Modeling Key Compromise Impersonation Attacks on Group Key Exchange Protocols

M. Choudary Gorantla, Colin Boyd, and Juan Manuel González Nieto

Information Security Institute, Faculty of IT, Queensland University of Technology
GPO Box 2434, Brisbane, QLD 4001, Australia.
Email: mc.gorantla@isi.qut.edu.au, {c.boyd,j.gonzaleznieto}@qut.edu.au

Abstract. A key exchange protocol allows a set of parties to agree upon a secret session key over a public network. Two-party key exchange (2PKE) protocols have been rigorously analyzed under various models considering different adversarial actions. However, the analysis of group key exchange (GKE) protocols has not been as extensive as that of 2PKE protocols. Particularly, the security attribute of key compromise impersonation (KCI) resilience has so far been ignored for the case of GKE protocols. We first model the security of GKE protocols addressing KCI attacks by both outsider and insider adversaries. We then show that a few existing protocols are not secure even against outsider KCI attacks. The attacks on these protocols demonstrate the necessity of considering KCI resilience for GKE protocols. Finally, we give a new proof of security for an existing GKE protocol under the revised model assuming random oracles.
Keywords. Group Key Exchange, Key Compromise Impersonation, Insider Attacks.

1 Introduction

A group key exchange (GKE) protocol allows a group of parties to agree upon a secret common session key over a public network. Although there had been GKE protocols earlier, Bresson et al. [10, 8, 9] were the first to analyze the security of GKE protocols under formal security models. These models define authenticated key exchange (AKE) security and mutual authentication as the desired notions of security against an outsider adversary i.e. assuming that the adversary is not part of the group. The notion of AKE-security demands that an outsider adversary should not learn the session key while the notion of mutual authentication requires that parties who complete the protocol execution should output identical session keys and that each party should be ensured of the identity of the other participating parties.

Katz and Shin [18] define insider security for GKE protocols by separating the requirements of mutual authentication in the presence of insiders into *agreement* and *security against insider impersonation attacks*. Their definition has been revisited by Bohli et al. [3] and Bresson and Manulis [11] under different corruption models. Bohli et al. also define the notion of *contributiveness* which requires that a proper subset of insiders should not predetermine the resulting session key. Bresson and Manulis strengthen this notion by considering strong corruptions where the ephemeral session state of an instance may also be revealed in addition to the long-term private key of the party.

All the models above (for both outsider and insider security) assume that a party will be fully under the control of the adversary once the party's long-term private key is compromised. These models however consider forward secrecy to limit the damage done by compromise of long-term private key after session completion. Another equally important security attribute related to compromise of the long-term private key of parties is key compromise impersonation (KCI) resilience. Informally, an adversary is said to impersonate a party B to another party A if B is honest and the protocol instance at A accepts the session with B as one of the session peers but there exists no such partnered instance at B [18]. In a successful KCI attack, an adversary with the knowledge of the long-term private key of a party A can impersonate B to A. Resilience to KCI attacks is often seen as a desired security attribute for two-party key exchange (2PKE) [22, 21]. However, it has so far been ignored for the case of group key exchange.

We argue that KCI resilience for GKE protocols is at least as important as it is for 2PKE protocols. For this purpose we illustrate two scenarios with different setup assumptions, where a KCI attack is a threat.

We first extend the peer-to-peer file sharing system scenario given for the two-party case by Ng [24, Section 4.2.2], to the group case. In these systems each user stores some data and allows access only to users with whom it wants to share the data. This can be achieved by executing a GKE protocol with the peers who have read access to the data and sending them the data encrypted using the established session key. Let A be the party who has some sensitive data. The goal of an adversary $\mathcal{A}$ is to access the data at A which is to be shared only with the users who have read access. Although the compromise of A's long-term private key helps $\mathcal{A}$ to impersonate A, $\mathcal{A}$ may not be able to access the data locally stored at A. However, if the GKE protocol used is vulnerable to KCI attacks, $\mathcal{A}$ can impersonate a party who has read access and decrypt the data using the session key. Note that in this scenario, the GKE protocol needs to have forward secrecy. Otherwise, compromising A's private key will enable the adversary to obtain the session key. On the other hand, having forward secrecy alone is not sufficient as $\mathcal{A}$ will be able to spoof the presence of an honest party if the protocol is not KCI resilient as discussed above.

The second one is a server-client scenario for the application given by Bresson et al. [7]. They propose a GKE protocol which allows a cluster of mobile devices (acting as clients) to agree upon a session key with a wireless gateway (acting as a server). The authors suggest that the established session key can be used along with a suitable protocol to secure IEEE 802.11 wireless networks. If the long-term private key of the gateway is compromised, an adversary can easily impersonate the gateway and allow any mobile device to access the wireless network. However, impersonating the gateway may be recognized by observing the logs, erasing which may require additional administrative rights depending on how the gateway is configured. On the other hand, if the GKE protocol is vulnerable to KCI attacks, the adversary can impersonate a legitimate mobile device and gain access to the wireless network without being detected. We indeed present a KCI attack on the protocol of Bresson et al. [7].

Outsider KCI Resilience. We call a *party* corrupted if the long-term private key of the party is compromised, while a *protocol instance* is called corrupted if the ephemeral session state of that instance is revealed. In an outsider KCI attack scenario for GKE protocol, an adversary $\mathcal{A}$ is allowed to compromise the long-term private keys of up to all parties except one. But, it is allowed neither to corrupt the protocol instances at any of parties nor to participate in the protocol on behalf of the corrupted parties. $\mathcal{A}$ is an outsider to the specific protocol execution in consideration as no session specific information is revealed. $\mathcal{A}$ is considered successful in mounting a KCI attack if it can impersonate any uncorrupted party to an uncorrupted instance at any of the corrupted parties. We consider the goal of $\mathcal{A}$ as an outsider is to break the confidentiality of the session key established. Hence, we modify the existing definition of AKE-security accordingly.

Insider KCI Resilience. A party is called an insider if the adversary corrupts the party and actively participates in the protocol on behalf of that party. In an insider KCI attack scenario, the goal of an adversary $\mathcal{A}$ is to impersonate an uncorrupted party B to an uncorrupted instance of a party A. Note that $\mathcal{A}$ is allowed to compromise the long-term private key of A, while all the parties except A and B can be insiders. We revise the existing definition of mutual authentication accordingly. It is easy to see that the revised definition implies mutual authentication with KCI resilience in the presence of no insiders.

Boyd and González Nieto (BG) proposed a one-round GKE protocol and proved the protocol secure under the AKE-security notion. Choo et al. [13] later presented unknown key share attacks on the BG protocol and suggested an improvement. We show that the improved protocol is not outsider KCI resilient. We also show that the tripartite key agreement protocol TAK-3 of Al-Riyami and Paterson [1] and the GKE protocol of Bresson et al. [7] are not secure against outsider KCI attacks.

