Sufficient Condition for Ephemeral Key-Leakage Resilient Tripartite Key Exchange

Atsushi Fujioka[1], Mark Manulis[2], Koutarou Suzuki[1], and Berkant Ustaoğlu[3]

[1] NTT Secure Platform Laboratories
3-9-11 Midori-cho Musashino-shi Tokyo 180-8585, Japan
{fujioka.atsushi,suzuki.koutarou}@lab.ntt.co.jp
[2] University of Surrey, Guildford, Surrey, GU2 7XH, United Kingdom
mark@manulis.eu
[3] Izmir Institute of Technology, Urla, Izmir, 35430, Turkey
bustaoglu@uwaterloo.ca

Abstract. Tripartite (Diffie-Hellman) Key Exchange (3KE), introduced by Joux (ANTS-IV 2000), represents today the only known class of group key exchange protocols, in which computation of unauthenticated session keys requires one round and proceeds with minimal computation and communication overhead. The first one-round authenticated 3KE version that preserved the unique efficiency properties of the original protocol and strengthened its security towards resilience against leakage of ephemeral (session-dependent) secrets was proposed recently by Manulis, Suzuki, and Ustaoglu (ICISC 2009).

In this work we explore sufficient conditions for building such protocols. We define a set of *admissible polynomials* and show how their construction generically implies 3KE protocols with the desired security and efficiency properties. Our result generalizes the previous 3KE protocol and gives rise to many new authenticated constructions, all of which enjoy forward secrecy and resilience to ephemeral key-leakage under the gap Bilinear Diffie-Hellman assumption in the random oracle model.

1 Introduction

Key Exchange (KE) protocols are crucial research topics with direct practical applications. Although KE was introduced back in 1976 [17], it was not until 1993 when Bellare and Rogaway [7] made the first step towards capturing the security requirements for these protocols in a formal way. Research efforts on provable security in KE protocols, in the public key setting, focused on two-party KE (2KE), e.g. [14,24,15,25,32,37,16], and group KE (GKE), e.g. [10,29,22,12,31,19], reaching out to other flavors such as password-based solutions [8,6,9,3] or flexible combinations of GKE and 2KE [30,1]. The security notion, shared by most KE flavors, takes its roots in [7] and is called authenticated key exchange (AKE) security. Although AKE-security has been modeled for different types of adversaries, the common idea for secure key exchange is *indistinguishability* of a test session key from a randomly chosen one.

W. Susilo, Y. Mu, and J. Seberry (Eds.): ACISP 2012, LNCS 7372, pp. 15–28, 2012.

Tripartite Key Exchange. A powerful GKE subclass of tripartite KE (3KE) emerged with the use of pairings in the work of Joux [20], where one communication round amongst three parties is sufficient to compute the session key. Each party communicates only one group element and performs one exponentiation and one pairing evaluation. The original protocol in [20] was unauthenticated and so efforts were taken to achieve protection against active attacks, without sacrificing the efficiency of the protocol. Adopting traditional authentication techniques such as digital signatures, as previously applied to unauthenticated 2KE Diffie-Hellman in [14] or GKE in [21,13,22], would require at least two rounds of communication to prevent replay attacks. 3KE protocols with at least two communication rounds have also been known in other authentication settings, e.g. with passwords [2]. The only way to preserve one communication round with constant bit communication complexity from [20] is to resort to an implicitly authenticated solution, in which session key is derived through a mixing of static (long-term) and ephemeral (session-dependent) secrets. Many attempts to achieve such authentication in 3KE, e.g. [36,4,28,27,26] failed (as detailed in [31]). So far the only implicitly authenticated 3KE protocol that provably fulfills this goal is by Manulis, Suzuki, and Ustaoglu [31].

Ephemeral Key Leakage. The security model from [31], stated in a more general GKE setting, considers a very strong attacker, that may adaptively compromise static and ephemeral secret keys used in the protocol sessions (with the restriction that at least one key per participant remains secret). Leakage of ephemeral secrets, typically the exponent used in computing the ephyemeral Diffie-Hellman key, could be damaging for implicitly authenticated protocols, where for better efficiency one may desire to pre-compute store ephemeral public keys off-line. Even if ephemeral secret keys are chosen (and erased) within the protocol session, attacks exploiting side-channels may threaten their secrecy. In general, motivation for considering leakage of ephemeral secrets in KE protocols stems from 2KE domain, e.g. as first mentioned in [14,24] and explicitly modeled in AKE-security definitions from [25,37]. Various efforts towards construction of 2KE leakage-resilient protocols have been taken, e.g. [25,34,37,33,23,18]. In general, modeling and designing ephemeral key-leakage resilient KE protocols should not be taken for granted — Cremers [16] demonstrated how various technical elements of 2KE models such as the notions of session ids and partnering as well as conditions for freshness of the test session may affect the strength of AKE-security definition with ephemeral key-leakage resilience, when it comes to comparability of models and 2KE protocols. The model in [31] is so-far the only GKE security model that focuses on ephemeral key-leakage in test sessions and has recently been applied in [38], for the analysis of a two-round explicitly authenticated ephemeral key-leakage resilient GKE protocol.

Sufficient Condition for Ephemeral Key-Leakage Resilience. Most of KE designs focus on concrete constructions, aiming to achieve particular security goals. Some goals can be obtained generically, using protocol compilers such

as to add authentication (without ephemeral key leakage-resilience) to unauthenticated KE protocols [22,13] or to obtain optional insider security [21,11,19] in GKE protocols. Yet another interesting direction is to search for sufficient conditions for achieving a security goal. Only recently, and for 2KE protocols only, sufficient conditions for ephemeral key-leakage resilience (in the eCK model [25]) have been identified by Fujioka and Suzuki [18]. Their key observation is that many eCK-secure implicitly authenticated 2KE protocols derive session keys from a shared secret group element of the form g^z, where g is the generator of a cycling group of prime order q and the exponent $z \in \mathbb{Z}_q^*$ can often be represented as a function that "mixes" products of static and ephemeral private keys. The authors introduced the concept of *admissible* polynomials over $\mathbb{Z}_q$ to describe which representations of z admit AKE-secure 2KE protocols in the eCK model, by offering a general reduction algorithm to the gap Diffie-Hellman (GDH) [35] problem (in the random oracle model). They could explain constructions of existing eCK-secure 2KE protocols and design new more efficient protocols. The beauty of their approach is that instead of designing an eCK-secure 2KE protocol from scratch it suffices to come up with a set of admissible polynomials.

