ID-based Tripartite Multiple Key Agreement Protocol Facilitating Computer Auditing and Transaction Refereeing

Hung-Yu Chien
Department of Information Management National Chi Nan University, Taiwan, R.O.C.

Abstract

Computer auditing and transaction refereeing require the system to keep genuine records. However, it is difficult for an auditor or a referee to on-line audit the contents or involve in the communication while the communication is kept confidential from others. This problem has a promising solution, when Joux proposed the first efficient tripartite key agreement protocol that enables three parties (that might include one referee) to establish a secure session key. However, several published schemes are not secure. This paper examines the weaknesses, and then proposes an ID-based tripartite multiple key agreement protocol to raise the level of security and improve the efficiency. The security is proved in a modified Bellare-Pointcheval-Rogaway model.

Key words: security, auditing, key agreement, bilinear pairing, elliptic curve.

便於線上稽核及交易仲裁之身分基礎式三方金鑰協定

簡宏宇
國立暨南大學資管系

摘要

為了便於電腦稽核或交易仲裁，系統需要保存真實資料。然而，通常通訊雙方會將通訊內容予以加密以防止不法份子之竊聽或攻擊；如此將造成稽核者或仲裁者無法線上即時予以監控或直接介入通訊。此一惱人的問題自從 Joux 設計出第一個高效率之三方式金鑰協定後，將得以有效的解決。一個三方式金鑰協定允許通訊的三方可以高效率的進行金鑰協議進而對彼此通訊做安全之加密，如此仲裁者或稽核者便可以安全地加入通訊雙方。然而，現今已發表之三方金鑰協定仍存在著諸多的安全弱點。此篇論文將檢視一些近日發表論文之弱點，並提出一新的機制以改善安全功能及效率。我們將在修訂之 Bellare-Pointcheval-Rogaway 模型中證明此協定的安全。

關鍵字：安全、稽核、金鑰協議、雙線性配對、橢圓曲線。

1. INTRODUCTION

To audit computer data or referee a transaction, it is required that the system could keep the genuine data and record the communication content. While in the insecure Internet, the data and the communication content are usually encrypted to prevent an adversary from hijacking or modifying the contents. Conventionally, the communication contents are encrypted using the shared key between the two communicating parties. This makes an auditor or a referee difficult to monitor or involve the secret communication. Now the dilemma has a promising solution, since Joux [5] proposed the first efficient tripartite key agreement protocol, where three parties can efficiently establish their session key. Since then, the tripartite key agreement protocol from pairing has drawn the attention of many researchers recently [1-6], not only because the three-party (or tripartite) case is the most common size for electronic conferences but also because it can be used to provide a range of services for two parties communicating. For example, a third party can be added to chair, or referee a conversation for auditing, or data recovery purposes. However, Joux's scheme [5] cannot resist the man-in-the-middle attack [4]. Al-Riyma-Paterson's certificate-based protocols [4] consume a large amount of computing, communication and storage to access and verify the public keys. Further, Shim [10] analyzed the insecurity of Nalla-Reddy's scheme [6], and Cheng et al. [9] pointed out the weaknesses of Al-Riyma-Paterson's scheme. Sun and Hsieh [7] pointed out the key compromise impersonation attack on Shim's scheme.

The possible attacks and desirable properties of the tripartite key agreement protocol consists of replay attack, impersonation attack, known key attack, man-in-the-middle attack, insider attack, key-compromise impersonation attack, unknown key share attack, forward secrecy, and TA-forward secrecy [1-6]. A protocol is vulnerable to the key-compromise impersonation attack if the attacker who has compromised the private key of one entity not only can impersonate the compromised entity but also other entities. *Forward secrecy* means that **c**ompromise of the long-term secret keys of one entity or more entities at some point in the future does not lead to the compromise of the communications in the past. This can be further classified as *partial forward secrecy* and *perfect forward secrecy*, where *partial forward secrecy* is the situation that only one entity's long-term key (or more entities but not all entities) is compromised and *perfect forward secrecy* is under the situation that all the involved entities' long-term keys (but not the *Trusted Authority*) are compromised. Under the In ID-based systems, each user's private key is known and issued by *Trusted Authorities* (*TA*); therefore, compromise of TA's private key leads to compromise of all the private keys issued by TAs, too. TA forward secrecy requires that

even if the TA's private key is compromised in the future does not lead to compromise of the communications in the past.

The insider attack on a tripartite key agreement protocol means that one of the three entities try to impersonate another one of the three entities. For example, B might try to fool A that they and C are participating in a protocol run, while in fact C does not. This attack could have serious consequences: for example, if C acts as an on-line escrow agent, an auditor or a referee [5]. If B could impersonate C to A, then B can communicate with or performing transactions with A; while A would do the transactions or communications only if C (the referee) is monitoring the contents on-line. With no referee involved, this might cause serious risk for A.

In this paper, we show the insider attack on Zhang-Liu-Kim's ID-based scheme (the ZLK scheme for short) [1, 3] and Shim's certificate-based scheme [2], and propose a new tripartite ID-based scheme that improves both security and efficiency. Our scheme is based on the elliptic curves and the bilinear pairings. And, its security is proved secure in terms of in-distinguishability [11] and resistance to the insider attack in a modified Bellare-Pointcheval-Rogaway model [12].

Related works

Although recent progress has been made on the use of formal models to prove the security of key exchange protocols, the area where formal model to prove the security of tripartite case is required further work. Especially, there is no formal model that caters for the insider attack and the key-compromise impersonation attack in the tripartite case, to our best knowledge. Therefore, to prove the security against the insider attack and the key-compromise impersonation attack in the tripartite case, we need to modify the existing model.

