyk, H.: "A Modular Approach to the Design and Analysis of Authentication and Key Exchange Protocols"; STOC'98, ACM Press, New York (1998) 419-428.

[Bellare et al. 2001] Bellare, M., Fischlin, M., Goldwasser, S., and Micali, S.: "Identification Protocols Secure against Reset Attacks"; EUROCRYPT 2001, Lect. Notes Comp. Sci. 2045, Springer, Berlin (2001) 495-511.

[Bellare et al. 2003] Bellare, M., Boldyreva, A., and Staddon, J.: "Randomness Re-use in Multi-recipient Encryption Schemes"; PKC'03, Lect. Notes Comp. Sci. 2567, Springer, Berlin (2003) 85-99.

[Blake-Wilson et al. 1997] Blake-Wilson, S., Johnson, D., and Menezes, A.: "Key Agreement Protocols and their Security Analysis"; Sixth IMA International Conference on Cryptography and Coding, Lect. Notes Comp. Sci. 1355, Springer, Berlin (1997) 30-45.

[Blake-Wilson et al. 1998] Blake-Wilson, S., and Menezes, A.: "Authenticated Diffie-Hellman Key Agreement Protocols"; SAC'98, Lect. Notes Comp. Sci. 1556, Springer, Berlin (1998) 339-361.

[Boyd 1997] Boyd, C.: "On Key Agreement and Conference Key Agreement"; ACISP'97, Lect. Notes Comp. Sci. 1270, Springer, Berlin (1997) 294-302.

[Boyd et al. 2003] Boyd, C., and Nieto, J.: "Round-Optimal Contributory Conference Key Agreement"; PKC'03, Lect. Notes Comp. Sci. 2567, Springer, Berlin (2003) 161-174.

[Bresson et al. 2001] Bresson, E., Chevassut, O., and Pointcheval, D.: "Provably Authenticated Group Diffie-Hellman Key Exchange - The Dynamic Case"; ASIACRYPT'01, Lect. Notes Comp. Sci. 2248, Springer, Berlin (2001) 290-309.

[Canetti et al. 2001] Canetti, R., and Krawczyk, H.: "Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels"; EUROCRYPT'01, Lect. Notes Comp. Sci. 2045, Springer, Berlin (2001) 453-474.

[Canetti et al. 2002] Canetti, R., and Krawczyk, H.: "Universally Composable Notions of Key Exchange and Secure Channels"; EUROCRYPT'02, Lect. Notes Comp. Sci. 2332, Springer, Berlin (2002) 337-351.

[Choo et al. 2005] Choo, K., Boyd, C., and Hitchcock, Y.: "Examining Indistinguishability-Based Proof Models for Key Establishment Protocols"; ASIACRYPT'05, Lect. Notes Comp. Sci. 3788, Springer, Berlin (2005) 585-604.

[Denning et al. 1981] Denning, D., and Sacco, G.: "Timestamps in Key Distribution Protocols"; Comm. ACM 24, 8, (1981) 533-536.

[Diffie et al. 1976] Diffie, W., and Hellman, M.: "New Directions in Cryptography"; IEEE Transactions on Information Theory, 22, 6, (1976) 644-654.

[Diffie et al. 1992] Diffie, W., Oorschot, P., and Wiener, M.: "Authentication and Authenticated Key Exchanges"; Designs, Codes, and Cryptography, Lect. Notes Comp. Sci. 2, Springer Netherlands (1992) 107-125.

[Goldreich et al. 1986] Goldreich, O., Goldwasser, S., and Micali, S.: "How to construct random functions"; Journal of the ACM, 33, 4, (1986) 210-217.

[Goldwasser et al. 1988] Goldwasser, S., Micali, S., and Rivest, R.: "A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks"; SIAM J. Comput. 17, 2, (1988) 281-308.

[Ingemarasson et al. 1982] Ingemarasson, I., Tang, D., and Wong, C.: "A Conference Key Distribution System"; IEEE Transactions on Information Theory, 28, 5, (1982) 714-720.

[Jeong et al. 2006] Jeong, I., Kwon, J., and Lee, D.: "A Diffie-Hellman Key Exchange Protocol without Random Oracles"; CANS'06, Lect. Notes Comp. Sci. 4301, Springer, Berlin (2006).

[Jun et al. 2006] Jun, Z., Yu, Z., Fanyuan, M., Dawu, G., and Yingcai, B.: "An extension of secure group communication using key graph"; Information Sciences, 176, 20, (October 2006) 3060-3078.

[Katz et al. 2003] Katz, J., and Yung, M.: "Scalable Protocols for Authenticated Group Key Exchange"; CRYPTO'03, Lect. Notes Comp. Sci. 2729, Springer, Berlin (2003) 110-125.

[Krawczyk 2005] Krawczyk, H.: "HMQV: A High-Performance Secure Diffie-Hellman Protocol"; CRYPTO'05, Lect. Notes Comp. Sci. 3621, Springer, Berlin (2005) 546-566.

[Kurosawa 2002] Kurosawa, K.: "Multi-recipient Public-Key Encryption with Shortened Ciphertext"; PKC'02, Lect. Notes Comp. Sci. 2274, Springer, Berlin (2002) 48-63.

[Law et al. 2003] Law, L., Menezes, A., Qu, M., Solinas, J., and Vanstone, S.: "An Efficient Protocol for Authenticated Key Agreement"; Designs, Codes and Cryptography, Lect. Notes Comp. Sci. 28, Springer Netherlands (2003) 119-134.

[Menezes 2005] Menezes, A.: "Another Look at HMQV"; http://eprint.iacr.org/2005/205, 2005.

[Menezes et al. 1995] Menezes, A., Qu, M., and Vanstone, S.: "Some new key agreement protocols providing mutual implicit authentication"; In proceedings of the second workshop on Selected Area in Cryptography, (1995) 22-32.

[Santis et al. 2006] Santis, A., Ferrara, A., and Masucci, B.: "Enforcing the security of a time-bound hierarchical key assignment scheme"; Information Sciences, 176, 12 (June), Elsevier Science (2006) 1684-1694.

[Shoup 2000] Shoup, V.: "On Formal Models for Secure Key Exchange"; Available at http://eprint.iacr.org.