Strongly Secure Authenticated Key Exchange Protocol Based on Computational Diffie-Hellman Problem

Hai Huang and Zhenfu Cao

Department of Computer Science and Engineering, Shanghai Jiaotong University, 800 Dongchuan Road, Shanghai, 200240, People's Republic of China
`chinesechess@sjtu.edu.cn; zfcao@cs.sjtu.edu.cn`

Abstract. Currently, there are a lot of authenticated key exchange (AKE) protocols in literature. However, the security proofs of this kind of protocols have been established to be a non-trivial task. The main issue is that without static private key it is difficult for simulator to fully support the SessionKeyReveal and EphemeralKeyReveal queries. Some proposals which have been proven secure either just hold in relatively weak models which do not fully support above-mentioned two queries or make use of the stronger gap assumption.
In this paper, using a new technique named twin Diffie-Hellman problem proposed by Cash, Kiltz and Shoup, we present a new AKE protocol based on the computational Diffie-Hellman (CDH) assumption, which is more standard than gap Diffie-Hellman (GDH) assumption. Moreover, our scheme is shown to be secure in strong security definition, the enhanced Canetti-Krawczyk (eCK) model introduced by LaMacchia, Lauter and Mityagin, which better supports the adversaries' queries than previous models.

Keywords: Authenticated key exchange, CDH problem, Twin DH problem, Trapdoor test, Provably secure

1 Introduction

Authenticated key exchange (AKE) is a cryptographic protocol, which enables two or more parties to establish a shared session key over an insecure channel. Later, the shared session key can be used to‘ efficiently ensure data integrity and confidentiality by symmetric primitives.

It is desirable for an AKE protocol to have the following attributes:

1.*Known-key security* (KKS): Each run of the protocol should result in a unique secret session key. It is reasonable to assume the adversary has the ability to learn the session keys except for one under attack. A protocol is said to be known-key secure if the compromise of one session key should not compromise other session keys.

2.*Forward security* (FS): If the static private key of an entity is compromised, the adversary can arbitrarily masquerade as that entity in future. However, we want to guarantee that when the static private key is compromised, the adversary can not obtain the session keys that were accepted before the compromise.

Protocols are said to provide perfect forward security (PFS) if the static private keys of all parties involved have been compromised without compromising the previously established session keys by these entities. However, if the adversary is actively involved with the choice of the DH values X, Y at a session, no two-message AKE protocol can achieve forward security, according to the result of HMQV [12]. So we define weak from of forward security (wFS).

3.*Key compromise impersonation resistance* (KCI): When the static private key of an entity, say A, is compromised, the adversary can arbitrarily masquerade as A in future. However, we want to guarantee that in this case the adversary cannot masquerade as another entity, say B, to communicate with A.

4.*Ephemeral key reveal resistance*: The adversary can obtain the ephemeral key of entities. Protocols are said to be ephemeral key reveal resistance if even when the adversary obtains the ephemeral key of entities the session key under attack still remains secure.

The design and analysis of secure AKE protocols have been established to be a non-trivial task. Bellare and Rogaway [1] firstly proposed a formal security model for authentication and key distribution. Since then, there have been several extensions to BR model [2–4]. All these models attempt to cover these desirable properties listed above as much as possible. Recently [5, 10] uniformly deal with these attacks in one model named eCK model, which better supports SessionKeyReveal and EphemeralKeyReveal queries compared with previous models.

1.1 Related Work

A great number of AKE protocols which focus on enhancing security or weakening the assumption are proposed in either ID-based setting or traditional PKI-based setting. In this paper, we focus on the latter in which the MQV protocol [11] is possibly the considerably efficient one. However, it is not known whether the MQV protocol can be proven secure in a strong security model. Later, based on MQV, Krawczyk proposes HMQV protocol [12], which has a formal security proof under GDH [15] and KEA1 [16] assumption. The security of HMQV covers KKS, resistance to KCI, resilience to the leakage of ephemeral private keys and wPFS etc.

K.Lauter and A.Mityagin propose an AKE protocol named KEA+ [13] under GDH assumption. However, their security model is weaker than that of HMQV. For instance, the adversary is not allowed to learn the static private keys of both two parties. In other words, KEA+ does not guarantee wPFS.

Kudla and Paterson [6] propose a modular proof approach to the design of AKE protocols, which makes use of gap assumption to keep the consistency of random oracle queries. By doing this, their method better supports the Session-KeyReveal queries.

Two recent AKE protocols NAXOS [5] and CMQV [10] are related to our work and both of them are shown secure in eCK model under GDH assumption. The trick for the AKE protocols to be proved secure in eCK model, as [5, 10] did, is to hash the ephemeral private key and static private key.