Ai-Riyama and Paterson [4] had proved the security of one of their protocols in a formal model; however, the model only allows passive adversary. The Canetti-Krawczyk proof model [13] allows different components to be proved secure separately, and then joins together to provide a secure key exchange. A protocol is first proved secure in their AM model, and then authenticators are applied on the protocol to derive a secure protocol in the UM model. This approach leads to a simpler, less error-prone proofs and the ability to construct a large number of secure protocols from a small much smaller number of basic secure components. Hichcock et al. [14] had modified the model for the tripartite case and had designed secure tripartite protocols. However, the insider attacks and the key-compromise impersonation attack are not considered in the proof since each entity always follows the protocols and a session with a corrupted entity is not considered as fresh for testing. Furthermore, the modular approach most of the time results in in-efficient

protocols, and heuristic optimization techniques are applied on the protocols to improve the performance. But, these optimization processes are not proved. In 1995, Bellare and Rogaway [11] analyzed a three-party server-based key distribution (3PKD) protocol, which was referred to as BR 95 model [15]. The most recent revision of the BR95 model is the BPR2000 model [12]. The partnership concept is used to capture the matching session among oracles. Partnership in the BR95 model is defined using the notation of a partner function, while that in the BPR2000 model is defined using the notation of session identifier. Choo et al. [15] combined the BR95 model and the BPR2000 model to design a secure 3PKD protocol. However, like the Canetti-Krawczyk model, every participating entity (not the adversary) is assumed to be honest and the insider attack is not considered. We, therefore, modify the BPR2000 model to prove the security of our protocol. Our protocol is secure in terms of in-distingiusihability, resistance to key-compromise impersonation attack and resistance to the insider attack relative to the Decisional Bilinear Diffie-Hellman assumption (DBDH) and the secure Hess signature scheme.

The rest of this paper is organized as follows. Section 2 introduces the preliminaries of bilinear pairings and the BDH/DBDH assumptions. Section 3 examines the weaknesses of the ZLK scheme, while Section 4 further points out the insider attack on Shim's scheme. Section 5 proposes our scheme. Section 6 proves the security. Section 7 analyzes its performance, which is followed by our conclusions in Section 8.

2. BILINEAR PAIRINGS AND THE BDH ASSUMPTION

Let G_1 be a cyclic additive group generated by P and G_2 be a cyclic multiplicative group. Both G_1 and G_2 have a prime order q. The discrete logarithm problems (DLP) in both G_1 and G_2 are assumed to be hard. Let $e: G_1 \text{x} G_1 \rightarrow G_2$ be a bilinear pairing [5] that has the following properties:

1. Bilinear: $e(P_1+P_2,Q)=e(P_1,Q)\cdot e(P_2,Q)$ and $e(P,Q_1+Q_2)=e(P,Q_1)\cdot e(P,Q_2)$;
2. Non-degenerate: There exists $P\in G_1$ such that $e(P,P)\neq 1$;
3. Computable: There exist efficient algorithms to compute $e(P,Q)$ for all $P,Q\in G_1$.

Bilinear Diffie-Hellman Problem (BDHP) for a bilinear pairing $e: G_1 \text{x} G_1 \rightarrow G_2$ is defined as follows: Given $P, aP, bP, cP \in G_1$, compute $e(P,P)^{abc}$, where a,b,c are random numbers from Z_q^*. It is commonly believed that the BDHP problem is hard.

Decision Bilinear Diffie-Hellman assumption (DBDH): Let $e: G_1 \text{x} G_1 \rightarrow G_2$ be a bilinear pairing and q be the order of G_1 and G_2. Let two probability distributions of tuples of seven elements, Q_0 and Q_1, be defined as :

$Q_0 = \{\langle G_1, G_2, P, aP, bP, cP, e(P,P)^{abc} \rangle : a,b,c \in_R Z_q\}$
$Q_1 = \{\langle G_1, G_2, P, aP, bP, cP, e(P,P)^{d} \rangle : a,b,c,d \in_R Z_q\}$.

Then the DBDH assumption states that Q_0 and Q_1 are computationally indistinguishable.

3. WEAKNESSES of THE ZLK KEY AGREEMENT SCHEME

The ZLK scheme has two versions: one is for sharing single key and the other is for sharing eight keys among the three entities. Here, we review the multiple key case, because one can easily derive the single key case from the multiple key version. The ZLK scheme [1, 3] involves a Trusted Authority (TA) and users. The TA is responsible for the set-up operation and the private key extraction operation.

Setup: TA sets up an additive group G_1 of prime order q and a cyclic multiplicative group G_2 of the same order q. Let P be a generator of G_1, and e: $G_1 \text{x} G_1 \rightarrow G_2$ a bilinear mapping satisfying the conditions in Section 2. Define two cryptographic hash functions $H: \{0,1\}^* \rightarrow Z_q$ and $H_1: \{0,1\}^* \rightarrow G_1$. TA owns the system's private key $s \in Z_q^*$ and the system's public key $P_{pub} = sP$. TA publishes $\{G_1, G_2, P, q, P_{pub}, H, H_1\}$.

Private key extraction: Let $A, B \text{ and } C$ be the three entities running the protocol. A has his public key/private key as $Q_A = H_1(ID_A) / S_A = sQ_A$, B has his public key/private key as $Q_B = H_1(ID_B)$ / $S_B = sQ_B$, and C has his public key/private key as $Q_C = H_1(ID_C) / S_C = sQ_C$.

The ZLK protocol: *A, B*, and *C* run the protocol as follows.

1. $A \rightarrow B, C$: $P_A = aP, P_A' = a'P, T_A = H(P_A, P_A')S_A + aP_A'$, where a and a' are two random numbers chosen by A.
2. $B \rightarrow A, C$: $P_B = bP, P_B' = b'P, T_B = H(P_B, P_B')S_B + bP_B'$, where b and b' are two random numbers chosen by B.
3. $C \rightarrow B, A$: $P_C = cP, P_C' = c'P, T_C = H(P_C, P_C')S_C + cP_C'$, where c and c' are two random numbers chosen by C.

