A Provable Secure ID-Based Explicit Authenticated Key Agreement Protocol without Random Oracles

Hai-Bo Tian[1,2], Willy Susilo[3], Yang Ming[4], and Yu-Min Wang[5]

[1]*School of Information Science and Technology, Sun Yat-Sun University, Guangzhou, China*

[2]*Guangdong Key Laboratory of Information Security Technology, Guangzhou 510275, P.R.China*

[3]*Centre for Computer and Information Security Research (CCISR) School of Computer Science and Software Engineering University of Wollongong, Australia*

[4]*School of Information Engineering, Chang'an University, Xi'an 710064, China*

[5]*State key Lab. on ISN, Xidian University, Xi'an, China*

E-mail: tianhb@mail.sysu.edu.cn; wsusilo@uow.edu.au; mingyang2001@sohu.com; ymwang@xidian.edu.cn

Received August 6th, 2007.

Abstract In this paper, we present an identity-based *explicit authenticated* key agreement protocol that is provably secure without random oracles. The protocol employs a new method to isolate a session key from key confirmation keys so that there is no direct usage of hash functions in the protocol. The protocol is proved secure without random oracles in a variant of Bellare and Rogaway style model, an exception to current proof method in this style model in the ID-based setting. We believe that this key isolation method is novel and can be further studied to construct more efficient protocols.

Keywords Cryptography, Identity-based, Key Agreement, Random Oracles

1 Introduction

This paper focuses on an identity based key agreement protocol with a standard proof. We introduce some concepts to parse the topic including an *explicit* authenticated key agreement protocol, an *identity-based* protocol, common security properties of key agreement protocols, proof models of such protocols and usage of random oracles in this field. Related works about *identity-based* key agreement protocols, security properties, proof models and usage of random oracles are embed in above concepts introduction parts.

An *explicit* authenticated key agreement protocol is a key agreement protocol which provides explicit key authentication [1]. A key

* This work is supported by National Natural Science Foundation of China under Grant No 60473027, also by Sun Yat-Sen university under Grant No 35000-2910025,35000-3171912

agreement protocol or mechanism is a key establishment technique in which a shared secret is derived by two (or more) parties as a function of information contributed by each of these, and ideally no party can predetermine the resulting value. Key establishment is a process or protocol whereby a shared secret becomes available to two or more parties for subsequent cryptographic use. Explicit key authentication is the property obtained when *both* implicit key authentication and key confirmation hold. Implicit key authentication is the property whereby one party is assured that no other party aside from a specifically identified second party (and possibly additional identified trusted parties) may access to a particular secret key. Finally, key confirmation is the property whereby one party is assured that a second party (possibly unidentified) actually has possession of a particular secret key.

A key agreement protocol is said to be *identity-based* (ID-based) if the identity information of the party involved is used as the party's public key. After Shamir proposed the idea of identity-based asymmetric key pairs [2], a few identity-based key agreement protocols based on Shamir's idea have been developed, such as [3-5]. However the practical ID-based protocols boomed after appeared the work of [6] and [7] based on pairing techniques, which include [8-16]. The practical protocols enjoy some security properties, such as perfect forward security (PFS), key compromise impersonation resilience (KCI) etc.

Usually, some security properties are used to evaluate the security of key agreement protocols, including known session key security (KSK), unknown key-share resilience (UKS), PFS, and KCI etc. By KSK, we mean that the compromise of one session key should not compromise the keys established in other sessions. UKS means that party A should not be able to be coerced into sharing a key with party C when in fact A thinks that she/he is sharing the key with some party B. PFS in the two-party case usually means that if their private keys are compromised, the secrecy of session keys previously established by the two parties should not be affected. If the condition is relaxed to only one principle, it is called partially forward security (P-FS). If the condition is restricted by adding the loss of the third trusted party's master key in the ID-based scenario, it is called master-key forward security (M-FS) [15]. By KCI, we mean that the compromise of party A's long-term private key should not enable the adversary to impersonate other parties to A. Some of the above security properties can be captured by a Bellare and Rogaway (BR) style model.

To the best of our knowledge, there are some models to prove ID-based protocols, including BR model [17], BRP model [18], BCP model [19], CK model [20], UC model [21] etc. Most ID-based protocols are proved in some variants of the BR model, such as protocols in [9,12-16]. Usually, an adversary in a BR style model is powered by some kinds of queries, such as Send, Reveal, Corrupt queries etc. The execution of a protocol is described as oracle responses to the adversary's queries. After polynomial bounded times queries, the adversary is expected to pass a test with a non-negligible probability. If the adversary cannot pass the test and the protocol transcripts satisfy some secure conditions, it is believed that the protocol is secure in the defined model. Roughly all BR style models are defined and used in the above fashion.

The original BR model provides us a good framework but it is not suitable for key agreement protocols. Blake-Wilson, Johnson, and Meneze (BJM) extended the BR model to the public key setting [22]. The KSK and UKS properties have been built into the BJM model. The KCI was built into another variant model proposed by Cheng et al in the definition of no-matching [23] for authenticated key agreement with key confirmation protocols. So one can prove a protocol secure with one fresh condition capturing KSK, UKS, and KCI properties [15]. For PFS, there is another independent fresh condition is defined, and another independent proof procedure is needed [15,23,24]. Another security property SSR also takes the way to define an independent fresh condition, which considers the leakage of temporal private keys [23-24]. Here we just give arguments about the PFS and SSR properties out of our proof model so that we can present a more clear proof procedure without random oracles in the model.

Blake-Wilson etc adopted the random oracle model (ROM) in their proof procedure. The powerful tool was proposed by Bellare and Rogaway. It is used almost in every key agreement protocols with key confirmation after Blake - Wilson etc's work, where hash functions are used to isolate a session key from confirmation keys. Recently ROM is debated for its uninstantiable property [25-27]. Following the conservative culture in cryptography [28], we believe that it is meaningful to provide a proof without ROM for key agreement protocols. At least, it can reveal what happened when ROM is absent. Note that a traditional Deffie-Hellman protocol was proved in [24] without ROM. Their proof lacks an obvious no-matching proof since their protocol was under the assumption of duplex channel, i.e. simultaneous message transmission.

Our Contributions

We fail to find some *direct* related works about identity based *explicit* key agreement protocols with a standard proof. In fact, this is the purpose of our protocol. We note the trend of stand proof for schemes and protocols. Also we note that there is no explicit authenticated key agreement protocols with a stand proof in the identity based cryptography field. Motivated by Gentry's excellent work[29], we are deliberated to design a protocol with a stand proof. We deem that this protocol design method can be applied further if some more efficient schemes than Gentry's are proposed.