More recently, some people focus on the AKE protocols in standard model. T.Okamoto [7] proposes a PKI-based AKE protocol secure in eCK model without random oracle assumption. Their protocol bases the security on decisional Diffie-Hellman (CDH) assumption and the existence of pseudo-random functions with pair-wise independent random sources (πPRF). Boyd et al. [9] propose a generic approach to the design of AKE protocols based on a CCA-secure key encapsulation mechanism (KEM) primitive. They show that the resulted protocol is secure in CK model if the underlying KEM scheme is CCA-secure. We also note that Jeong et al. [8] propose an AKE protocol without random oracle in the traditional PKI-based setting. However, their protocol fails to achieve KCI resistance and thus their security model is weaker than ours.

Cash, Kiltz and Shoup [14] recently proposed a new computational problem called twin Diffie-Hellman problem, a nice feature of which not enjoyed by ordinary Diffie-Hellman problem is that the twin Diffie-Hellman problem remains hard, even with access to a decision oracle that recognizes solutions to the problem. At the heart of their method is the "trapdoor test" that allows us to implement an effective decision oracle for the twin Diffie-Hellman problem, without knowing the corresponding discrete logarithm.

As one application of the trapdoor test technique, they present a new variant of non-interactive Diffie-Hellman key exchange protocol.

1.2 Our Contributions

In this paper, based on the trapdoor test technique, we present a new interactive AKE protocol for the traditional PKI-based setting. We show that our protocol is secure in eCK model and bases its security on standard CDH assumption. Compared to previous AKE protocols based on gap assumption, our proposal has a more standard one. On the other hand, compared to other AKE protocols without gap assumption, our proposal has advantages over them either in efficiency or security model.

1.3 Organization

The paper is organized as follows. In section 2, we will review the related preliminaries. In section 3 we review the eCK model. Then we propose our scheme in section 4. In section 5, we will give its security proof in the eCK model. In section 6 we compare the performance and security between previous AKE protocols and ours. Finally, concluding remarks are made in section 7.

2 Preliminaries

Let the value k be the security parameter. Let $G = \langle g \rangle$ be the cyclic group of prime order q. Define $CDH(U, V) := Z$, where $U = g^u, V = g^v$, and $Z = g^{uv}$.
CDH Assumption. For any probabilistic polynomial time algorithm A,

$$Pr[A(q, g, U = g^u, V = g^v) = CDH(U, V)] \leq \epsilon(k).$$

where $u, v \in \mathbb{Z}_q$, and where $\epsilon(k)$ is negligible. The probability is taken over the coin tosses of A, the choice of q, g and the random choices of u and v in $\mathbb{Z}_q$.

Theorem 1 (Trapdoor Test [14]). *Let $G = \langle g \rangle$ be a cyclic group of prime order q, generated by $g \in G$. Suppose X_1, r, s are mutually independent random variables where X_1 takes values in G, and each of r, s is uniformly distributed over $\mathbb{Z}_q$, and define the random variable $X_2 := g^s / X_1^r$. Further, suppose that $\hat{Y}, \hat{Z}_1, \hat{Z}_2$ are random variables taking values in G, each of which is defined as some function of X_1 and X_2. Then we have:*
(i) X_2 is uniformly distributed over G;
(ii) X_1 and X_2 are independent;
(iii) if $X_1 = g^{x_1}$ and $X_2 = g^{x_2}$, then the probability that the truth value of

$$\hat{Z_1}^r \hat{Z}_2 \stackrel{?}{=} \hat{Y}^s \tag{1}$$

does not agree with the truth value of

$$\hat{Z}_1 \stackrel{?}{=} \hat{Y}^{x_1} \bigwedge \hat{Z}_2 \stackrel{?}{=} \hat{Y}^{x_2} \tag{2}$$

is at most $1/q$; moreover, if (2) *holds, then* (1) *certainly holds.*

Intuitionally, theorem 1 means that the simulator can use (1) to judge whether (2) holds (Knowing the discrete logarithm $\hat{y}$ of $\hat{Y}$, the adversary can computes $\hat{Z}_1, \hat{Z}_2$ itself, while the simulator cannot). This technique is essential for us to implement the effective decision oracle without knowing the corresponding discrete logarithm.

3 Security Model

Our basic security model is the eCK model. For more details of this model, see [5, 10].

Participants. We model the protocol participants as a finite set P of fixed size with each ID_i being a probabilistic polynomial time (PPT) Turing machine. Each protocol participant $ID_i \in P$ may execute a polynomial number of protocol instances in parallel. We will refer to s-th instance of principal ID_i communicating with peer ID_j as $\Pi^s_{i,j} (i, j \in N)$ (*a session*).

Adversary Model. The adversary M is modeled as a PPT Turing machine and has full control of the communication network and may eavesdrop, delay, replay, alter and insert messages at will. We model the adversary's capability by providing it with oracle queries.