Now A verifies the received data by checking whether Equation (1) holds. If so, he/she computes the eight session keys using Equation (2). Similarly, B / C verifies Equation (3)/(5) and computes session keys using Equation (4)/(6), respectively.

$$e(T_B + T_C, P) = e(H(P_B, P_B')Q_B + H(P_C, P_C')Q_C, P_{pub})e(P_B, P_B')e(P_C, P_C') \quad (1)$$

$$\begin{aligned} &K_A^1 = e(P_B, P_c)^a, K_A^2 = e(P_B, P_C')^a, K_A^3 = e(P_B', P_C)^a, K_A^4 = e(P_B', P_C')^a \\ &K_A^5 = e(P_B, P_C)^{a'}, K_A^6 = e(P_B, P_C')^{a'}, K_A^7 = e(P_B', P_C)^{a'}, K_A^8 = e(P_B', P_C')^{a'} \end{aligned} \quad (2)$$

$$e(T_A + T_C, P) = e(H(P_A, P_A')Q_A + H(P_C, P_C')Q_C, P_{pub})e(P_A, P_A')e(P_C, P_C') \quad (3)$$

$$\begin{aligned} &K_B^1 = e(P_A, P_C)^b, K_B^2 = e(P_A, P_C')^b, K_B^3 = e(P_A', P_C)^b, K_B^4 = e(P_A', P_C')^b \\ &K_B^5 = e(P_A, P_C)^{b'}, K_B^6 = e(P_A, P_C')^{b'}, K_B^7 = e(P_A', P_C)^{b'}, K_B^8 = e(P_A', P_C')^{b'} \end{aligned} \quad (4)$$

$$e(T_B + T_A, P) = e(H(P_B, P_B')Q_B + H(P_A, P_A')Q_A, P_{pub})e(P_B, P_B')e(P_A, P_A') \quad (5)$$

$$\begin{aligned} &K_C^1 = e(P_B, P_A)^c, K_C^2 = e(P_B, P_A')^c, K_C^3 = e(P_B', P_A)^c, K_C^4 = e(P_B', P_A')^c \\ &K_C^5 = e(P_B, P_A)^{c'}, K_C^6 = e(P_B, P_A')^{c'}, K_C^7 = e(P_B', P_A)^{c'}, K_C^8 = e(P_B', P_A')^{c'} \end{aligned} \quad (6)$$

Security weaknesses of the ZLK multiple key agreement protocol

- **Insider attack**: An insider, say *B*, might just replay *C*'s messages to impersonate *C* and shares session keys with *A*, if *A* does not log and check the past records. To log and check the past records is not practical. However, even if the protocol really checks the past records, an insider can still launch an insider attack as follows. Assume A has sent his correct message as in the above. Now B, the inside attacker, tries to fool A that C participates in the protocol run as follows.

1. B randomly chooses $c, k, w \in Z_q^*$, and computes $P_C = kP_{pub}$, $P_C' = cP$, $P_B = Q_C$ and $P_B' = -H(P_C, P_C')P_{pub}$.
2. B computes $T_B = H(P_B, P_B')S_B + (kc - w)P_{pub}$ and $T_C = wP_{pub}$.
3. B broadcasts $\{P_B, P_B', T_B\}$, and sends $\{P_C, P_C', T_C\}$ as C's message.

After authenticating the broadcast messages using Equation (1), A will accept the messages and will compute the session keys. We show this result as follows.

Theorem 1. A will accept B's forged messages (in the above plotted attack) after *A* checking the verification equation (1).

Proof: To verify the authenticity, A will check Equation (1) as follows.

$$
\begin{aligned}
&e(H(P_B,P_B')Q_B + H(P_C,P_C')Q_C, P_{pub}) \cdot e(P_B,P_B') \cdot e(P_C,P_C') \\
&= e(H(P_B,P_B')Q_B + H(P_C,P_C')Q_C, P_{pub}) \cdot e(Q_C, -H(P_C,P_C')P_{pub})) \cdot e(kP_{pub}, cP) \\
&= e(H(P_B,P_B')Q_B + H(P_C,P_C')Q_C, P_{pub}) \cdot e(-H(P_C,P_C')Q_C, P_{pub}) \cdot e((kc)P, P_{pub}) \\
&= e(H(P_B,P_B')Q_B + H(P_C,P_C')Q_C - H(P_C,P_C')Q_C + (kc)P, P_{pub}) \\
&= e(H(P_B,P_B')Q_B + (kc)P, P_{pub}) = e(H(P_B,P_{B'})Q_B + (kc-w)P + wP, P_{pub}) \\
&= e(H(P_B,P_{B'})S_B + (kc-w)P_{pub} + wP_{pub}, P) \\
&= e(T_B + T_C, P)
\end{aligned}
$$

So, Equation (1) holds and A believes that he has authenticated the messages. Furthermore, the insider attacker B can share four common session keys with A. From Equation (2) and (4), B can computes $e(P_B,P_A)^c = e(P_B,P_C')^a = K_A^2$, $e(P_B',P_A)^c = e(P_B',P_C')^a = K_A^4$, $e(P_B,P_A')^c = e(P_B,P_C')^{a'} = K_A^6$, and $e(P_B',P_A')^c = e(P_B',P_C')^{a'} = K_A^8$. That is, the insider B fools A into accepting the forged messages and sharing four session keys with him, even though C does not involve in the communication.

- **Replay attack**: The verification equations do not check the freshness of the received messages. An attacker can, therefore, resend old messages of one entity (for example, the referee) to fool the communicating party that the entity (the referee) is actively participating the communication, even though he cannot derive the new session keys. This attack might result in serious problems. We give an example. An entity, say *A*, would communicate with *B* only if *C* (the referee) is actively involved in the communication. *B* might replay *C*'s messages to fool *A*.

4. WEAKNESSES OF THE SHIM SCHEME

Even though Sun and Hsieh [7] have pointed out the key-compromise impersonation attack on Shim's scheme, the insider attack has not been noticed. Shim's scheme is certificated-based, where $Y_A = aP$, $Y_B = bP$, and $Y_C = cP$ are the public keys of *A, B,* and *C*, respectively. $Cert_A$ $Cert_B$, and $Cert_C$ are the corresponding certificates. *A, B, and C* broadcasts the data in (7), and computes $K_A = e(T_B,T_C)^{axe(Y_B,Y_C)^a} = e(P,P)^{abcxyze(P,P)^{abc}}$, $K_B = e(T_A,T_C)^{bye(Y_A,Y_C)^b} = e(P,P)^{abcxyze(P,P)^{abc}}$, and $K_C = e(T_B,T_A)^{cze(Y_B,Y_A)^c} = e(P,P)^{abcxyze(P,P)^{abc}}$, respectively.

$$
\begin{cases}
A \to B, C: \ T_A = xY_A, Cert_A; \\
B \to A, C: \ T_B = yY_B, Cert_B \\
C \to B, A: \ T_C = zY_C, Cert_C
\end{cases} \tag{7}
$$

- **Key-compromise impersonation attack:** Sun and Hsieh [7] have showed the key-compromise impersonation attack on Shim's scheme.

