Modeling Key Compromise Impersonation Attacks on Group Key Exchange Protocols

M. CHOUDARY GORANTLA, Infosys Ltd., India
COLIN BOYD and JUAN MANUEL GONZÁLEZ NIETO, Queensland University of Technology, Australia
MARK MANULIS, TUDarmstadt and CASED, Germany

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, an important security attribute called *key compromise impersonation (KCI) resilience* has been completely ignored for the case of GKE protocols. Informally, a protocol is said to provide KCI resilience if the compromise of the long-term secret key of a protocol participant A does not allow the adversary to impersonate an honest participant B to A. In this paper, we argue that KCI resilience for GKE protocols is at least as important as it is for 2PKE protocols.

Our first contribution is revised definitions of security for GKE protocols considering KCI attacks by both outsider and insider adversaries. We also give a new proof of security for an existing two-round GKE protocol under the revised security definitions assuming random oracles. We then show how to achieve insider KCIR in a generic way using a known compiler in the literature. As one may expect, this additional security assurance comes at the cost of an extra round of communication. Finally, we show that a few existing protocols are not secure against outsider KCI attacks. The attacks on these protocols illustrate the necessity of considering KCI resilience for GKE protocols.

Categories and Subject Descriptors: C.2.0 [**Computer-Communication Networks**]: General—*Security and protection, data communications*; E.3 [**Data Encryption**]: *Public key cryptosystems*

General Terms: Security, Algorithms, Design, Theory

Additional Key Words and Phrases: Group key exchange, key compromise impersonation, insider attacks

ACM Reference Format:
Gorantla, M. C., Boyd, C., González Nieto, J. M., and Manulis, M. 2011. Modeling key compromise impersonation attacks on group key exchange protocols. ACM Trans. Info. Syst. Sec. 14, 4, Article 28 (December 2011), 24 pages.
DOI = 10.1145/2043628.2043629 http://doi.acm.org/10.1145/2043628.2043629

1. INTRODUCTION

A group key exchange (GKE) protocol allows a set of parties to agree upon a secret common session key by allowing the parties to exchange a few messages among themselves. In the real world, where the parties may communicate over a public channel like the Internet, it is prudent to assume the presence of an active adversary who can

A preliminary version of this work appeared in Gorantla et al. [2009a, 2009b].
M. C. Gorantla was partially supported by a Queensland University of Technology (QUT) write-up scholarship; M. Manulis was supported by grant MA 4957 of the German Research Foundation (DFG).
The majority of the work by M. C. Gorantla was done while he was a Ph.D. student at QUT.
Corresponding author's address: M. C. Gorantla, Security and Privacy Group, Infosys Labs, Infosys Ltd., Gachibowli, Hyderabad 500032, India; email: mc.gorantla@gmail.com.

DOI 10.1145/2043628.2043629 http://doi.acm.org/10.1145/2043628.2043629

modify, delete messages or even inject new messages during the protocol execution instead of just a passive eavesdropper. Security against active adversaries requires the usage of long-term keys that can authenticate the parties possessing them. In this article, we concentrate on GKE protocols that use long-term public-private keys for authentication.

Although there had been GKE protocols earlier [Ingemarsson et al. 1982; Steer et al. 1990; Burmester and Desmedt 1994, 1997; Steiner et al. 1996; Becker and Wille 1998], Bresson et al. [2001a, 2001b, 2002] were the first to analyze the security of GKE protocols under formal security models. All these models consider the security of a GKE protocol by allowing the parties to have multiple instances running distinct concurrent executions of the protocol. The security of GKE protocols is modeled by taking into account different adversarial actions (e.g., compromise of sensitive information or private keys, actively interfering with the protocol messages) during the run of the protocol and by defining the security goal of the protocol under such adversarial behavior.

The important security goals generally considered for GKE protocols are session key security and security against impersonation attacks. Informally, a protocol is said to provide session key security if no party other than the legitimate protocol participants have the knowledge of the session key. An adversary is said to impersonate a party B to another party A if B is honest (i.e., the adversary does not have access to the long-term private key of B and B runs the protocol as per the given specification) and the instance at A accepts the session with B as one of the session peers but there exists no such instance at B that accepts the session with A [Katz and Shin 2005].

The property of session key security for GKE protocol is typically modeled through a security notion called authenticated key exchange security (AKE-security) whereas impersonation attacks are covered by the notion of mutual authentication (MA-security). Since the session key cannot be kept confidential from a protocol participant, AKE-security makes sense only against an outsider adversary, that is, an adversary who is not one of the legitimate participants of the protocol run. In addition to considering AKE-security, the models of Bresson et al. [2001a, 2001b, 2002] also consider mutual authentication in the presence of an outsider adversary. Informally, mutual authentication requires that (1) the parties who complete the protocol execution should output identical session keys and (2) each party should be ensured of the identities of the other parties acting in the protocol. Note that the requirement (2) essentially means there should be no impersonation attacks.

Katz and Shin [2005] were the first to model the security for GKE protocols against insiders that is, malicious protocol participants. In particular, they defined a notion of security against insider impersonation attacks. Note that we cannot prevent an insider from knowing the value of the session key since an insider is a legitimate protocol participant (albeit with malicious intentions). The definition of Katz and Shin [2005] has been revisited by Bohli et al. [2007] and Bresson and Manulis [2008] under different corruption models. Specifically, Bohli et al. consider only the *weak* corruption model in which the adversary is allowed to reveal only the long-term key of a party whereas Bresson and Manulis consider a *strong* corruption model in which sensitive session-specific information used in computing the session key of an instance of a party may also be revealed in addition to the long-term private key of the party. An additional notion considered by Bohli et al. is *contributiveness*, which requires that a proper subset of insiders should not predetermine the resulting session key. Bresson and Manulis strengthened this notion by considering the strong corruption model. In this work, we concentrate on AKE-security and MA-security notions but do not explore the notion of contributiveness.

All the models discussed till now (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 has been compromised. These models also consider forward secrecy to limit the damage done by compromise of the long-term private key of a party after session completion. Another equally important security attribute related to compromise of the long-term private keys of parties is key compromise impersonation resilience (KCIR). In a successful key compromise impersonation (KCI) attack, an adversary with the knowledge of the long-term private key of a party A can impersonate an honest party B to A. Resilience to KCI attacks is often seen as a desired security attribute for 2PKE [Law et al. 2003; Krawczyk 2005]. However, it has so far been ignored for GKE protocols. In this article, we model KCI attacks by outsider and insider adversaries on GKE protocol. Specifically, we extend the notion of AKE-security by considering KCI resilience against outsider adversaries and MA-security by considering KCI resilience against both outsider and insider adversaries. As specified above, MA-security covers impersonation attacks. Hence, our new notion of MA-security with KCIR in the presence of insiders covers insider impersonation attacks.

1.1. The Importance of KCIR for GKE Protocols

Just and Vaudenay [1996] were the first to consider KCI attacks for key exchange protocols. They highlighted the importance of KCIR for 2PKE protocols in a scenario wherein a rogue engineer setting up an automated teller machine (ATM) obtains the long-term private key that is to be stored at the ATM terminal. By launching a KCI attack, the engineer would be able to impersonate an honest user (e.g., a customer or a data center terminal) and establish a session key with the ATM terminal. This allows the engineer to compromise any sensitive information that may be encrypted using that session key. Note that in this scenario, the engineer does not need to have access to the ATM terminal after the setup.

Many 2PKE protocols have been later analyzed under KCIR, and KCI attacks have also been modeled formally in the security models for 2PKE protocols [Krawczyk 2005; LaMacchia et al. 2007; Ustaoglu 2008]. However, KCIR so far has been neither modeled nor considered as a desirable attribute for GKE protocols. It is worth noting that the probability of a party's long-term private key getting compromised is higher in a GKE protocol than it is in a 2PKE protocol since the number of active participants in a GKE protocol is more than two. Hence, KCIR seems to be more important for GKE protocols than it is for 2PKE protocols. Manulis [2007, Section 6.1.2] argued that KCIR would be important only for GKE protocols that establish session keys which could be further used for authentication purpose only. We now illustrate two more scenarios with different setup assumptions to highlight the importance of KCIR for any GKE protocol irrespective of the purpose of establishing the session key.

We first extend the peer-to-peer file sharing system scenario given for the two-party case by Ng [2005, Section 4.2.2] to the group case. In these systems, each user stores some data and allows access only to users whom it wants to share the data with. This can be achieved by first executing a GKE protocol with the peers who have read access to the data and then 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 play the role of A during the protocol, $\mathcal{A}$ may not be able to access the locally stored data of A. However, if the GKE protocol does not provide KCIR, $\mathcal{A}$ can launch a KCI attack by compromising the long-term private key of A, impersonating a party who has read access to the data at A and then computing the session key with A. $\mathcal{A}$ can now trivially decrypt the data sent by A using the session key. Note that in this scenario, the GKE protocol needs to have forward secrecy to achieve KCIR. 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 since $\mathcal{A}$ will be able to spoof the presence of an honest party if the protocol is not KCI resilient.

The second scenario is a server-client application given by Bresson et al. [2003]. They proposed a GKE protocol which would allow a cluster of mobile devices (acting as clients) to agree upon a session key with a wireless gateway (acting as a server). The authors suggested that the established session key could 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 play the role of the gateway and allow any mobile device to access the wireless network. However, if the adversary allows illegitimate clients to connect to the gateway it 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. [2003] in Section 6.

1.2. Contributions

1.2.1. Outsider KCIR. We first define what we mean by an outsider adversary. We call a *party* corrupted if the long-term private key of the party is compromised, while a *party instance* is called corrupted if the secret ephemeral session state of that instance is revealed. An adversary $\mathcal{A}$ is called an outsider to a particular protocol session if any secret information specific to that session is not revealed. Note an outsider adversary $\mathcal{A}$ may compromise the long-term private keys of the parties but it is allowed neither to corrupt the instances of any of the parties associated to the protocol session nor to participate in the protocol session on behalf of the corrupted parties (i.e., it is not allowed to execute the party instances with its knowledge of the corresponding long-term private keys of the corrupted parties).