The main difference of our protocol design method lies in the MAC key and session key generation and isolation fashion, which makes it possible that there is no direct usage of hash functions in our protocol. Let's explain our design procedure step by step. To exclude ROM in ID-based protocols, we firstly adopt a private key generation method where hash functions are not needed. Gentry in EuroCrypt 2006 proposed an IND-CPA ID-based encryption scheme [29], which can be proven secure without random oracles. His method is adopted here. Secondly we need another method to isolate a session key from confirmation keys. We use key materials of a session key as confirmation keys if key materials and the session key can construct a hard problem. For example, considering the tuple (g, g^x, g^y, g^{xy}), we can use (g^x, g^y) as confirmation keys and g^{xy} as the session key. Then the Deffie-Hellman problem isolates confirmation keys from the session key. At last, we use MTI serials (C0) protocol to hide confirmation keys from an adversary.

Then we elaborate to give the stand proof. A key step is to show the indistinguishability of random confirmation keys from real confirmation keys in a protocol run. With such a conclusion, we can deduce adversary's no-matching advantage to a MAC forger's advantage. With the authentication conclusion, we can further construct a simulator to solve a hard problem, who plays a test game with an adversary, so as to deduce adversary's advantage to the simulator's advantage. The proof steps are similar with Blake-Wilson etc's work but with a big difference that there is no ROM.

Roadmap

The rest of this paper is organized as follows. The introduction of bilinear maps, and complexity assumption of our protocol are reviewed in Section 2. In Section 3, we present our ID-EAKA protocol. The security model, proof and security properties of the protocol are provided in Section 4. Section 5 concludes the paper.

2 Preliminaries

In this section, we review the definition of bilinear maps and related complexity assumptions.

2.1 Bilinear Maps

Basic notations that are used throughout this paper are as follows.

1. G and G_T are two (multiplicative) cyclic groups of prime order p;
2. g is a generator of G;
3. e: $G \times G \rightarrow G_T$ is a bilinear map.

Let G and G_T be two groups as above. A bilinear map is a map e: $G \times G \rightarrow G_T$ with the following properties:

1. Bilinear: for all u, v in G and a, b in Z_p, we have $e(u^a, v^b) = e(u, v)^{ab}$;

2. Non-degenerate: $e(g, g) \neq 1$.

We say that G is a bilinear group if the group action in G can be computed efficiently and there exists a group G_T and an efficiently computable bilinear map e: $G \times G \rightarrow G_T$ as above. Here the bilinear map is symmetric since $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

2.2 Complexity Assumptions

The security of our protocol is based on a complexity assumption that is known as a truncated version of the decisional augmented bilinear Diffie-Hellman exponent assumption in [29] (truncated decisional ABDHE).

Truncated *q*-ABDHE

The problem is that given a vector of q+3 elements

$$(g', g'^{(\alpha^{q+2})}, g, g^{\alpha}, g^{\alpha^2}, ..., g^{(\alpha^q)}) \in G^{q+3}$$

as input, outputs $e(g, g')^{(\alpha^{q+1})}$. We use g_i and g'_i to denote $g^{(\alpha^i)}$ and $g'^{(\alpha^i)}$ below. An algorithm A has advantage ϵ in solving a truncated q-ABDHE problem if $\Pr[A(g', g'_{q+2}, g, g_1, ..., g_q) = e(g_{q+1}, g')] \geq \epsilon$, where the probability is over the random choice of generators g, g' in G, the random choice of α in Z_p, and the random bits used by A.

The assumption is that there is no such an probability polynomial time (p.p.t) algorithm A has a non-negligible advantage ϵ.

Truncated decisional *q*-ABDHE

The problem of decisional version of truncated q-ABDHE is defined as one would expect. An algorithm A that outputs $b \in \{0,\ 1\}$ has advantage ϵ in solving a truncated decisional q-ABDHE problem if

$$\left| \begin{array}{l} \Pr[A(g', g'_{q+2}, g, g_1, ..., g_q, e(g_{q+1}, g')) = 0] \\ - \Pr[A(g', g'_{q+2}, g, g_1, ..., g_q, Z) = 0] \end{array} \right| \geq \epsilon$$

where the probability is over the random choice of generators g, g' in G, the random choice of α in Z_p, the random choice of $Z \in G_T$, and the random bits consumed by A.

The truncated decisional $(t,\ \epsilon,\ q)$-ABDHE assumption holds in G if no t-time algorithm has advantage at least ϵ in solving the truncated decisional q-ABDHE problem in G.

Remarks

We note the truncated q-ABDHE problem was introduced by Gentry [29]. The normal version, which is not truncated, is called the q-ABDHE problem. The q-ABDHE problem has additional $(q-1)$ input terms, which seems easier to solve than the truncated version. The q-ABDHE problem is similar with the q-BDHE problem used in [30,31]. The difference is that the q-ABDHE problem has an additional input term $g'^{(\alpha^{q+2})}$. Gentry argued that introducing the additional term did not appear to ease the computation of $e(g, g')^{\alpha^{q+1}}$, since the input vector was missing the term $g^{(\alpha^{-1})}$ [29].

2.3 MAC Algorithm

We use the MAC security definition in [32], where a practical *one key CBC MAC* scheme is defined. We need the unforgeable definition here.

A MAC algorithm is a map $MAC : \mathcal{K}_{MAC} \times \{0,1\}^* \rightarrow \{0,1\}^n$, where $\mathcal{K}_{MAC}$ is a set of keys and we write $MAC_K(\cdot)$ for $MAC(K, \cdot)$. We say that an adversary $A^{MAC_K(\cdot)}$ *forges* if A outputs $(M, MAC_K(M))$ where A never queried M to its oracle $MAC_K(\cdot)$. The advantage is defined as

$$\begin{cases} Adv_{MAC}(A) \overset{\text{def}}{=} \Pr(K \overset{R}{\leftarrow} \mathcal{K}_F : A^{MAC_K(\cdot)}\ forges) \\ Adv_{MAC}(t,q,u) \overset{\text{def}}{=} \max_{A}\{Adv_{MAC}(A)\} \end{cases}$$

where the maximum is over all adversaries who run in time at most t, make at most q queries, and each query is at most u bits. We say that a MAC algorithm is secure if $Adv_{MAC}(t,q,u)$ is sufficiently small.

3 The ID-EAKA Protocol

There are three entities involved in our protocol: two users Alice and Bob who wish to establish an authenticated shared secret session key, and a PKG who generates user private keys using its public/private key pairs.

The PKG generates its public/private key pairs as follows. Let G and G_T be groups of order p, and let $e : G \times G \rightarrow G_T$ be the bilinear map. The PKG picks randomly generators g, $h \in G$ and $\alpha \in Z_p$. It sets $g_1 = g^\alpha \in G$ and $g_T = e(g,g) \in G_T$. The public/private key pairs are given by public key = (g, g_1, h, g_T, MAC), private key = α, where MAC is a public MAC algorithm enjoying unforgeable property. Note that $MAC : \mathcal{K}_{MAC} \times \{0,1\}^* \rightarrow \{0,1\}^n$. We assume that the key set $\mathcal{K}_{MAC}$ is the group G_T. Certainly, we also can assume that there is a public algorithm to uniformly map elements in G_T to the key set $\mathcal{K}_{MAC}$. For simplicity, we just use G_T as $\mathcal{K}_{MAC}$ in the protocol description.