- **EphemeralKeyReveal($\Pi^s_{i,j}$)** The adversary obtains the ephemeral private key of $\Pi^s_{i,j}$. These queries are motivated by practical scenarios, such as if session-specific secret information is stored in insecure memory on device or if the random number generator of party is corrupted.
- **SessionKeyReveal($\Pi^s_{i,j}$)** The adversary obtains the session key for a session s of ID_i, provided that the session holds a session key.

– **StaticKeyReveal**(ID_i) The adversary obtains the static private key of ID_i.
– **EstablishParty**(ID_i) The query models that the adversary can arbitrarily register the public key on behalf of the party ID_i. In this way the adversary totally controls the party ID_i. Parties against whom the adversary does not issue this query are called *honest*.
– **Send**$(\Pi^s_{i,j}, m)$ The adversary sends the message m to the session s executed by ID_i communicating with ID_j and get a response according to the protocol specification.
– **Test**$(\Pi^s_{i,j})$ Only one query of this form is allowed for the adversary. Provided that session key is defined, the adversary M can execute this query at any time. Then with probability 1/2 the session key and with probability 1/2 a uniformly chosen random value $\zeta \in \{0,1\}^k$ is returned.

Definition 1 (Matching Session). *Let $\Pi^s_{i,j}$ be a completed session with public output (ID_i, X, Y, ID_j), where ID_i is the owner of the session, ID_j is the peer, and X is ID_i's outgoing message, Y is ID_j's outgoing message. The session $\Pi^t_{j,i}$ is called the* matching session *of $\Pi^s_{i,j}$, if either $\Pi^t_{j,i}$ is completed with public output (ID_j, Y, X, ID_i) or it is incomplete with $((ID_j, Y, *, ID_i))$.*

Definition 2 (Freshness). *Let instance $\Pi^s_{i,j}$ be a completed session, which was executed by a honest party ID_i with another honest party ID_j. We define $\Pi^s_{i,j}$ to be* fresh *if none of the following three conditions hold:*

– *The adversary M reveals the session key of $\Pi^s_{i,j}$ or of its matching session (if latter exists).*
– *ID_j is engaged in session $\Pi^t_{j,i}$ matching to $\Pi^s_{i,j}$ and M either makes queries:*
*-both **StaticKeyReveal(ID_i)** and **EphemeralKeyReveal($\Pi^s_{i,j}$)**; or*
*-both **StaticKeyReveal(ID_j)** and **EphemeralKeyReveal($\Pi^t_{j,i}$)**.*
– *No sessions matching to $\Pi^s_{i,j}$ exist and M either makes queries:*
*-both **StaticKeyReveal(ID_i)** and **EphemeralKeyReveal($\Pi^s_{i,j}$)**; or*
*-**StaticKeyReveal(ID_j)**.*

Definition 3 (AKE Security). *As a function of the security parameter k, we define the advantage $Adv^{AKE}_{M,\Sigma}(k)$ of the PPT adversary M in attacking protocol Σ as*

$$Adv^{AKE}_{M,\Sigma}(k) \overset{def}{=} |Succ^{AKE}_{M,\Sigma}(k) - \tfrac{1}{2}|$$

*Here $Succ^{AKE}_{M,\Sigma}$ is the probability that the adversary queries **Test** oracle to a* fresh *instance $\Pi^s_{i,j}$, outputs a bit $\hat{b}$ such that $\hat{b} = b$, where the bit b is used by the **Test** oracle.*

We call the protocol Σ to be AKE secure if for any PPT adversary M the function is negligible.

Remark: The original CK model does not cover KCI attacks. The eCK model covers KCI attacks resistance, weak forward secrecy and ephemeral key reveal resistance etc. Moreover, in eCK model the adversary's ability is extended to the extent such that the adversary is allowed to reveal any static private key and ephemeral private key of parties involved except for both static private key and ephemeral private key of one of parties involved. We note that recently Boyd et al.'s research [9] compare these two models. Their conclusion is that the eCK is not stronger than the CK model. The essential difference comes from the fact the CK model allows session state reveal queries. This give the adversary complete information about the state of a given session at any entity, including all ephemeral values, but also any other value during the computation, while eCK model only allows the adversary to access to ephemeral values. However, we also note that the previous AKE protocols claiming to use CK model does not allow the adversary to get access to complete state information, e.g. HMQV.