Our Contributions. We identify sufficient conditions for ephemeral key-leakage resilience of (implicitly authenticated) one-round 3KE protocols, that is conditions under which the protocol can achieve AKE-security (with forward secrecy) from [31]. Technically, we build on the work from [18] and adopt their notion of *admissible* polynomials. The main difference to the 2KE case is that we work with three parties and that one-round 3KE protocols generally require bilinear maps, and hence our definition of "admissible" is different. In particular, our admissible polynomials are of degree three and involve six variables as opposed to polynomials of degree two and four variables from [18]. We show that our conditions on such polynomials are sufficient by providing a generic framework for the design of implicitly authenticated one-round 3KE protocols with ephemeral key-leakage resilience and forward secrecy in the model from [31] under the gap Bilinear Diffie-Hellman (gap BDH) assumption [5] in the random oracle model. This framework explains the 3KE protocol from [31] and gives rise to many further 3KE protocols, all of which are resilient to the leakage of ephemeral session secrets and enjoy forward secrecy.

2 The Model and Security Definitions

We recall the security model from [31], termed *g-eCK model*. This model extends strongly-authenticated key exchange model for two-party protocols from [32] to the group setting and it is described using notations and terminology of the state-of-the-art GKE model [19].

Protocol Participants and Initialization. Let $\mathcal{U} := \{U_1, \ldots, U_N\}$ be a set of potential protocol participants and each user $U_i \in \mathcal{U}$ is assumed to hold a static private/public key pair (s_i, S_i) generated by some algorithm $Gen(1^\kappa)$ on a security parameter 1^κ during the initialization phase.

Protocol Sessions and Instances. Any subset of $\mathcal{U}$ can decide at any time to execute a new protocol session and establish a common group key. Participation of some $U \in \mathcal{U}$ in multiple sessions is modeled through a number of *instances* $\{\Pi_U^s \mid s \in [1 \dots n], U \in \mathcal{U}\}$, i.e., the Π_U^s is the s-th session of U. Each instance is invoked via a message to U with a *partner id* [1] $\mathtt{pid}_U^s \subseteq \mathcal{U}$, which encompasses the identities of all the intended session participants (note that $\mathtt{pid}_U^s$ also includes U). Then, we say that U owns the instance Π_U^s. In the invoked session, Π_U^s *accepts* if the protocol execution was successful, in particular Π_U^s holds then the computed *group key* K_U^s.

Session State. During the session execution, each participating Π_U^s creates and maintains a *session id* $\mathtt{sid}_U^s$ and an associated internal state $\mathtt{state}_U^s$ which in particular is used to maintain ephemeral secrets used by Π_U^s during the protocol execution. We say that U *owns* session $\mathtt{sid}_U^s$ if the instance Π_U^s was invoked at U. Note that the integer s is an internal parameter in the model, used to differentiate amongst the invoked sessions at U, since at the onset of the instance, there may not be enough information to create $\mathtt{sid}_U^s$; until $\mathtt{sid}_U^s$ is created, the instance is identified via $\mathtt{pid}_U^s$ and the outgoing ephemeral public key which is unique per user except with negligible probability. Furthermore, we assume that instances that accepted or aborted delete all information in their respective states.

Partnering. Two instances Π_U^s and $\Pi_{U_*}^t$ are called *partnered* or *matching* if $\mathtt{sid}_U^s \subseteq \mathtt{sid}_{U_*}^t$ or $\mathtt{sid}_{U_*}^t \subseteq \mathtt{sid}_U^s$ and $\mathtt{pid}_U^s = \mathtt{pid}_{U_*}^t$. The first condition models the fact that if session ids are computed during the protocol execution, e.g., from the exchanged messages, then their equality should be guaranteed only at the end of the protocol, i.e., upon the acceptance of Π_U^s and $\Pi_{U_*}^t$.

Note also that the notion of partnering is self-inclusive in the sense that any Π_U^s is partnered with itself. If the protocol allows a user U to initiate sessions with U, then the equality $\mathtt{pid}_U^s = \mathtt{pid}_{U_*}^t$ is a multi-set equality.

Adversarial Model. The adversary $\mathcal{A}$, modeled as a PPT machine, can schedule the protocol execution and mount own attacks via the following queries:

- $\mathsf{AddUser}(U, S_U)$: This query allows $\mathcal{A}$ to introduce new users. In response, if $U \notin \mathcal{U}$ (due to the uniqueness of identities) then U with the static public key S_U is added to $\mathcal{U}$; Note that $\mathcal{A}$ is not required to prove the possession of the corresponding secret key s_U [2].
- $\mathsf{Send}(\Pi_U^s, m)$: With this query, $\mathcal{A}$ can deliver a message m to Π_U^s whereby U denotes the identity of its sender. $\mathcal{A}$ is then given the protocol message generated by Π_U^s in response to m (the output may also be empty if m is not required or if Π_U^s accepts). A special invocation query of the form $\mathsf{Send}(U, (\text{'start'}, U_1, \dots, U_n))$ with $U \in \{U_1, \dots, U_n\}$ creates a new instance

[1] Invocation should include the order of users and perhaps some additional information.

[2] In our security argument, we will only assume that S_U chosen by $\mathcal{A}$ must come from the ephemeral public key space, e.g., element of $\mathbb{G}$.

Π_U^s with $\mathtt{pid}_U^s = \{U_1, \ldots, U_n\}$ and provides $\mathcal{A}$ with the first protocol message.
- $\mathsf{SessionKeyReveal}(\Pi_U^s)$: This query models the leakage of session group keys and provides $\mathcal{A}$ with K_U^s. It is answered only if Π_U^s has accepted.
- $\mathsf{StaticKeyReveal}(U)$: This query provides $\mathcal{A}$ with the static private key s_U.
- $\mathsf{StateReveal}(\Pi_U^s)$: $\mathcal{A}$ is given the ephemeral secret information contained in $\mathtt{state}_U^s$ at the moment the query is asked. Note that the protocol specifies what the state contains.
- $\mathsf{Test}(\Pi_U^s)$: This query models the indistinguishability of the session group key according to the privately flipped bit τ. If $\tau = 0$ then $\mathcal{A}$ is given a random session group key, whereas if $\tau = 1$ the real K_U^s. The query can be queried only once and requires that Π_U^s has accepted.

Correctness. A GKE protocol is said to be *correct* if in the presence of a benign[3] adversary all instances invoked for the same protocol session accept with the same session group key.