The PKG generates user keys as follows. To generate a private key for identity $ID \in Z_p$, the PKG generates random $r_{ID} \in Z_p$, and outputs the private key $d_{ID} = (r_{ID}, h_{ID})$, where $h_{ID} = (hg^{-r_{ID}})^{1/(\alpha - ID)}$. If $ID = \alpha$, the PKG aborts.

With user keys, Alice and Bob run the following protocol to establish a shared session key with *explicit* key authentication. We use ID_A and ID_B to denote the identification strings of Alice and Bob. Figure 1 depicts the protocol. The detail procedure is as follows.

1. Alice uniformly at random selects $x \in Z_p$, computes $M_{11} = (g_1 g^{-ID_B})^x$ and $M_{12} = g_T^x$. Alice sends $M_1 = ID_A||M_{11}||M_{12}$ to Bob, where symbol || denotes concatenation.

2. Bob uniformly at random selects $y \in Z_p$, computes $M_{211} = (g_1 g^{-ID_A})^y$, $M_{212} = g_T^y$, and $M_{22} = MAC_{KM_{BA}}(M_1||ID_B||M_{211}||M_{212})$ where $KM_{BA} = (M_{12})^{r_{ID_B}} e(M_{11},$

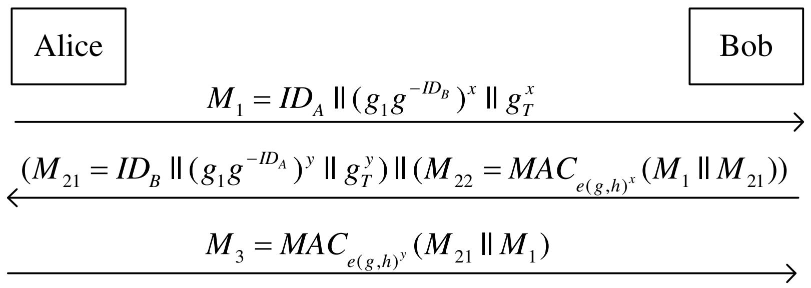
Alice
Bob
$M_1 = ID_A \| (g_1 g^{-ID_B})^x \| g_T^x$
$(M_{21} = ID_B \| (g_1 g^{-ID_A})^y \| g_T^y) \| (M_{22} = MAC_{e(g,h)^x}(M_1 \| M_{21}))$
$M_3 = MAC_{e(g,h)^y}(M_{21} \| M_1)$

Figure 1: The ID-Based Explicit Authenticated Key Agreement Protocol

h_{ID_B}). Let M_{21} denote $ID_B||M_{211}||M_{212}$. Bob sends $M_2 = M_{21}||M_{22}$ to Alice.

3. Alice computes $KVM_{AB} = e(g,h)^x$, $V_{M_{22}}$=$MAC_{KVM_{AB}}(M_1||M_{21})$. If $M_{22} \neq V_{M_{22}}$, Alice rejects and aborts the protocol. Else if $M_{22} = V_{M_{22}}$, Alice accepts, computes $KM_{AB} = e(M_{211}, h_{ID_A})(M_{212})^{r_{ID_A}}$, sets $K_{AB} = {KM_{AB}}^x$ as the session key. Then Alice computes $M_3 = MAC_{KM_{AB}}(M_{21}||M_1)$, and sends M_3 to Bob.

4. Bob computes $KVM_{BA} = e(g,h)^y$, $V_{M_3} = MAC_{KVM_{BA}}(M_{21}||M_1)$. If $M_3 \neq V_{M_3}$, Bob rejects and aborts the protocol. Else Bob accepts and sets $K_{BA} = {KM_{BA}}^y$ as the session key.

4 Security analysis

This section presents a security model, the proof in the model and security properties of our protocol.

4.1 Security Model

Our security model is based on the model of Blake-Wilson etc[22] for key agreement protocols and a no-matching definition in [23]. The no-matching definition is also adopted in [15].

In the model, an oracle $\Pi^s_{i,j}$ models the behavior of a party with identity i carrying out a protocol session in the belief that it is communicating with a party with identity j for the s-th time, where $i, j \in I, s \in N_1$. The total number of possible parties is denoted by symbol $|I|$ and the total session number is denoted by symbol $|N_1|$. One oracle instance is used only for one time, which maintains a variable *view* consisting of the oracle's protocol transcripts so far.

An adversary is modeled by a probabilistic polynomial time Turing machine that is assumed to have complete control over all communication links in the network and to interact with parties via oracle accesses. The adversary A is allowed to execute any of the following queries.

- *Corrupt* (i). This allows the adversary to get the long term private key of the party

i. If party i doesn't exist, the system will setup a private key for the party, and send the private key to the adversary.

- *Send* ($\Pi^s_{i,j}$, X). The adversary sends a message X to the oracle $\Pi^s_{i,j}$. The system will give an output of $\Pi^s_{i,j}$ to the adversary as response. If $X = \lambda$, the party i is asked to initiate a session s with party j, where λ is an empty string.

- *Reveal* ($\Pi^s_{i,j}$). This asks the oracle $\Pi^s_{i,j}$ to reveal whatever session key it currently holds.

An oracle exists in one of the following several possible states:

- ***Accepted***: an oracle has accepted if it decides to accept, holding a session key, after receipt of properly formulated messages.

- ***Rejected***: an oracle has rejected if it decides not to establish a session key and to abort the protocol.

- ***Unsettled***: an oracle is unsettled if it has not made any decision to accept or reject.

- ***Revealed***: an oracle is opened if it has answered a *Reveal* query.

- ***Corrupted***: an oracle is corrupted if it has involved in a *Corrupt* query.

By $\Pi^{s'}_{j,i}$, *matching oracle* of $\Pi^s_{i,j}$, we mean that every message that $\Pi^s_{i,j}$ sends out is subsequently delivered to $\Pi^{s'}_{j,i}$, with the response of $\Pi^{s'}_{j,i}$ to this message being returned to the $\Pi^s_{i,j}$ as the next message. The detail definition can be found in [17] or [22].

By *No-Matching*$(\cdot)$ event, we mean that when our protocol is running against an adversary, there exists an oracle $\Pi^s_{i,j}$ which has accepted but there is no oracle $\Pi^t_{j,i}$ which has engaged in a matching conversation to $\Pi^s_{i,j}$, where j has never been corrupted.

By *fresh oracle* $\Pi^s_{i,j}$, we mean that the oracle $\Pi^s_{i,j}$ is *Accepted*, not *Revealed*, party j is not *Corrupted*, the oracle $\Pi^t_{j,i}$ is not *Revealed* if $\Pi^t_{j,i}$ is a matching oracle of $\Pi^s_{i,j}$.

A *Test* query is defined for session key secrecy.