One way to modify the BG protocol to make it secure against KCI attacks is by applying the compiler of Katz and Shin (KS) [18]. A KS-compiled protocol is shown to guarantee mutual authentication in the presence of insiders. If one applies the KS-compiler to the improved BG protocol it is easy to show that the resulting protocol will be both outsider and insider KCI resilient in the random oracle model. However the resulting two-round protocol will not provide forward secrecy as the BG protocol itself does not have this property. Hence, we show that the two-round protocol of Bohli et al. [3], which already has forward secrecy, satisfies our new definitions. For the sake of completeness, we also show that this protocol is secure under the notion of contributiveness proposed by Bresson and Manulis [11]. The contributions of this paper are:

1. Modeling KCI attacks on GKE protocols by presenting new outsider and insider security notions
2. KCI attacks on the protocols of Boyd and González Nieto [6], Al-Riyami and Paterson [1] and Bresson et al. [7]
3. A new proof of security for the protocol of Bohli et al. [3] in the random oracle model

Organization. In Section 2 we present new notions of AKE-security and mutual authentication considering KCI attacks by outsiders and insiders respectively. Section 3 presents outsider KCI attacks on the improved Boyd and González Nieto's protocol, Al-Riyami and Paterson's protocol and Bresson et al.'s protocol. In Section 4, we show that the protocol of Bohli et al. [3] is insider secure i.e. that it satisfies the new notions AKE-security and mutual authentication in addition to existing notion of contributiveness. We conclude our paper in Section 5 with a comparison among existing GKE protocols.

2 Model

Let $\mathcal{U} = \{U_1, \ldots, U_n\}$ be a set of n parties. The protocol may be run among any subset of these parties. Each party is assumed to have a pair of long-term public and private keys, (PK_U, SK_U) generated during an initialization phase prior to the protocol run. A GKE protocol π executed among n users is modeled as a collection of n programs running at the n different parties in $\mathcal{U}$. Each instance of π within a party is defined as a session and each party may have multiple such sessions running concurrently.

Let π_U^i be the i-th run of the protocol π at party $U \in \mathcal{U}$. Each protocol instance at a party is identified by a unique session ID. We assume that the session ID is derived during the run of the protocol. The session ID of an instance π_U^i is denoted by sid_U^i. We assume that each party knows who the other participants are for each protocol instance. The partner ID pid_U^i of an instance π_U^i, is a set of identities of the parties with whom π_U^i wishes to establish a common group key. Note that pid_U^i includes the identity of U itself.

An instance π_U^i enters an *accepted* state when it computes a session key sk_U^i. Note that an instance may terminate without ever entering into an accepted state. The information of whether an instance has terminated with acceptance or without acceptance is assumed to be public. Two instances π_U^i and $\pi_{U'}^j$ at two different parties U and U' respectively are considered *partnered* iff (1) both the instances have accepted, (2) $\mathsf{sid}_U^i = \mathsf{sid}_{U'}^j$, and (3) $\mathsf{pid}_U^i = \mathsf{pid}_{U'}^j$.

The communication network is assumed to be fully controlled by an adversary $\mathcal{A}$, which schedules and mediates the sessions among all the parties. $\mathcal{A}$ is allowed to insert, delete or modify the protocol messages. If the adversary honestly forwards the protocol messages among all the participants, then all the instances are partnered and output identical session keys. Such a protocol is called a correct GKE protocol. In addition to controlling the message transmission, $\mathcal{A}$ is allowed to ask the following queries.

- $\mathsf{Execute}(\mathsf{pid})$ prompts a complete execution of the protocol among the parties in pid. $\mathcal{A}$ is given all the protocol messages, modeling passive attacks.
- $\mathsf{Send}(\pi_U^i, m)$ sends a message m to the instance π_U^i. If the message is only pid, the instance π_U^i is initiated with partner ID pid. The response of π_U^i to any Send query is returned to $\mathcal{A}$.
- $\mathsf{RevealKey}(\pi_U^i)$ If π_U^i has accepted, $\mathcal{A}$ is given the session key sk_U^i established at π_U^i.
- $\mathsf{Corrupt(U)}$ The long-term secret key SK_U of U is returned to $\mathcal{A}$. Note that this query returns neither the session key (if computed) nor the internal state.
- $\mathsf{RevealState}(\pi_U^i)$ The internal state of U is returned to $\mathcal{A}$. We assume that the internal state is erased once π_U^i has accepted. Hence, a $\mathsf{RevealState}$ query to an accepted instance returns nothing.
- $\mathsf{Test}(\pi_U^i)$ A random bit b is secretly chosen. If $b = 1$, $\mathcal{A}$ is given sk_U^i established at π_U^i. Otherwise, a random value chosen from the session key probability distribution is given. Let K_b be the challenge key given to $\mathcal{A}$. Note that a Test query is allowed only on an accepted instance.

2.1 AKE Security

We present a revised notion of AKE-security by taking KCI attacks into account. As this is a notion of outsider security, we assume that all the participants are honest i.e. all the parties execute the protocol honestly.

The notion of freshness is central to the definition of AKE-security. We define the notion of freshness by considering KCI attack scenarios based on a corresponding notion for two-party key exchange given by Krawczyk [21]. Informally, a session is considered fresh if the session key is not trivially compromised.

Freshness. An instance π_U^i is fresh if the following conditions hold:

1. the instance π_U^i or any of its partners has not been asked a $\mathsf{RevealKey}$ after their acceptance
2. the instance π_U^i or any of its partners has not been asked a $\mathsf{RevealState}$ before their acceptance
3. If $\pi_{U'}^j$ is a partner of π_U^i and $\mathcal{A}$ asked $\mathsf{Corrupt}(U')$, then any message that $\mathcal{A}$ sends to π_U^i on behalf of $\pi_{U'}^j$ must come from $\pi_{U'}^j$ intended to π_U^i .

The last condition requires that the adversary be an outsider, i.e. it must be passive for any partner that it corrupts.

Definition 1 (AKE-Security with KCI resilience)**.** An adversary $\mathcal{A}_{ake}$ against the AKE-security notion is allowed to make $\mathsf{Execute}$, Send, $\mathsf{RevealState}$, $\mathsf{RevealKey}$ and $\mathsf{Corrupt}$ queries in Stage 1. $\mathcal{A}_{ake}$ makes a Test query to an instance π_U^i at the end of Stage 1 and is given a challenge key K_b as described above. It can continue asking queries in Stage 2. Finally, $\mathcal{A}_{ake}$ outputs a bit b' and wins the AKE security game if (1) $b' = b$ **and** (2) the instance π_U^i that was asked Test query remained fresh till the end of $\mathcal{A}_{ake}$'s execution. Let $\mathsf{Succ}_{\mathcal{A}_{ake}}$ be the success probability of $\mathcal{A}_{ake}$ in winning the AKE security game. The advantage of $\mathcal{A}_{ake}$ in winning this game is $\mathsf{Adv}_{\mathcal{A}_{ake}} = |2 \cdot \Pr[\mathsf{Succ}_{\mathcal{A}_{ake}}] - 1|$. A protocol is called AKE-secure if $\mathsf{Adv}_{\mathcal{A}_{ake}}$ is negligible in the security parameter k for any polynomial time $\mathcal{A}_{ake}$.

The definition of freshness takes care of the KCI attacks as it does allow $\mathcal{A}_{ake}$ to corrupt the owner of the test protocol instance. Note that if the adversary is active with respect to a partner to the test instance π_U^i, then that party cannot have been corrupted, otherwise π_U^i is not fresh. The definition also takes forward secrecy into account as it allows $\mathcal{A}_{ake}$ to obtain the long-term private keys of all the parties. In this case, $\mathcal{A}_{ake}$ must be passive with respect to all partners of π_U^i.