Freshness. The classical notion of freshness of some instance Π_U^s is traditionally used to define the goal of AKE-security by specifying the conditions for the $\mathsf{Test}(\Pi_U^s)$ query. For example, the model in [22] defines an instance Π_U^s that has accepted as fresh if none of the following is true: (1) at some point, $\mathcal{A}$ asked $\mathsf{SessionKeyReveal}$ to Π_U^s or to any of its partnered instances; or (2) a query $\mathsf{StaticKeyReveal}(U_*)$ with $U_* \in \mathtt{pid}_U^s$ was asked before a Send query to Π_U^s or any of its partnered instances.

Unfortunately, these restrictions are not sufficient for our purpose since Π_U^s becomes immediately unfresh if the adversary gets involved into the protocol execution via a Send query after having learned the static key s_{U_*} of some user U_* those instance participates in the same session as Π_U^s.

The recent model in [12] defines freshness using the additional $\mathsf{AddUser}$ and $\mathsf{StateReveal}$ queries as follows. According to [12], an instance Π_U^s that has accepted is fresh if none of the following is true: (1) $\mathcal{A}$ queried $\mathsf{AddUser}(U_M, S_{U_M})$ with some $U_* \in \mathtt{pid}_U^s$; or (2) at some point, $\mathcal{A}$ asked $\mathsf{SessionKeyReveal}$ to Π_U^s or any of its partnered instances; or (3) a query $\mathsf{StaticKeyReveal}(U_*)$ with $U_* \in \mathtt{pid}_U^s$ was asked before a Send query to Π_U^s or any of its partnered instances; or (4) $\mathcal{A}$ queried $\mathsf{StateReveal}$ to Π_U^s or any of its partnered instances at some point after their invocation but before their acceptance.

Although this definition is already stronger than the one in [22] it is still insufficient for the main reason that it excludes the leakage of ephemeral secrets of instances in the period between the protocol invocation and acceptance. Also this definition of freshness does not model key compromise impersonation attacks.

The recent update of the freshness notion in [19] addressed the lack of key compromise impersonation resilience. In particular, it modifies the above condition (3) by requiring that if there exists an instance $\Pi_{U_*}^t$ which is partnered

[3] Benign adversary executes an instance of the protocol and faithfully delivers messages without any modification.

with Π_U^s and $\mathcal{A}$ asked $\mathsf{StaticKeyReveal}(U_*)$ then all messages sent by $\mathcal{A}$ to Π_U^s on behalf of $\Pi_{U_*}^t$ must come from $\Pi_{U_*}^t$ intended for Π_U^s. This condition should allow the adversary to obtain static private keys of users prior to the execution of the attacked session while requiring its benign behavior with respect to the corrupted user during the attack.

Yet, this freshness requirement still prevents the adversary from obtaining ephemeral secrets of participants during the attacked session. What is needed is a freshness condition that would allow the adversary to corrupt users and reveal the ephemeral secrets used by their instances in the attacked session at will for the only exception that it does not obtain both the static key s_{U_*} *and* the ephemeral secrets used by the corresponding instance of U_*; otherwise security can no longer be guaranteed. In the following we define freshness taking into account all the previously mentioned problems.

Definition 1. *An accepted instance Π_U^s is fresh if none of the following is true:*

1. *$\mathcal{A}$ queried $\mathsf{AddUser}(U_*, S_{U_*})$ with some $U_* \in \mathtt{pid}_U^s$; or*
2. *$\mathcal{A}$ queried $\mathsf{SessionKeyReveal}$ to Π_U^s or any of its accepted partnered instances; or*
3. *$\mathcal{A}$ queried both $\mathsf{StaticKeyReveal}(U_*)$ with $U_* \in \mathtt{pid}_U^s$ and $\mathsf{StateReveal}(\Pi_{U_*}^t)$ for some instance $\Pi_{U_*}^t$ partnered with Π_U^s; or*
4. *$\mathcal{A}$ queried $\mathsf{StaticKeyReveal}(U_*)$ with $U_* \in \mathtt{pid}_U^s$ and there exists no instance $\Pi_{U_*}^t$ partnered with Π_U^s.*

Note that since $U \in \mathtt{pid}_U^s$ and since the notion of partnering is self-inclusive Condition 3 prevents the simultaneous corruption of static and ephemeral secrets for the corresponding instance Π_U^s as well. In case when users are allowed to own two partnering instances i.e., they can initiate protocols with themselves the last condition should be modified to say that the number of instances of U equals the number of times U appears in $\mathtt{pid}_U^s$. Note also that the above definition captures key-compromise impersonation resilience through Condition 4: $\mathcal{A}$ is allowed to corrupt participants of the test session in advance but then must ensure that instances of such participants have been honestly participating in the test session. In this way we exclude the trivial break of security where $\mathcal{A}$ reveals static keys of users prior to the test session and then actively impersonates those users during the session. On the other hand, as long as $\mathcal{A}$ remains benign with respect to such users their instances will still be considered as fresh.

Definition 2 (g-eCK Security). *Let P be a correct GKE protocol and τ be a uniformly chosen bit. We define the adversarial game $\mathsf{Game}_{\mathcal{A},\mathsf{P}}^{\text{ake-}\tau}(\kappa)$ as follows: after initialization, $\mathcal{A}$ interacts with instances via queries. At some point, $\mathcal{A}$ queries $\mathsf{Test}(\Pi_U^s)$, and continues own interaction with the instances until it outputs a bit τ'. If Π_U^s to which the Test query was asked is* fresh *at the end of the experiment then we set $\mathsf{Game}_{\mathcal{A},\mathsf{P}}^{\text{ake-}\tau}(\kappa) = \tau'$. We define*

$$\mathsf{Adv}_{\mathcal{A},\mathsf{P}}^{\text{ake}}(\kappa) = |2\Pr[\tau = \tau'] - 1|$$

and denote with $\mathsf{Adv}^{\mathrm{ake}}_{\mathsf{P}}(\kappa)$ *the maximum advantage over all PPT adversaries* $\mathcal{A}$*. We say that a GKE protocol* P *provides* g-eCK security *if this advantage is negligible.*

3 Sufficient Condition for Secure Tripartite Protocols

We identify now sufficient conditions for a 3KE protocol to satisfy g-eCK security from Definition 2. Technically, we build upon [18] and their notion of *admissible* polynomials. We extend definition of admissible polynomials to account for the specifics of 3KE protocols and then present a framework for the generic design of g-eCK secure one-round 3KE protocols out of those polynomials.