- *Test*($\Pi^s_{i,j}$). If an oracle $\Pi^s_{i,j}$ is fresh, an adversary can make a test query to it. To answer the query, the oracle flips a fair coin $b \leftarrow \{0, 1\}$, and returns the session key holding by oracle $\Pi^s_{i,j}$ if $b = 0$, or else a random key sampled from key space if $b = 1$.

After Test query, the adversary can continue making queries to oracles except the Corrupt query to the party j, and the Reveal query to oracle $\Pi^s_{i,j}$ and its possible matching oracle $\Pi^t_{j,i}$.

To complete the function of Test query, the advantage of an adversary is defined. After all possible queries are made, the adversary output a bit b'. The advantage is defined as:

$$Adv = |Pr[b' = b] - 1/2|$$

To define an explicit authenticated key agreement protocol, we should prove that the protocol satisfies the following goals:

1. Correctness. If two oracles are matching, then both of them are accepted and have the same session key which is distributed uniformly at random in the session key sample space.
2. Secrecy. *Adv* is negligible.
3. Authentication. The probability of *No-Matching*$(\cdot)$ is negligible.

Remarks

Another query is about *State*$(\cdot)$[24]. These queries are disabled in the above model so that the model cannot capture the SSR property or known session-specific temporary information security. A protocol satisfying SSR property means that the protocol session key is produced together by long term secret key and temporal key material[24]. This fashion itself has advantages and disadvantages[22]. Since the session key of our protocol is produced solely by temporal key materials, we are intended to exclude the special query.

4.2 Security Proof

The three goals are separately proved in three theorems. The first is dedicated for Correctness, the second for Authentication and the last for Secrecy. The second conclusion is used in the proof of the last theorem.

Theorem 4.1. *If two oracles are matching, then both of them are accepted and have a same session key which is distributed uniformly at random in the session key sample space.*

Proof. Suppose two oracles $\Pi^s_{i,j}$ and $\Pi^t_{j,i}$. Assume the oracle $\Pi^s_{i,j}$ receives the Send ($\Pi^s_{i,j}$, λ) query. Then the oracle $\Pi^s_{i,j}$ acts as an initiator and $\Pi^t_{j,i}$ as a responder. Before the initiator accepts, the initiator has a *view* (M_1, M_2) which is identical to the *view* of responder because the initiator and responder are matching. At that point,

$$KM_{ji} = e(M_{11}, h_j)(M_{12})^{r_j} =$$
$$e((g_1 g^{-j})^x, h_j)((g_T)^x)^{r_j} = \boldsymbol{e(g,h)^x} = KVM_{ij}$$

and the initiator and responder has identical vector $(M_1||M_{21})$, so the equality $M_{23} = V_{M_{23}}$ holds. The initiator will accept according to the protocol and give the last message to the responder.

Before the responder accepts, the responder has a *view* (M_1, M_2, M_3) which is identical to the *view* of initiator. At that point,

$$KM_{ij} = e(M_{211}, h_i)(M_{212})^{r_i} =$$
$$e((g_1 g^{-i})^y, h_i)((g_T)^y)^{r_i} = \boldsymbol{e(g,h)^y} = KVM_{ji}$$

Similarly, the responder will also accept.

The session key is $K_{ji} = KM^y_{ji} = \boldsymbol{e(g,h)^{xy}} = KM^x_{ij} = K_{ij}$, where $e(g,h)$ can be determined by public parameters. The session key is distributed uniformly in G_T since the exponent x and y are selected uniformly during the protocol execution. □

Theorem 4.2. *The probability of No-Matching$(\cdot)$ is negligible.*

Proof. Note that our *No-Matching*$(\cdot)$ event only requires one party remaining not Corrupted. So we divide the proof into two parts. The first part is for an initiator and the second part is for a responder.

Case 1: the probability of *No-Matching*$(\cdot)$ for an initiator is negligible.

The proof includes two phases. The phase one is to conclude the indistinguishability of two distributions $\{M_1||M_{21}||M_{22}\}$ and $\{M_1||M_{21}||MAC_{K\leftarrow\mathcal{K}}(\cdot)\}$. $\{M_1||M_{21}||M_{22}\}$ is a set of bit string which is concatenated by protocol messages M_1, M_{21} and M_{22}. $\{M_1||M_{21}||MAC_{K\leftarrow\mathcal{K}}(\cdot)\}$ is a set of bit string which is concatenated by protocol messages M_1, M_{21} and a MAC tag computed by a random MAC key and $M_1||M_{21}$. The phase two is to reduce an adversary's advantage to a MAC forger's advantage.

Phase 1.1: Suppose there is a p.p.t algorithm D. D can distinguish $\{M_1||M_{21}||M_{22}\}$ and $\{M_1||M_{21}||MAC_{K\leftarrow\mathcal{K}}(\cdot)\}$ with a non-negligible advantage and without the private key of the message M_1's reception party. Then we can construct an algorithm B to solve the truncated decisional q-ABDHE problem.

B takes as input a challenge (g', g'_{q+2}, g, g_1, …, g_q, Z), where Z is either $e(g_{q+1}, g')$ or a random element in G_T.

B simulates a PKG as follows. B generates a random polynomial $f(z) \in Z_p[z]$ of degree q. It sets $h = g^{f(\alpha)}$, computing h from (g, g_1, …, g_q). Other public parameters g_T and MAC are defined the same as those in the protocol specification. The public parameters are (g, g_1, h, g_T, MAC). There is no master-key belonging to B.

B generates user keys as follows. To generate a private key for identity $ID \in Z_p$, if $ID = \alpha$, B uses α to solve the truncated decisional q-ABDHE problem immediately. If $ID \neq \alpha$, let $F_{ID}(z)$ denote the $(q-1)$ degree polynomial $(f(z) - f(ID))/(z - ID)$. B computes (r_{ID}, h_{ID}) to be ($f(ID)$, $g^{F_{ID}(\alpha)}$). This is a valid private key for ID, since $g^{F_{ID}(\alpha)} = g^{(f(\alpha)-f(ID))/(\alpha-ID)} = (hg^{-f(ID)})^{1/(\alpha-ID)}$ as required. Note that if Corrupt queries are less than $(q-1)$ times, the generated private key has identical distribution as in a real protocol context because of the randomly selected $f(z)$.

Let $f_2(z) = z^{q+2}$ and let $F_{2,j}(z) = (f_2(z) - f_2(j))/(z-j)$, which is a polynomial of degree $q+1$. Then B generates $M_1^*||M_{21}^*||M_{22}^*$ by protocol simulation, where $M_1^*||M_{21}^*||M_{22}^*$ denotes a special bit string to feed algorithm D. Let $M_1^* = M_{11}^*||M_{12}^*$. $M_{11}^* = g'^{(f_2(\alpha)-f_2(j))x^*}$, $M_{12}^* = Z^{x^*} \cdot e(g', \prod_{l=0}^{q} g^{F_{2,j,l}\alpha^l})^{x^*}$. M_{21}^*, M_{22}^* and related MAC key KM_{ji}^* are calculated according to protocol specifications. Let $s^* = (log_g g')F_{2,j}(\alpha)$. Then if $z = e(g_{q+1}, g')$, $M_{11}^* = (g_1 g^{-j})^{s^* x^*}$, $M_{12}^* = g_T^{s^* x^*}$, which are the same as M_{11} and M_{12} in a real protocol run where participant j selected a random exponential value $s^* x^*$. If $z \neq e(g_{q+1}, g')$, M_1^* is not a valid protocol message.