2.2 Mutual Authentication

Katz and Shin [18] first presented a definition of insider security that models impersonation attacks and ensures agreement on the session key in the presence of insiders. Bohli et al. [3] revisited this notion in weak corruption model, where session state is not revealed. They also presented insider attacks on the protocols of Katz and Yung [19] and Kim et al. [20] that violate integrity of the protocols. Later, Bresson and Manulis [11] unified the insider security notions of Katz and Shin into their definition of mutual authentication. They also considered session state reveal queries separately from the corrupt queries. We strengthen the definition of Bresson and Manulis by considering KCI attacks by insiders.

Definition 2 (Mutual Authentication with KCI resilience)**.** An adversary $\mathcal{A}_{ma}$ against the mutual authentication of a correct GKE protocol π is allowed to ask $\mathsf{Execute}$, Send, $\mathsf{RevealState}$, $\mathsf{RevealKey}$ and $\mathsf{Corrupt}$ queries. $\mathcal{A}_{ma}$ violates the mutual authentication property of the GKE protocol if at some point during the protocol run, there exists an uncorrupted instance π_U^i (although the party U may be corrupted) that has accepted with a key sk_U^i and another party $U' \in \mathsf{pid}_U^i$ that is uncorrupted at the time π_U^i accepts such that

1. there is no instance $\pi_{U'}^j$ with $(\mathsf{pid}_{U'}^j, \mathsf{sid}_{U'}^j) = (\mathsf{pid}_U^i, \mathsf{sid}_U^i)$ **or**
2. there is an instance $\pi_{U'}^j$ with $(\mathsf{pid}_{U'}^j, \mathsf{sid}_{U'}^j) = (\mathsf{pid}_U^i, \mathsf{sid}_U^i)$ that has accepted with $sk_{U'}^j \neq sk_U^i$.

Let $\mathsf{Succ}_{\mathcal{A}_{ma}}$ be the success probability of $\mathcal{A}_{ma}$ in winning the mutual authentication game. A protocol is said to provide mutual authentication in the presence of insiders if $\mathsf{Succ}_{\mathcal{A}_{ma}}$ is negligible in the security parameter k for any polynomial time $\mathcal{A}_{ma}$.

The difference between the above definition and that of Bresson and Manulis is that we allow $\mathcal{A}_{ma}$ to obtain the long-term private key of U_i, but $\mathcal{A}_{ma}$ is not allowed to execute the protocol on U_i's behalf. $\mathcal{A}_{ma}$ is considered successful in an insider KCI attack against U_i if it violates above definition of mutual authentication.

It is easy to see that if a protocol does not satisfy earlier definitions [18, 3, 11], it also does not satisfy Definition 2. Many existing protocols [3, 11] which are proven secure under the definitions in the corresponding papers also seem to satisfy our definition. Note that this is not the general case as the adversary in our definition clearly has additional power.

2.3 Contributiveness

To ensure complete covering of insider security notions we present below the notion of contributiveness by Bresson and Manulis [11]. A GKE protocol secure under this notion resists the key control attacks by Pieprzyk and Wang [25] where a proper subset of insiders tries to predetermine the resulting session key.

Definition 3 (Contributiveness)**.** An adversary $\mathcal{A}_{con}$ against the contributiveness of correct GKE protocol π is allowed ask Execute, Send, RevealKey, RevealState and Corrupt queries. It operates in two stages prepare and attack as follows:

prepare. $\mathcal{A}_{con}$ queries the instances of π and outputs some state information ζ along with a key $\tilde{k}$.
At the end of prepare stage, a set Π is built such that Π consists of uncorrupted instances which have been asked either Execute or Send queries.
attack. On input (ζ, Π), $\mathcal{A}_{con}$ interacts with the instances of π as in the prepare stage.

At the end of this stage $\mathcal{A}_{con}$ outputs (U, i) and wins the game if an instance π_U^i at an uncorrupted party U has terminated accepting $\tilde{k}$ with $\pi_U^i \notin \Pi$.

Let $\mathsf{Succ}_{\mathcal{A}_{con}}$ be the success probability of $\mathcal{A}_{con}$ in winning the above game. A protocol is said to provide contributiveness in the presence of insiders, if $\mathsf{Succ}_{\mathcal{A}_{con}}$ is negligible in the security parameter k for any polynomial time $\mathcal{A}_{con}$.

3 KCI Attacks on Existing Protocols

We present KCI attacks on the protocols of Boyd and González Nieto [6], Al-Riyami and Paterson [1] and Bresson et al. [7]. We speculate that there are many GKE protocols in the literature which are not secure against KCI attacks. By selecting these three protocols, we are able to demonstrate the importance of considering resilience to KCI attacks for GKE protocols under different setup assumptions. Note that the Boyd and González Nieto protocol is a contributory GKE protocol where each party is assumed to have equal resources. The Al-Riyami and Paterson protocol is a GKE protocol with the group size three, while the protocol of Bresson et al. assumes a server with high computational resources and many computationally restricted clients.

3.1 Boyd and González Nieto's protocol [6]

Boyd and González Nieto [6] (BG) proposed a one-round GKE protocol and proved it AKE-secure in the Bellare-Rogaway model [2] adapted to the group setting. Later, Choo et al. [13] presented an unknown key share attack on the BG protocol in a multi-user setting. They also presented an improved BG protocol that resists unknown key share attacks but do not give any formal security proof. We now briefly describe the protocol.

Let $\mathcal{U} = \{U_1, U_2, \ldots, U_n\}$ be the set of participants. All the users agree upon a distinguished user for each execution of the protocol. Without loss of generality let U_1 be the distinguished user. The protocol uses a public key encryption scheme $\mathcal{PE} = (\mathcal{K}_e, \mathcal{E}, \mathcal{D})$, where $\mathcal{K}_e$, $\mathcal{E}$ and $\mathcal{D}$ are the key generation, encryption and decryption algorithms. It also uses a signature scheme $\Sigma = (\mathcal{K}_s, \mathcal{S}, \mathcal{V})$, where $\mathcal{K}_s$, $\mathcal{S}$ and $\mathcal{V}$ are the key

Round 1

$U_1 \rightarrow * : \mathcal{U}, \mathcal{S}_{SK_{s_1}}(\mathcal{U}, \mathcal{E}_{PK_{e_2}}(N_1, U_1), \ldots, \mathcal{E}_{PK_{e_n}}(N_1, U_1))$

$U_1 \rightarrow * : \mathcal{E}_{PK_{e_i}}(N_1, U_1)$ for $1 < i \leq n$

$U_i \rightarrow * : U_i, N_i$ for $1 < i \leq n$

Key Computation

$sid = \mathcal{U} \| \mathcal{S}_{SK_{s_1}}(\mathcal{U}, \mathcal{E}_{PK_{e_2}}(N_1, U_1), \ldots, \mathcal{E}_{PK_{e_n}}(N_1, U_1)) \| \mathcal{E}_{PK_{e_i}}(N_1, U_1) \| U_i \| N_i$

The session key is $sk_i^U = \mathcal{H}(N_1 \| sid)$

Fig. 1. Improved Boyd-González Nieto Protocol [13]

generation, signature and verification algorithms. Each user is issued with a key pair for each of the schemes. Let (SK_{e_i}, PK_{e_i}) and (SK_{s_i}, PK_{s_i}) be the private-public key pairs for the encryption and signature schemes.