3.1 Admissible Polynomials

We define *admissible* polynomials in Definition 3 with respect to multivariate polynomials with six variables over $\mathbb{Z}_q$ and state three conditions that, as we will see, are sufficient for building g-eCK secure one-round 3KE protocols. The first condition from Definition 3 says that each term of polynomial $p^{(i)}$ has degree three and that either u_0 or u_1, either v_0 or v_1, and either w_0 or w_1 appear in each term. The second condition says that there exist four polynomials $p^{(i)}, p^{(j)}, p^{(k)}, p^{(l)}$ such that the four corresponding vectors of the coefficients of their terms containing a specific variable, are linearly independent. The third condition says that for each polynomial $p^{(i)}$ the corresponding polynomial, which consists of the terms containing specific variables, is a product of three linear polynomials.

Definition 3 (Admissible Polynomials). *We say* m $(m \geq 4)$ *polynomials* $p^{(i)} \in \mathbb{Z}_q[u_0, u_1, v_0, v_1, w_0, w_1]$ $(i = 1, ..., m)$ *are* admissible *if the following conditions are satisfied.*

1. *For any* i $(= 1, ..., m)$*, the following condition holds*

$$p^{(i)}(u_0, u_1, v_0, v_1, w_0, w_1) = \sum_{\alpha,\beta,\gamma=0,1} d^{(i)}_{\alpha,\beta,\gamma} u_\alpha v_\beta w_\gamma,$$

   where $d^{(i)}_{\alpha,\beta,\gamma} \in \mathbb{Z}_q$.
2. *We denote*

$$V^{(i)}_{\alpha,*,*} = (d^{(i)}_{\alpha,0,0}, d^{(i)}_{\alpha,0,1}, d^{(i)}_{\alpha,1,0}, d^{(i)}_{\alpha,1,1}),$$
$$V^{(i)}_{*,\beta,*} = (d^{(i)}_{0,\beta,0}, d^{(i)}_{0,\beta,1}, d^{(i)}_{1,\beta,0}, d^{(i)}_{1,\beta,1}),$$
$$V^{(i)}_{*,*,\gamma} = (d^{(i)}_{0,0,\gamma}, d^{(i)}_{0,1,\gamma}, d^{(i)}_{1,0,\gamma}, d^{(i)}_{1,1,\gamma}).$$

   For any α $(= 0, 1)$*, there exist distinct indices* i, j, k, l $(1 \leq i < j < k < l \leq m)$*, s.t.*

$$V^{(i)}_{\alpha,*,*},\ V^{(j)}_{\alpha,*,*},\ V^{(k)}_{\alpha,*,*},\ V^{(l)}_{\alpha,*,*}$$

are linearly independent, and for any β $(= 0, 1)$*, there exist distinct indices* i, j, k, l $(1 \leq i < j < k < l \leq m)$*, s.t.*

$$V^{(i)}_{*,\beta,*},\ V^{(j)}_{*,\beta,*},\ V^{(k)}_{*,\beta,*},\ V^{(l)}_{*,\beta,*}$$

are linearly independent, and for any γ $(= 0, 1)$*, there exist distinct indices* i, j, k, l $(1 \leq i < j < k < l \leq m)$*, s.t.*

$$V^{(i)}_{*,*,\gamma},\ V^{(j)}_{*,*,\gamma},\ V^{(k)}_{*,*,\gamma},\ V^{(l)}_{*,*,\gamma}$$

are linearly independent.

3. *We denote*

$$P^{(i)}_{\alpha,*,*} = d^{(i)}_{\alpha,0,0} u_\alpha v_0 w_0 + d^{(i)}_{\alpha,0,1} u_\alpha v_0 w_1 + d^{(i)}_{\alpha,1,0} u_\alpha v_1 w_0 + d^{(i)}_{\alpha,1,1} u_\alpha v_1 w_1,$$

$$P^{(i)}_{*,\beta,*} = d^{(i)}_{0,\beta,0} u_0 v_\beta w_0 + d^{(i)}_{0,\beta,1} u_0 v_\beta w_1 + d^{(i)}_{1,\beta,0} u_1 v_\beta w_0 + d^{(i)}_{1,\beta,1} u_1 v_\beta w_1,$$

$$P^{(i)}_{*,*,\gamma} = d^{(i)}_{0,0,\gamma} u_0 v_0 w_\gamma + d^{(i)}_{0,1,\gamma} u_0 v_1 w_\gamma + d^{(i)}_{1,0,\gamma} u_1 v_0 w_\gamma + d^{(i)}_{1,1,\gamma} u_1 v_1 w_\gamma.$$

For any i $(= 1, ..., m)$*, the following condition holds: for any* α $(= 0, 1)$*,* $P^{(i)}_{\alpha,*,*}$ *is expressed as*

$$P^{(i)}_{\alpha,*,*} = \ell^{(i)}_{\alpha,*,*}(u_0, u_1) \ell'^{(i)}_{\alpha,*,*}(v_0, v_1) \ell''^{(i)}_{\alpha,*,*}(w_0, w_1),$$

where $\ell^{(i)}_{\alpha,*,*}(u_0, u_1)$*,* $\ell'^{(i)}_{\alpha,*,*}(v_0, v_1)$*,* $\ell''^{(i)}_{\alpha,*,*}(w_0, w_1)$ *are linear combinations of* (u_0, u_1)*,* (v_0, v_1)*,* (w_0, w_1)*, respectively, and for any* β $(= 0, 1)$*,* $P^{(i)}_{*,\beta,*}$ *is expressed as*

$$P^{(i)}_{*,\beta,*} = \ell^{(i)}_{*,\beta,*}(u_0, u_1) \ell'^{(i)}_{*,\beta,*}(v_0, v_1) \ell''^{(i)}_{*,\beta,*}(w_0, w_1),$$

where $\ell^{(i)}_{*,\beta,*}(u_0, u_1)$*,* $\ell'^{(i)}_{*,\beta,*}(v_0, v_1)$*,* $\ell''^{(i)}_{*,\beta,*}(w_0, w_1)$ *are linear combinations of* (u_0, u_1)*,* (v_0, v_1)*,* (w_0, w_1)*, respectively, and for any* γ $(= 0, 1)$*,* $P^{(i)}_{*,*,\gamma}$ *is expressed as*

$$P^{(i)}_{*,*,\gamma} = \ell^{(i)}_{*,*,\gamma}(u_0, u_1) \ell'^{(i)}_{*,*,\gamma}(v_0, v_1) \ell''^{(i)}_{*,*,\gamma}(w_0, w_1),$$

where $\ell^{(i)}_{*,*,\gamma}(u_0, u_1)$*,* $\ell'^{(i)}_{*,*,\gamma}(v_0, v_1)$*,* $\ell''^{(i)}_{*,*,\gamma}(w_0, w_1)$ *are linear combinations of* (u_0, u_1)*,* (v_0, v_1)*,* (w_0, w_1)*, respectively.*

In Section 3.2 we construct a g-eCK secure 3KE protocol from admissible polynomials, where parties compute m shared secrets $Z_i = g_T^{p^{(i)}}$ $(= 1, ..., m)$. The above three conditions will be utilized in the security proof of the designed protocol. Roughly, the first condition ensures that each user is able to compute the shared secret group elements. The second condition enables the simulator to extract a BDH solution from the challenge test session. The third conditions ensures simulator can verify that shared secret group elements are correctly formed. We refer to the proof of Theorem 1 for further details and provide in the following some examples of admissible polynomials.