In an outsider KCI attack scenario for a GKE protocol, an adversary $\mathcal{A}$ is allowed to compromise the long-term private keys of all parties except one. $\mathcal{A}$ is considered successful in mounting a KCI attack if it can impersonate an uncorrupted instance of an uncorrupted party to an uncorrupted instance of any of the corrupted parties. One of the goals of $\mathcal{A}$ by launching a KCI attack as an outsider is to break the confidentiality of the established session key. Hence, we modify an existing definition of AKE-security taking KCI attacks into account. Since AKE-security concerns only about outsider adversaries, we simply write *AKE-security with KCIR* to refer to AKE-security with KCIR against outsider adversaries.

Another goal of an outsider adversary is to violate the notion of MA-security. Thus, we enrich the security notions for GKE by embedding a definition of outsider KCIR into MA-security. We demonstrate the usefulness of the new notion by showing that the generic transformation of Bresson et al. [2001a] to achieve outsider MA-security is not sufficient to attain MA-security with outsider KCIR.

We also highlight a separation between the two outsider KCIR notions with an example protocol. Katz and Shin [2005] informally mentioned that their generic compiler (described in Figure 2) could provide KCIR. In our earlier work [Gorantla et al. 2009a], we too speculated that the Katz-Shin compiler (KS-compiler) when applied to the Boyd-González Nieto (BG) protocol [Boyd and González Nieto 2003] would result in a GKE protocol secure under both AKE-security and MA-security with KCIR. However, we show that the KS-compiler can enhance only MA-security to MA-security with outsider KCIR but not AKE-security the same way. Specifically, there exists AKE-secure GKE protocols for which the respective KS-compiled versions do not achieve AKE-security with KCIR.

1.2.2. Insider KCIR. A party is called an insider if the adversary corrupts the party and actively participates in a protocol session on behalf of that party. In an insider KCI attack scenario, $\mathcal{A}$ is considered successful if it impersonates an uncorrupted instance of an uncorrupted party B to an uncorrupted instance at a party A in the presence of insiders. 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. Note that it is impossible to prevent an insider adversary $\mathcal{A}$ from learning the session key. Hence, we consider violation of MA-security as the only goal of $\mathcal{A}$. We revise an existing definition of MA-security considering KCI attacks by insider adversaries. It is straightforward to see that MA-security with insider KCIR implies MA-security with outsider KCIR. However, from the separation between the outsider KCIR notions discussed earlier, MA-security with insider KCIR does not imply AKE-security with (outsider) KCIR. Hence, we have the following relations among the AKE-security and MA-security notions for GKE protocols.

$$\text{AKE-security with KCIR} \Longrightarrow \text{AKE-security}$$

$$\begin{tikzcd}
\text{MA-security with insider KCIR} \arrow[r, Rightarrow] \arrow[d, Rightarrow] & \text{MA-security with outsider KCIR} \\
\text{MA-security} &
\end{tikzcd}$$

In this figure, the KCIR notions are the ones proposed in this paper while the AKE-security and MA-security notion are from Bresson and Manulis [2008]. Note that the MA-security notion of Bresson and Manulis [2008] covers insider impersonation attacks. Hence, from the above relation our insider KCIR notion also covers insider impersonation attacks.

It seems that a protocol needs to have at least two rounds to satisfy AKE-security with KCIR. As we discussed earlier, forward secrecy seems to be a necessary condition to achieve AKE-security with KCIR. Since we do not know of any one-round GKE protocol with forward secrecy, we need at least two rounds to achieve AKE-security with KCIR. Intuitively, satisfying MA-security needs at least two rounds. In a one-round protocol, the parties cannot get the assurance that the other parties have actually participated in the protocol. Furukawa et al. [2008] formally established a lower bound of two rounds for GKE protocols that satisfy MA-security. Although their analysis is in the universal composability (UC) framework [Canetti 2001], the lower bound seems to apply for the game-based definitions as well. We show that the two-round GKE protocol of Bohli et al. [2007] satisfies the notions of AKE-security with KCIR and also MA-security with insider KCIR. We also give a generic proof of security showing that the KS-compiler can be used in a generic way to achieve MA-security with insider KCIR.

1.2.3. KCI Attacks on Existing GKE Protocols. Choo et al. [2005] presented unknown key share attacks on the BG protocol and then suggested an improvement to the protocol. We show that the improved BG protocol does not satisfy our notion of AKE-security with KCIR. We also present KCI attacks on the tripartite key agreement protocol TAK-3 of Al-Riyami and Paterson [2003] and the GKE protocol of Bresson et al. [2003].

Specific contributions of this article are:

(1) New outsider and insider security notions modeling KCI attacks on GKE protocols;
(2) A proof of security in the new model (both AKE-security and MA-security notions with KCIR) for the protocol of Bohli et al. [2007] in the random oracle model;
(3) A generic proof of security showing that the KS-compiler guarantees MA-security with insider KCIR;

(4) KCI attacks on the protocols of Boyd and González Nieto [2003], Al-Riyami and Paterson [2003], and Bresson et al. [2003].

Outline. We describe some background concepts in Section 2. In Section 3, we present new notions of AKE-security and MA-security considering KCI attacks by outsider and insider adversaries. In Section 4, we show that the protocol of Bohli et al. [2007] satisfies the new notions of AKE-security and MA-security. We show a generic way for any AKE-secure protocol to achieve MA-security with insider KCIR at an additional round of communication in Section 5. Section 6 presents KCI attacks on the improved BG protocol, Al-Riyami and Paterson's protocol and Bresson et al.'s protocol. Finally, we conclude the article in Section 7 with a comparison among existing GKE protocols.

2. PRELIMINARIES

We now describe a computational assumption and a cryptographic tool that will be used in the proofs of GKE protocols later in the paper.

2.1. Negligible Function

Definition 2.1 (*Negligible [Goldreich 2001]*). We call a function $\mu : \mathbb{N} \to \mathbb{R}$ negligible if for every positive polynomial $p(\cdot)$ there exists an N such that for all $k > N$

$$\mu(k) < \frac{1}{p(k)}$$

Informally, an event is negligible in a variable k if it happens with a probability that is asymptotically less than the inverse of any polynomial in k. The advantage of an adversary against a hard problem or a cryptographic algorithm is generally computed as a function of the given security parameter k. A typical security parameter for a cryptographic algorithm is the length of the key employed by it.

2.2. Computational Diffie Hellman Assumption

Let $\mathbb{G}$ be a cyclic group of prime order p and let g be an arbitrary generator of $\mathbb{G}$. The Computational Diffie Hellman (CDH) problem is to compute g^{ab} given a random instance (g, g^a, g^b) for $a, b \in \mathbb{Z}_p$.

We say that the CDH assumptions hold in $\mathbb{G}$ if for all probabilistic polynomial time (PPT) algorithms, the probability of solving the CDH problem is negligible in the security parameter k (e.g. order of the group).

2.3. Digital Signature

Goldwasser et al. [1988] first defined a formal notion of security for digital signatures, called *existential unforgeability under chosen message attack* (UF-CMA). We now review the definition of digital signature and its formal security notion.

A digital signature scheme consists of three algorithms:

KeyGen is a PPT algorithm that takes as input the security parameter k and outputs a public-private key pair (pk, sk). The public key also specifies the message space $\mathcal{M}$ and a signature space $\mathcal{S}$ which define the set of all messages that can be submitted to the Sign algorithm and the set of all signatures that can be submitted to the Verify algorithm respectively.

Sign is an algorithm which takes as input $m \in \mathcal{M}$ and a secret key sk. It outputs a signature $\sigma \in \mathcal{S}$. The Sign algorithm could be either deterministic or probabilistic.

Verify is a deterministic algorithm that takes as input a message-signature pair (m, σ) and a public key pk and outputs a boolean value: true or false. The given message-signature pair (m, σ) is said to be valid if Verify outputs true.

It is necessary for any digital signature scheme to have the correctness property i.e., for all valid key pairs (pk, sk), if $\sigma = \mathsf{Sign}(m, sk)$ for any $\sigma \in \mathcal{S}$, then $\mathsf{Verify}(m, \sigma, pk) = \mathsf{true}$.

We now describe the notion of UF-CMA for digital signature schemes. Informally, UF-CMA security for signature schemes requires that an adversary cannot produce a signature on a message that has not been given as input to the signing algorithm.

Definition 2.2. A digital signature scheme is UF-CMA secure if the advantage of any PPT adversary in the following game is negligible in k, where k is the security parameter.

Setup. The challenger runs the KeyGen algorithm and obtains a key pair (pk, sk). The public key pk is given to the adversary.
Sign Queries. The adversary issues these queries with input messages $m_1, \ldots, m_q$, for a q polynomial in k. Each sign query is replied to by executing the Sign algorithm on the input message. The queries may be asked adaptively that is, the message m_i is allowed to be selected after obtaining the responses to messages $m_1, \ldots, m_{i-1}$.
Forgery. Eventually, the adversary outputs a signature (m, σ). It wins the UF-CMA game if (1) σ is a valid signature on m under pk and (2) m has not been an input to any of the sign queries i.e., $m \notin \{m_1, \ldots, m_q\}$.

The advantage of an adversary in the UF-CMA game is defined to be the same as its probability of success in the game.

3. SECURITY MODEL

We describe the communication model in which the parties would execute a GKE protocol, the adversarial model which describes the actions that an adversary is allowed to perform and new security notions considering both outsider and insider KCI attacks.

3.1. Communication Model

Let $\mathcal{U} = \{U_1, \ldots, U_n\}$ be a set of n parties. The protocol π may be run among any subset of these parties. Each party U_i for $i \in [1, n]$ is assumed to have a pair of long-term public and private keys, (pk_i, sk_i) generated during an initialization phase prior to the protocol run. A GKE protocol π executed among $\tilde{n} \leq n$ users is modeled as a collection of $\tilde{n}$ programs running at the $\tilde{n}$ distinct parties in $\mathcal{U}$. Each party may have multiple instances running concurrent sessions of the protocol.