In the protocol, the distinguished user U_1 chooses a nonce $N_1 \xleftarrow{R} \{0,1\}^k$ and encrypts it along with its identity for each of the other parties. U_1 signs all these ciphertexts together with the set of identities of all the users $\mathcal{U}$. The set $\mathcal{U}$, the signature computed and the ciphertexts are then broadcast. All the parties $U_i \in \mathcal{U}, U_i \neq U_1$ broadcast their nonces $N_i \xleftarrow{R} \{0,1\}^k$ along with their identities. A user computes the session ID as the concatenation of all the outgoing and incoming protocol messages. A key derivation function $\mathcal{H}$ is used to compute the session key with the nonce N_1 and the session ID as input. As there is no restriction on who should send a protocol message first, the protocol can be completed in one round. The protocol message transmission and session key computation are presented in Figure 1. The users U_i, $i \neq 1$, verify the signature of U_1 and decrypt N_1 before computing the session key.

We show that the improved BG protocol in Figure 1 is not secure against KCI attacks. An attack can be mounted by corrupting any user except the distinguished user U_1. Let us assume that U_2 is corrupted. An adversary $\mathcal{A}$ can impersonate U_1 just by replaying a message from a previous successful execution of the protocol. The nonce selected by U_1 in the replayed message can be decrypted using the private key of U_2. Thus $\mathcal{A}$ can easily win the AKE-security game by selecting the test session at U_2.

A straightforward improvement to the protocol in Figure 1 could be by asking all users $U_i \neq U_1$ to encrypt their nonces with the public keys of other users and broadcast the messages. Although this thwarts the above KCI attack on the protocol, the improved protocol cannot be proven secure under the AKE-security notion. To see why, note that in the above attack scenario we have considered the simple case where the long-term private key of only one user other than U_1 is compromised. If we assume that more than one user (other than U_1) is corrupted, then the adversary can impersonate U_1 in the same way as described above and successfully mount a KCI attack. We leave open the task of constructing a one-round GKE protocol that resists KCI attacks.

3.2 Al-Riyami and Paterson's Protocol [1]

Al-Riyami and Paterson [1] proposed a series of tripartite key agreement (TAK) protocols based on Joux's protocol [17]. While the authors do not provide a definition of KCI resilience for a TAK protocol, they claim that the protocol TAK-3 is secure against KCI attacks. However we below present a KCI attack on TAK-3.

The system parameters are $(q, \mathbb{G}_1, \mathbb{G}_T, P, e, H)$, where q is a prime number, $\mathbb{G}_1$ and $\mathbb{G}_T$ are groups of order q, P is a generator of $\mathbb{G}_1$, $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ is an admissible bilinear map and H is a hash function that maps to the key space. Let (x, xP), (y, yP) and (z, zP) be the private-public key pairs of three users A, B and C respectively, where $x, y, z \in \mathbb{Z}_q^*$. The parties are issued certificates for their public key, which bind an identity to the corresponding public key. Let $Cert_A$, $Cert_B$ and $Cert_C$ be the certificates issued for the public keys of A, B and C respectively.

As the part of the protocol the users A, B and C select the ephemeral secret keys $a, b, c \xleftarrow{R} \mathbb{Z}_q^*$ respectively. The protocol message transmission and key computation is shown in Figure 2.

We now show that the protocol in Figure 2 is not KCI resilient as per our AKE-security definition. Let us assume that the adversary $\mathcal{A}$ has compromised the long-term private keys x and y of the parties

Round 1

$A \rightarrow B, C : aP \| Cert_A$
$B \rightarrow A, C : bP \| Cert_B$
$C \rightarrow A, B : cP \| Cert_C$

Key Computation:

$A : K_A = e(yP, cP)^x \cdot e(bP, zP)^x \cdot e(yP, zP)^a \cdot e(bP, cP)^a$
$B : K_B = e(aP, zP)^y \cdot e(xP, cP)^y \cdot e(xP, zP)^b \cdot e(aP, cP)^b$
$C : K_C = e(aP, yP)^z \cdot e(xP, bP)^z \cdot e(xP, yP)^c \cdot e(aP, bP)^c$

The session key is $K_{ABC} = K_A = K_B = K_C = e(P, P)^{(xy)c+(xz)b+(yz)a+abc}$

Fig. 2. TAK-3 protocol of Al-Riyami and Paterson with forward secrecy [1]

A and B respectively. $\mathcal{A}$ can impersonate an honest user C by sending a message $c'P \| Cert_C$ for a known $c' \in \mathbb{Z}_q^*$. It can compute the same key that A and B computes with its knowledge of x, y and c' as $K' = e(yP, c'P)^x \cdot e(bP, zP)^x \cdot e(aP, zP)^y \cdot e(aP, bP)^{c'}$. It can now easily win the AKE-security game by selecting a test session at either A or B.

The key derivation of TAK-3 protocol is similar to the MTI/A0 protocol [23]. Al-Riyami and Paterson also proposed another tripartite variant TAK-4 whose key derivation is based on the MQV [22]. The two-party protocols MTI/A0 and the MQV are secure against KCI attacks. It is interesting to see that TAK-3 protocol is vulnerable to KCI attacks while TAK-4 protocol appears to resist them.

3.3 Bresson et al.'s Protocol [7]

Round 1:

1. Each U_i selects $x_i \xleftarrow{R} \mathbb{Z}_q^*$, computes $y_i = g_i^x$, $\alpha_i = y^{x_i}$ and $\sigma_1 = \mathcal{S}_{SK_i}(y_i)$
2. Each U_i sends (y_i, σ_i) to the server S

Round 2:

1. The base station first verifies all the incoming signatures.
2. It then computes $\alpha_i = y_i^x$, initializes a counter $c \in \{0, 1\}^{k_1}$ and the shared secret key $K = \mathcal{H}_0(c \| \alpha_1 \| \dots \| \alpha_n)$
3. It also computes for each party U_i, $K_i = K \oplus \mathcal{H}_1(c \| \alpha_i)$
4. S sends (c, K_i) to the user U_i for all $i \in [1, n]$

Key Computation:

1. Each user computes $K = K_i \oplus \mathcal{H}_1(c \| \alpha_i)$
2. The session key is computed by the server and the parties as $sk = \mathcal{H}(K \| \mathcal{G}_c \| S)$

Fig. 3. Bresson et al.'s GKE protocol [7]

In the protocol of Bresson et al. [7], a group of n parties computes a common session key with a mobile gateway S acting as a server. The system parameters are $(q, \mathbb{G}, g, \mathcal{H}, \mathcal{H}_0, \mathcal{H}_1, k_0, k_1)$, where q is a prime number chosen based on a security parameter k, $\mathbb{G}$ is a finite cyclic group of order q, g is an arbitrary generator of $\mathbb{G}$. The hash functions $\mathcal{H}, \mathcal{H}_0$ and $\mathcal{H}_1$ map to bit strings of length k, k_0 and k_1 respectively. The server is assumed to have a private-public key pair $(x, y = g^x)$ where $x \xleftarrow{R} \mathbb{Z}_q^*$. It also assumed to know the group of parties $\mathcal{G}_c$ with whom it communicates. Each party U_i is issued a private-public key pair (SK_i, PK_i) for a signature scheme $\Sigma = (\mathcal{K}_s, \mathcal{S}, \mathcal{V})$. The protocol execution is described in Figure 3.