Example 1

$$p^{(1)} = u_0v_0w_0,\ p^{(2)} = u_0v_0w_1,\ p^{(3)} = u_0v_1w_0,\ p^{(4)} = u_0v_1w_1,$$
$$p^{(5)} = u_1v_0w_0,\ p^{(6)} = u_1v_0w_1,\ p^{(7)} = u_1v_1w_0,\ p^{(8)} = u_1v_1w_1.$$

Example 2

$$p^{(1)} = u_0v_0w_0 + u_1v_1w_1,\quad p^{(2)} = u_0v_1w_1 + u_1v_0w_0,$$
$$p^{(3)} = u_1v_0w_1 + u_0v_1w_0,\quad p^{(4)} = u_1v_1w_0 + u_0v_0w_1.$$

Example 3. This example essentially explains the construction behind the one-round 3KE protocol by Manulis, Suzuki, and Ustaoglu [31].

$$p^{(1)} = (u_0 + Du_1)(v_0 + v_1)(w_0 + w_1),\ p^{(2)} = (u_0 + u_1)(v_0 + Ev_1)(w_0 + w_1),$$
$$p^{(3)} = (u_0 + u_1)(v_0 + v_1)(w_0 + Fw_1),\ p^{(4)} = (u_0 + Du_1)(v_0 + Ev_1)(w_0 + Fw_1),$$

where $D, E, F \neq 1$.

Example 4

$$p^{(1)} = (u_0 + u_1)(v_0 + v_1)(w_0 + w_1)\ , p^{(2)} = u_0v_1w_1 + u_1v_0w_0,$$
$$p^{(3)} = u_1v_0w_1 + u_0v_1w_0\ , p^{(4)} = u_1v_1w_0 + u_0v_0w_1.$$

3.2 Proposed 3KE Protocol

We now propose the 3KE protocol $\Pi_{p^{(1)},\ldots,p^{(m)}}$ constructed from admissible polynomials $p^{(i)}$ $(i = 1, \ldots, m)$. We then prove in Theorem 1 that if polynomials $p^{(i)}$ $(i = 1, \ldots, m)$ satisfy the conditions of admissible polynomials, the proposed 3KE protocol $\Pi_{p^{(1)},\ldots,p^{(m)}}$ is g-eCK secure, i.e., we provide a sufficient condition for building g-eCK secure 3KE protocols.

The proposed 3KE protocol $\Pi_{p^{(1)},\ldots,p^{(m)}}$ is described as follows. Let $p^{(i)}$ $(i = 1, \ldots, m)$ be admissible polynomials. Let κ be the security parameter. Let $\mathbb{G}$ and $\mathbb{G}_T$ be cyclic groups of prime order q. Let $e : \mathbb{G} \times \mathbb{G} \mapsto \mathbb{G}_T$ be a non-degenerate bilinear map, called pairing, from group $\mathbb{G} \times \mathbb{G}$ to group $\mathbb{G}_T$. Let g and $g_T = e(g, g)$ be a generator of $\mathbb{G}$ and $\mathbb{G}_T$, respectively. Let $H : \{0,1\}^* \to \{0,1\}^\kappa$ be cryptographic hash function modeled as a random oracle. Let P be the protocol identifier of the protocol $\Pi_{p^{(1)},\ldots,p^{(m)}}$.

For a user U_A, we set U_A's static and ephemeral keys $A_0 = g^{a_0}$ and $A_1 = g^{a_1}$, respectively, and the lowercase letters are the private keys.

In the description, users U_A, U_B, and U_C communicate with each other, and compute the session key.

1. U_A selects a random ephemeral private key $a_1 \in_U \mathbb{Z}_q$, computes the ephemeral public key $A_1 = g^{a_1}$, stores ephemeral private key a_1 as state information, and broadcasts $(\mathsf{P}, (U_A, U_B, U_C), U_A, A_1)$ to U_B and U_C.
2. U_B selects a random ephemeral private key $b_1 \in_U \mathbb{Z}_q$, computes the ephemeral public key $B_1 = g^{b_1}$, stores ephemeral private key b_1 as state information, and broadcasts $(\mathsf{P}, (U_A, U_B, U_C), U_B, B_1)$ to U_C and U_A.

3. U_C selects a random ephemeral private key $c_1 \in_U \mathbb{Z}_q$, computes the ephemeral public key $C_1 = g^{c_1}$, stores ephemeral private key c_1 as state information, and broadcasts $(\mathsf{P}, (U_A, U_B, U_C), U_C, C_1)$ to U_A and U_B.
4. Upon receiving $(\mathsf{P}, (U_A, U_B, U_C), U_B, B_1)$ and $(\mathsf{P}, (U_A, U_B, U_C), U_C, C_1)$, U_A verifies $B_1, C_1 \in \mathbb{G}$, computes m shared secrets

$$Z_i = \prod_{\beta,\gamma=0,1} e(B_\beta, C_\gamma)^{(d^{(i)}_{0,\beta,\gamma} a_0 + d^{(i)}_{1,\beta,\gamma} a_1)} \quad (i = 1, \ldots, m),$$

obtains the session key $K = H(Z_1, \ldots, Z_m, \mathsf{P}, U_A, A_0, A_1, U_B, B_0, B_1, U_C, C_0, C_1)$, and completes the session.
5. Upon receiving $(\mathsf{P}, (U_A, U_B, U_C), U_C, C_1)$ and $(\mathsf{P}, (U_A, U_B, U_C), U_A, A_1)$, U_B verifies $C_1, A_1 \in \mathbb{G}$, computes m shared secrets

$$Z_i = \prod_{\gamma,\alpha=0,1} e(C_\gamma, A_\alpha)^{(d^{(i)}_{\alpha,0,\gamma} b_0 + d^{(i)}_{\alpha,1,\gamma} b_1)} \quad (i = 1, \ldots, m),$$

obtains the session key $K = H(Z_1, \ldots, Z_m, \mathsf{P}, U_A, A_0, A_1, U_B, B_0, B_1, U_C, C_0, C_1)$, and completes the session.
6. Upon receiving $(\mathsf{P}, (U_A, U_B, U_C), U_A, A_1)$ and $(\mathsf{P}, (U_A, U_B, U_C), U_B, B_1)$, U_C verifies $A_1, B_1 \in \mathbb{G}$, computes m shared secrets

$$Z_i = \prod_{\alpha,\beta=0,1} e(A_\alpha, B_\beta)^{(d^{(i)}_{\alpha,\beta,0} c_0 + d^{(i)}_{\alpha,\beta,1} c_1)} \quad (i = 1, \ldots, m),$$

obtains the session key $K = H(Z_1, \ldots, Z_m, \mathsf{P}, U_A, A_0, A_1, U_B, B_0, B_1, U_C, C_0, C_1)$, and completes the session.