- **Insider attack**: An insider, say A, now tries to masquerade as C to B as follows. The symbol $C(A)$ denotes that A masquerades as C. After broadcasting the data in (8), A and B share the session key $K_A = e(T_B, T_C)^{axe(Y_B, Y_C)^a} = e(P,P)^{abaxyze(P,P)^{abc}}$ and $K_B = e(T_A, T_C)^{bye(Y_A, Y_C)^b} = e(P,P)^{abaxyze(P,P)^{abc}}$, where $K_A = K_B$. The insider attack succeeds. Please notice that B wrongly believes that C is involved in the communication.

$$\begin{cases} A \to B, C: \ T_A = xY_A, Cert_A; \\ B \to A, C: \ T_B = yY_B, Cert_B \\ C(E) \to B, A: \ \boxed{T_C = zY_A, Cert_C} \end{cases} \quad (8)$$

- **Replay attack:** The Shim scheme does not check the freshness of the broadcast data. It is, therefore, vulnerable to the replay attack as discussed in Section 3.

5. THE IMPROVED SCHEME

Based on Hess's ID-based signature scheme [8] which was proved secure against adaptively chosen messages attacks relative to the Diffie-Helleman problem in the random model, we proposes an ID-based tripartite authenticated multiple key agreement protocol. We additionally define the following hash functions, and introduce Hess's ID-based signature scheme as follows.

$h_1()$: $\{0, 1\}^* \times G_2 \to Z_q^*$; $H_1()$: $\{0, 1\}^* \to Z_q^*$.

Hess's ID-based signature scheme

Assume A be the signer. To sign a message m, A randomly chooses an integer k, computes $r = e(P,P)^k$, $v = h_1(m,r)$ and $u = vS_A + kP$. Then, the signature is (m, v, u). To verify the signature, the verifier checks whether the equation $v \stackrel{?}{=} h_1(m, e(u,P)e(Q_A, -P_{pub})^v)$ holds. If so, he accepts the signature. The correctness of the equation can be checked as follows. $h_1(m, e(u,P)e(Q_A, -P_{pub})^v) = h_1(m, e(vS_A + kP, P)e(-vS_A, P)) = h_1(m, e(kP,P)) = h_1(m,r) = v$. Please notice that, for fixed signers, the values $e(P,P)$ and $e(Q_A, -P_{pub})$ can be pre-computed, and then the computational cost for signature and verification is 2 exponentiation operations, 2 scalar multiplications and 1 pairing.

Improved ID-based tripartite multiple key agreement scheme

The proposed scheme adopts the same set-up and private key extraction as the ZLK scheme, but the communication messages and the verification process are modified as follows. In the communication, *A, B*, and *C* should include the session identifier (which is defined as that in the BPR2000 model), the entities' identities, ephemeral public keys in the broadcast and in the calculation of their signatures. The inclusion of the entities' identities, ephemeral public keys and the session identifier is to prevent the insider attack among parallel sessions of the same group. In the following, *sid* denotes the session identifier of the current session. There are two rounds where the entities broadcast their ephemeral public keys the first run and the entities broadcast their confirmation (signatures) on the session and ephemeral public keys in the second round.

1.1. $A \rightarrow B, C: sid, A, B, C, P_A, P_A'$

A computes $P_A = aP, P_A' = a'P$, where k_A, a and a' are random numbers chosen by A. A sends (sid, A, B, C, P_A, P_A') to B and C.

1.2. $B \rightarrow A, C: sid, B, C, A, P_B, P_B'$

B computes $P_B = bP, P_B' = b'P$, where k_B, b and b' are random numbers.

1.3. $C \rightarrow B, A: sid, C, A,$ B, P_C, P_C'

C computes $P_C = cP, P_C' = c'P'$, where k_C, c and c' are random numbers.

2.1. $A \rightarrow B, C: sid, v_A, u_A$

A computes $m_A = H_1(sid, A, B, C, P_A, P_A', P_B, P_B', P_C, P_C')$, $r_A = e(P,P)^{k_A}$, $v_A = h_1(m_A, r_A)$ and $u_A = v_A S_A + k_A P$, where k_A is a random number chosen by A. A sends (sid, v_A, u_A) to B and C.

2.2. $B \rightarrow A, C: sid, v_B, u_B$

B computes $m_B = H_1(sid, A, B, C, P_A, P_A', P_B, P_B', P_C, P_C')$, $r_B = e(P,P)^{k_B}$, $v_B = h_1(m_B, r_B)$ and $u_B = v_B S_B + k_B P$, where k_B is a random number.

2.3. $C \rightarrow B, A: sid, v_C, u_C$

C computes $m_C = H_1(sid, A, B, C, P_A, P_A', P_B, P_B', P_C, P_C')$, $r_C = e(P,P)^{k_C}$, $v_C = h_1(m_C, r_C)$ and $u_C = v_C S_C + k_C P$, where k_C is a random number.