4 Strongly Secure Authenticated Key Exchange Protocol Based on Computational Diffie-Hellman Problem

Parameters

Let k be the security parameter. The p is a large prime such that $p-1$ is divisible by another large prime q. The g is a generator of order q. We also denote by $G = \langle g \rangle$ cyclic group of order q and by G^* the set of non-identity elements in G. Let $H_1 : \{0,1\}^* \to \mathbb{Z}_q^*$ and $H : \{0,1\}^* \to \{0,1\}^k$ be hash functions modeled as random oracles. In the following parts we omit the operation " $\pmod p$ " to simplify the expression. The party $\hat{A}$'s static private key is a_1, a_2 and its public key is $A_1 = g^{a_1}, A_2 = g^{a_2}$. Similarly, the party $\hat{B}$'s static private key is b_1, b_2 and its public key is $B_1 = g^{b_1}, B_2 = g^{b_2}$. The protocol follows below.

$$
\begin{array}{lcl}
\hat{A} & & \hat{B} \\
A_1 = g^{a_1},\ \ A_2 = g^{a_2} & & B_1 = g^{b_1},\ \ B_2 = g^{b_2} \\
\tilde{x} \leftarrow_R \mathbb{Z}_q,\ x = H_1(\tilde{x}, a_1, a_2) & & \tilde{y} \leftarrow_R \mathbb{Z}_q,\ y = H_1(\tilde{y}, b_1, b_2) \\
 & \xrightarrow{X = g^x} & \\
 & \xleftarrow{Y = g^y} & \\
sid = (X, Y, \hat{A}, \hat{B}) & & sid = (X, Y, \hat{A}, \hat{B}) \\
Z_1 = (YB_1)^x,\ Z_2 = (YB_2)^x & & Z_1 = X^{y+b_1},\ Z_2 = X^{y+b_2} \\
Z_3 = Y^{x+a_1},\ Z_4 = Y^{x+a_2} & & Z_3 = (XA_1)^y,\ Z_4 = (XA_2)^y \\
SK = H(Z_1, Z_2, Z_3, Z_4, sid) & & SK = H(Z_1, Z_2, Z_3, Z_4, sid)
\end{array}
$$

Fig. 1. Our proposed protocol

Protocol description

1. $\hat{A}$ chooses an ephemeral private key $\tilde{x} \in \mathbb{Z}_q$ at random, computes $x = H_1(\tilde{x}, a_1, a_2)$ and ephemeral public key $X = g^x$. Then $\hat{A}$ destroys x and send X to $\hat{B}$. Similarly, $\hat{B}$ randomly chooses $\tilde{y} \in \mathbb{Z}_q$ at random, computes

$y = H_1(\tilde{y}, b_1, b_2)$ and ephemeral public key $Y = g^y$. Then $\hat{B}$ destroys y and send Y to $\hat{A}$.

2. Upon receiving X, party $\hat{B}$ verifies that $X \in G^*$. If so, $\hat{B}$ computes $y = H_1(\tilde{y}, b_1, b_2)$, $Z_1 = X^{y+b_1}$, $Z_2 = X^{y+b_2}$, $Z_3 = (XA_1)^y$, $Z_4 = (XA_2)^y$ and $SK = H(Z_1, Z_2, Z_3, Z_4, sid)$, where $sid = (X, Y, \hat{A}, \hat{B})$. $\hat{B}$ keeps SK as the established session key.
3. Similarly, upon receiving Y, $\hat{A}$ checks if $Y \in G^*$. If so, $\hat{A}$ computes $x = H_1(\tilde{x}, a_1, a_2)$, $Z_1 = (YB_1)^x$, $Z_2 = (YB_2)^x$, $Z_3 = Y^{x+a_1}$, $Z_4 = Y^{x+a_2}$ and $SK = H(Z_1, Z_2, Z_3, Z_4, sid)$, where $sid = (X, Y, \hat{A}, \hat{B})$. $\hat{A}$ keeps SK as the established session key.

5 Security Proof

Theorem 2. *Suppose that the CDH assumption for (G, g) holds, H, H_1 are random oracles, then the proposed scheme in Figure 1 is a secure AKE protocol in eCK model.*

Proof. Let k denote the security parameter. Assume that the adversary M activates at most $n(k)$ honest parties and $s(k)$ sessions in each party. Assume that the adversary succeeds with non-negligible probability in the environment described in Section 3. Since $H(\cdot)$ is modeled as a random oracle, after the adversary queries Test oracle, it has only two possible ways to distinguish a session key from a random string.

CASE 1 Forging attack: At some point in its run, the adversary M queries H on the value $(Z_1, Z_2, Z_3, Z_4, X, Y, \hat{A}, \hat{B})$ in the Test session owned by $\hat{A}$ communicating with $\hat{B}$. Clearly, in this case M computes these values Z_1, Z_2, Z_3, Z_4.

CASE 2 Key-replication attack: The adversary M forces a non-matching session to have the same session key with the Test session. In this case, the adversary M can simply learn the session key by querying the non-matching session.