Now B takes the simulated message to play a game with D. The game is that a fair coin is made by B and then the simulated message $M_1^*||M_{21}^*||M_{22}^*$ or a special message $M_1||M_{21}||MAC_{K\leftarrow\mathcal{K}}(\cdot)$ in $\{M_1||M_{21}||MAC_{K\leftarrow\mathcal{K}}(\cdot)\}$ is given to D according to the value of the fair coin. We use the symbol $M_1^b||M_{21}^b||M_{22}^b$ to denote D's input. Note that B can feed D its input in an interactive way. For example, B simulates all participants, runs all oracles according to protocol specifications except oracles $\Pi_{i,j}^s$ and $\Pi_{j,i}^t$. D can corrupt any parties except j. When D sends λ to an oracle $\Pi_{i,j}^s$, M_1^b is responded. When M_1^b is firstly received by $\Pi_{j,i}^t$, $M_{21}^b||M_{22}^b$ is included in the response.

If $Z = e(g_{q+1}, g')$, the simulated message is a qualified real message as we stated previously. By our assumption, D should have a non-negligible advantage to win the game. However if $Z \neq e(g_{q+1}, g')$, D has no advantage, which will be argued shortly. By the advantage differences, B can solve the truncated decisional q-ABDHE problem.

We argue D's zero advantage when $Z \neq e(g_{q+1}, g')$ as follows. First of all, KM_{ji}^* in the simulated message is just a uniformly random and independent value from the viewpoint of D since the private key of party j, (r_j, h_j), is not disclosed to D, and the first part of the private key, r_j, is a uniformly random and independent value. So the MAC tag part M_{22}^* in the simulated message and $MAC_{K\leftarrow\mathcal{K}}(\cdot)$ in the special message are indistinguishable for D. We continue to say that it is also impossible for D to distinguish M_1^* in the simulated message from M_1 in the special message. If D can distinguish them, then D can distinguish a simulated message from a real protocol message, which can be used to distinguish the challenge directly. Note that whether the simulated message is a real protocol message depending on the value of Z.

To conclude phase 1.1, we say that if truncated decisional q-ABDHE problem is hard, the private key of M_1's reception party is not disclosed and the number of disclosed private keys is less than $(q-1)$, two distributions $\{M_1||M_{21}||M_{22}\}$ and $\{M_1||M_{21}||MAC_{K\leftarrow\mathcal{K}}(\cdot)\}$ are indistinguishable.

Phase 1.2: Suppose the probability of *No-Matching*$(\cdot)$ for an initiator is non-negligible. Then there is an adversary A who can make an oracle $\Pi_{i,j}^s$ accepted with a non-negligible probability while there is no matching oracle.

B is now a chosen message MAC attacker. B accesses a MAC oracle and obtains MAC tags from the oracle. B's task is to give out a qualified MAC tag which is not generated by the MAC oracle. B runs A by protocol simulation.

According to our protocol, B sets parameters and runs the protocol on behalf of all participants. B picks parties $\{i, j\}$ and a session s, guessing that A will succeed against initiator $\Pi_{i,j}^s$ oracle.

B answers all A's queries itself according to protocol specifications if party j is not related. Note that the Corrupt query about party j is not allowed. When the reception oracle of a message M_1 is a session of party j, B will use its MAC oracle to compute the MAC tag in message M_2 responding to M_1. When a message M_2 claims coming from j, and the intended reception oracle is not $\Pi^s_{i,j}$, B will recompute a tag using its MAC oracle to continue play the game with A. While the intended reception oracle is indeed $\Pi^s_{i,j}$, B will take the responding M_2 as a valid forgery.

Let's analyze B's advantage. First of all, A cannot distinguish whether B's MAC oracle is used because of the conclusion of phase 1.1. So If B's guessing is correct with a probability $1/|I|^2|N_1|$, A should have non-negligible advantage to make oracle $\Pi^s_{i,j}$ accepted while there is no matching oracle. According to protocol specification, accepted oracle $\Pi^s_{i,j}$ means that the MAC tag in M_2 is the same as the MAC tag computed locally by oracle $\Pi^s_{i,j}$. In the simulation scenario, it means the MAC tag should be a valid one. So if the $M_1||M_{21}$ has never been queried to B's MAC oracle, B can success with a non-negligible probability.

If the $M_1||M_{21}$ has been queried, B must do it on behalf of an initiator oracle $\Pi^{s'}_{i,j}$ or a responder oracle $\Pi^{t'}_{j,i}$, where i is determined by the ID in the exponent part of M_{21}. The initiator oracle should be independent of oracle $\Pi^s_{i,j}$, that is, $s' \neq s$. However, since a random value x is used to generate M_1, the event is negligible that two independent oracles select one same value $x \in Z_p$. The responder oracle $\Pi^{t'}_{j,i}$ must have received M_1 before the oracle needs a MAC operation. It is negligible for the oracle $\Pi^{t'}_{j,i}$ to receive M_1 before $\Pi^s_{i,j}$ really produced it since the random value x is embedded in message M_1. However, if oracle $\Pi^{t'}_{j,i}$ received message M_1 after $\Pi^s_{i,j}$ produced it, the oracle $\Pi^{t'}_{j,i}$ has a matching conversation to $\Pi^s_{i,j}$. To conclude, the probability that $M_1||M_{21}$ has been queried is negligible.

To conclude phase 1.2, we give a more concrete expression to show B's advantage. While the advantage of A is ϵ, the advantage of B is $\frac{\epsilon}{|I|^2|N_1|} - \epsilon_1$, where ϵ_1 is the probability that message $M_1||M_{21}$ has been queried to B's MAC oracle. Since B's advantage should be negligible, it is clear the probability ϵ should be negligible.

Case 2: the probability of *No-Matching*$(\cdot)$ for a responder is negligible.

Again, there are two phases. The first phase is to conclude the indistinguishability of two distributions $\{M_1||M_2||M_3\}$ and $\{M_1||M_2||MAC_{K\leftarrow\mathcal{K}}(\cdot))\}$. The second phase is to reduce an adversary's advantage to a MAC forger's advantage.