To verify the data from B and C, A checks whether the following two equations (9 & 10) holds. Likewise, *B* and *C* perform similar verifications. After authenticating the messages from the other two entities, $A, B,$ and C share the session keys specified in Equations (11).

$$v_B \stackrel{?}{=} h_1(H_1(sid, A, B, C, P_A, P_A', P_B, P_B', P_C, P_C'), e(u_B, P)e(Q_B, P_{pub})^{v_B}) \quad (9)$$

$$v_C \stackrel{?}{=} h_1(H_1(sid, A, B, C, P_A, P_A' P_B, P_B' P_C, P_C'), e(u_C, P)e(Q_C, P_{pub})^{v_C}) \quad (10)$$

$$\begin{aligned} &K^1_{A,B,C} = e(P_B, P_c)^a, K^2_{A,B,C} = e(P_B, P_C')^a, K^3_{A,B,C} = e(P_B', P_C)^a, \\ &K^4_{A,B,C} = e(P_B', P_C')^a, K^5_{A,B,C} = e(P_B, P_C)^{a'}, K^6_{A,B,C} = e(P_B, P_C')^{a'}, \\ &K^7_{A,B,C} = e(P_B', P_C)^{a'}, K^8_{A,B,C} = e(P_B', P_C')^{a'} \end{aligned} \quad (11)$$

6. THE SECURITY

The model of the security

To our best knowledge, there is no formal model that captures the insider attack and key-compromise impersonation attack in the tripartite key agreement protocol. To capture the security, we should consider the in-distinguishability property [11-13], and the resistance to key-compromise impersonation and the insider attack. All the models of BR95 and BPR2000, a session with corrupted entities is not considered as fresh; therefore, it cannot model the key-compromise impersonation and the insider attack. We, therefore, prove the in-distinguishability in a modified model, and gain the advantage of insider attack and key-compromise impersonation attack related to the advantage of forging advantage of underlying signature scheme. Regarding the in-distinguishability, we adopt the BPR2000 model with some modifications- (1) extension to the tripartite case, and (2) extension for the Corrupt query.

The in-distinguishability

In the model, the adversary $\mathcal{A}$ is a probabilistic machine that controls all the communications that take place between parties by interacting with a set of $\Pi^i_{U_1,U_2,U_3}$ oracles ($\Pi^i_{U_1,U_2,U_3}$ is defined to be the *ith* instantiation of an entity U_1 in a specific run, and U_2 and U_3 are the entities with whom U_1 wishes to establish a session key). The pre-defined oracle queries are described informally as follows.

-**Send**(U_1, U_2, U_3, i, m) allows $\mathcal{A}$ to send some message m of his choice to $\Pi^i_{U_1,U_2,U_3}$ at will. $\Pi^i_{U_1,U_2,U_3}$, upon receiving the query, will compute what the protocol specification demands and return to $\mathcal{A}$ the response message and/or decision. If $\Pi^i_{U_1,U_2,U_3}$ has either accepted with some session key or terminated, this will be made known to $\mathcal{A}$.

Reveal(U_1, U_2, U_3, i) query allows $\mathcal{A}$ to expose an old session key that has been

previously accepted. $\Pi^{i}_{U_1,U_2,U_3}$, upon receiving the query and if it has accepted and holds some session key, will send this session key back to $\mathcal{A}$.

Corrupt(U_1, K_E**)** query allows $\mathcal{A}$ to corrupt the entity U_1 at will, and thereby learn the complete internal state of the entity. The corrupt query also allows $\mathcal{A}$ to overwrite the long-term key of the corrupted entity to the value of his choice (i.e., K_E). This query can be used to model the real world scenarios of an insider co-operating with the adversary or an insider who has been completely compromised by the adversary.

Test(U_1, U_2, U_3, i**)** query**:** If $\Pi^{i}_{U_1,U_2,U_3}$ has accepted with some session key and is being asked a Test(U_1, U_2, U_3, i**)**, then depending on a random bit b, $\mathcal{A}$ is given either the actual session key or a session key drawn randomly from the session key distribution.

-In addition, the hashing functions $h_1()$ and $H_1()$ are modeled as three random oracles with domains and ranges specified as the hashing functions.

The definition of security depends on the notations of partnership of oracles and in-distinguishability. In the BPR2000 model, partnership of oracles is defined using SIDs (session identifiers). The definition of partnership is used in the definition of security to restrict the adversary's Reveal and Corrupt queries to oracles that are not partners of the oracles whose key the adversary is trying to guess.

Definition 1. Extension of BPR2000 Definition of Partnership: Three oracles $\Pi^{i}_{U_1,U_2,U_3}$, $\Pi^{j}_{U_2,U_1,U_3}$ and $\Pi^{k}_{U_3,U_2,U_1}$ are BPR2000 partners if, and only if, the three oracles have accepted the same session key with the same SID, have agreed on the same set of entities, and no other oracles besides $\Pi^{i}_{U_1,U_2,U_3}$, $\Pi^{j}_{U_2,U_1,U_3}$ and $\Pi^{k}_{U_3,U_2,U_1}$ have accepted with the same SID.

Definition of security in both BR95 and BPR2000 also depend on the notation of freshness of the oracle to whom the *Test* query is sent. For $\Pi^{i}_{U_1,U_2,U_3}$ to be fresh, the adversary in the BR95 model is not restricted from sending Corrupt queries to entities apart from the entities of oracles $\Pi^{i}_{U_1,U_2,U_3}$ and its partner oracles $\Pi^{j}_{U_2,U_1,U_3}$ and $\Pi^{k}_{U_3,U_2,U_1}$ (if such partners exist). We, therefore, adopt the definition of freshness of BR95 model.

Definition 2. Extension of BR95 Definition of Freshness: $\Pi^{i}_{U_1,U_2,U_3}$ is fresh (or it holds a fresh session key) at the end of execution, if, and only if, oracle $\Pi^{i}_{U_1,U_2,U_3}$ has accepted with or without a partner oracles $\Pi^{j}_{U_2,U_1,U_3}$ and $\Pi^{k}_{U_3,U_2,U_1}$, all the oracles $\Pi^{i}_{U_1,U_2,U_3}$, $\Pi^{j}_{U_2,U_1,U_3}$ and $\Pi^{k}_{U_3,U_2,U_1}$ (if such an partner oracles exist) have not been sent a *Reveal* query, and the entities U_1, U_2 and U_3 of oracles $\Pi^{i}_{U_1,U_2,U_3}$, $\Pi^{j}_{U_2,U_1,U_3}$ and $\Pi^{k}_{U_3,U_2,U_1}$ (if such partners exist) have not been sent a *Corrupt* query.

Security in both the BR95 and BPR2000 models is defined using the game *G*, played between the adversary A and a collections of $\Pi^{i}_{U_x,U_y,U_z}$ oracles for players U_x, U_y and $U_z \in \{U_1,U_2,...,U_{N_P}\}$ and instances $i \in \{1,...,l\}$. The adversary $\mathcal{A}$ runs the game simulation *G* with setting as follows.