All users U_A, U_B, and U_C compute the same shared secrets

$$Z_i = g_T^{p^{(i)}(a_0, a_1, b_0, b_1, c_0, c_1)} \quad (i = 1, \ldots, m),$$

and so compute the same session key K.

The outlined 3KE protocol $\Pi_{p^{(1)},\ldots,p^{(m)}}$ requires exactly m shared secrets, 4 pairing operations at most, and $4m+1$ exponential operations at most (including the exponentiation for the ephemeral public key).

3.3 Security

For the security of the proposed protocol, we need[4] the gap Bilinear Diffie-Hellman (gap BDH) assumption [5] described below. Let BCDH : $\mathbb{G}^3 \to \mathbb{G}_T$ s.t. BCDH$(g^u, g^v, g^w) = e(g, g)^{uvw}$, and BDDH : $\mathbb{G}^4 \to \{0, 1\}$ be a predicate which takes an input $(g^u, g^v, g^w, e(g, g)^x)$ and returns bit 1 if $uvw = x \bmod q$ and bit 0 otherwise. An adversary $\mathcal{A}$ is given input $U, V, W \in_U \mathbb{G}$ selected uniformly random and oracle access to BDDH$(\cdot, \cdot, \cdot, \cdot)$ oracle, and tries to compute BCDH(U, V, W). For adversary $\mathcal{A}$, we define advantage

[4] Gap BDH assumption is used since in bilinear groups no BDDH oracle is available. Using twin BDH technique we could also rely on BDH instead of gap BDH.

$$Adv^{\mathrm{gapBDH}}(\mathcal{A}) = \Pr[U, V, W \in_R \mathbb{G}, \mathcal{A}^{\mathrm{BDDH}(\cdot,\cdot,\cdot,\cdot)}(U, V, W) = \mathrm{BCDH}(U, V, W)],$$

where the probability is taken over the choices of U, V, W and $\mathcal{A}$'s random tape.

Definition 4 (gap BDH assumption). *We say that $\mathbb{G}$ and $\mathbb{G}_T$ satisfy the gap BDH assumption if, for all polynomial-time adversaries $\mathcal{A}$, advantage $Adv^{\mathrm{gapBDH}}(\mathcal{A})$ is negligible in security parameter κ.*

Theorem 1. *If $\mathbb{G}$ and $\mathbb{G}_T$ are groups where the gap BDH assumption holds, H is a random oracle, and $p^{(i)}$ $(i = 1, \ldots, m)$ are admissible polynomials, the proposed 3KE protocol $\Pi_{p^{(1)},\ldots,p^{(m)}}$ constructed from $p^{(i)}$ $(i = 1, \ldots, m)$ is secure in the g-eCK model.*

Proof (Sketch). From the first condition of admissible polynomials, all users U_A, U_B, and U_C can compute the shared secrets as follows. User U_A, who knows secret keys a_0, a_1, can compute shared secrets

$$Z_i = \prod_{\beta,\gamma=0,1} e(B_\beta, C_\gamma)^{(d^{(i)}_{0,\beta,\gamma} a_0 + d^{(i)}_{1,\beta,\gamma} a_1)} = g_T^{\sum_{\alpha,\beta,\gamma=0,1} d^{(i)}_{\alpha,\beta,\gamma} a_\alpha b_\beta c_\gamma},$$

user U_B, who knows secret keys b_0, b_1, can compute shared secrets

$$Z_i = \prod_{\gamma,\alpha=0,1} e(C_\gamma, A_\alpha)^{(d^{(i)}_{\alpha,0,\gamma} b_0 + d^{(i)}_{\alpha,1,\gamma} b_1)} = g_T^{\sum_{\alpha,\beta,\gamma=0,1} d^{(i)}_{\alpha,\beta,\gamma} a_\alpha b_\beta c_\gamma},$$

and user U_C, who knows secret keys c_0, c_1, can compute shared secrets

$$Z_i = \prod_{\alpha,\beta=0,1} e(A_\alpha, B_\beta)^{(d^{(i)}_{\alpha,\beta,0} c_0 + d^{(i)}_{\alpha,\beta,1} c_1)} = g_T^{\sum_{\alpha,\beta,\gamma=0,1} d^{(i)}_{\alpha,\beta,\gamma} a_\alpha b_\beta c_\gamma}.$$

The gap BDH solver $\mathcal{S}$ extracts the answer g_T^{uvw} of an instance $(U = g^u, V = g^v, W = g^w)$ of the gap BDH problem using adversary $\mathcal{A}$. For instance, we assume the case that test session $\mathtt{sid}^*$, owner of which is user U_A, has no partnered sessions $\overline{\mathtt{sid}^*}$, owners of which are users U_B and U_C, adversary $\mathcal{A}$ is given a_0, and adversary $\mathcal{A}$ does not obtain a_1, b_0 and c_0 from the condition of the freshness. In this case, solver $\mathcal{S}$ can perfectly simulate StaticKeyReveal query by selecting random a_0 and setting $A_0 = g^{a_0}$, and solver $\mathcal{S}$ embeds the instance as $A_1 = U$ $(= g^u)$, $B_0 = V$ $(= g^v)$ and $C_0 = W$ $(= g^w)$ to extract g_T^{uvw} from the shared secrets $Z_i = g_T^{p^{(i)}}$ $(i = 1, \ldots, m)$.