The input to the key derivation function $H(\cdot)$ includes all information contained in *sid*. Since two non-matching sessions can not have same parties and same ephemeral public keys except for negligible probability and H is modeled as random oracle, the success probability of key replication attack is negligible.

The rest of this section is mainly devoted to the analysis of the CASE 1. We consider two complementary subcases:

CASE 1.1: No honest party owns a matching session to the Test session.

CASE 1.2: The Test session has a matching session owned by another honest party.

5.1 The analysis of CASE 1.1

In this case, following the standard approach, we will show how to construct CDH problem solver S that uses an adversary M who succeeds with non-negligible probability in CASE 1.1. The solver S is given a pair of CDH challenge (U, V), where $U, V \in G$. Its task is to compute $CDH(U, V) = U^v = V^u$. With probability at least $\frac{1}{n(k)^2}$, S guesses the adversary M will select one party denoted by $\hat{A}$

as the owner of the session $\hat{s}$ and the other party denoted by $\hat{B}$ as the peer. With probability at least $\frac{1}{s(k)}$, S guesses the adversary M will select the session $\hat{s}$ as Test session. Furthermore, S randomly chooses $s, r \in \mathbb{Z}_q$, assigns static public key $B_1 = V, B_2 = g^s/V^r$ for $\hat{B}$, and random static key pairs for the remaining $n(k) - 1$ parties (including $\hat{A}$). When the adversary M activates a party whose static key S possesses, S follows the protocol description. We next mainly discuss the action of S when the adversary M makes queries related to party $\hat{B}$ (because S does not know $\hat{B}$'s static private key). Without loss of generality, we assume that $\hat{B}$ is the responder.

- $H(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, \hat{Z}_4, X, Y, ID_i, ID_j)$: S maintains an initially empty list H^{list} with entries of the form $(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, \hat{Z}_4, X, Y, ID_i, ID_j, h)$. S simulates the oracle in usual way except for queries of the form $(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, \hat{Z}_4, X, Y, \hat{C}, \hat{B})$, where $\hat{C}$ is $\hat{B}$'s peer and may not be honest. The simulator S responds to these queries in the following way:
  - If $(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, \hat{Z}_4, X, Y, \hat{C}, \hat{B})$ is already there, then S responds with stored value h.
  - Otherwise, S looks in L^{list} (maintained in the Send query) for the entries of the form $(X, Y, \hat{C}, \hat{B}, *)$. If finds it, S computes

$$\bar{Z}_1 = \hat{Z}_1/X^y \tag{3}$$

$$\bar{Z}_2 = \hat{Z}_2/X^y \tag{4}$$

    Note that the value $\hat{Z}_i (i = 1, 2, 3, 4)$ is correctly generated iff $\hat{Z}_1 = (YB_1)^x$, $\hat{Z}_2 = (YB_2)^x$, $\hat{Z}_3 = Y^{x+c_1}$, $\hat{Z}_4 = Y^{x+c_2}$ which is equivalent to $\bar{Z}_1 = X^{b_1}$, $\bar{Z}_2 = X^{b_2}$, $Z_3 = (XC_1)^y$, $Z_4 = (XC_2)^y$. Thus S judges whether ${\bar{Z}_1}^r \bar{Z}_2$ equals X^s (Theorem 1), $\hat{Z}_3$ equals $(XC_1)^y$ and $\hat{Z}_4$ equals $(XC_2)^y$.
    - ∗ If the predicate evaluates to 0, S chooses $h \in \{0,1\}^k$ at random, sends it to M and stores the new tuple $(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, \hat{Z}_4, X, Y, \hat{C}, \hat{B}, h)$ in H^{list}.
    - ∗ Otherwise, it returns from L^{list} the stored value SK to M, stores the new tuple $(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, \hat{Z}_4, X, Y, \hat{C}, \hat{B}, SK)$ in H^{list}.
  - Otherwise (no such entries exist), S chooses $h \in \{0,1\}^k$ at random, sends it to M and stores the new tuple $(\hat{Z}_1, \hat{Z}_2, \hat{Z}_3, \hat{Z}_4, X, Y, \hat{C}, \hat{B}, h)$ in H^{list}.
- **StaticKeyReveal**(ID_i):
  - If $ID_i = \hat{B}$, then simulator fails (S does not know the corresponding static private key b_1, b_2).
  - Otherwise, S returns the corresponding static private key to M.
- **EstablishParty**(ID_i): The adversary M registers public key. In this case, S does not know adversary's private key.
- **EphemeralKeyReveal**$(\Pi^s_{i,j})$: The simulator returns the stored ephemeral private key $\tilde{y}$ to M.
- **Send**$(\Pi^s_{i,j}, m)$: S maintains an initially empty list L^{list} with entries of the form (X, Y, ID_i, ID_j, SK).