We now show that the protocol in Figure 3 is not secure against KCI attacks as per Definition 1. If an adversary $\mathcal{A}$ obtains the long-term private key x of the server S, it can impersonate any honest user in $\mathcal{G}_c$

to S as follows: $\mathcal{A}$ simply replays a message (y_i, σ_i) of party U_i from an earlier successful execution of the protocol. The server sends back (c, K_i). $\mathcal{A}$ can compute the shared secret K as $K = K_i \oplus \mathcal{H}_1(c \| \alpha_i)$ where α_i is computed as y_i^x with its knowledge of the private key x. Thus it can win the AKE-security game by choosing a test session at S.

4 An insider secure GKE protocol

Bohli et al. [3] showed that the protocol of Kim et al. [20] was insecure in the presence of insiders and then modified the protocol as shown in Figure 4. The improved protocol was shown to satisfy their definitions of outsider and insider security. We briefly review the protocol here.

Let $\{U_1, \ldots, U_n\}$ be the set of parties who wish to establish a common group key. It is assumed that the parties are ordered in a logical ring with U_{i-1} and U_{i+1} being the left and right neighbors of U_i for $1 \leq i \leq n$, $U_0 = U_n$ and $U_{n+1} = U_1$. During the initialization phase, a cyclic group $\mathbb{G}$ of prime order q, an arbitrary generator g of $\mathbb{G}$ and the description of a hash function H that maps to $\{0,1\}^k$ are chosen. Each party is assumed to have a long-term private and public key pair for a public key signature scheme. Figure 4 outlines the execution of the protocol after initialization.

At a high level, the protocol in Figure 4 embeds the protocol of Boyd and González Nieto [6] in the first round of Burmester and Desmedt [12] (BD) protocol with the Katz and Yung [19] signature-based compiler applied. However, there are non-trivial and crucial changes done to the resulting protocol to enable it to achieve forward secrecy and contributiveness. As in the Boyd and González Nieto [6] (BG) protocol the parties choose their shares k_i's in the first round and all except one party send their shares in plain with the message broadcast in Round 1. Unlike the BG protocol, the nth user (or the distinguished user) sends only a commitment to its share instead of encrypting it with the long-term public keys of other users. The parties compute pair-wise CDH components using the y_i's sent in the first round similar to the BD protocol. These *ephemeral* values are used to encrypt the share of the distinguished user in the second round, which can be decrypted by the other users using the pair-wise CDH components they have computed. This enables the protocol to achieve forward secrecy unlike the BG protocol. The session key is finally computed in a way similar to the BG protocol using the shares from all the users, which guarantees contributiveness unlike the BD protocol [25]. The signature based authentication ensures security against impersonation attacks.

We now show that the protocol in Figure 4 is KCI resilient as per our new definitions and also contributory as per the definition of Bresson and Manulis [11].

Theorem 1. *The protocol in Figure 4 is AKE-secure as per Definition 1 assuming that the CDH assumption holds in $\mathbb{G}$, the signature scheme is UF-CMA secure and that H is a random oracle. The advantage of $\mathcal{A}_{ake}$ is upper bounded by*

$$2\left(n^2 \mathsf{AdvCMA}_{\Sigma} + \frac{(3q_s + q_r)^2}{2^k} + \frac{q_s^2}{2^k} + nq_sq_r\mathsf{SuccCDH} + \frac{q_sq_r}{2^k}\right)$$

where n is the number of participants, AdvCMA_{Σ} is the advantage of a polynomial adversary against the UF-CMA security of the signature scheme, $\mathsf{SuccCDH}$ is the probability of solving CDH in $\mathbb{G}$ and k is the security parameter. q_s and q_r are the upper bounds on the number of Send and random oracle queries respectively that $\mathcal{A}_{ake}$ can ask.

Proof. We give the proof in a sequence of games. Let S_i be the event that $\mathcal{A}_{ake}$ wins the AKE-security game in Game i and τ_i be the advantage of $\mathcal{A}_{ake}$ in Game i i.e. $\tau_i = |2 \cdot \Pr[S_i] - 1|$.

We use the following game hopping technique suggested by Dent [14] for indistinguishability games and recently used by Boyd et al. [5]. Consider an event E that may occur during $\mathcal{A}_{ake}$'s execution such that E is detectable by the simulator, E is independent of S_i, Game i and Game $i+1$ are identical unless E occurs, and $Pr[S_{i+1}|E] = \frac{1}{2}$. Then we have:

Round 1:

Computation

1. Each U_i chooses $k_i \xleftarrow{R} \{0,1\}^k$, $x_i \xleftarrow{R} \mathbb{Z}_q$ and computes $y_i = g^{x_i}$. U_n additionally computes $H(k_n)$
2. Each U_i except U_n sets $M_i^I = k_i \| y_i$, while U_n sets $M_n^I = H(k_n) \| y_n$
3. Each U_i computes a signature σ_i^I on $M_i^I \| \mathsf{pid}_i$.

Broadcast Each U_i broadcasts $M_i^I \| \sigma_i^I$.

Check Each U_i checks all signatures σ_j^I of incoming messages $M_j^I \| \sigma_j^I$ for $j \neq i$

Round 2:

Computation

1. Each U_i computes $t_i^L = H(y_{i-1}^{x_i})$, $t_i^R = H(y_{i+1}^{x_i})$, $T_i = t_i^L \oplus t_i^R$ and $\mathsf{sid}_i = H(\mathsf{pid} \| k_1 \| \dots \| k_{n-1} \| H(k_n))$. U_n additionally computes $\mathsf{mask}_n = k_n \oplus t_n^R$.
2. Each U_i except U_n sets $M_i^{II} = T_i \| \mathsf{sid_i}$ while U_n sets $M_n^{II} = \mathsf{mask}_n \| T_n \| \mathsf{sid_n}$
3. Each U_i computes a signature σ_i^{II} on M_i^{II}.

Broadcast Each U_i broadcasts $M_i^{II} \| \sigma_i^{II}$.

Check

1. Each U_i verifies the incoming the signatures σ_j^{II} on the corresponding message M_j^{II} for each $j \in [1,n]$ and $j \neq i$ also checks that $T_1 \oplus \cdots \oplus T_n \stackrel{?}{=} 0$ and $\mathsf{sid}_i \stackrel{?}{=} \mathsf{sid}_j$
2. Each U_i for $i < n$, extracts $k_n = \mathsf{mask}_n \oplus T_1 \oplus \cdots \oplus T_{i-1} \oplus t_i^L$ and checks the commitment $H(k_n)$ sent in Round 1 for the k_n extracted.