Phase 2.1: It is similar with the proof in case 1. The adversary D now is limited not to obtain the private key of the message M_2's reception party. The simulator B simulates a PKG and generates user keys the same as it does in case 1. The number of disclosed private keys is limited to be less than $(q-1)$. B

generates $M_1^*||M_2^*||M_3^*$ by protocol simulation as follows. B firstly selects a party i as the message M_2 reception party. Then B generates message $M_{211}^*||M_{212}^*$, $M_{211}^* = g'^{(f_2(\alpha)-f_2(i))y}$, $M_{212}^* = Z^y \cdot e(g', \prod_{l=0}^{q} g^{F_{2,i,l}\alpha^l})^y$. M_1^*, M_{22}^*, and M_3^* are calculated according to the protocol specification.

B then plays a game as in case 1 with D except that the used simulated message is $M_1^*||M_2^*||M_3^*$. Again D has zero advantage when $Z \neq e(g_{q+1}, g')$. At last, if the truncated decisional q-ABDHE problem is hard, the private key of the message M_2's reception party is not disclosed, and the number of disclosed keys are less than $(q-1)$, two distributions $\{M_1||M_2||M_3\}$ and $\{M_1||M_2||MAC_{K\leftarrow\mathcal{K}}(\cdot)\}$ are indistinguishable.

Phase 2.2: Again an adversary A is assumed. A chosen message attacker B for a MAC algorithm is used. B now picks parties $\{j, i\}$ and a session t, guessing that A will succeed against a responder $\Pi_{j,i}^t$ oracle. B plays a game with A similar with what they done in case 1 except the replacement of identity j by identity i, M_1 in case 1 by M_2, M_2 in case 1 by M_3, $\Pi_{i,j}^s$ in case 1 by $\Pi_{j,i}^t$. The MAC verification for M_{22} in case 1 now doesn't need a MAC oracle. The same replacement can be used to analyze B's advantage to conclude that if the message $M_{21}||M_1$ has never been queried to B's MAC oracle, B can success with a non-negligible probability.

If the $M_{21}||M_1$ has been queried, B must do it on behalf of an initiator oracle $\Pi_{i,j}^{s'}$ where j is determined by the ID in the exponent part of message M_1 or a responder oracle $\Pi_{j,i}^{t'}$ with $t' \neq t$. Since a random value y is embedded in the M_{21} message, the probability is negligible of two independent responder oracles selecting one same value $y \in Z_q$. Note that $M_{21}||M_1$ should be generated by the responder oracle $\Pi_{j,i}^t$ for verification, where M_{21} is a locally stored message and M_1 is received by $\Pi_{j,i}^t$. If the initiator oracle $\Pi_{i,j}^{s'}$ has queried the same message $M_{21}||M_1$, the oracle should form this message by locally stored M_1 and received M_{21}. So the first flow generated by $\Pi_{i,j}^{s'}$ is received by $\Pi_{j,i}^t$ except a negligible probability that another initiator oracle selecting the same random value x in the exponent part of M_1. The M_{21} in the second flow generated by $\Pi_{j,i}^t$ is received by $\Pi_{i,j}^{s'}$ except a negligible probability that another responder oracle selecting the same random value y in the exponent part of M_{21}. While the initiator and responder agreed on the $M_{21}||M_1$, the MAC tag M_{22} generated by $\Pi_{j,i}^t$ should be the same as the verification MAC tag generated by $\Pi_{i,j}^{s'}$. So the M_2 is generated by $\Pi_{j,i}^t$ and received by $\Pi_{i,j}^{s'}$. Also the initiator and responder should have the same view on M_3 if they have the same view on $M_{21}||M_1$. So the initiator oracle $\Pi_{i,j}^{s'}$ and responder oracle $\Pi_{j,i}^t$ have a matching conversation, contradicting to the no-matching assumption.

To conclude phase 2.2, we also give an expression to show B's advantage. While the advantage of A is ϵ, the advantage of B is $\frac{\epsilon}{|I|^2|N_1|} - \epsilon_2$, where ϵ_2 is the probability that message $M_{21}||M_1$ has been queried to B's MAC oracle. Since B's advantage should be negligible, the probability ϵ should also be negligible.

At last, we conclude that the probability of *No-Matching*$(\cdot)$ for a responder (an initiator) is negligible if the private key of an initiator (a responder) is not Corrupted, the number of Corrupted keys is less than $q-1$, and the truncated decisional q-ABDHE problem is hard. □

Theorem 4.3. *The Adv is negligible.*

Proof. Let A be an adversary who has non-negligible *Adv* in the defined model. We construct an algorithm B to solve the truncated decisional q-ABDHE problem. B takes as input a random truncated decisional q-ABDHE challenge $(g', g'_{q+2}, g, g_1, \ldots, g_q, Z)$, where Z is either $e(g_{q+1}, g')$ or a random element in G_T.

B simulates a PKG as follows. B generates a random polynomial $f(z) \in Z_p[z]$ of degree q. It sets $h = g^{f(\alpha)}$, computing h from $(g, g_1, \ldots, g_q)$. Other public parameters g_T and MAC are defined the same as those in the protocol specification. The public parameters are (g, g_1, h, g_T, MAC). There is no master-key belonging to B.

B generates user keys as follows. To generate a private key for an identity $ID \in Z_p$, if $ID = \alpha$, B uses α to solve the truncated decisional problem immediately. If $ID \neq \alpha$, let $F_{ID}(z)$ denote the $(q-1)$ degree polynomial $(f(z) - f(ID))/(z - ID)$. B computes (r_{ID}, h_{ID}) to be $(f(ID), g^{F_{ID}(\alpha)})$. This is a valid private key for ID, since $g^{F_ID(\alpha)} = g^{(f(\alpha)-f(ID))/(\alpha-ID)} = (hg^{-f(ID)})^{1/(\alpha-ID)}$ as required.

B answers adversary queries as follows.

- *Send*$(\Pi^s_{i,j}, X)$. Firstly B guesses that the oracle $\Pi^s_{i,j}$ should be fresh and be tested. Generally, suppose that $\Pi^s_{i,j}$ is the initiator. Again let $f_2(z) = z^{q+2}$ and let $F_{2,j}(z) = (f_2(z) - f_2(j))/(z - j)$, which is a polynomial of degree $q + 1$. B then simulates the protocol for $\Pi^s_{i,j}$ according to the protocol specification except that:

  1. $M_{11} = g'^{(f_2(\alpha)-f_2(j))x}$ and $M_{12} = Z^x \cdot e(g', \prod_{l=0}^{q} g^{F_{2,j,l}\alpha^l})^x$;
  2. B finds out who received M_1 from $\Pi^s_{i,j}$ and who sent M_2 to $\Pi^s_{i,j}$. If B finds an oracle $\Pi^t_{j,i}$, B directly use KM_{JI} as KVM_{IJ} to compute the value $V_{M_{22}}$. If B decides to set oracle $\Pi^s_{i,j}$ as Accepted, B uses the temporal value y in $\Pi^t_{j,i}$ to compute $K_{ij} = KVM^y_{ij}$. Else B stops the game with a *Fail* output.