-**Stage 1**: $\mathcal{A}$ is able to send *Send*, *Reveal* and *Corrupt* queries in the simulation.

-**Stage 2**: At some point during *G*, $\mathcal{A}$ will choose a fresh session and send a Test query to the fresh oracle associated with the test session. Depending on the randomly chosen bit *b*, $\mathcal{A}$ is given either the actual session key or a session key drawn from the session key distribution.

-**Stage 3**: $\mathcal{A}$ continues making any Send, Reveal and Corrupt oracle queries to its choice.

-**Stage 4**: Eventually, $\mathcal{A}$ terminates the game simulation and output its guess bit *b'*.

Success of $\mathcal{A}$ in *G* is measured in terms of $\mathcal{A}$'s advantage in distinguishing whether $\mathcal{A}$ receives the real key or a random value. Let the advantage function of $\mathcal{A}$ be denoted by $Adv^{A}(k)$, where k is the security parameter and $Adv^{A}(k)$=2Pr[*b*=*b'*]-1.

Key-compromise impersonation

The participating entities (except the adversary) are always considered honest in all of the BR95 model, the BPR2000 model and the Canetti-Krawczyk model, and a session with any corrupted entity is not considered as fresh for testing. It, therefore, cannot capture the key-compromise impersonation attack. However, we can gain the advantage of key-compromise impersonation to that of forging a signature with the private key. In our tripartite scheme, the adversary who has compromised U_1's private key and should try to impersonate U_2 to both U_1 and U_3. And, the adversary should generate U_2's signature on the fresh *sid* and ephemeral public keys. Therefore, his advantage of impersonation is directly related to the advantage of forging U_2's signature.

Insider attack

For the tripartite case involving entities U_1, U_2 and U_3, we consider the following two scenarios are non-sense: (1) U_1 and U_2 co-operatively impersonate U_3 to themselves, and (2) U_1 impersonates U_2 and U_3 simultaneously to himself. So, the only meaningful attack scenarios are like that U_1 impersonates U_2 to U_3 such that U_3 wrongly believes that itself, U_1 and U_2 will share the same key. In our protocol, U_3 will complete the protocol and compute the session key if only if U_3 has validated the second run message from U_1 and U_2. Of course, U_1 (the inside attacker) can generate the valid message in the second run. But, to generate valid message on behalf of U_2, U_1 should generate the valid signature on the session-bound data (*sid*, U_1, U_2, U_3, $P_{U_1}, P_{U_1}', P_{U_2}, P_{U_2}', P_{U_3}, P_{U_3}'$). So, the inside attacker U_1's advantage in impersonating U_2 is the same as that advantage of forging U_2's signature. Since Hess's signature is secure against adaptively chosen message attack and U_3's ephemeral public keys P_{U_3} and P_{U_3}' are random and fresh, the advantage is negligible. The detailed advantage is given in the

proof of Theorem 1. Now we are ready to define the security.

Definition 3 (Secure tripartite key agreement protocol): A tripartite key agreement protocol is secure in our model if the following thee requirements are satisfied:

1. **Validity**: When the protocol is run among three oracles in the absence of a malicious adversary, the three oracles accept the same key.
2. **Indistinguishabilit**y: For all probabilistic, polynomial-time adversaries A, $Adv^{A}(k)$ is negligible.
3. **Security against insider impersonation and key-compromise impersonation**: Even an insider (and a key-compromise impersonator) cannot impersonate another entity to the third entity and complete the session run with the third one.

6.2 Security proof

Theorem 1. The proposed tripartite key agreement protocol is secure in the sense of Definition 3 if the Hess's digital signature scheme is secure against the adaptively chosen message attack and the DBDH is hard.

Proof:

1. The validity is straightforward due to our protocol specification.
2. The security against insider impersonation (and the key-compromise impersonation) is equivalent to the security of the Hess's signature scheme. This has been discussed in Section 6.1, and the detailed advantage will be given in proving the in-distinguishability.
3. So, we concentrate on the in-distinguishability. The general notation of this in-distinguishable proof is to assume an adversary A who can gain a non-negligible advantage in distinguishing the test key in the game, and use A to construct a distinguisher D that distinguishes between the distributions Q_0 and Q_1, with non-negligible probability.

To easier present the reduction, we examine the case of single key agreement (that is, each entity just sends one ephemeral public key and the session key is $e(P,P)^{abc}$). The result can be easily extended to the multiple key agreement case. The proof can be divided into two cases since the adversary A can either gain its advantage against the protocol by forging a participating entity's signature or gain its advantage against the protocol without forging a participating entity's signature.

Case 1. A gains its advantage by forging a participating entity's signature.

We denote by $\Pr[Succ^{Sig}(k)]$ the probability of a successful signature forgery under adaptively chosen message attack, and define an event *SigForgery* to be an event that at some point in the game A asks a Send(U_1, U_2, U_3, i, $(sid, v_{U_1,i}, u_{U_1,i})$) query to some

partner oracles $\Pi^{j}_{U_2,U_1,U_3}$ or $\Pi^{k}_{U_3,U_2,U_1}$ such that the oracle accept, but the signature value $(v_{U_1,i}, u_{U_1,i})$ used in the query was not previously output by a fresh oracle. We construct an adaptive *Signature* forger F against the message authentication scheme using $\mathcal{A}$ in the following game G_F.

-**Stage 1**: F is provided permanent access to the *Signature* oracle O_U associated with its private key of U throughout the game G_F.

- F randomly chooses an entity $\overline{U} \in \{U_1,...,U_{N_P}\}$. $\overline{U}$ is F's guess at which $\mathcal{A}$ will choose for the *SigForgery*.
- F generates a list of public key/private key pairs for the entities $\{U_1,...,U_{N_P}\} \setminus \{\overline{U}\}$.

-**Stage 2**: F runs $\mathcal{A}$ and answers all queries from $\mathcal{A}$. This can be easily done since F can respond to all oracle queries as required using the keys chosen in Stage 1 and O_U. F also records all the signatures it receives from O_U. If, during the execution, $\mathcal{A}$ make an oracle query that includes a forged signature for $\overline{U}$, then F outputs the signature forgery as its own and halts. Otherwise, F halts as $\mathcal{A}$ halts.