From the second condition of admissible polynomials, solver $\mathcal{S}$ can extract the answer of the gap BDH instance as follows. From the second condition, there exist i, j, k, l $(1 \le i, j, k, l \le m)$, s.t., $V^{(i)}_{1,*,*}$, $V^{(j)}_{1,*,*}$, $V^{(k)}_{1,*,*}$, and $V^{(l)}_{1,*,*}$ are linearly independent. Using knowledge of a_0, solver $\mathcal{S}$ can compute

$$Z'_I = g_T^{a_1(d^{(I)}_{1,0,0} b_0 c_0 + d^{(I)}_{1,0,1} b_0 c_1 + d^{(I)}_{1,1,0} b_1 c_0 + d^{(I)}_{1,1,1} b_1 c_1)}$$

$$= Z_I / (e(B_0, C_0)^{d^{(I)}_{0,0,0} a_0} e(B_0, C_1)^{d^{(I)}_{0,0,1} a_0} e(B_1, C_0)^{d^{(I)}_{0,1,0} a_0} e(B_1, C_1)^{d^{(I)}_{0,1,1} a_0})$$

for the indices $I = i, j, k, l$. Solver $\mathcal{S}$ can compute $g_T^{a_1 b_0 c_0}$ from Z'_i, Z'_j, Z'_k, Z'_l since $V^{(i)}_{1,*,*}$, $V^{(j)}_{1,*,*}$, $V^{(k)}_{1,*,*}$, and $V^{(l)}_{1,*,*}$ are linearly independent, and successfully outputs the answer $g_T^{a_1 b_0 c_0} = g_T^{uvw}$ of the gap BDH problem.

From the third condition of admissible polynomials, solver $\mathcal{S}$ can check whether the shared secrets are correctly formed w.r.t. static and ephemeral public keys, and can simulate H and $\mathsf{SessionKeyReveal}$ queries consistently. More precisely, in the simulation of the $H(Z_1, \ldots, Z_m, \mathsf{P}, U_A, A_0, A_1, U_B, B_0, B_1, U_C, C_0, C_1)$ query, solver $\mathcal{S}$ must check that the shared secrets Z_i $(i = 1, \ldots, m)$ are correctly formed, and if so return session key K that is consistent with the previously answered $\mathsf{SessionKeyReveal}(\mathsf{P}, U_X, U_A, A_0, A_1, U_B, B_0, B_1, U_C, C_0, C_1)$ $(X = A, B, C)$ queries. For all i $(= 1, \ldots, m)$, solver $\mathcal{S}$ performs the following procedure. Using the knowledge of a_0, solver $\mathcal{S}$ can compute

$$Z'_i = g_T^{a_1(d^{(i)}_{1,0,0} b_0 c_0 + d^{(i)}_{1,0,1} b_0 c_1 + d^{(i)}_{1,1,0} b_1 c_0 + d^{(i)}_{1,1,1} b_1 c_1)}$$

$$= Z_i / (e(B_0, C_0)^{d^{(i)}_{0,0,0} a_0} e(B_0, C_1)^{d^{(i)}_{0,0,1} a_0} e(B_1, C_0)^{d^{(i)}_{0,1,0} a_0} e(B_1, C_1)^{d^{(i)}_{0,1,1} a_0})$$

Then, solver $\mathcal{S}$ can check if shared secret Z'_i is correctly formed w.r.t. the static and ephemeral public keys, by asking BDDH oracle

$$\mathrm{BDDH}(g^{\ell^{(i)}_{1,*,*}(a_0,a_1)}, g^{\ell'^{(i)}_{1,*,*}(b_0,b_1)}, g^{\ell''^{(i)}_{1,*,*}(c_0,c_1)}, Z'_i) = 1,$$

since the third condition of admissible polynomials holds, and this implies Z_i is correctly formed. Here solver $\mathcal{S}$ can compute $g^{\ell^{(i)}_{1,*,*}(a_0,a_1)}$, $g^{\ell'^{(i)}_{1,*,*}(b_0,b_1)}$, and $g^{\ell''^{(i)}_{1,*,*}(c_0,c_1)}$, since $\ell^{(i)}_{1,*,*}(a_0, a_1)$, $\ell'^{(i)}_{1,*,*}(b_0, b_1)$, and $\ell''^{(i)}_{1,*,*}(c_0, c_1)$ are linear.

□

4 Conclusion

We presented a sufficient condition for constructing one-round ephemeral key-leakage resilient 3KE protocols where parties are equipped with a static public key and an ephemeral public key, each comprised of only one group element, and where key derivation is performed via a single call to the hash function, modeled as a random oracle. Technically, the proposed 3KE protocol can be seen as a combination of several two-dimensional versions of the original (unauthenticated) tripartite key exchange protocol from [20]. The protocol gives rise to a framework for the design of efficient ephemeral key-leakage resilient one-round 3KE protocols in the model from [31] by choosing different admissible polynomials. The amount of work for proving security of all those protocols essentially reduces to proving that chosen polynomials are admissible according to the conditions stated in this paper.

References

1. Abdalla, M., Chevalier, C., Manulis, M., Pointcheval, D.: Flexible Group Key Exchange with On-demand Computation of Subgroup Keys. In: Bernstein, D.J., Lange, T. (eds.) AFRICACRYPT 2010. LNCS, vol. 6055, pp. 351–368. Springer, Heidelberg (2010)