- If $\mathit{\Pi}_{i,j}^s$ is the Test session, then simulator returns U to M (We set the ephemeral public key of Test session owned by $\hat{A}$ to be U).
- If $ID_i = \hat{B}$ (For convenience, we set $ID_j = \hat{C}$ and $X = m$.)
  - ∗ S chooses $\tilde{y}, y \leftarrow_R \mathbb{Z}_q$ and returns $Y = g^y$ to the adversary M.
  - ∗ S looks in H^{list} for entry of the form $(*, *, *, *, X, Y, \hat{C}, \hat{B}, *)$. If finds it, S computes $\bar{Z}_1 = \hat{Z}_1/X^y = X^{b_1}$ and $\bar{Z}_2 = \hat{Z}_2/X^y = X^{b_2}$, where $\hat{Z}_i(i = 1, 2, 3, 4)$ are first four elements from entry above. Note that the value $\hat{Z}_i(i = 1, 2, 3, 4)$ is correctly generated iff $\hat{Z}_1 = (YB_1)^x$, $\hat{Z}_2 = (YB_2)^x$, $\hat{Z}_3 = Y^{x+c_1}$, $\hat{Z}_4 = Y^{x+c_2}$ which is equivalent to $\bar{Z}_1 = X^{b_1}$, $\bar{Z}_2 = X^{b_2}$, $\hat{Z}_3 = (XC_1)^y$, $\hat{Z}_4 = (XC_2)^y$.
    Thus S judges whether $\bar{Z}_1{}^r\bar{Z}_2$ equals X^s (Theorem 1), Z_3 equals $(XC_1)^y$ and Z_4 equals $(XC_2)^y$.
    - · If the predicate evaluates to 0, S chooses $SK \in \{0, 1\}^k$ at random and stores the new tuple $(X, Y, \hat{C}, \hat{B}, SK)$ in L^{list}.
    - · Otherwise, S stores the new tuple $(X, Y, \hat{C}, \hat{B}, h)$ in L^{list} (The value h is from H^{list}).
  - ∗ Otherwise, S chooses $SK \in \{0, 1\}^k$ at random and stores the new tuple $(X, Y, \hat{C}, \hat{B}, SK)$ in L^{list}.
- Otherwise $(ID_i \neq \hat{B})$,
  - ∗ S chooses $\tilde{y}, y \leftarrow_R \mathbb{Z}_q$ and returns $Y = g^y$ to the adversary M.
  - ∗ S chooses $SK \in \{0, 1\}^k$ at random and stores the new tuple (X, Y, ID_i, ID_j, SK) in L^{list}.

– **SessionKeyReveal**$(\mathit{\Pi}_{i,j}^s)$:
- If $\mathit{\Pi}_{i,j}^s$ is the Test session, then simulator fails.
- Otherwise, S returns the stored value SK in L^{list} to M.

– **Test**$(\mathit{\Pi}_{i,j}^s)$:
- If $\mathit{\Pi}_{i,j}^s$ is not the Test session, S aborts.
- Otherwise, S randomly chooses $\zeta \in \{0, 1\}^k$ and returns it to M.

As the attack that adversary M mounts is Forging attack, if M succeeds, it must have queried oracle H on the first two inputs $Z_1^* = (Y^*B_1)^x = (Y^*V)^u, Z_2^* = (Y^*B_2)^x = (Y^*g^s/V^r)^u$, where $X = U$ is the outgoing message of Test session, Y^* is the incoming message from the adversary. To solve $CDH(U, V)$ problem, for each entry in H^{list}, S proceeds with following steps:

S computes

$$\frac{Z_2^*}{Z_1^*} = \frac{(Y^*B_2)^u}{(Y^*B_1)^u} = (\frac{B_2}{B_1})^u = g^{(s-b_1r-b_1)u} \tag{5}$$

From (5), knowing s, r, the simulator S gets

$$\bar{Z}_1 = (g^{(s-b_1r-b_1)u}/U^s)^{\frac{-1}{r+1}} = B_1^u = V^u \tag{6}$$

This contradicts the CDH assumption.

On the other hand, because the adversary cannot reveal both $\hat{A}$'s ephemeral key and static key, the probability that the adversary makes queries H_1 on $(\tilde{x}, a_1, a_2)$ is negligible. Also, the adversary can not reveal $\hat{B}$'s static key. So the simulation of S is accurate and its success probability is

$$Pr[S] \geq \frac{1}{s(k)n(k)^2t(k)}p_1(k) \tag{7}$$

where $p_1(k)$ is the probability of the event that CASE 1.1 occurs and the adversary M succeeds in this case. $t(k)$ is the polynomial bound on the number of distinct H calls made by the adversary M.