Let π_i^j be the instance at a party $U_i \in \mathcal{U}$ for j-th run of the protocol π. Each session of the protocol is identified by a unique session ID. We assume that the session ID is derived during the run of the protocol for example, as concatenation of the messages exchanged among all the parties along with their identities. The session ID of an instance π_i^j is denoted by sid_i^j. We assume that each party knows who the other participants are for each protocol run. The partner ID pid_i^j of an instance π_i^j, is a set of identities of the parties whom π_i^j wishes to establish a common group key with. Note that pid_i^j includes the identity of U_i itself.

An instance π_i^j enters an *accepted* state when it computes a session key sk_i^j. 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_i^j and $\pi_{i'}^{j'}$ at two different parties U_i and $U_{i'}$ respectively are considered *partnered* iff (1) both the instances have accepted, (2) $\mathsf{sid}_i^j = \mathsf{sid}_{i'}^{j'}$ and (3) $\mathsf{pid}_i^j = \mathsf{pid}_{i'}^{j'}$.

3.2. Adversarial Model

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. It can also start multiple new instances at any of the parties, modeling the ability of the parties to engage in many sessions simultaneously. $\mathcal{A}$ is considered passive if it faithfully forwards the protocol messages among all the participants without any modifications.

In addition to controlling the message transmission, $\mathcal{A}$ is allowed to ask the following queries.

—Execute(pid). prompts a complete execution of the protocol among the parties in pid. Note that pid here is a set of parties among whom $\mathcal{A}$ wishes to initialize a protocol execution. $\mathcal{A}$ is given all the protocol messages, modeling passive attacks.
—Send(π_i^j, m). sends a message m to the instance π_i^j. If the message is only pid, the instance π_i^j is initiated with partner ID pid. The response of π_i^j to any Send query is returned to $\mathcal{A}$.
—RevealKey(π_i^j). If π_i^j has accepted, $\mathcal{A}$ is given the session key sk_i^j established at π_i^j.
—Corrupt(U_i). The long-term secret key sk_i of U_i is returned to $\mathcal{A}$. Note that this query returns neither the session key (if already computed) nor the internal state.
—RevealState(π_i^j). The internal state of U_i is returned to $\mathcal{A}$. We assume that the internal state is erased once π_i^j has accepted. Hence, a RevealState query to an accepted instance returns nothing.
—Test(π_i^j). A random bit b is secretly chosen. If $b = 1$, $\mathcal{A}$ is given $K_1 = sk_i^j$ established at π_i^j. Otherwise, a random value K_0 chosen from the session key probability distribution is given. Note that $\mathcal{A}$ is allowed to issue the Test query only once during its execution. The query can only be issued on an accepted instance.

Definition 3.1 (*Correctness*). A GKE protocol is called *correct* if the instance π_i^j and its partnered instances at parties in pid_i^j output identical session keys in the presence of a passive adversary.

Remark 3.2. The earlier security models for GKE have assumed that if a party's long-term key is compromised, then the party will be completely under the control of the adversary [Bresson et al. 2001a, 2001b, 2002; Katz and Shin 2005; Bohli et al. 2007; Bresson and Manulis 2008]. However, in some scenarios the adversary may not be able to actively control the party after obtaining its long-term key. As illustrated in Section 1.1, this can happen for a variety of reasons for instance, the adversary no longer has access to the terminal or such an activity can be detected. Hence, in our adversarial model we assume that even if a party has been issued a Corrupt query, it is not under the control of the adversary that is, the corrupted party still follows the protocol and does computations as per the protocol specification. Our modelling of Corrupt query is similar to LaMacchia et al.'s [2007] model for 2PKE protocol. We now describe some of the other differences with the earlier GKE security models.

3.2.1. Corrupted Parties, Corrupted Instances. We call a *party* U_i corrupted if it has been issued a Corrupt query, while a *party instance* π_i^j is called corrupted if a RevealState(π_i^j) query has been asked. Note that there exists an uncorrupted instance π_i^j at a corrupted party U_i when U_i has been corrupted but π_i^j has not.

3.2.2. Insiders. A party U_i is called an *insider* to a particular protocol run if both the party U_i and the instance π_i^j are corrupted **or** if the adversary issues a Corrupt(U_i)

query and then plays the role of U_i in the instance π_i^j, that is, either the output of Send queries issued to π_i^j are not computed as per the protocol specification or the output of Send queries issued to π_i^j are modified while issuing Send queries to other instances of parties in pid_i^j.

Note that the security models for GKE protocols so far have not distinguished between corrupted parties and insiders. In these models, a party was deemed an insider if it had been the subject of a $\mathsf{Corrupt}$ query.

3.3. 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 execute the protocol honestly.

We now define the notion of freshness considering KCI attack scenarios and based on a corresponding notion for 2PKE given by Krawczyk [2005].

Definition 3.3 (*Freshness*). An instance π_i^j is called fresh if the following conditions hold:

(1) the instance π_i^j or any of its partners have not been asked a $\mathsf{RevealKey}$ after their acceptance;
(2) the instance π_i^j or any of its partners have not been asked a $\mathsf{RevealState}$ before their acceptance;
(3) If $\pi_{i'}^{j'}$ is a partner of π_i^j and $\mathcal{A}$ has asked $\mathsf{Corrupt}(U_{i'})$ query, then only pid_i^j or the output of a Send query to $\pi_{i'}^{j'}$ has been used as input to the Send queries issued to π_i^j.

Note that the last condition implies that the adversary is not allowed to modify messages from the instance $\pi_{i'}^{j'}$ when the party $U_{i'}$ has been corrupted.

Definition 3.4 (*AKE-security with KCI resilience*). An adversary $\mathcal{A}_{\mathsf{ake}}$ against the AKE-security notion is allowed to make $\mathsf{Execute}$, Send, $\mathsf{RevealState}$, $\mathsf{RevealKey}$ and $\mathsf{Corrupt}$ queries in Stage 1. $\mathcal{A}_{\mathsf{ake}}$ makes a single Test query to an instance π_i^j at the end of Stage 1 and is given a challenge key K_b as described in Section 3.2. It can continue asking queries in Stage 2. Finally, $\mathcal{A}_{\mathsf{ake}}$ outputs a bit b' and wins the AKE-security game if (1) $b' = b$ **and** (2) the instance π_i^j that is asked Test query remains fresh till the end of $\mathcal{A}_{\mathsf{ake}}$'s execution. Let $\mathsf{Succ}_{\mathcal{A}_{\mathsf{ake}}}$ be the event that $\mathcal{A}_{\mathsf{ake}}$ wins the AKE-security game. The advantage of $\mathcal{A}_{\mathsf{ake}}$ in winning this game is $\mathsf{Adv}_{\mathcal{A}_{\mathsf{ake}}} = |2 \cdot \Pr[\mathsf{Succ}_{\mathcal{A}_{\mathsf{ake}}}] - 1|$. A protocol is called AKE-secure with KCIR if $\mathsf{Adv}_{\mathcal{A}_{\mathsf{ake}}}$ is negligible in the security parameter k for any PPT $\mathcal{A}_{\mathsf{ake}}$.

The definition of freshness takes care of the KCI attacks as it does allow $\mathcal{A}_{\mathsf{ake}}$ to corrupt the owner of the test session. Note that if the adversary is active with respect to a partner $\pi_{i'}^{j'}$ to the test instance π_i^j (i.e., modifies the output of any Send queries issued to $\pi_{i'}^{j'}$), then that party cannot have been corrupted; otherwise π_i^j is not fresh. The definition also takes forward secrecy into account as it allows $\mathcal{A}_{\mathsf{ake}}$ to obtain the long-term private keys of all the parties. In this case, as per the definition $\mathcal{A}_{\mathsf{ake}}$ must not modify the outputs of instances of parties in pid_i^j.

Remark 3.5. It is clear that if a GKE protocol does not have forward secrecy, the AKE-security of the session key can be compromised by revealing the long-term key of a protocol participant. An adversary can perform a KCI attack on GKE protocols without forward secrecy by replaying messages of past successful executions or even by relaying messages from an honest party. The KCI attacks we present on Boyd and González Nieto [2003] and Bresson et al. [2003] protocols work in the same way. As the AKE-security notion with KCIR implies that at most $n-1$ corruptions are allowed, it is necessary for a protocol realizing this notion to have at least partial forward secrecy when $n-1$ parties are corrupted or full forward secrecy (Please see Gorantla [2010, Chapter 4] for definitions on forward secrecy for GKE). However, as will be evident by our attack on Al-Riyami and Paterson's protocol [Al-Riyami and Paterson 2003], having forward secrecy alone is not sufficient for a GKE protocol to have AKE-security with KCIR.

3.4. Mutual Authentication

We now strengthen the definition of Bresson and Manulis [2008] by considering KCI attacks by both outsiders and insiders.

Definition 3.6 (*MA-security with Outsider KCIR*). An adversary $\mathcal{A}_{ma}$ against the MA-security of a correct GKE protocol π is allowed to ask Execute, Send, RevealState, RevealKey, and Corrupt queries. $\mathcal{A}_{ma}$ violates the MA-security with outsider KCIR notion of the GKE protocol if at some point during the protocol run, there exists an uncorrupted instance π_i^j (although the party U_i may be corrupted) that has accepted with a key sk_i^j and another party $U_{i'} \in \mathsf{pid}_i^j$ that is uncorrupted at the time π_i^j accepts such that there are no insiders in pid_i^j and

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

Let $\mathsf{Succ}_{\mathcal{A}_{ma}}$ be the success probability of $\mathcal{A}_{ma}$ in winning the above security game. A protocol is said to provide MA-security with outsider KCIR if $\mathsf{Succ}_{\mathcal{A}_{ma}}$ is negligible in the security parameter k for any PPT $\mathcal{A}_{ma}$.