2. Abdalla, M., Fouque, P.-A., Pointcheval, D.: Password-Based Authenticated Key Exchange in the Three-Party Setting. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 65–84. Springer, Heidelberg (2005)
3. Abdalla, M., Pointcheval, D.: A Scalable Password-Based Group Key Exchange Protocol in the Standard Model. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 332–347. Springer, Heidelberg (2006)
4. Al-Riyami, S.S., Paterson, K.G.: Tripartite Authenticated Key Agreement Protocols from Pairings. In: Paterson, K.G. (ed.) Cryptography and Coding 2003. LNCS, vol. 2898, pp. 332–359. Springer, Heidelberg (2003)
5. Baek, J., Safavi-Naini, R., Susilo, W.: Efficient Multi-receiver Identity-Based Encryption and Its Application to Broadcast Encryption. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 380–397. Springer, Heidelberg (2005)
6. Bellare, M., Pointcheval, D., Rogaway, P.: Authenticated Key Exchange Secure against Dictionary Attacks. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 139–155. Springer, Heidelberg (2000)
7. Bellare, M., Rogaway, P.: Entity Authentication and Key Distribution. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 232–249. Springer, Heidelberg (1994)
8. Bellovin, S.M., Merritt, M.: Augmented Encrypted Key Exchange: A Password-Based Protocol Secure against Dictionary Attacks and Password File Compromise. In: ACM CCS 1993, pp. 244–250. ACM (1993)
9. Boyko, V., MacKenzie, P.D., Patel, S.: Provably Secure Password-Authenticated Key Exchange Using Diffie-Hellman. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 156–171. Springer, Heidelberg (2000)
10. Bresson, E., Chevassut, O., Pointcheval, D., Quisquater, J.-J.: Provably Authenticated Group Diffie-Hellman Key Exchange. In: ACM CCS 2001, pp. 255–264. ACM Press (2001)
11. Bresson, E., Manulis, M.: Contributory Group Key Exchange in the Presence of Malicious Participants. IET Information Security 2(3), 85–93 (2008)
12. Bresson, E., Manulis, M.: Securing Group Key Exchange against Strong Corruptions. In: ACM ASIACCS 2008, pp. 249–260. ACM Press (2008); full version in Intl. J. Applied Cryptography in 2008
13. Bresson, E., Manulis, M., Schwenk, J.: On Security Models and Compilers for Group Key Exchange Protocols. In: Miyaji, A., Kikuchi, H., Rannenberg, K. (eds.) IWSEC 2007. LNCS, vol. 4752, pp. 292–307. Springer, Heidelberg (2007)
14. Canetti, R., Krawczyk, H.: Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 453–474. Springer, Heidelberg (2001)
15. Choo, K.-K.R.: Secure Key Establishment. Advances in Information Security, vol. 41. Springer (2009)
16. Cremers, C.J.F.: Examining Indistinguishability-Based Security Models for Key Exchange Protocols: The case of CK, CK-HMQV, and eCK. In: ASIACCS 2011, pp. 80–91. ACM, New York (2011)
17. Diffie, W., Hellman, M.E.: New Directions in Cryptography. IEEE Transactions on Information Theory IT-22(6), 644–654 (1976)
18. Fujioka, A., Suzuki, K.: Designing Efficient Authenticated Key Exchange Resilient to Leakage of Ephemeral Secret Keys. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 121–141. Springer, Heidelberg (2011)
19. Gorantla, M.C., Boyd, C., González-Nieto, J.M., Manulis, M.: Modeling key compromise impersonation attacks on group key exchange protocols. ACM Trans. Inf. Syst. Secur. 14(4), 28 (2011)

20. Joux, A.: A one round protocol for tripartite Diffie–Hellman. Journal of Cryptology 17(4), 263–276 (2004)
21. Katz, J., Shin, J.S.: Modeling Insider Attacks on Group Key-Exchange Protocols. In: ACM CCS 2005, pp. 180–189. ACM Press (2005)
22. Katz, J., Yung, M.: Scalable Protocols for Authenticated Group Key Exchange. J. Cryptology 20(1), 85–113 (2007)
23. Kim, M., Fujioka, A., Ustaoğlu, B.: Strongly Secure Authenticated Key Exchange without NAXOS' Approach. In: Takagi, T., Mambo, M. (eds.) IWSEC 2009. LNCS, vol. 5824, pp. 174–191. Springer, Heidelberg (2009)
24. Krawczyk, H.: HMQV: A High-Performance Secure Diffie-Hellman Protocol. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 546–566. Springer, Heidelberg (2005)
25. LaMacchia, B.A., Lauter, K., Mityagin, A.: Stronger Security of Authenticated Key Exchange. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) ProvSec 2007. LNCS, vol. 4784, pp. 1–16. Springer, Heidelberg (2007)
26. Lim, M.-H., Lee, S., Lee, H.: Cryptanalysis on improved one-round Lin-Li's tripartite key agreement protocol. Cryptology ePrint Archive, Report 2007/411
27. Lim, M.-H., Lee, S., Park, Y., Lee, H.: An Enhanced One-Round Pairing-Based Tripartite Authenticated Key Agreement Protocol. In: Gervasi, O., Gavrilova, M.L. (eds.) ICCSA 2007, Part II. LNCS, vol. 4706, pp. 503–513. Springer, Heidelberg (2007)
28. Lin, C.-H., Lin, H.-H.: Secure one-round tripartite authenticated key agreement protocol from Weil pairing. In: AINA 2005, vol. 2, pp. 135–138. IEEE (2005)
29. Manulis, M.: Security-Focused Survey on Group Key Exchange Protocols. Cryptology ePrint Archive, Report 2006/395 (2006), `http://eprint.iacr.org/2006/395`
30. Manulis, M.: Group Key Exchange Enabling On-Demand Derivation of Peer-to-Peer Keys. In: Abdalla, M., Pointcheval, D., Fouque, P.-A., Vergnaud, D. (eds.) ACNS 2009. LNCS, vol. 5536, pp. 1–19. Springer, Heidelberg (2009)
31. Manulis, M., Suzuki, K., Ustaoglu, B.: Modeling Leakage of Ephemeral Secrets in Tripartite/Group Key Exchange. In: Lee, D., Hong, S. (eds.) ICISC 2009. LNCS, vol. 5984, pp. 16–33. Springer, Heidelberg (2010)
32. Menezes, A., Ustaoglu, B.: Comparing the Pre- and Post-specified Peer Models for Key Agreement. In: Mu, Y., Susilo, W., Seberry, J. (eds.) ACISP 2008. LNCS, vol. 5107, pp. 53–68. Springer, Heidelberg (2008)
33. Moriyama, D., Okamoto, T.: An eCK-Secure Authenticated Key Exchange Protocol without Random Oracles. In: Pieprzyk, J., Zhang, F. (eds.) ProvSec 2009. LNCS, vol. 5848, pp. 154–167. Springer, Heidelberg (2009)
34. Okamoto, T.: Authenticated Key Exchange and Key Encapsulation in the Standard Model. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 474–484. Springer, Heidelberg (2007)
35. Okamoto, T., Pointcheval, D.: The Gap-Problems: A New Class of Problems for the Security of Cryptographic Schemes. In: Kim, K.-C. (ed.) PKC 2001. LNCS, vol. 1992, pp. 104–118. Springer, Heidelberg (2001)
36. Shim, K.: Efficient one round tripartite authenticated key agreement protocol from Weil pairing. IET Electronics Letters 39(2), 208–209 (2003)
37. Ustaoglu, B.: Comparing *SessionStateReveal* and *EphemeralKeyReveal* for Diffie-Hellman Protocols. In: Pieprzyk, J., Zhang, F. (eds.) ProvSec 2009. LNCS, vol. 5848, pp. 183–197. Springer, Heidelberg (2009)
38. Zhao, J., Gu, D., Gorantla, M.C.: Stronger security model of group key agreement. In: ASIACCS 2011, pp. 435–440. ACM (2011)