5.2 The Analysis of CASE 1.2

In this case, given the CDH instance U, V, where $U, V \in G$. Its task is to compute $CDH(U,V) = U^v = V^u$. With probability at least $\frac{2}{s(k)^2}$, F guesses that the adversary M will select one of two sessions as Test session and the other as matching session. We assume that the owner of Test session is $\hat{A}$ and owner of matching session is $\hat{B}$. The simulator S sets the ephemeral public key of Test session and its matching session to be U, V respectively. The simulations of $\hat{A}$ and $\hat{B}$ (for which S knows corresponding private key) is trivial. If M succeeds in the Test session, then M must have queried the value $Z_1^* = (YB_1)^x = (VB_1)^u, Z_2^* = (YB_2)^x = (VB_1)^u$, where $X = U$ is the outgoing message of Test session, $Y = V$ is the outgoing message of its matching session. From one of these values, say Z_1^*, knowing B's static private b_1, b_2, S can computes $CDH(U,V) = Z_1^*/U^{b_1} = V^u = U^v$.

Because the adversary cannot reveal both $\hat{A}(\hat{B})$'s ephemeral key and static key, the probability that the adversary makes queries H_1 on $(\tilde{x}, a_1, a_2)((\tilde{y}, b_1, b_2))$ is negligible. So the simulation of S is accurate and its success probability is

$$Pr[S] \geq \frac{2}{s(k)^2t(k)}p_2(k) \tag{8}$$

where $p_2(k)$ is the probability of the event that CASE 1.2 occurs and the adversary M succeeds in this case. $t(k)$ is the polynomial bound on the number of distinct H calls made by the adversary M.

Together with (7),(8), the success probability of S is

$$Pr[S] \geq \max\{\frac{1}{s(k)n(k)^2t(k)}p_1(k), \frac{2}{s(k)^2t(k)}p_2(k)\} \tag{9}$$

where $p_1(k), p_2(k)$ are defined in (7),(8) respectively. $t(k)$ is the polynomial bound on the number of distinct H calls made by the adversary M.

If the adversary M succeeds with non-negligible probability in any case above, we can also solve the CDH problem with non-negligible probability, which contradicts the assumed security of CDH problem.

So we complete the proof of Theorem 2.

6 Comparison of performance and security

In Table 1 we compare our protocol with several popular traditional PKI-based AKE protocols in term of efficiency, security model and underlying hardness assumptions. For simplicity, we do not take into account subgroup validation and speedup trick that may be applicable. Also, we just consider expensive operations and denote by “E” the exponentiation in G. We denote by “Enc” the CCA-secure encryption algorithm of KEM and by “Dec” the corresponding decryption algorithm. Also, we denote by “ROM” the random oracle model and by “ Standard” the standard model. CK denotes Canetti-Krawczyk security [4] without perfect forward secrecy. BR denotes the Bellare-Rogaway model [1], where no EphemeralKeyReveal queries are allowed. KCI denotes security against key-compromise impersonation. wPFS denotes weak perfect forward secrecy.

Protocol	*Computation*	*Security Model*	*Assumption*
KEA+ [13]	3E	CK,KCI	GDH, ROM
HMQV [12]	2.5E	CK, KCI, wPFS	GDH, KEA1, ROM
NAXOS [5]	4E	eCK	GDH, ROM
CMQV [10]	3E	eCK	GDH, ROM
Kudla-Paterson[6]	3E	BR, KCI	GDH, ROM
Jeong-Katz-Lee[8]	3E	BR, wPFS	DDH, secure MACs, Standard
Okamato[7]	8E	eCK	DDH, πPRF, Standard
Boyd et al.[9]	1Enc+1Dec+2E	CK, KCI, wPFS	*[1], Standard
Our scheme	5E	eCK	CDH, ROM

Table 1. Protocol comparison

Compared with the KEA+ and Kudla-Paterson our scheme has advantages in both hardness assumption and security model. Compared with HMQV, the CDH assumption of our scheme is more standard than GDH and KEA1 assumption. While NAXOS and CMQV are shown secure in eCK model, both of which base their security on stronger GDH assumption.

While Jeong-Katz-Lee’s protocol is efficient in standard model, their protocol fails to achieve KCI resistance and thus their security model is weaker than ours. On the other hand, while Okamoto’s protocol is secure in eCK model without random oracle assumption, their protocol bases the security on decisional Diffie-Hellman (DDH) problem and another assumption of existence of πPRF, whose construction from a fundamental primitive like a one-way function or (trapdoor) one-way permutation is an open problem [7].

The comparison between Boyd et al.’s protocol and ours is a bit complicated. Since their protocol is generic, it can be instantiated using any combination of KEM as long as they are CCA secure. If the underlying KEM is instantiated using any KEM scheme in standard model, obviously, our protocol in random oracle is more efficient. On the other hand, if the underlying KEM is instantiated using any KEM scheme in random oracle, say, twin Elgamal encryption scheme [14], which is CCA-secure based on CDH assumption, the operations of

[1] The assumption of this protocol depends on that of underlying KEM.