  For any other Send queries that are not related to the guessed oracle, B will act exactly according to the protocol specification.

- *Corrupt*(i). If $i \neq \alpha$, B gives the private key of i as response. Else if $i = \alpha$, B solves the truncated decisional problem.

- $Reveal(\Pi^s_{i,j})$. B gives the session key currently held by the oracle $\Pi^s_{i,j}$. Note before the oracle $\Pi^s_{i,j}$ is Accepted, the session key is λ.

- $Test(\Pi^s_{i,j})$. If B made a wrong guess, B stops the game with a *Fail* output. Else B flips a fair coin $b \leftarrow \{0,1\}$, and returns the session key holding by the oracle $\Pi^s_{i,j}$ if $b = 0$, or else a random key sampled from the key space if $b = 1$.

If the adversary really shows its advantage, B will guess $Z = e(g_{q+1}, g')$. If the adversary has no advantage at all, B will guess $Z \neq e(g_{q+1}, g')$.

Analysis

If B does not stop before the output event, the simulation is indistinguishable. Firstly, if the number of *Corrupt* queries is less than $(q-1)$, the generated private key has identical distribution as in a real protocol context because of the random selected $f(z)$. Secondly, the outputs of other queries are generated according to the protocol specification or the model rules if we don't consider the oracles $\Pi^s_{i,j}$ and $\Pi^t_{j,i}$. Thirdly, the adversary can not distinguish behaviors of oracles $\Pi^s_{i,j}$ and $\Pi^t_{j,i}$ in the simulation from behaviors of them in a real protocol context. Assume $s = (log_g g')F_{2,J}(\alpha)$, then related messages and values are compared in table 1.

It is clear that if $Z = e(g', g_{q+1})$, messages and related values are reasonably to be real protocol messages and real protocol values. However, if $Z \neq e(g', g_{q+1})$, the messages and values are not qualified to claim as protocol messages and values. However, if the adversary can distinguish the simulation from real protocol context, then the adversary can distinguish whether $Z = e(g_{q+1}, g')$, which contradicts the truncated decision q-ABDHE assumption.

Now we calculate the probability that B does not stop. First of all, if B has made a wrong guess, B stops. There are at most $|I|^2|N_1|$ oracles. So the probability of B's right guess is at least $1/|I|^2|N_1|$. Even if B have made a right guess, B may stops due to the lost of $\Pi^s_{i,j}$ oracle's matching oracle. However from theorem 2, we know that if j has not been Corrupted, the number of Corrupted keys is less than $q-1$, and q-ABDHE problem is hard, the probability of *No-Matching*$(\cdot)$ for $\Pi^s_{i,j}$ is negligible. Here we use ϵ_3 to denote the negligible probability.

In general, B will have a probability $\frac{1-\epsilon_3}{|I|^2|N_1|}$ to justify adversary A's advantage. When $Z = e(g', g_{q+1})$, the adversary should show its advantage to B. When $Z \neq e(g', g_{q+1})$, the session key $K_{i,j}$ is $e(g,h)^{x_{snz}y}(Z/e(g', g_{q+1}))^{\frac{r_j x_{snz} y}{s}}$ or just a random value in G_T. Since r_j is not disclosed to the adversary, the value $K_{i,j}$ is just a random value from the view of the adversary. So there is no advantage for the adversary at all. Now B's strategy works. However, B should have no advantage so that A's advantage should be zero too. $\square$

	M_{11}	M_{12}	KM_{JI} or KVM_{IJ}
Real	$g^{(\alpha-J)x_r}$	$e(g,g)^{x_r}$	$e(g,h)^{x_r}$
Simu& $Z = e(g', g_{q+1})$	$g^{(\alpha-J)x_{sz}}$	$e(g,g)^{x_{sz}}$	$e(g,h)^{x_{sz}}$
Simu& $Z \neq e(g', g_{q+1})$	$g^{(\alpha-J)x_{snz}}$	$e(g,g)^{x_{snz}}(Z/e(g', g_{q+1}))^{\frac{x_{snz}}{s}}$	$e(g,h)^{x_{snz}}(Z/e(g', g_{q+1}))^{\frac{r_J x_{snz}}{s}}$

Table 1 Messages and values in different scenarios

4.3 Performance

First of all, let's show our protocol performance and the reason of the performance. In our protocol, the computation load for an initiator is the same as the load for a responder. The computation for one oracle includes 4 times exponentiation operations, 2 times MAC operations, and 2 times pair operations. We think that the computation load is a cost to obtain a standard proof. In fact, our protocol is as practical as Gentry's encryption scheme. Provided a more efficient ID-based encryption scheme with standard proof, it is easy to give a more efficient protocol with the same protocol design and proof method.

As we have said that we failed to find some direct related works, we found no *explicit* authentication protocols with a stand proof in the ID-based field to be compared with ours. We note that the protocol in [24] has similar goals with ours but in a traditional field. Thanks to many advantages of ID-based cryptography, such as no need for certificates etc, our protocol has some advantages in application over the protocol in [24] but not in efficiency.

4.4 Security Properties

We consider the following common security properties.

- *Known session keys.* The Reveal query is designed to capture the notion. The fresh condition has never restricted adversary's Reveal ability to any oracles except the tested oracle and its possible matching oracle.

- *Unknown key share.* Suppose that a $\Pi^s_{i,e}$ oracle and a $\Pi^t_{j,i}$ oracle holding the same session key. An adversary could simply reveal the key held by $\Pi^s_{i,e}$, and pick $\Pi^t_{j,i}$ as the tested oracle. In this way, the adversary defeat the secrecy goal in the model.

- *Impersonation attack resistance.* If a $\Pi^s_{i,e}$ oracle accepted, the authentication goal assures there is only one matching oracle $\Pi^t_{e,i}$ becasue the probabilities of *No-Matching* event and *Multi-matching*[17] event are all negligible. So an impersonation attack can appear only with a negligible probability.

- *Key compromise impersonation resilience.* Note that the authentication goal is proved while only one party's private is limited not to be Corrupted. So even one player's private key is Corrupted, nobody can cheat the player to be accepted with a impersonated ID.

- *Perfect forward secrecy.* Here we just informally claim that our protocol enjoy the property. The session key in our protocol is just related to two temporal random values in Z_p. The session key has no relation to long term keys. So even long term keys are Corrupted, it just means that MAC keys can be obtained. Even an adversary knows $e(g,h)^x$ and $e(g,h)^y$, there is still a computation hard problem to obtain $e(g,h)^{xy}$.

- *Session State Reveal.* The property considers what happened when temporal state values are revealed. Apparently, the leakage of temporal value x or y in a session means that an adversary can impersonate a responder or an initiator in that session. If all temporal values are revealed, the session key will be disclosed. However the bad result is limited to this session only. One session with a new temporal value will not be affected by the leakage of temporal values in another session.