Key Computation

Each U_i computes the session key $sk_i = H(\mathsf{pid}_i \| k_1 \| \dots \| k_n)$

Fig. 4. GKE protocol of Bohli et al. [3]

$$\begin{aligned}
\Pr[S_{i+1}] &= \Pr[S_{i+1}|E]\Pr[E] + \Pr[S_{i+1}|\neg E]\Pr[\neg E] && (1)\\
&= \frac{1}{2}\Pr[E] + \Pr[S_i|\neg E]\Pr[\neg E] && (2)\\
&= \frac{1}{2}(1 - \Pr[\neg E]) + \Pr[S_i]\Pr[\neg E] && (3)\\
&= \frac{1}{2} + \Pr[\neg E]\left(\Pr[S_i] - \frac{1}{2}\right) && (4)\\
\text{Hence } \tau_{i+1} &= 2|\Pr[S_{i+1}] - \frac{1}{2}| = 2|\Pr[\neg E]\left(\Pr[S_i] - \frac{1}{2}\right)| && (5)\\
&= \Pr[\neg E]\tau_i && (6)
\end{aligned}$$

Game 0. This is the original AKE-security game as per the Definition 1. By definition we have

$$Adv_{\mathcal{A}_{ake}} = |2 \cdot \Pr[S_0] - 1| = \tau_0 \qquad (7)$$

Game 1. This is the same as the previous game except that the simulation fails if an event Forge occurs. Hence

$$|\Pr[S_1] - \Pr[S_0]| \leq \Pr[\mathsf{Forge}] \qquad (8)$$

$$\begin{aligned}
\tau_0 = |2 \cdot \Pr[S_0] - 1| &\leq |2 \cdot \Pr[S_0] - 2 \cdot \Pr[S_1]| + |2 \cdot \Pr[S_1] - 1| && (9)\\
&\leq 2\Pr[\mathsf{Forge}] + \tau_1 && (10)
\end{aligned}$$

The event Forge occurs when $\mathcal{A}_{ake}$ issues a Send query with a message of the form (M_i, σ_i) such that U_i is not corrupted and the message has previously not been an output of an instance at U_i. Note that in

a KCI attack, $\mathcal{A}_{ake}$ corrupts up to $n-1$ parties but it has to remain passive on behalf of the corrupted users. Hence Forge represents successful forgery of honest users' signatures.
If this event occurs we can use $\mathcal{A}_{ake}$ to forge a signature for a given public key in a chosen message attack as follows: The given public key is assigned to one of the n parties. All other parties are initialized as normal according to the protocol. All queries to the parties can be easily answered by following the protocol specification since all secret keys are known, except for the private key corresponding to the public key of the forgery attack game. In the latter case the signing oracle that is available as part of the chosen message attack can be used to simulate the answers.
The probability of $\mathcal{A}_{ake}$ not corrupting this party is $\geq \frac{1}{n}$. The probability of $\mathcal{A}_{ake}$ outputting a valid forgery on behalf of this user is also $\geq \frac{1}{n}$. Hence $\mathsf{AdvCMA}_{\Sigma} \geq \frac{1}{n^2} \cdot \Pr[\mathsf{Forge}]$. Rewriting the equation we have

$$\Pr[\mathsf{Forge}] \leq n^2 \cdot \mathsf{AdvCMA}_{\Sigma} \tag{11}$$

Game 2. This game is the same as the previous game except that the simulation fails if an event $\mathsf{Collision}$ occurs.

$$|\Pr[S_2] - \Pr[S_1]| \leq \Pr[\mathsf{Collision}] \tag{12}$$

$$\tau_1 = |2 \cdot \Pr[S_1] - 1| \leq |2 \cdot \Pr[S_1] - 2 \cdot \Pr[S_2]| + |2 \cdot \Pr[S_2] - 1| \tag{13}$$

$$\leq 2\Pr[\mathsf{Collision}] + \tau_2 \tag{14}$$

The event $\mathsf{Collision}$ occurs when the random oracle H produces a collision for any of its inputs. Each Send query requires at most 3 queries to the random oracle. Hence the total number of random oracle queries are bounded by $(3q_s + q_r)$. The probability of $\mathsf{Collision}$ is

$$\Pr[\mathsf{Collision}] \leq \frac{(3q_s + q_r)^2}{2^k} \tag{15}$$

Game 3. This game is the same as the previous game except that the simulation fails if an event Repeat occurs. Hence

$$|\Pr[S_3] - \Pr[S_2]| \leq \Pr[\mathsf{Repeat}] \tag{16}$$

$$\tau_2 = |2 \cdot \Pr[S_2] - 1| \leq |2 \cdot \Pr[S_2] - 2 \cdot \Pr[S_3]| + |2 \cdot \Pr[S_3] - 1| \tag{17}$$

$$\leq 2\Pr[\mathsf{Repeat}] + \tau_3 \tag{18}$$

The event Repeat occurs when an instance at a party U_i chooses a nonce k_i that was chosen by another instance at U_i. As there are a maximum q_s instances that may have chosen a nonce k_i, we have

$$\Pr[\mathsf{Repeat}] \leq \frac{q_s^2}{2^k} \tag{19}$$

Game 4. This game is the same as the previous game except that at the beginning of the game a value $\bar{s}$ is chosen at random in $\{1, \ldots, q_s\}$, where q_s is an upper bound to the maximum number of protocol sessions activated by the adversary. $\bar{s}$ represents a guess as to the protocol session in which the adversary is going to be tested. If the adversary does not choose the $\bar{s}^{th}$ session to ask the Test query, then the guess it wrong and the game is aborted.
The probability of aborting due to an incorrect choice of $\bar{s}$ is $1 - 1/q_s$. This event could be detected in the previous game if it also chose $\bar{s}$ in the same way. Therefore from Equation 6 we have

$$\tau_4 = \frac{1}{q_s}\tau_3 \implies \tau_3 = q_s\tau_4 \tag{20}$$

Game 5. This game differs from the previous game in how the Send queries are answered in the test session. Note that in this test session, the adversary is an outsider and moreover is passive with respect to all the parties. An active adversary producing valid signatures on behalf of uncorrupted parties would have caused Game 1 to halt.

In round 1 of the test session in Game 5, all messages y_i are chosen at random from $\mathbb{G}$. In round 2, all $t_i^R(= t_{i+1}^L)$ are assigned random values from $\{0,1\}^k$. All other computations are performed as in Game 4.

Since $H(\cdot)$ is modeled as a random oracle, the only way that any adversary can distinguish between Game 4 and 5 is if for at least one value of i it queries $y_i^{x_{i+1}}$ $(= y_{i+1}^{x_i})$ to the random oracle, where x_i and x_{i+1} are discrete logs of y_i and y_{i+1} respectively. Let Ask be such an event.

$$|\Pr[S_5] - \Pr[S_4]| \le \Pr[\mathsf{Ask}] \tag{21}$$

$$\begin{aligned} \tau_4 = |2 \cdot \Pr[S_4] - 1| &\le |2 \cdot \Pr[S_4] - 2 \cdot \Pr[S_5]| + |2 \cdot \Pr[S_5] - 1| && (22)\\ &\le 2\Pr[\mathsf{Ask}] + \tau_5 && (23)\end{aligned}$$

If Ask occurs, we can use $\mathcal{A}_{ake}$ to solve the CDH problem in $\mathbb{G}$. Given a CDH instance $(g, A = g^a, B = g^b)$, this can be plugged into a simulation of Game 5 as follows. Firstly, choose at random a party in the test session U_i. Then for the test session assign $y_i = A$ and $y_{i+1} = B$. If the event Ask occurs, the probability that a randomly chosen entry Z, from the random oracle table is a pair-wise CDH is at least $\frac{1}{q_r}$. Further, the probability of Z being the correct solution to the given instance (g, g^a, g^b) is at least $\frac{1}{n}$. Hence, $\mathsf{SuccCDH} \ge \frac{1}{nq_r}\Pr[\mathsf{Ask}]$. Rewriting the equation we have

$$\Pr[\mathsf{Ask}] \le nq_r\mathsf{SuccCDH} \tag{24}$$

Game 6. This game is the same as the previous game except that in the test session the game halts if $\mathcal{A}_{ake}$ asks a H-query with the corresponding input $(\mathsf{pid}_i\|k_1\|\ldots\|k_n)$.