The preceding definition implies that $\mathcal{A}_{ma}$ must be passive for any corrupted party in pid_i^j. Otherwise $\mathcal{A}_{ma}$ would be considered an insider as per the definition of insider in Section 3.2.2. Note that in a protocol execution with n parties, the above definition also implies that $\mathcal{A}_{ma}$ is allowed to corrupt up to $n-1$ parties.

Definition 3.7 (*MA-security with Insider KCIR*). An adversary $\mathcal{A}_{ma}$ against MA-security of a correct GKE protocol π is allowed to ask Execute, Send, RevealState, RevealKey and Corrupt queries. $\mathcal{A}_{ma}$ violates the MA-security with insider KCIR notion of the GKE protocol if at some point during the protocol run, there exists an uncorrupted instance π_i^j (although the party U_i may be corrupted) that has accepted with a key sk_i^j and another party $U_{i'} \in \mathsf{pid}_i^j$ that is uncorrupted at the time π_i^j accepts such that

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

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

The only difference between Definition 3.6 and Definition 3.7 is that the former does not allow the presence of insiders while the latter does. Similarly, the difference

Round 1.

Computation.

(1) Each U_i chooses $k_i \xleftarrow{R} \{0,1\}^k$, $x_i \xleftarrow{R} \mathbb{Z}_p$ and computes $y_i = g^{x_i}$. $U_{\tilde{n}}$ additionally computes $H(k_{\tilde{n}})$.

(2) Each U_i except $U_{\tilde{n}}$ sets $M_i^I = k_i \| y_i$, while $U_{\tilde{n}}$ sets $M_{\tilde{n}}^I = H(k_{\tilde{n}}) \| y_{\tilde{n}}$.

(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 $\mathsf{sid}_i = H(\mathsf{pid} \| k_1 \| \dots \| k_{\tilde{n}-1} \| H(k_{\tilde{n}}))$. $U_{\tilde{n}}$ additionally computes $\mathsf{mask}_{\tilde{n}} = k_{\tilde{n}} \oplus t_{\tilde{n}}^R$.

(2) Each U_i except $U_{\tilde{n}}$ sets $M_i^{II} = T_i \| \mathsf{sid_i}$ while $U_{\tilde{n}}$ sets $M_{\tilde{n}}^{II} = \mathsf{mask}_{\tilde{n}} \| T_{\tilde{n}} \| \mathsf{sid}_{\tilde{n}}$.

(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 $j \in [1, \tilde{n}]$ and $j \neq i$ also checks that $T_1 \oplus \cdots \oplus T_{\tilde{n}} \overset{?}{=} 0$ and $\mathsf{sid}_i \overset{?}{=} \mathsf{sid}_j$.

(2) Each U_i for $i < \tilde{n}$, extracts $k_{\tilde{n}} = \mathsf{mask}_{\tilde{n}} \oplus T_1 \oplus \cdots \oplus T_{i-1} \oplus t_i^L$ and checks the commitment $H(k_{\tilde{n}})$ sent in Round 1 for the $k_{\tilde{n}}$ extracted.

Key Computation.

. Each U_i computes the session key $\kappa_i = H(\mathsf{pid}_i \| k_1 \| \dots \| k_{\tilde{n}})$.

Fig. 1. GKE Protocol of Bohli et al.

between Definition 3.7 and the definition of MA-security by Bresson and Manulis [2008] 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 the MA-security as per Definition 3.7.

Our definition is stronger than the earlier definitions [Katz and Shin 2005; Bohli et al. 2007; Bresson and Manulis 2008], since it captures the additional goal of KCIR in the presence of insiders by giving the adversary extra power. This implies that if a protocol does not satisfy earlier definitions, it also cannot satisfy Definition 3.7.

4. AN INSIDER SECURE GKE PROTOCOL

Kim et al. [2004] presented a two-round GKE protocol based on the Burmester-Desmedt protocol [Burmester and Desmedt 1994]. Bohli et al. [2007] showed that the protocol of Kim et al. [2004] was insecure in the presence of insiders and then modified the protocol to present the one shown in Figure 1. The improved protocol of Bohli et al. was shown to satisfy their definitions of outsider and insider security. We briefly review the protocol here.

Let $\{U_1, \dots, U_{\tilde{n}}\}$ 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 $1 \leq i \leq \tilde{n}$, $U_0 = U_{\tilde{n}}$ and $U_{\tilde{n}+1} = U_1$. During the initialization phase, a cyclic group $\mathbb{G}$ of prime order p, 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 1 outlines the execution of the protocol after the initialization phase.

At a high level, the protocol in Figure 1 embeds the BG protocol in the first round of Burmester and Desmedt [1994] (BD) protocol. However, there are nontrivial and crucial changes done to the resulting protocol to enable it to achieve forward secrecy.

As in the 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 $\tilde{n}^{th}$ user (or the distinguished user) sends only a commitment to its share instead of encrypting it with the long-term public keys of the other users. The parties compute pairwise 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 pairwise 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. The signature based authentication ensures security against impersonation attacks.

We now show that the protocol in Figure 1 is KCI resilient as per our new definitions.

THEOREM 4.1. *The protocol in Figure 1 is AKE-secure as per Definition 3.4 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 $\mathcal{A}_{\mathsf{ake}}$ is upper bounded by*

$$2\left(n \cdot Adv_{\mathcal{A}^{\mathsf{cma}}} + \frac{(3q_e + 3q_s + q_r)^2 + (q_e + q_s)^2 + (q_e + q_s)q_r}{2^k} + \tilde{n}(q_e + q_s)q_r \mathsf{SuccCDH}\right)$$

where $\tilde{n} \leq n$ is the number of participants, n is the total number of public keys in the system, $Adv_{\mathcal{A}^{\mathsf{cma}}}$ is the advantage of a PPT adversary $\mathcal{A}^{\mathsf{cma}}$ against the UF-CMA security of the signature scheme, SuccCDH *is the probability of solving the CDH in $\mathbb{G}$ and k is the security parameter. q_e, q_s and q_r are the upper bounds on the number of* Execute, Send *and random oracle queries that $\mathcal{A}_{\mathsf{ake}}$ can ask respectively.*

PROOF. We give the proof in a sequence of games. In each game, the execution of the protocol on behalf of the uncorrupted parties is simulated for the adversary. Let S_i be the event that $\mathcal{A}_{\mathsf{ake}}$ wins the AKE-security game in Game i.

Game 0. This is the original AKE-security game as per the Definition 3.4. By definition we have

$$Adv_{\mathcal{A}_{\mathsf{ake}}} = |2 \cdot \Pr[S_0] - 1|. \tag{1}$$

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

$$|\Pr[S_0] - \Pr[S_1]| \leq \Pr[\mathsf{Forge}]. \tag{2}$$

The event Forge occurs when $\mathcal{A}_{\mathsf{ake}}$ 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, $\mathcal{A}_{\mathsf{ake}}$ is allowed to corrupt up to $\tilde{n} - 1$ parties but not allowed to modify messages output by the instances at the corrupted users. Hence Forge represents successful forgery of honest users' signatures.

If this event occurs we can use $\mathcal{A}_{\mathsf{ake}}$ 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 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 $\mathcal{A}_{\mathsf{ake}}$ not corrupting this party is at least $\frac{1}{n}$. Hence $Adv_{\mathcal{A}^{\mathsf{cma}}} \geq \frac{1}{n} \cdot \Pr[\mathsf{Forge}]$. Rewriting the inequality we have

$$\Pr[\mathsf{Forge}] \leq n \cdot Adv_{\mathcal{A}^{\mathsf{cma}}}. \tag{3}$$

Game 2. This game is the same as the previous game except that the simulation fails if an event Collision occurs.

$$|\Pr[S_1] - \Pr[S_2]| \leq \Pr[\mathsf{Collision}]. \tag{4}$$

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

$$\Pr[\mathsf{Collision}] \leq \frac{(3q_e + 3q_s + q_r)^2}{2^k}. \tag{5}$$

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

$$|\Pr[S_2] - \Pr[S_3]| \leq \Pr[\mathsf{Repeat}]. \tag{6}$$

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 $q_e + q_s$ instances that may have chosen a nonce k_i, we have

$$\Pr[\mathsf{Repeat}] \leq \frac{(q_e + q_s)^2}{2^k}. \tag{7}$$

Game 4. This game is the same as the previous game except that at the beginning of the game a value t is chosen at random in $\{1, \ldots, q_e + q_s\}$, where $q_e + q_s$ is an upper bound on the number of protocol sessions activated by the adversary. t represents a guess as to the protocol session in which the adversary is going to be tested. If the adversary does not choose the t^{th} session to ask the Test query, then the game outputs a random bit and aborts. Let Guess be the event that the guess is correct. The event Guess happens with the probability $1/(q_e + q_s)$. The probability of aborting due to an incorrect choice of t is $1 - 1/(q_e + q_s)$. Note that if Guess occurs, this game and the previous game are indistinguishable. Thus, we have

$$\begin{aligned} \Pr[S_4] &= \Pr[S_4|\mathsf{Guess}]\Pr[\mathsf{Guess}] + \Pr[S_4|\neg\mathsf{Guess}]\Pr[\neg\mathsf{Guess}] \\ &= \Pr[S_3]\frac{1}{(q_e + q_s)} + \frac{1}{2}\left(1 - \frac{1}{(q_e + q_s)}\right). \end{aligned} \tag{8}$$

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 faithfully forwards the messages among the parties. Otherwise either the session would not have been accepted or 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 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.