Since $\overline{U}$ is randomly chosen from the N_P entities, the probability that $\overline{U}$ is the entity for whom $\mathcal{A}$ generates a forgery is at least $1/N_P$. Therefore, the success probability of F is $\Pr[Succ^F(k)] \geq \Pr[SigForgery(k)]/N_P$. Hence,

$$N_P \cdot \Pr[Succ^F(k)] \geq \Pr[SigForgery(k)]. \quad (12)$$

Since Hess's signature is secure against adaptively chosen message attack and the N_P is polynomial in k, the $\Pr[SigForgery(k)]$ is negligible.

Case 2. $\mathcal{A}$ gains its advantage without forging a participating entity's signature.

This part assumes that $\mathcal{A}$ gains its advantage without forging a participating entity's signature. We use $\mathcal{A}$ to construct a distinguisher D for the DBDH problem. The input to D is denoted by $(G_1, G_2, P, e, \alpha, \beta, \gamma, \delta)$ and is chosen from Q_0 and Q_1 (each with probability 1/2). Let l be an upper bound on the number of sessions invoked by $\mathcal{A}$ among any three entities. The objective of D is to correctly guess the challenge θ in the game simulation G_{DBDH}. The distinguisher D uses the adversary $\mathcal{A}$ as a subroutine and proceeds as follows.

-**Stage 1**: D is given the challenge $(G_1, G_2, P, e, \alpha, \beta, \gamma, \delta)$ that is chosen from Q_0 and Q_1 (each with probability 1/2).

-**Stage 2**: D randomly chooses his target entities $A, B, C \in \{U_1,...,U_{N_P}\}$, and randomly chooses a target session $u \in \{1,2,...,l\}$. He chooses a list of public key/private key pair for all entities $\{U_1,...,U_{N_P}\}$. Define $\overline{N} = c(\binom{N_P}{3}) \cdot l$.

- **Stage 3**: D runs $\mathcal{A}$ with public parameters (G_1, G_2, P, e) to determine his guessing bit θ'. During the simulation, D should answer $\mathcal{A}$'s queries and maintain the Send-list as follows.

Send(U_1, U_2, U_3, i,m) query:

1. If (($\{U_1, U_2, U_3\} \neq \{A, B, C\}$) OR ($i \neq u$)) AND m=*, then randomly chooses an integer $w \in Z_q^*$ and outputs the outgoing message as (i, U_1, U_2, U_3, wP). D also records the data (i, U_1, U_2, U_3, (w, wP)) in his send-list.
2. If the set ($\{U_1, U_2, U_3\}=\{A, B, C\}$) AND m=* AND u=i, then
   a. If $U_1 = A$, output the outgoing message as (i, U_1, U_2, U_3, α).
   b. If $U_1 = B$, output the outgoing message as (i, U_1, U_2, U_3, β).
   c. If $U_1 = C$, output the outgoing message as (i, U_1, U_2, U_3, γ).
3. If m has the form (w_1P, w_2P, w_3P), use U_1's corresponding private key to generate the signature on (i, U_1, U_2, U_3, w_1P, w_2P, w_3P) and output the signature (v_{U_1}, u_{U_1}).
4. If m has the form (v_{U_1}, u_{U_1}), then use U_1's public key to verify the signature. If the verification succeeds, then the signature must be previously generated by D. D outputs decision = "*accept*"; otherwise, outputs "*reject*".
5. In all other cases the input to the Send query is invalid, so D randomly chooses a bit θ' as its response and hand it to the challenger.

Reveal(U_1, U_2, U_3, i) query:

1. $\{U_1, U_2, U_3\}=\{A, B, C\}$ AND ($\Pi^i_{U_1,U_2,U_3}$ has accepted) AND (it forms the target session), then D randomly chooses a bit θ' as its response and hand it to the challenger.
2. If this is not the target session AND ($\Pi^i_{U_1,U_2,U_3}$ has accepted), then compute and output the session key sk (compute the sk using the data in the Send-list).
   In all other cases the input to the Revel query is invalid, so D randomly chooses a bit θ' as its response and hand it to the challenger.

Corrupt(U , K) query:

If $U \in \{A, B, C\}$, then D randomly chooses a bit θ' as its response and hand it to the challenger.

If $U \in \{U_1,...,U_{N_P}\} \backslash \{A, B, C\}$, then D hands in all internal of U to $\mathcal{A}$, and updates U 's key pair as K.

Test(U_1, U_2, U_3, i) query:

If $\{U_1, U_2, U_3\}=\{A, B, C\}$ AND (this is the target session) AND (the last flow that $\Pi^i_{U_1,U_2,U_3}$ received had the form (v_{U_1}, u_{U_1}) is a valid signature on (i, U_1, U_2, U_3, α, β, γ), then D will answer the query with δ, else D randomly chooses a bit θ' as its response and hand it to the challenger. After making a *Test* query and getting an answer δ from D, $\mathcal{A}$ continues interacting with the protocol an eventually outputs a guess bit b'. D then outputs his guess bit θ'=b' as its response to the challenger.

The probability that $\mathcal{A}$ chooses the target session as the *Test* session is $1/(\overline{N})$ since *D* randomly chooses the entities (*A, B, C*) and the session *u*. It is easy to verify that when the target session and the *Test* session are different or the target session is not fresh, *D* outputs a random bit θ', so the probability of success is 1/2. For the case the *Test* session is the target session, then the probability of *D*'s success is the same as $\mathcal{A}$. Hence, *D*'s success probability is as follows.

$$\Pr[Succ^{DBDH}(k)] = \frac{\overline{N}-1}{2\overline{N}} + \frac{\Pr[b'=b \mid \neg SigForgery]}{\overline{N}}$$

$$= \frac{\overline{N}-1}{2\overline{N}} + \frac{(Adv^{A}(k) \mid \neg SigForgery)+1}{2\overline{N}}$$

$$2\Pr[Succ^{DBDH}(k)]-1 = \frac{(Adv^{A}(k) \mid \neg SigForgery)}{\overline{N}}$$

$$\overline{N} \cdot Adv^{DBDH}(k) = (Adv^{A}(k) \mid \neg SigForgery) \quad (13)$$

The $\overline{N}$ is polynomial in *k*, so $(Adv^{A}(k) \mid \neg SigForgery)$ is negligible if the DBDH is hard. From (12) an (13), we have proved the in-distinguishability and the theorem.