5 Conclusion

We proposed an ID-based protocol and a standard proof. The protocol employs a new method to isolate session keys from key confirmation keys. Due to the method, there is no direct usage of hash functions in the protocol and there is no random oracles in the proof procedure.

References

[1] A.J. Menezes, P.C. van Oorschot, S.A. Vanstone. Handbook of Applied Cryptography. CRC Press, 1997.

[2] Shamir. Identity-based cryptosystems and signatures schemes. In *G.T. Blakey and D. Chaum, editors, Advanced in Cryptography – Proceedings of Crypto'84*, LNCS 196, pp. 48–53. Spring-Verlag, 1985.

[3] E. Okamoto. Proposal for identity-based key distribution system. *Electronics Letters*, Vol.22, pp. 1283–1284. 1986.

[4] M. Girault and J. Paillés. An identity-based scheme providing zero-knowledge authentication and authenticated key exchange. In *Proceedings of ESORICS 90*, pages 173–184. 1990.

[5] K. Tanaka and E. Okamoto. Key distribution system for mail systems using ID-related information directory. *Computers and Security*, Vol.10, pp. 25–33. 1991.

[6] A. Joux. A one round protocol for tripartite Diffie-Hellman. In *proceedings of Algorithmic number theory symposium, ANTS-IV*, LNCS 1838, pp. 385–394. 2000.

[7] R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems based on pairing. In *proceedings of 2000 symposium on Cryptography and Information Security*, SCIS 2000.

[8] N. P. Smart. Identity-based authenticated key agreement protocol based on Weil pairing. *Electronics Letters* Vol.38, No.13, pp. 630–632. 2002.

[9] L. Chen and C. Kudla. Identity based authenticated key agreement protocols from pairing. In *proceedings of 16th IEEE Security Foundations Workshop*, pp. 219–233. IEEE Computer Society Press, 2003.

[10] M. Scott. Authenticated ID-based key exchange and remote log-in with insecure token and PIN number. *Cryptography ePrint Archive*, 2002/164, 2002.

[11] K. Shim. Efficient ID-based authenticated key agreement protocol based on the Weil pairing. *Electronics Letters*. Vol.39, No.8, pp. 653–654. 2003.

[12] P. McCullagh and P. Barreto. A new two-party identity-based authenticated key agreement. In *Proceedings of CT-RSA 2005*, LNCS 3376, pp. 262–274. Springer-Verlag, 2005.

[13] K. R. Choo, C. Boyd, and Y. Hitchcock. On Session Key Construction in Provably-Secure Key Establishment Protocols. *First International Conference on Cryptology in Malaysia – Mycrypt 2005*, LNCS 3715, pp. 116–131. Springer-Verlag. 2005.

[14] Y. Wang. Efficient Identity-Based and Authenticated Key Agreement Protocol. *Cryptography ePrint Archive*, 2005/108, 2005.

[15] Z. Cheng, L. Chen, R. Comley, and T. Tang. Identity-Based Key Agreement with Unilateral Identity Privacy Using Pairings. In *2nd Information Security Practice and Experience Conference – ISPEC 2006*, LNCS 3903. Springer-Verlag, 2006.

[16] K. Y. Choi, J. Y. Hwang, D. H. Lee, and I. S. Seo. ID-based Authenticated Key Agreement for Low-Power Mobile Devices. In *Tenth Australasian Conference on Information Security and Privacy – ACISP 2005*, LNCS 2005, pp. 494–505. Springer-Verlag, 2005.

[17] M. Bellare, and P. Rogaway. Entity Authentication and Key Distribution. In *Advances in Cryptology – Crypto 1993*, LNCS 773, pp. 110–125. Springer-Verlag, 1994.

[18] M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated Key Exchange Secure Against Dictionary Attacks. In *Advances in Cryptology – Eurocrypt 2000*, LNCS 1807, pp. 139–155. Springer-Verlag, 2000.

[19] E. Bresson, O. Chevassut, and D. Pointcheval. Provably Authenticated Group Diffie–Hellman Key Exchange–The Dynamic Case. In *Advances in Cryptology – Asiacrypt 2001*, LNCS 2248, pp. 209–223. Springer-Verlag, 2001.

[20] R. Canetti, and H. Krawczyk. Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. In *Advances in Cryptology – Eurocrypt 2001*, LNCS 2045, pp. 453–474. Springer-Verlag, 2001.

[21] R. Canetti. Universally composable security: a new paradigm for cryptographic protocols. Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on 8-11 Oct. 2001 Page(s):136 - 145.

[22] S. Blake-Wilson, D. Johnson, and A. Menezes. Key agreement protocols and their security analysis. *Proceedings of the sixth IMA International Conference on Cryptography and Coding*, LNCS 1355, pp. 30–45. Springer-Verlag, 1997.

[23] Z. Cheng, M. Nistazakis, R. Comley, and L. Vasiu. On The Indistinguishability-Based Security Model of Key Agreement Protocols-Simple Cases. *Cryptography ePrint Archive*, 2005/129, 2005.

[24] I.R. Jeong, J.O. Kwon, and D.H. Lee. A Diffie-Hellman Key Exchange Protocol Without Random Oracles. In *D.Pointcheval, Y. Mu, and K. Chen editors, CANS 2006*, LNCS 4301, pp.37-54. Springer-Verlag, 2006.

[25] R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. In *proceedings of the 30th Annual Symposium on the Theory of Computing (STOC'98)*, pages 209–218. ACM Press, 1998.

[26] M. Bellare, A. Boldyreva, and A. Palacio. A uninstantiable random-oracle-model scheme for a hybrid-encryption problem. In *C. Cachin and J. Camenisch, editor, Advance in Cryptology – Proceedings of EUROCRYPT2004*, Lecture Notes in Computer Science 3027, pages 171–188. Springer-Verlag, 2004.

[27] N. Koblitz. Another Look at "Provable Security". *Journal of Cryptography.* Vol 20, No. 1, pp. 3-37. 2007.

[28] W. Mao. Modern cryptography: theory and practice. Prentice-Hall PTR. 2003.

[29] C. Gentry. Practical Identity-Based Encryption Without Random Oracles. In S. Vaudenay, editor, *proceedings of EUROCRYPT 2006*, LNCS 4004, pp. 445-464. Springer-Verlag, 2006.

[30] D. Boneh, X. Boyen, and E.-J. Goh. Hierarchical Identity Based Encryption with Constant Size Ciphertext. In *Advances in Cryptology – Eurocrypt 2005*, LNCS 3494, pages 440–456. Springer-Verlag, 2005.

[31] D. Boneh, C. Gentry, and B. Waters. Collusion-Resistant Broadcast Encryption with Short Ciphertexts and Private Keys. In *Advances in Cryptology – Crypto 2005*, LNCS 3621, pages 258–275. Springer-Verlag, 2005.

[32] T. Iwata and K. Kurosawa. OMAC: One-Key CBC MAC. In *T. Johansson editor, FSE 2003*, LNCS 2887, pp. 129-153. Springer-Verlag, 2003.