$$|\Pr[S_4] - \Pr[S_5]| \leq \Pr[\mathsf{Ask}]. \tag{9}$$

If Ask occurs, we can use $\mathcal{A}_{\mathsf{ake}}$ to solve the CDH problem in $\mathbb{G}$. Given a CDH instance $(g, A = g^a, B = g^b)$, this can be plugged into the 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 $\frac{1}{\tilde{n}}$. Hence, $\mathsf{SuccCDH} \geq \frac{1}{\tilde{n}q_r} \Pr[\mathsf{Ask}]$. Rewriting the equation we have

$$\Pr[\mathsf{Ask}] \leq \tilde{n}q_r \mathsf{SuccCDH}. \tag{10}$$

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

Note that in Round 2 of the protocol $\mathsf{mask}_{\tilde{n}}$ is computed as XOR of $k_{\tilde{n}}$ and $t_{\tilde{n}^R}$. Since both $k_{\tilde{n}}$ and $t_{\tilde{n}}^R$ are uniformly distributed in $\{0, 1\}^k$ both the values are information theoretically hidden in $\mathsf{mask}_{\tilde{n}}$. Hence, the protocol messages in round 2 of the test session carry no information about $k_{\tilde{n}}$.The best any adversary can do is to guess $k_{\tilde{n}}$ with a probability $\frac{1}{2^k}$. Hence, the probability that $\mathcal{A}_{\mathsf{ake}}$ asks the right H-query for the test session is at most $\frac{q_r}{2^k}$.

$$|\Pr[S_5] - \Pr[S_6]| \leq \frac{q_r}{2^k}. \tag{11}$$

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. Hence,

$$\Pr[S_6] = \frac{1}{2}. \tag{12}$$

By combining Equations (1)–(12), we have the claimed advantage of $\mathcal{A}_{\mathsf{ake}}$, which is negligible in k. □

THEOREM 4.2. *The protocol in Figure 1 satisfies mutual authentication as per Definition 3.7 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*

$$n \cdot Adv_{\mathcal{A}^{\mathsf{cma}}} + \frac{(3q_e + 3q_s + q_r)^2}{2^k} + \frac{(q_e + q_s)^2}{2^k},$$

where $\tilde{n} \leq n$ is the number of participants, n is the total number of public keys in the system, $Adv_{\mathcal{A}^{\mathsf{cma}}}$ is the advantage of a PPT adversary $\mathcal{A}^{\mathsf{cma}}$ against the UF-CMA security of the signature scheme and k is the security parameter. q_e, q_s and q_r are the upper bounds on the number of Execute*,* 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 3.7. By definition we have

$$Adv_{\mathcal{A}_{ma}} = \Pr[S_0]. \tag{13}$$

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 4.1.

$$|\Pr[S_0] - \Pr[S_1] \leq \Pr[\mathsf{Forge}] \leq n \cdot Adv_{\mathcal{A}^{\mathsf{cma}}}. \tag{14}$$

Game 2. This game is the same as the previous game except that the simulation fails if an event Collision occurs, where Collision is the same event described in Game 2 of Theorem 4.1.

$$|\Pr[S_1] - \Pr[S_2]| \leq \Pr[\mathsf{Collision}] \leq \frac{(3q_e + 3q_s + q_r)^2}{2^k}. \tag{15}$$

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 4.1.

$$|\Pr[S_2] - \Pr[S_3]| \leq \Pr[\mathsf{Repeat}] \leq \frac{(q_e + q_s)^2}{2^k}. \tag{16}$$

In Game 3, we have eliminated all the bad events Forge, Collision and Repeat. Hence, if Game 3 does not abort, all the honest partnered parties compute the same key, that is, $\Pr[S_3] = 0$.

By combining Equations (13)–(16), we have the claimed advantage of $\mathcal{A}_{ma}$, which is negligible in k. □

5. ACHIEVING MA-SECURITY WITH INSIDER KCIR

Bresson et al. [2001a] proposed a generic transformation that turns an AKE-secure GKE protocol π into a protocol π' that provides MA-security in the presence of an outsider adversary. Yet, their notion of MA-security did not consider KCIR. The transformation uses the well known technique of constructing an "authenticator" using the shared session key established in π. It works as follows: Let κ_i be the session key computed by U_i in protocol π. The protocol π' requires an additional round in which each party U_i computes a message $auth_i = \mathcal{H}(\kappa_i, i)$, where $\mathcal{H}$ is a hash function (modeled as random oracle in the proof) and broadcasts it to all the other parties. Each party verifies the incoming messages using the session key established at their end. If the verification is successful, π' terminates with each party U_i accepting the session key $\kappa'_i = \mathcal{H}(\kappa_i, 0)$.

We show that the above transformation does not necessarily guarantee MA-security with outsider KCIR. For example, consider a protocol π which does not have forward secrecy like the BG protocol [Boyd and González Nieto 2003] or our one-round GKE protocol [Gorantla et al. 2009]. Definition 3.6 implies that an adversary against MA-security with outsider KCIR can issue up to $n - 1$ Corrupt queries but must then remain passive on behalf of the corrupted users. As the protocol π does not have forward secrecy, corrupting a single party U_i is enough to obtain the session key κ_i. The adversary can now easily impersonate an uncorrupted party U_j in protocol π' by computing $auth_j = \mathcal{H}(\kappa_i, j)$. Hence, transformations based on shared keys cannot be used to obtain MA-security with outsider KCIR.

We now show that the KS-compiler [Katz and Shin 2005] can be generically used to achieve MA-security with both outsider and insider KCIR when applied to any GKE protocol with AKE-security. For this generic construction, a protocol that satisfies the basic notion of AKE-security without forward secrecy suffices, that is, the protocol does not need to have forward secrecy. Hence, the KS-compiler can be applied to the one-round GKE protocols [Boyd and González Nieto 2003; Gorantla et al. 2009] to obtain two-round GKE protocols that provide MA-security with insider KCIR.

The KS-compiler is reviewed in Figure 2. Katz and Shin [2005] showed that this compiler provides AKE-security and MA-security in the presence of insiders, yet without considering KCI attacks. Here, we show that this compilation technique is also sufficient to obtain MA-security with outsider and insider KCIR. Recall that MA-security

Let F be a collision-resistant PRF, and assume that v_0 is output by $\mathsf{Sample}(k)$ and publicly-known. Let $v_1 \neq v_0$ also be publicly-known.

Initialization Phase. During the initialization phase of π', each player U_i runs $\mathsf{Gen}(k)$ to generate long-term verification/signing keys (PK_{s_i}, SK_{s_i}) (in addition to any keys needed for π).

The Protocol. Players run protocol π. If U_i would terminate without accepting in π, then it terminates without accepting in π'. Otherwise, if U_i would accept (in protocol π) with $(\mathsf{sid}_i, \mathsf{pid}_i, \kappa_i)$, this player performs the following additional steps:

(1) U_i computes $\mathsf{ack}_i = F_{\kappa_i}(v_0)$ and $\kappa_i' = F_{\kappa_i}(v_1)$. Next, U_i erases all its local state except for ack_i, κ_i', sid_i and pid_i. Then, U_i computes a signature $\sigma_i \leftarrow \mathsf{Sign}_{SK_{s_i}}(U_i, \mathsf{sid}_i, \mathsf{pid}_i, \mathsf{ack}_i)$ and sends the message (U_i, σ_i) to all players in pid_i.

(2) Upon receipt of $|\mathsf{pid}_i - 1|$ messages (U_j, σ_j) from all other players $U_j \in \mathsf{pid}_i \setminus \{U_i\}$, player U_i checks that $\mathsf{Vrfy}_{PK_{s_j}}((U_j, \mathsf{sid}_i, \mathsf{pid}_i, \mathsf{ack}_i), \sigma_j) = 1$ for all $U_j \in \mathsf{pid}_i$. Assuming all verifications succeed, U_i accepts, erases its internal state, and outputs $(\mathsf{sid}_i, \mathsf{pid}_i, \kappa_i')$. If any of the verifications do not succeed, U_i terminates without accepting (and with no output).

Fig. 2. Katz and Shin compiler.

with insider KCIR implies MA-security with outsider KCIR i.e. given an adversary against MA-security with outsider KCIR, one can construct an adversary against MA-security with insider KCIR. For this reason we only need to prove that the compiled protocol guarantees MA-security with insider KCIR.

5.1. The Katz-Shin Compiler

Katz and Shin [2005] proposed a compiler that would turn any AKE-secure GKE protocol into a universally composable GKE protocol that achieves MA-security in the presence of insiders. The compiler uses messages authenticated with signatures generated by long-term private keys of the parties. Let $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ be a public key signature scheme which is existentially unforgeable under chosen message attack, where Gen is an algorithm that generates a signing key pair, Sign is a signing algorithm and Verify is a verification algorithm. The compiler also uses a pseudorandom function family [Goldreich et al. 1986] $\mathcal{F}$ with *collision-resistance*. A formal definition of collision-resistant pseudorandom function family is given below.

Definition 5.1 (*Collision-resistant PRF [Katz and Shin 2005]*). Let $\mathcal{F} = \{F^k\}$ with $F^k = \{F_s\}_{s \in \{0,1\}^k}$ be a pseudorandom function family (PRF). We say that $\mathcal{F}$ is collision-resistant PRF if there is an efficient procedure Sample such that the following is negligible in k for all PPT adversaries $\mathcal{A}$:

$$\Pr\left[v_0 \leftarrow \mathsf{Sample}(k);\ s, s' \leftarrow \mathcal{A}(k, v_0) \ \ : \ \ s, s' \in \{0,1\}^k \bigwedge s \neq s' \bigwedge F_s(v_0) = F_{s'}(v_0)\right]$$

Informally, the definition requires that for all k there exists an (efficiently computable) v_0 such that the function defined by $g(x) \stackrel{\text{def}}{=} F_x(v_0)$ is collision-resistant. Katz and Shin also sketched a way of constructing collision-resistant PRF in the standard model from any one-way permutation. Note that the above definition of collision-resistance for PRFs is different from the (standard) collision resistance considered for keyed hash functions. This property was earlier called *fixed-value-key-binding* property of a PRF ensemble [Fischlin 1999].

In Figure 2, we present the KS-compiler. The users first compute a session key κ by executing an initial GKE protocol π. The compiler starts by using the session key as a seed to the PRF with two publicly known strings to compute an acknowledgment message ack and a session key sk'. The message ack is then signed with the user's signing key and the signature along with the user's identity is broadcast to all other users, which serves as key confirmation message. If all the incoming signatures verify correctly, the compiled protocol π' accepts sk' as the session key; otherwise, π' terminates without accepting.

