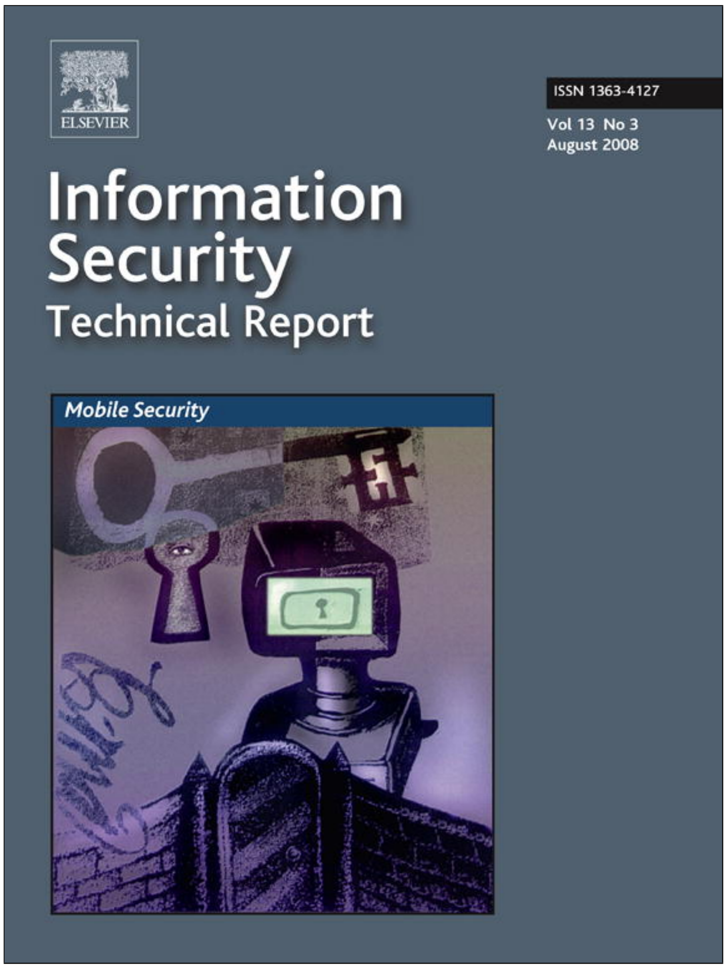

ELSEVIER
ISSN 1363-4127
Vol 13 No 3
August 2008
Information Security Technical Report
Mobile Security

INFORMATION SECURITY TECHNICAL REPORT 13 (2008) 158–164

available at www.sciencedirect.com

ScienceDirect

www.compseconline.com/publications/prodinf.htm

Information Security Technical Report

Secure authenticated group key agreement protocol in the MANET environment

Chan Yeob Yeun[a,*], Kyusuk Han[b], Duc Liem Vo[b], Kwangjo Kim[b]

[a]Khalifa University of Science, Technology and Research, P.O. Box 573, Sharjah, United Arab Emirates
[b]Information and Communications University, 119, Munjiro, Yuseong-gu, Daejeon, 305-732, South Korea

Keywords:
MANET
Detect cheating
Identifying malicious insiders
Authenticated group key agreement

A B S T R A C T

Due to dynamic and infrastructure-less nature of Mobile Ad hoc Network (MANET) environment, there exist number of threats as mobile devices and nodes could freely move around in MANET such as eavesdropping of communications channels, modification of sensitive m-commerce transactions, Denial of Service(DoS), vulnerabilities of impersonation by malicious insiders etc. In this paper, we propose a novel authenticated group key agreement protocol for end-to-end security in the MANET environment without any infrastructure that is based on Burmester and Desmedt group key agreement protocol (Burmester M, Desmedt Y. A secure and efficient conference key distribution system. Advances in Cryptology – EuroCrypt '94, Lecture Notes in Computer Science 1995;950:275–86) and their variants (Choi KY, Hwang JY, Lee DH. Efficient ID-based group key agreement with bilinear maps. Public Key Cryptography – PKC, Lecture Notes in Computer 2004;2947:130–144)].
We also design practical enhancements of BD and Choi et al.'s protocols that not only detect, but also identify malicious insiders by using the trusted arbiter who only involves in the protocol if the cheating has been occurred.

1. Introduction

In recent year, a MANET (Mobile Ad Hoc Network) is omnipresent emerges in our life. The mobile devices called nodes in MANET. The MANET has some characteristics: infrastructure-less, mobility, dynamic topology, resource constraint. In the MANET environment, each node can decide to join and leave the network by itself and there communicate each other without infrastructure. The communication depends on each node corporate to forward packet called multi-hop communication. Besides each node has mobility, it can lead to change topology quickly and low connectivity each other.

Due to dynamic and infrastructure-less nature of the Mobile Ad hoc Network (MANET) environment, there exist number of threats as mobile devices and nodes could freely move around in MANET such as eavesdropping of communications channels, modification of sensitive m-commerce transactions, Denial of Service(DoS), vulnerabilities of impersonation by malicious insiders and etc.

Moreover, infrastructure-less nature of MANET, Light weight asymmetric techniques such as ID-based crypto systems could provide intelligent facilities for securing MANET environments. ID-based systems require no explicit public key available and the key is constructed from public available information. It is an asymmetric system where unique name plays the role of the public key. These characteristics of ID-based techniques make it very suitable for the MANET security architecture and applications.

* Corresponding author.
E-mail addresses: cyeun@kustar.ac.ae (C.Y. Yeun), hankyusuk@icu.ac.kr (K. Han), vdliem@icu.ac.kr (D.L. Vo), kkj@icu.ac.kr (K. Kim).