Because the protocol messages in round 2 of the test session carry no information about k_n, the best any adversary can do is to guess k_n with a probability $\frac{1}{2^k}$. Hence, the probability that $\mathcal{A}_{ake}$ asks the right H-query for the test session is at most $\frac{q_r}{2^k}$.

$$|\Pr[S_6] - \Pr[S_5]| \le \frac{q_r}{2^k} \tag{25}$$

$$\begin{aligned} \tau_5 = |2 \cdot \Pr[S_5] - 1| &\le |2 \cdot \Pr[S_6] - 2 \cdot \Pr[S_6]| + |2 \cdot \Pr[S_6] - 1| && (26)\\ &\le 2\frac{q_r}{2^k} + \tau_6 && (27)\end{aligned}$$

If the adversary does not query the random oracle H on the correct input, then the adversary has no advantage in distinguishing the real session key from a random one and so

$$\tau_6 = 0.$$

By combining equations 7 to 25, we have the claimed advantage of $\mathcal{A}_{ake}$, which is negligible in k.

□

Theorem 2. *The protocol in Figure 4 satisfies mutual authentication as per Definition 2 assuming that the signature scheme is UF-CMA secure and that H is a random oracle. The advantage of $\mathcal{A}_{ma}$ is upper bounded by*

$$n^2 \cdot \mathsf{AdvCMA}_\Sigma + \frac{(3q_s + q_r)^2}{2^k} + \frac{q_s^2}{2^k}$$

where n is the number of participants, AdvCMA_Σ is the advantage of a polynomial adversary against the UF-CMA security of the signature scheme and k is the security parameter. q_s and q_r are the upper bounds on the number of Send *and random oracle queries respectively that $\mathcal{A}_{ma}$ can ask.*

Proof. We give the proof in a sequence of games. Let S_i be the event that $\mathcal{A}_{ma}$ violates the mutual authentication definition in Game i.

Game 0. This is the original mutual authentication game as per the Definition 2. By definition we have

$$Adv_{\mathcal{A}_{ma}} = \Pr[S_0] \tag{28}$$

Game 1. This game is the same as the previous game except that the simulation fails if an event Forge occurs, where Forge is the same event described in Game 1 of Theorem 1.

$$|\Pr[S_1] - \Pr[S_0] \leq \Pr[\mathsf{Forge}] \leq n^2 \cdot \mathsf{AdvCMA}_{\Sigma} \tag{29}$$

Game 2. This game is the same as the previous game except that the simulation fails if an event $\mathsf{Collision}$ occurs, where $\mathsf{Collision}$ is the same event described in Game 2 of Theorem 1.

$$|\Pr[S_2] - \Pr[S_1]| \leq \Pr[\mathsf{Collision}] \leq \frac{(3q_s + q_r)^2}{2^k} \tag{30}$$

Game 3. This game is the same as the previous game except that the game now aborts if an event Repeat occurs, where Repeat is the same event described in Game 3 of Theorem 1.

$$|\Pr[S_3] - \Pr[S_2]| \leq \Pr[\mathsf{Repeat}] \leq \frac{q_s^2}{2^k} \tag{31}$$

If Game 3 does not abort, all the honest partnered parties compute the same key. Hence, $\Pr[S_3] = 0$. By combining the Equations 28 to 31, we have the claimed advantage of $\mathcal{A}_{ma}$, which is negligible in k.

□

Theorem 3. *The protocol in Figure 4 satisfies contributiveness as per Definition 3 assuming that H is a random oracle. The advantage of $\mathcal{A}_{con}$ is upper bounded by*

$$\frac{q_s^2 + 2 \cdot q_r}{2^k}$$

where n is the number of participants and k is the security parameter. q_s and q_r are the upper bounds on the number of Send *and random oracle queries respectively that $\mathcal{A}_{ake}$ can ask.*

Proof. We give the proof in a sequence of games. Let S_i be the event that $\mathcal{A}_{con}$ violates the definition of contributiveness in Game i.

Game 0. This is the original game of contributiveness and as per the Definition 3. By definition we have

$$Adv_{\mathcal{A}_{con}} = \Pr[S_0] \tag{32}$$

Game 1. This game is the same as the previous game except that the simulation fails if an event Repeat occurs, where Repeat is the same event described in Game 3 of Theorem 1.

$$|\Pr[S_1] - \Pr[S_0]| \leq \Pr[\mathsf{Repeat}] \leq \frac{q_s^2}{2^k} \tag{33}$$

Game 2. This game is the same as the previous game except that the simulation fails if $\mathcal{A}_{con}$ can find a collision for the input k_n.

$$|\Pr[S_2] - \Pr[S_1]| \leq \frac{q_r}{2^k} \tag{34}$$

Game 3. This game is the same as the previous game except that the simulation fails if $\mathcal{A}_{con}$ finds a collision for the keying material input $(\mathsf{pid}_i \| k_1 \| \ldots \| k_n)$.

$$|\Pr[S_3] - \Pr[S_1]| \leq \frac{q_r}{2^k} \tag{35}$$

If Game 3 does not abort, the output of the random oracle is uniformly distributed. Hence, $\Pr[S_3] = 0$. By combining the equations 32 to 35, we have the claimed advantage of $\mathcal{A}_{con}$, which is negligible in k.

□

5 Conclusion

Table 1 gives a comparison of the security of some of the existing GKE protocols. The terms "AKE" refers to AKE-security, "AKE-FS" refers to AKE-security with forward secrecy and "AKE-KCIR" refers to AKE-security with KCI resilience. Similarly "MA" refers to mutual authentication and "MA-KCIR" refers to mutual authentication with KCI resilience. The entry "Yes*" says that the corresponding protocol appears to be secure under the notion but there is no formal proof. The last column in the table says whether the protocol is proven in the random oracle model or in the standard model.

It can be observed from the table that only the two-round protocol of Bohli et al. is proven to satisfy all the desired notions of security in the random oracle model. The two-round protocol obtained after applying the KC-compiler to the improved Boyd and González Nieto protocol appears to satisfy all the desired notions except forward secrecy. Another two-round protocol that appears to resist KCI attacks is that of Furukawa et al. Their protocol is proven in the universal composability framework without assuming random oracles. It is not known whether the protocol of Furukawa et al. satisfies contributiveness in the presence of insiders. The protocol of Bresson and Manulis appears to resist KCI attacks and their protocol is also proven secure in standard model. However, it has three rounds of communication.