1Enc+1Dec+2E need 3+2+2=7 exponentiations totally while our scheme needs 5 exponentiations.

7 Conclusion

For the simulator to better support the SessionKeyReveal and EphemeralKeyReveal queries, a lot of AKE protocols base their security on the stronger gap assumption, which is a basic technique for the simulator to keep the consistency of random oracle. In this paper, based on the twin Diffie-Hellman problem proposed by Cash, Kiltz and Shoup, a new traditional PKI-based AKE protocol is proposed and its security is proved in eCK model based on CDH assumption.

Compared to previous AKE protocols based on gap assumption, our proposal has more standard one, i.e. CDH assumption. On the other hand, compared to other AKE protocols without gap assumption, our proposal has advantages over them in either efficiency or hardness assumption.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments. This work was supported in part by the National High Technology Development Program of China under Grant No. 2006AA01Z424, National 973 Program under Grant No. 2007CB311201, and the National Natural Science Foundation of China under Grant No. 60673079.

References

1. M.Bellare, P.Rogaway, Entity authentication and key distribution, Advances in Cryptology-CRYPTO'93, volume 773 of Lecture Notes in Computer Science, pages 232-249. Springer, 1993.
2. M.Bellare, P. Rogaway, Provably Secure Session Key Distribution: The Three Party Case, In Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing (STOC'95), pages 57-66. ACM Press, 1995.
3. M.Bellare, D.Pointcheval, and P. Rogaway, Authenticated key exchange secure against dictionary attacks, In Advances in Cryptology-EUROCRYPT'00, volume 1807 of Lecture Notes in Computer Science, pages 139-155. Springer, May 2000.
4. R.Canetti, H.Krawczyk, Analysis of key-exchange protocols and their use for building secure channels, In Advances in Cryptology-EUROCRYPT'01, volume 2045 of Lecture Notes in Computer Science, pages 453-474. Springer, 2001.
5. B.LaMacchia, K.Lauter, A.Mityagin, Stronger security of authenticated key exchange, ProvSec 2007, Lecture Notes in Computer Science, 4784 (2007), 1-16.
6. C. Kudla and K. Paterson. Modular security proofs for key agreement protocols. In Advances in Cryptology -Asiacrypt 2005, Springer-Verlag LNCS 3788, 549-565, 2005.
7. T.Okamoto. Authenticated key exchange and key encapsulation in the standard model. In Advances in Cryptology-ASIACRYPT 2007, volume 4833 of Lecture Notes in Computer Science, pages 474-484. Springer, 2007. Full version in Cryptology ePrint Archive, Report 2007/473, http://eprint. iacr.org/.

8. I. R. Jeong, J. Katz, and D. H. Lee. One-round protocols for two-party authenticated key exchange. In Applied Cryptography and Network Security, Second International Conference, ACNS 2004, volume 3089 of Lecture Notes in Computer Science, pages 220-232. Springer, 2004.
9. Colin. Boyd, Yvonne Cliff, Juan Gonzalez Nieto, and Kenneth G. Paterson. Efficient one-round key exchange in the standard model. Information Security and Privacy 2008, Volume 5107 of Lecture Notes in Computer Science, pages 69-83. Springer. Full version in http://eprint.iacr.org/2008/007.
10. B.Ustaoglu, Obtaining a secure and efficient key agreement protocol from (H)MQV and NAXOS, Designs, Codes and Cryptography, 46 (2008), 329-342.
11. L.Law, A.Menezes, M.Qu, J.Solinas, S.Vanstone, An efficient protocol for authenticated key agreement, Designs, Codes and Cryptography 28(2003), 119-134.
12. H.Krawczyk, HMQV: A high-performance secure Diffie-Hellman protocol, Advances in Cryptology CRYPTO 2005, LNCS 3621, 546-566, Full version available at http://eprint.iacr.org/2005/176.
13. K.Lauter, A.Mityagin, Security analysis of KEA authenticated key exchange, Public Key Cryptography PKC 2006, LNCS 3958, 378-394.
14. D.Cash, E.Kiltz, V.Shoup, The Twin Diffie-Hellman problem and applications, Advances in Cryptlogy EUROCRYPT 2008, Full version available at http://eprint.iacr.org/2008/067.
15. T.Okamoto, D.Pointcheval, The Gap-Problems: A new class of problems for the security of cryptographic schems, Public Key Cryptography PKC 2001,LNCS 1992(2001),104-118.
16. M. Bellare, A. Palacio, The knowledge-of-exponent assumptions and 3-round zero knowledge Protocols, Crypto'04, LNCS 3152.