5.2. Security Proof

THEOREM 5.2. *If π is an AKE-secure GKE protocol, then applying the KS-compiler will result in a GKE protocol π' that preserves the AKE-security of π and also achieves MA-security with insider KCIR as per Definition 3.7.*

PROOF. Note that the KS-compiler does not enhance the security of π in terms of the AKE-security notion. It only preserves the AKE-security of π in the compiled protocol π'. On the other hand, it enhances the security of π by enabling the compiled protocol to achieve MA-security with insider KCIR. The proof of this theorem follows from Claims 5.3 and 5.4 below. □

CLAIM 5.3. *The advantage of an adversary $\mathcal{A}_{\mathsf{ake}}{}'$ against the AKE-security of π' is upper bounded by*

$$(q_e + q_s) \cdot (Adv_{\mathcal{A}_{\mathsf{ake}}} + Adv_{\mathcal{A}^{\mathsf{prf}}})$$

where $Adv_{\mathcal{A}_{\mathsf{ake}}}$ is the advantage of a PPT adversary $\mathcal{A}_{\mathsf{ake}}$ against the AKE-security of π and $Adv_{\mathcal{A}^{\mathsf{prf}}}$ is the advantage of a PPT adversary $\mathcal{A}^{\mathsf{prf}}$ against the pseudorandomness of the PRF $\mathcal{F}$ used in the KS-compiler. q_e and and q_s are the upper bounds on the number of Execute *and* Send *queries that $\mathcal{A}_{\mathsf{ake}}{}'$ can ask respectively.*

PROOF. We show that π' preserves the AKE-security of π. The proof is given as a sequence of games. Let S_i be the success probability of $\mathcal{A}'_{\mathsf{ake}}$ in Game i.

Game 0. This is the original AKE-security game. By definition we have

$$Adv'_{\mathcal{A}_{\mathsf{ake}}} = |2 \cdot \Pr[S_0] - 1|. \tag{17}$$

Game 1. This game is the same as the previous game except that at the beginning of the game a value t is chosen at random in $\{1, \ldots, q_e + q_s\}$, where $q_e + q_s$ is an upper bound on the number of protocol sessions activated by $\mathcal{A}'_{\mathsf{ake}}$. t represents the guess of $\mathcal{A}_{\mathsf{ake}}$ on the test session that is going to be chosen by $\mathcal{A}'_{\mathsf{ake}}$. If $\mathcal{A}'_{\mathsf{ake}}$ does not choose the t^{th} session to ask the Test query, then $\mathcal{A}_{\mathsf{ake}}$ outputs a random bit and aborts. Let Guess be the event that the guess is correct.
From Game 4 of Theorem 4.1, we have

$$\begin{aligned}\Pr[S_1] &= \Pr[S_1|\mathsf{Guess}]\Pr[\mathsf{Guess}] + \Pr[S_1|\neg\mathsf{Guess}]\Pr[\neg\mathsf{Guess}] \\ &= \Pr[S_0]\frac{1}{(q_e+q_s)} + \frac{1}{2}\left(1 - \frac{1}{(q_e+q_s)}\right).\end{aligned} \tag{18}$$

Game 2. This is identical to the previous game except that the input random seed κ_{i^*} of F during the test session is replaced by a uniformly random string from $\{0, 1\}^k$. We claim that

$$|\Pr[S_1] - \Pr[S_2]| \leq Adv_{\mathcal{A}_{\mathsf{ake}}}. \tag{19}$$

The behavior of a distinguisher between Game 1 and Game 2 can be perfectly simulated by $\mathcal{A}_{\mathsf{ake}}$. Particularly, the queries asked by $\mathcal{A}'_{\mathsf{ake}}$ in this game can be answered by $\mathcal{A}_{\mathsf{ake}}$ as follows: $\mathcal{A}_{\mathsf{ake}}$ initially starts by selecting the signature key pairs to be used by π' as part of the KS-compiler, for all the users in the protocol.

With the knowledge of these keys, $\mathcal{A}_{\mathsf{ake}}$ can trivially simulate the Send, RevealState, RevealKey and Corrupt queries of $\mathcal{A}'_{\mathsf{ake}}$. When $\mathcal{A}'_{\mathsf{ake}}$ chooses the test session in the t-th session, $\mathcal{A}_{\mathsf{ake}}$ uses its own challenge key K_b (which it would have got from its own protocol interaction) to output a challenge key for $\mathcal{A}'_{\mathsf{ake}}$ as follows: $\mathcal{A}_{\mathsf{ake}}$ computes $\kappa_{i^*} = F_{K_b}(v_1)$ and returns κ_{i^*} to $\mathcal{A}'_{\mathsf{ake}}$.

Let θ' be the output guess bit of $\mathcal{A}'_{\mathsf{ake}}$ for the test query. $\mathcal{A}_{\mathsf{ake}}$ simply forwards θ' to its challenger as its guess for the bit b. On the other hand, if $\theta' = 1$ (guess for real key), the distinguisher identifies the game as Game 1; otherwise as Game 2. Note that the advantage of the distinguisher is the same as that of $\mathcal{A}_{\mathsf{ake}}$, which in turn is the same as the advantage of $\mathcal{A}'_{\mathsf{ake}}$ in this game. Hence, we have the claimed advantage for the distinguisher.

Game 3. This is identical to the previous game except that the output of the PRF in the test session is replaced by a random value uniformly chosen from $\{0, 1\}^k$. Note that if F is not a PRF, $\mathcal{A}'_{\mathsf{ake}}$ can distinguish the output of F from a random string. We have,

$$|\Pr[S_2] - \Pr[S_3]| \leq Adv_{\mathcal{A}^{\mathsf{prf}}}. \tag{20}$$

Since, the key in this game is uniformly distributed $\mathcal{A}'_{\mathsf{ake}}$ gets no advantage, that is, $\Pr[S_3] = 0$.

From Equations (17) to (20), we have the claimed advantage for $\mathcal{A}'_{\mathsf{ake}}$. □

CLAIM 5.4. *The advantage of an adversary $\mathcal{A}_{ma}$ against the MA-security with insider KCIR of π' is upper bounded by*

$$n \cdot Adv_{\mathcal{A}^{\mathsf{cma}}} + Adv_{\mathcal{A}^{\mathsf{coll}}}$$

where n is the total number of public keys in the system, $Adv_{\mathcal{A}^{\mathsf{cma}}}$ is the advantage of a PPT adversary $\mathcal{A}^{\mathsf{cma}}$ against the unforgeability of the signature scheme under chosen message attack and $Adv_{\mathcal{A}^{\mathsf{coll}}}$ is the advantage of a PPT adversary $\mathcal{A}^{\mathsf{coll}}$ against the collision resistance of the PRF $\mathcal{F}$ used in the KS-compiler.

PROOF. The second part of the proof shows that the KS-compiler enhances the security of π by enabling the compiled protocol to achieve MA-security with insider KCIR. The proof is again given in a sequence of games. Let S_i be the event that $\mathcal{A}_{ma}$ wins the MA-security game in Game i.

Game 0. This is the original MA-security game as per Definition 3.4. We have

$$Succ_{\mathcal{A}_{ma}} = \Pr[S_0]. \tag{21}$$

Game 1. This game is identical to the previous game except that the simulation fails when $\mathcal{A}_{ma}$ issues a Send query that contains a valid signature σ_i on the message $(U_i, \mathsf{sid}_i, \mathsf{pid}_i, \mathsf{ack}_i)$ such that the message has not been previously output by an oracle π_i^j and U_i has not been corrupted. Let Forge be such an event. From Game 1 of Theorem 4.1, we have

$$|\Pr[S_0] - \Pr[S_1]| \leq \Pr[\mathsf{Forge}] \leq n \cdot Adv_{\mathcal{A}^{\mathsf{cma}}}. \tag{22}$$

Game 2. This is the same as the previous game except that the simulation fails if a collision occurs in F. Let Collision be the event.

$$|\Pr[S_1] - \Pr[S_2]| \leq \Pr[\mathsf{Collision}]. \tag{23}$$

Collision occurs when two honest parties U_i and U_j compute keys κ_i and κ_j such that

$$\mathsf{ack}_i = F_{\kappa_i}(v_0) = F_{\kappa_j}(v_0) = \mathsf{ack}_i \text{ but, } \kappa'_i = F_{\kappa_i}(v_1) \neq F_{\kappa_j}(v_1) = \kappa'_j.$$

Hence, we have

$$\Pr[\mathsf{Collision}] \leq Adv_{\mathcal{A}^{\mathsf{coll}}}. \tag{24}$$

In this game all the honest parties output identical session keys. Hence, we have

$$\Pr[S_2] = 0. \tag{25}$$

By combining Equations (21) to (25), we have the claimed advantage for $\mathcal{A}_{ma}$. □

Remark 5.5. Note that the protocol obtained after applying the KS-compiler cannot achieve forward secrecy if the base protocol does not. Hence, as discussed in Remark 3.5, such a protocol cannot achieve AKE-security with KCIR. However, from Theorem 5.2 it is evident that forward secrecy is not necessary for a GKE protocol to achieve MA-security with insider KCIR.

6. KCI ATTACKS ON EXISTING PROTOCOLS

We now present KCI attacks on the protocols of Boyd and González Nieto [2003], Al-Riyami and Paterson [2003] and Bresson et al. [2003]. 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 BG protocol, though role asymmetric, 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 being three, while the protocol of Bresson et al. assumes a server with high computational resources and many computationally restricted clients.

6.1. Boyd and González Nieto's Protocol

The BG protocol [Boyd and González Nieto 2003] was proven AKE-secure in the Bellare-Rogaway model [Bellare and Rogaway 1993] adapted to the group setting. Later, Choo et al. [2005] presented an unknown key share attack on the BG protocol in a multiuser setting. They also presented an improved BG protocol that resists unknown key share attacks but did not give any formal security proof. We first briefly describe the improved version of 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 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 respectively.