	AKE	AKE-FS	AKE-KCIR	MA	MA-KCIR	Contributiveness	Model
Boyd and González Nieto [6] (BG)	Yes	No	No	No	No	honest	ROM
Katz and Yung [19]	Yes	Yes	Yes*	honest	honest	No	Std.
Bresson et al. [7]	Yes	Yes	No	honest	honest	unknown	ROM
BG Protocol + KS-compiler [18]	Yes	No	Yes*	Yes*	Yes*	Yes*	ROM
Bohli et al. [3]	Yes	Yes	Yes	Yes	Yes	Yes	ROM
Bresson and Manulis [11]	Yes	Yes	Yes*	Yes	Yes*	Yes	Std.
Furukawa et al. [16]	Yes	Yes	Yes*	Yes	Yes*	unknown	Std.

Table 1. Security comparison among existing GKE protocols

Our work models KCI attacks on GKE protocols in the presence of both outsiders and insiders. We have shown that there exist protocols which are not secure against KCI attacks. We have then shown that an existing protocol satisfies the new definitions. We hope that our work helps future protocols to be analyzed for KCI resilience.

We have not considered GKE protocols with special properties like robustness [15] and deniability [4]. Modeling KCI attacks on these types of protocols is an interesting open task. An open question is whether we can construct one-round AKE-secure GKE protocols that can have KCI resilience or forward secrecy. Constructing GKE protocols which do not use signature based authenticators seems to be another interesting problem.

Acknowledgment

This work has been supported by the Australian Research Council Discovery Project grant DP0773348.

References

1. Al-Riyami, S.S., Paterson, K.G.: Tripartite Authenticated Key Agreement Protocols from Pairings. In: Cryptography and Coding, 9th IMA International Conference. Volume 2898 of LNCS., Springer (2003) 332–359
2. Bellare, M., Rogaway, P.: Entity Authentication and Key Distribution. In: Advances in Cryptology–CRYPTO'93. Volume 773 of LNCS., Springer (1993) 232–249
3. Bohli, J.M., Gonzalez Vasco, M.I., Steinwandt, R.: Secure group key establishment revisited. Int. J. Inf. Sec. **6**(4) (2007) 243–254

4. Bohli, J.M., Steinwandt, R.: Deniable Group Key Agreement. In: Progressin Cryptology - VIETCRYPT 2006. Volume 4341 of LNCS., Springer (2006) 298–311
5. Boyd, C., Cliff, Y., González Nieto, J.M., Paterson, K.G.: Efficient One-Round Key Exchange in the Standard Model. In: Information Security and Privacy, 13th Australasian Conference, ACISP 2008. Volume 5107 of LNCS., Springer (2008) 69–83
6. Boyd, C., González Nieto, J.M.: Round-Optimal Contributory Conference Key Agreement. In: Public Key Cryptography–PKC'03. Volume 2567 of Lecture Notes in Computer Science., Springer (2003) 161–174
7. Bresson, E., Chevassut, O., Essiari, A., Pointcheval, D.: Mutual Authentication and Group Key Agreement for Low-Power Mobile Devices. In: Proc. of MWCN 03, World Scientific Publishing (October 2003) 5962
8. Bresson, E., Chevassut, O., Pointcheval, D.: Provably Authenticated Group Diffie-Hellman Key Exchange - The Dynamic Case. In: Advances in Cryptology–ASIACRYPT'01. Volume 2248 of LNCS., Springer (2001) 290–309
9. Bresson, E., Chevassut, O., Pointcheval, D.: Dynamic Group Diffie-Hellman Key Exchange under Standard Assumptions. In: Advances in Cryptology–EUROCRYPT'02. Volume 2332 of LNCS., Springer (2002) 321–336
10. Bresson, E., Chevassut, O., Pointcheval, D., Quisquater, J.J.: Provably authenticated group Diffie-Hellman key exchange. In: CCS'01: Proceedings of the 8th ACM conference on Computer and Communications Security, ACM (2001) 255–264
11. Bresson, E., Manulis, M.: Securing Group Key Exchange against Strong Corruptions. In: Proceedings of ACM Symposium on Information, Computer and Communications Security (ASIACCS'08), ACM Press (2008) 249–260
12. Burmester, M., Desmedt, Y.: A Secure and Efficient Conference Key Distribution System (Extended Abstract). In: EUROCRYPT. (1994) 275–286
13. Choo, K.K.R., Boyd, C., Hitchcock, Y.: Errors in computational complexity proofs for protocols. In Roy, B.K., ed.: Advances in Cryptology - ASIACRYPT 2005. Volume 3788 of LNCS., Springer (2005) 624–643
14. Dent, A.W.: A note on game-hopping proofs. Cryptology ePrint Archive, Report 2006/260 (2006) `http://eprint.iacr.org/`.
15. Desmedt, Y., Pieprzyk, J., Steinfeld, R., Wang, H.: A Non-malleable Group Key Exchange Protocol Robust Against Active Insiders. In: Information Security–ISC'06. Volume 4176 of LNCS., Springer (2006) 459–475
16. Furukawa, J., Armknecht, F., Kurosawa, K.: A Universally Composable Group Key Exchange Protocol with Minimum Communication Effort. In: Security and Cryptography for Networks–SCN 2008. Volume 5229 of LNCS., Springer (2008) 392–408
17. Joux, A.: A One Round Protocol for Tripartite Diffie-Hellman. In: Algorithmic Number Theory, 4th International Symposium. Volume 1838 of LNCS., Springer (2000) 385–394
18. Katz, J., Shin, J.S.: Modeling insider attacks on group key-exchange protocols. In: Proceedings of the 12th ACM Conference on Computer and Communications Security–CCS'05, ACM (2005) 180–189
19. Katz, J., Yung, M.: Scalable Protocols for Authenticated Group Key Exchange. In: Advances in Cryptology–CRYPTO'03. Volume 2729 of LNCS., Springer (2003) 110–125
20. Kim, H.J., Lee, S.M., Lee, D.H.: Constant-Round Authenticated Group Key Exchange for Dynamic Groups. In: Advances in Cryptology–ASIACRYPT'04. Volume 3329 of LNCS., Springer (2004) 245–259
21. Krawczyk, H.: HMQV: A High-Performance Secure Diffie-Hellman Protocol. In: Advances in Cryptology–CRYPTO'05. Volume 3621 of LNCS., Springer (2005) 546–566
22. Law, L., Menezes, A., Qu, M., Solinas, J.A., Vanstone, S.A.: An Efficient Protocol for Authenticated Key Agreement. Des. Codes Cryptography **28**(2) (2003) 119–134
23. Matsumoto, T., Takashima, Y., Imai, H.: On seeking smart public-keydistribution systems. Trans. IECE of Japan **E69** (1986) 99–106
24. Ng, E.M.: Security Models and Proofs for Key Establishment Protocols. Master's thesis, University of Waterloo (2005)
25. Pieprzyk, J., Wang, H.: Key Control in Multi-party Key Agreement Protocols. In: Workshop on Coding, Cryptography and Combinatorics (CCC 2003). Volume 23 of Progress in Computer Science and Applied Logic (PCS). (2003) 277–288