7. PERFORMANCE ANALYSIS

Table 1 summarizes the comparisons of our schemes and its counterparts. Among the operations, the pairing operation is the most expensive. From the comparisons, our scheme demands less pairing operations. Please also notice that our scheme can be easily extended to share only one session key by sending one ephemeral public value per entity or to share n^3 keys by sending *n* ephemeral public values per entity. While in the ZLK scheme, it is not known whether it is feasible to extend to more than eight keys.

Regarding the security, both the ZLK scheme and the Shim's scheme is vulnerable to the insider attack, and Shim's scheme is further vulnerable to key-compromise impersonation attack. Other considered security properties include *forward secrecy* and *TA forward secrecy*. Our scheme and the ZLK scheme are ID-based, while the Shim's scheme is not. All the three schemes are secure regarding the forward secrecy property.

From the table, we can see that our scheme is more computational efficient than the ZLK scheme and Shim's scheme, in addition to its security robustness and scalability. .

8. CONCLUSIONS

This paper has showed the security weaknesses of the Zhang-Liu-Kim tripartite key agreement protocol and the Shim tripartite schemes. We also have proposed an ID-based tripartite multiple key agreement protocol to raise security level and improve its efficiency

and scalability. The security of the proposed scheme is proved in the modified BPR2000 model relative to the Decisional Bilinear Diffie-Hellman problem and the security of the Hess' signature scheme.

Table 1. Comparisons of tripartite key agreement protocols

	ID-ZLK	ID-MKEY-ZLK	Shim	Our scheme		
Number of session keys	1	8	1	1, 8, …, n^3		
Insider impersonation	Y	Y	Y	N		
Key compromise impersonation	N	N	Y	N		
Forward secrecy	Y	Y	Y	Y		
KGC forward secrecy	Y	Y	Y	Y		
Number of round	1	1	1	2		
Computation of one party	5 E_{pair} 5 E_{scalar} 1 E_{add} 3 $F_{\exp}$	(8 keys) 8 E_{pair} 6 E_{scalar} 3 E_{add} 8 $F_{\exp}$	5 E_{pair} 5 E_{scalar} 2 E_{add} 1 $F_{\exp}$	(1 key) 3 E_{pair} 3 E_{scalar} 1 E_{add} 4 $F_{\exp}$	(8 keys) 6 E_{pair} 4 E_{scalar} 1 E_{add} 11 $F_{\exp}$	(n^3 keys) n^2+2 E_{pair} n+2 E_{scalar} 1 E_{add} n^3+3 $F_{\exp}$

** E_{pair} denotes one pairing operation on the elliptic curves (E); E_{scalar} denotes one scalar multiplication on the curves; E_{add} denote one elliptic curve point addition; $F_{\exp}$ denotes one exponentiation on G_2.

References

1. Liu, S., Zhang, F., Chen, K. ID-based tripartite key agreement protocol with pairing. In Proc. IEEE ISIT 2003, Yokohama, Japan, 2003, pp. 136.
2. Shim, K. Efficient one round tripartite authenticated key agreement protocol from Weil pairing. Electron. Lett., 2003, 39(2): 208-209.
3. Zhang, F., Liu, S., and Kim, K. ID-based one-round authenticated tripartite key agreement protocol with pairings. Cryptology eprint Archive, Report 2002/122.
4. Al-Riyami, S. S. and Paterson, K. G., "Tripartite Authenticated Key Agreement Protocols from Pairings", IMA Conference on Cryptography and Coding, LNCS 2898, Springer-Verlag (2003), pp. 332-359.
5. Joux, A. A one round protocol for tripartite Diffie-Hellman. ANTS IV, LNCS1838, Spring-Verlag, 2000, pp. 385-394.
6. Nalla, D. and Reddy, K.C. ID-based tripartite authenticated key agreement protocols from pairings. Cryptology eprint Archive, Report 2003/004.

7. Sun, H.-M. and Hsieh, B.-T., “Security Analysis of Shim's Authenticated Key Agreement Protocols from Pairings", Cryptology ePrint Archive, Report 2003/113.
8. Hess, F., “Efficient identity based signature schemes based on pairings”, SAC 2002, LNCS2595, pp. 310-324, Springer-Verlag, 2002.
9. Cheng, Z., Vasiu, L., and Comley, R., “Pairing-based one-round tripartite key agreement protocols”, Cryptology ePrint Archive, Report 2004/079, available at http://eprint.iacr.org/2004/079/.
10. Shim, K. “A man-in-the-middle attack on Nalla-Reddy’s ID-based tripartite authenticated key agreement protocol,” Cryptology ePrint Archive, Report 2003/115.
11. Bellare, M., Rogaway, P., “Provably secure session key distribution: The three party case”, in 27th ACM Symposium on the Theory of Computing, pp. 57-66, ACM press, 1995.
12. Bellare, M., Pointcheval, D., and Rogaway, P., “Authenticated key exchange secure against dictionary attacks”, Eurocrypt 2000, LNCS 1807, Springer, pp. 139-155, 2000.
13. Canetti, R., Krawczyk, H., “Analysis of key-exchange protocols and their use for building secure channels”, in Eurocrypt 2001, LNCS 2045, pp. 451-472, Springer, 2001.
14. Hitchcock, Y., Boyd, C., Nieto, J.M.G., “Tripartite key exchange in the Canetti-Krawczyk proof model”, in 5th International Conference on Cryptology in India - Indocrypt 2004. Springer-Verlag.
15. Choo, K.K.R., Boyd, C., Hitchcock, Y., Greg, M., “On session identifiers in provably secure protocols”, in Fourth Conference on Security in Communication Networks - SCN 2004, LNCS 3352, pp. 352-267, Springer-Verlag.