In the protocol, the distinguished user U_1 chooses a nonce $N_1 \overset{R}{\leftarrow} \{0,1\}^k$ 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 $N_i \overset{R}{\leftarrow} \{0,1\}^k$ along with their identities. Each user computes the session ID as the concatenation of all the outgoing and incoming protocol messages. A key derivation function $\mathcal{H}$ is then 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 a single round. The protocol message transmission and session key computation are presented in Figure 3. The users $U_i, i \neq 1$, verify the signature of U_1 and decrypt N_1 before computing the session key.

We now show that the improved BG protocol in Figure 3 is not secure against KCI attacks. An attack can be mounted by corrupting any user except the distinguished

Round 1.
- $U_1 : N_1 \xleftarrow{R} \{0,1\}^k$.
- $U_1 \to * : \mathcal{U}, \mathcal{S}_{SK_{s_1}}(\mathcal{U}, \mathcal{E}_{PK_{e_2}}(N_1, U_1), \ldots, \mathcal{E}_{PK_{e_n}}(N_1, U_1))$.
- $U_1 \to * : \mathcal{E}_{PK_{e_i}}(N_1, U_1)$ for $1 < i \leq n$.
- $U_i \to * : U_i, N_i$ for $1 < i \leq n$.

Key Computation.
- $sid = \mathcal{U} \| \mathcal{S}_{SK_{s_1}}(\mathcal{U}, \mathcal{E}_{PK_{e_2}}(N_1, U_1), \ldots, \mathcal{E}_{PK_{e_n}}(N_1, U_1)) \| \mathcal{E}_{PK_{e_i}}(N_1, U_1) \| U_i \| N_i$ for $1 < i \leq n$.
- The session key is $\kappa = \mathcal{H}(N_1 \| sid)$.

Fig. 3. Improved Boyd-González Nieto protocol.

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. Note that instance at U_2 would be still considered fresh as per our AKE-security notion since Corrupt(U_2) in our model only reveals the long-term key of U_2.

A straightforward improvement to the protocol in Figure 3 could be asking all users $U_i \neq U_1$ to encrypt their nonces with the public keys of the other users and broadcast the messages. Although the revised protocol resists the KCI attack that we described on the BG protocol, it still cannot be proven secure under our notion of AKE-security with KCIR. To see why, note that in the previous 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) are corrupted as allowed by Definition 3.4, then the adversary can impersonate U_1 in the same way as described above and successfully mount a KCI attack.

6.2. Al-Riyami and Paterson's Protocol

Al-Riyami and Paterson [2003] proposed a series of tripartite key agreement (TAK) protocols based on Joux's protocol [Joux 2000]. While the authors did not provide a definition of KCIR for a TAK protocol, they claimed that the protocol TAK-3 was secure against KCI attacks. However, we now present a KCI attack on TAK-3. We slightly alter the notation while describing their protocol.

The system parameters are $(p, \mathbb{G}_0, \mathbb{G}_1, g, e, H)$, where p is a prime number, $\mathbb{G}_0$ and $\mathbb{G}_1$ are groups of order p, g is a generator of $\mathbb{G}_0$, $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_1$ is an admissible bilinear map [Boneh and Franklin 2001], and H is a hash function that maps to the session key space. Let (x, g^x), (y, g^y) and (z, g^z) be the private-public key pairs of three users A, B and C respectively, where $x, y, z \in \mathbb{Z}_p^*$. Each party is issued a certificate for its public key, which binds 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 $a, b, c \xleftarrow{R} \mathbb{Z}_p^*$ respectively. The protocol message transmission and key computation are shown in Figure 4.

We now show that the protocol in Figure 4 is not KCI resilient as per Definition 3.4. Let us assume that the adversary $\mathcal{A}$ has compromised the long-term private keys x and y of the parties A and B respectively. $\mathcal{A}$ can impersonate an honest user C by sending a message $g^{c'} \| Cert_C$ for a known $c' \in \mathbb{Z}_p^*$. It can compute the same key that A and B computes with its knowledge of x, y and c' as $K' = e(g^y, g^{c'})^x \cdot e(g^b, g^z)^x \cdot e(g^a, g^z)^y \cdot$

Round 1.
. $A \to B, C : g^a \| Cert_A$.
. $B \to A, C : g^b \| Cert_B$.
. $C \to A, B : g^c \| Cert_C$.
Key Computation.
. $A : K_A = e(g^y, g^c)^x \cdot e(g^b, g^z)^x \cdot e(g^y, g^z)^a \cdot e(g^b, g^c)^a$.
. $B : K_B = e(g^a, g^z)^y \cdot e(g^x, g^c)^y \cdot e(g^x, g^z)^b \cdot e(g^a, g^c)^b$.
. $C : K_C = e(g^a, g^y)^z \cdot e(g^x, g^b)^z \cdot e(g^x, g^y)^c \cdot e(g^a, g^b)^c$.

The session key is $\kappa = K_A = K_B = K_C = e(g, g)^{(xy)c+(xz)b+(yz)a+abc}$.

Fig. 4. TAK-3 protocol of Al-Riyami and Paterson.

Round 1.
(1) Each U_i selects $x_i \overset{R}{\leftarrow} \mathbb{Z}_p^*$, computes $y_i = g_i^x$, $\alpha_i = y^{x_i}$ and $\sigma_1 = \mathcal{S}_{SK_{s_i}}(y_i)$.
(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 $K = \mathcal{H}_0(c\|\alpha_1\| \ldots \|\alpha_n)$.
(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 $\kappa = \mathcal{H}(K\|\mathcal{G}_c\|S)$.

Fig. 5. Bresson et al.'s GKE protocol.

$e(g^a, g^b)^{c'}$. It can now easily win the AKE-security game by selecting the test session at either A or B.

The key computation process of TAK-3 protocol is similar to the MTI/A0 protocol [Matsumoto et al. 1986]. Al-Riyami and Paterson also proposed another tripartite variant TAK-4 whose key derivation is based on the MQV [Law et al. 2003]. Both 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.

6.3. Bresson et al.'s Protocol

In the protocol of Bresson et al. [2003], a group of n parties computes a common session key with a mobile gateway S acting as a server. The system parameters are $(p, \mathbb{G}, g, \mathcal{H}, \mathcal{H}_0, \mathcal{H}_1, k_0, k_1)$, where p is a prime number chosen based on a security parameter k, $\mathbb{G}$ is a finite cyclic group of order p, 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 lengths k, k_0 and k_1 respectively. The server is assumed to have a private-public key pair $(x, y = g^x)$ where $x \overset{R}{\leftarrow} \mathbb{Z}_p^*$. It is 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_{s_i}, PK_{s_i}) for a signature scheme $\Sigma = (\mathcal{K}_s, \mathcal{S}, \mathcal{V})$. The protocol execution is described in Figure 5.

We now show that the protocol in Figure 5 is not secure against KCI attacks as per Definition 3.4. 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

Table I. Security and Efficiency Comparison among Existing GKE Protocols

	Rounds	AKE	AKE-FS	AKE-KCIR	MA	MA-Out-KCIR	MA-In-KCIR	Model
Boyd and González Nieto [2003]	1	Yes	No	No	No	No	No	ROM
Katz and Yung [2003]	3	Yes	Yes	Yes*	honest	honest	honest	Std.
Bohli et al. [2007]	2	Yes	Yes	Yes	Yes	Yes	Yes	ROM
Dutta and Barua [2008]	2	Yes	Yes	Yes*	honest	honest	honest	Std.
Bresson and Manulis [2008]	3	Yes	Yes	Yes*	Yes	Yes*	Yes*	Std.
Furukawa et al. [2008]	2	Yes	Yes	Yes*	Yes	Yes*	Yes*	Std.
Any KS-compiled Protocol π'	$\#\pi + 1$	From π	From π	From π	Yes	Yes	Yes	Std.

$\#\pi$ refers to the number of rounds in the base protocol π. The terms "AKE" refers to AKE-security without forward secrecy and KCIR, "AKE-FS" refers to AKE-security with forward secrecy and "AKE-KCIR" refers to AKE-security with both forward secrecy and KCIR. Similarly "MA" refers to MA-security without any KCIR while "MA-Out-KCIR" and "MA-In-KCIR" refer to MA-security with outsider and insider KCIR respectively. The entry "Yes*" says that the corresponding protocol appears to be secure under the notion but there is no formal proof. The entry "From π" says that the corresponding notion is preserved from the base protocol. The last column in the table says whether the protocol is proven secure in the random oracle model or in the standard model.

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 the test session at S.

6.4. On Achieving AKE-security with KCIR

In Sections 6.1, 6.2, and 6.3 we have shown that there exist a few GKE protocols that do not achieve AKE-security with KCIR. As discussed in Remark 3.5, it is necessary for a protocol to have at least partial forward secrecy when $n-1$ parties are corrupted (called $(n-1)$–*partial forward secrecy*) or full forward secrecy in order to achieve AKE-security with KCIR. However, it is not a sufficient condition, as evident by our attack on Al-Riyami and Paterson's protocol [Al-Riyami and Paterson 2003] in Section 6.2. It is also clear that the KS-compiler does not enhance the AKE-security of a GKE protocol to AKE-security with KCIR. For example, if we apply the result of Theorem 5.2 to the BG protocol [Boyd and González Nieto 2003], which is AKE-secure, the resultant protocol will still have only AKE-security. The attack in Section 6.1 will still be applicable to the resultant KS-compiled BG protocol. Hence, the KS-compiler cannot be used in a generic way to achieve AKE-security with KCIR.

Since at least $(n-1)$–partial forward secrecy is required for a protocol to achieve AKE-security with KCIR, it will be interesting to come up with a definition of AKE-security that covers $(n-1)$–partial forward secrecy and outsider KCIR and then propose a GKE protocol realizing such a definition. We also leave it an open problem to generically construct a GKE protocol that has AKE-security with KCIR from one with either $(n-1)$–partial forward secrecy or full forward secrecy.

7. CONCLUSION

Table I gives a comparison of the security of some of the existing GKE protocols. 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. 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. The protocol of Bresson and Manulis appears to resist KCI attacks and their protocol is also proven secure in the standard model. However, it has three rounds of communication. As shown in Section 5, any KS-compiled protocol preserves the security of the base protocol and enables the compiled protocol to achieve MA-security with insider KCIR. The BG protocol or our protocol in Gorantla et al. [2009] may be used as the base protocol.

In this article, we have modeled KCI attacks by both outsider and insider adversaries on GKE protocols. An existing protocol has been then proven secure under our new definitions. Additionally, we have shown that the Katz-Shin compiler can be generically used to achieve insider KCIR. Finally, we have also shown that there exist protocols which are not secure against KCI attacks. Hence, we recommend all future GKE protocols to be analyzed for outsider and insider KCIR.

REFERENCES

Al-Riyami, S. S. and Paterson, K. G. 2003. Tripartite authenticated key agreement protocols from pairings. In *Proceedings of the 9th IMA International Conference on Cryptography and Coding*. Lecture Notes in Computer Science, vol. 2898, Springer, 332–359.

Becker, K. and Wille, U. 1998. Communication complexity of group key distribution. In *Proceedings of the ACM Conference on Computer and Communications Security*. ACM Press, 1–6.

Bellare, M. and Rogaway, P. 1993. Entity authentication and key distribution. In *Proceedings of the Annual Cryptology Conference (CRYPTO'93)*. Lecture Notes in Computer Science, vol. 773. Springer, 232–249.

Bohli, J.-M., Gonzalez Vasco, M. I., and Steinwandt, R. 2007. Secure group key establishment revisited. *Int. J. Inform. Secur. 6*, 4, 243–254.

Boneh, D. and Franklin, M. K. 2001. Identity-based encryption from the Weil pairing. In *Proceedings of the Annual Cryptology Conference (CRYPTO'01)*. Lecture Notes in Computer Science, vol. 2139. Springer, 213–229.

Boyd, C. and González Nieto, J. M. 2003. Round-optimal contributory conference key agreement. In *Proceedings of the IACR International Conference on Practice and Theory of Public Key Cryptography (PKC'03)*. Lecture Notes in Computer Science, vol. 2567. Springer, 161–174.

Bresson, E., Chevassut, O., Essiari, A., and Pointcheval, D. 2003. Mutual authentication and group key agreement for low-power mobile devices. In *Proceedings of the IFIP Joint Conference on Mobile and Wireless Communications Networks (MWCN'03)*. World Scientific Publishing, 59–62.

Bresson, E., Chevassut, O., and Pointcheval, D. 2001b. Provably authenticated group Diffie-Hellman KEY EXCHANGE—The dynamic case. In *Proceedings of the Annual Cryptology Conference (ASIACRYPT'01)*. Lecture Notes in Computer Science, vol. 2248, Springer, 290–309.

Bresson, E., Chevassut, O., and Pointcheval, D. 2002. Dynamic Group Diffie-Hellman Key Exchange under Standard Assumptions. In *Proceedings of the Annual Cryptology Conference (EUROCRYPT'02)*. Lecture Notes in Computer Science, vol. 2332. Springer, 321–336.

Bresson, E., Chevassut, O., Pointcheval, D., and Quisquater, J.-J. 2001a. Provably authenticated group Diffie-Hellman key exchange. In *Proceedings of the 8th ACM Conference on Computer and Communications Security (CCS'01)*. ACM, 255–264.

Bresson, E. and Manulis, M. 2008. Securing group key exchange against strong corruptions. In *Proceedings of the ACM Symposium on Information, Computer and Communications Security (ASIACCS'08)*. ACM Press, 249–260.

Burmester, M. and Desmedt, Y. 1994. A secure and efficient conference key distribution system (extended abstract). In *Proceedings of the Annual Cryptology Conference (EUROCRYPT'94)*. 275–286.

Burmester, M. and Desmedt, Y. 1997. Efficient and secure conference-key distribution. In *Proceedings of the International Workshop on Security Protocols*. Springer-Verlag, 119–129.

Canetti, R. 2001. Universally compos able security: A new paradigm for cryptographic Protocols. In *Proceedings of the Annual Symposium on Foundations of Computer Science*. 136–145. http://eprint.iacr.org/2000/067.

Choo, K.-K. R., Boyd, C., and Hitchcock, Y. 2005. Errors in computational complexity proofs for protocols. In *Proceedings of the Annual Cryptology Conference (ASIACRYPT'05)*. Lecture Notes in Computer Science, vol. 3788, Springer, 624–643.

Dutta, R. and Barua, R. May 2008. Provably secure constant round contributory group key agreement in dynamic setting. *IEEE Trans. Inform. Theory 54*, 5, 2007–2025.

Fischlin, M. 1999. Pseudorandom function tribe ensembles based on one-way permutations: improvements and applications. In *Proceedings of the Annual Cryptology Conference (EUROCRYPT'99)*. J. Stern Ed. Lecture Notes in Computer Science, vol. 1592, Springer, 432–445.

FURUKAWA, J., ARMKNECHT, F., AND KUROSAWA, K. 2008. A universally composable group key exchange protocol with minimum communication effort. In *Proceedings of the 6th International Conference on Security and Cryptography for Networks (SCN'08)*. Lecture Notes in Computer Science, vol. 5229, Springer, 392–408.

GOLDREICH, O. 2001. *Foundations of Cryptography (Basic Tools)*. Cambridge University Press.

GOLDREICH, O., GOLDWASSER, S., AND MICALI, S. 1986. How to construct random functions. *J. ACM 33*, 4, 792–807.

GOLDWASSER, S., MICALI, S., AND RIVEST, R. L. 1988. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J Comput. 17*, 2,281–308.

GORANTLA, M. C. 2010. Design and analysis of group key exchange protocols. Ph.D. thesis, Queensland University of Technology, Brisbane, Australia.

GORANTLA, M. C., BOYD, C., AND GONZÁLEZ NIETO, J. M. 2009a. Modeling key compromise impersonation attacks on group key exchange protocols. In *Proceedings of the IACR International Conference on Practice and Theory of Public Key Cryptography*, S. Jarecki and G. Tsudik, eds. Lecture Notes in Computer Science, vol. 5443, Springer, 105–123.

GORANTLA, M. C., BOYD, C., GONZÁLEZ NIETO, J. M., AND MANULIS, M. 2009b. Generic one round group key exchange in the standard model. In *Proceedings of the 12th International Conference on Information Security and Cryptology (ICISC'09)*. Springer.

INGEMARSSON, I., TANG, D. T., AND WONG, C.-K. 1982. A conference key distribution system. *IEEE Trans. Inform. Theory 28*, 5, 714–719.

JOUX, A. 2000. A one round protocol for tripartite Diffie-Hellman. In *Proceedings of the International Symposium on Algorithmic Number Theory*. Lecture Notes in Computer Science, vol. 1838, Springer, 385–394.

JUST, M. AND VAUDENAY, S. 1996. Authenticated multi-party key agreement. In *Proceedings of the Annual Cryptology Conference (ASIACRYPT'96)*. Lecture Notes in Computer Science, vol. 1163, Springer, 36–49.

KATZ, J. AND SHIN, J. S. 2005. Modeling insider attacks on group key-exchange protocols. In *Proceedings of the 12th ACM Conference on Computer and Communications Security (CCS'05)*. ACM, 180–189.

KATZ, J. AND YUNG, M. 2003. Scalable protocols for authenticated group key exchange. In *Proceedings of the Annual Cryptology Conference (CRYPTO'03)*. Lecture Notes in Computer Science, vol. 2729, Springer, 110–125.

KIM, H.-J., LEE, S.-M., AND LEE, D. H. 2004. Constant-round authenticated group key exchange d dynamic groups. In *Proceedings of the Annual Cryptology Conference (ASIACRYPT'04)*. Lecture Notes in Computer Science, vol. 3329, Springer, 24–259.

KRAWCZYK, H. 2005. HMQV: A high-performance secure Diffie-Hellman protocol. In *Proceedings of the Annual Cryptology Conference (CRYPTO'05)*. Lecture Notes in Computer Science, vol. 3621, Springer, 546–566.

LAMACCHIA, B. A., LAUTER, K., AND MITYAGIN, A. 2007. Stronger security of authenticated key exchange. In *Proceedings of the 1st International Conference on Provable Security (ProvSec'07)*. W. Susilo, J. K. Liu, and Y. Mu, Eds., Lecture Notes in Computer Science, vol. 4784, Springer, 1–16.

LAW, L., MENEZES, A., QU, M., SOLINAS, J. A., AND VANSTONE, S. A. 2003. An efficient protocol for authenticated key agreement. *Des. Codes Cryptog. 28*, 2, 119–134.

MANULIS, M. 2007. *Provably Secure Group Key Exchange*. IT Security Series, vol. 5. Europaischer Universitatsverlag.

MATSUMOTO, T., TAKASHIMA, Y, AND IMAI, H. 1986. On seeking smart public-key distribution systems. *Trans. IECE Japan E69*, 99–106.

NG, E. M. 2005. Security models and proofs for key establishment protocols. M.S. thesis, University of Waterloo.

STEER, D. G., STRAWCZYNSKI, L., DIFFIE, W., AND WIENER, M. J. 1990. A secure audio teleconference system. In *Proceedings of the Annual Cryptology Conference (CRYPTO'88)*. S. Goldwasser, Ed., Lecture Notes in Computer Science, vol. 403, Springer, 520–528.

STEINER, M., TSUDIK, G., AND WALDNER, M. 1996. Diffie-Hellman key distribution extended to group communication. In *Proceedings of the ACM Conference on Computer and Communications Security*. 31–37.

USTAOGLU, B. 2008. Obtaining a secure and efficient key agreement protocol from (H)MQV and NAXOS *Des. Codes Cryptog. 46*, 3, 329–342.

Received June 2010; revised February 2011, June 2011; accepted September 2011