1363-4127/$ – see front matter Crown Copyright © 2008 Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.istr.2008.10.002

In this paper, we propose a novel authenticated group key agreement protocol for end-to-end security in the MANET environment without any infrastructure that is based on Burmester and Desmedt group key agreement protocol (Burmester and Desmedt) and their variants (Choi et al., 2004) that is based on ID-based crypto system.

An authenticated group key agreement protocol (AGKA) enables two or more participants who want secure communication share a common secret key. After Burmester and Desmedt (BD) proposed the conference key agreement protocol in (Burmester and Desmedt), there are many studies on the group key agreement (Just and Vadenay, 1996; Nam, 2007; Nam et al., 2006; Steiner et al., 1998).

Our motivation to write this paper is to find the efficient way for detecting and identifying the attackers, that enables not only stopping the repetition of attacks, but also warning or even terminating the attackers from the protocol in the real MANET applications. For example, initiating the tele-conference, if the attack is detected, then the malicious insiders are easily identified, and the repetition of such attacks are also prevented.

Therefore, we propose the practical enhancements of BD (Burmester and Desmedt) and Choi et al. (2004) authenticated group key agreement protocols that not only detect, but also identify malicious insiders by using the trusted arbiter if the cheating has occurred in MANET. Since the malicious cheaters will continuously try to make the group key agreement have the incorrect result in MANET, the identifying and removing the cheaters are necessary for the practical use. With the assistance of the trusted arbiter who could be a trusted mobile operator or service providers, we easily identify the cheaters and prevent the further attacks on the group key agreement protocol in MANET. Thus, our protocol provides any participants to detect the cheating, also identify the malicious insiders by the trusted arbiter regardless of the number of cheaters.

2. Group key agreement protocols

In this section, we review the original BD protocol and their inherited Choi et al.'s protocol that is based on an ID-based crypto system.

2.1. *BD conference key protocol*

Burmester–Desmedt (Burmester and Desmedt) provided several conference keying protocols, in which, the protocol works in the broadcast model is quite popular. The summary of the protocol follows:

(1) *One-time setup*. An appropriate prime p and generator α of $\mathbb{Z}_p^*$ are selected, and authentic copies of these are provided to each of n system users.

(2) *Conference key generation*. Any group of $t \le n$ users (typically $t \ll n$), derive a common conference key K as follows. (Without loss of generality, the users are labeled U_0 through U_t-1, and all indices j indicating users are taken modulo t.)

   (a) Each U_i selects a random integer r_i, $1 \le r_i \le p-2$, computes $z_i = \alpha^{r_i} \bmod p$, and sends z_i to each of the other $t-1$ group members. (Assume that U_i has been notified a priori, of the indices j identifying other conference members.).

   (b) Each U_i after receiving z_{i-1} and z_{i+1}, computes $X_i = (z_{i+1}/z_{i-1})^{r_i} \bmod p$ (note $X_i = \alpha^{r_{i+1}r_i - r_i r_{i-1}}$), and sends X_i to each of the other $t-1$ group members.

   (c) After receiving X_j, $1 \le j \le t$ excluding $j = i$, U_i computes $K = K_i$ as $K_i = (z_{i-1})^{tr_i} \cdot X_i^{t-1} \cdot X_{i+1}^{t-2} \ldots X_{i+(t-3)}^2 \cdot X_{i+(t-2)}^1 \bmod p$.

Just and Vadenay (1996) shows the generalization and the lack of key authentication feature in BD. Several attacks are shown in (Tang and Mitchell, 2005).

2.2. *Choi et al.'s ID-based AGKA scheme*

Choi et al.'s ID-based authenticated group key agreement scheme (Choi et al., 2004) is the variation of BD protocol. The process *One-time setup* in BD is modified as followings.

- **Setup:** The Key Generation Center (KGC) sets up the group G_1 and G_2, where G_1 is a cyclic additive group of prime order q and G_2 is a cyclic multiplicative group of same order q and chooses a random number $s \in \mathbb{Z}_q^*$ and sets $P_{\text{pub}} = sP$. KGC keeps s as the master key, which is known only by itself. KGC also defines $H : \{0,1\}^* \to \mathbb{Z}_q$ and $H_1 : \{0,1\}^* \to \mathbb{G}_1$, where H and H_1 are cryptographic hash functions.
- **Extraction:** A user submits his identity information ID $\in \{0, 1\}^*$ to KGC. KGC computes the user's private key $S_{\text{ID}} = sQ_{\text{ID}}$ and sends it to the user via a secure channel, here $Q_{\text{ID}} = H_1(\text{ID})$.

Let $\{U_i | i = 1, 2,\ldots, n\}$ be a set of n users who would like to share a session key. Suppose ID_i denotes the identity information of the user U_i. The indices are subject to modulo n. Let U_i's long-term public key and private key be $Q_i = H_1(\text{ID}_i)$ and $S_i = sQ_i$, respectively.

2.2.1. *Round 1*

Each user U_i picks a random integer $a_i \in \mathbb{Z}_q^*$ and computes $P_i = a_i P$, $h_i = H(P_i)$ and $T_i = a_i P_{\text{pub}} + h_i S_i$. Each user U_i broadcasts $\langle P_i, T_i \rangle$ to all others and keeps a_i secret.

2.2.2. *Round 2*

Upon the receipt of $\langle P_{i-1}, T_{i-1} \rangle$, $\langle P_{i+1}, T_{i+1} \rangle$ and $\langle P_{i+2}, T_{i+2} \rangle$, each user U_i verifies as follows:

$$e\left(\sum_{k \in \{-1,1,2\}} T_{i+k},\ P\right) = e\left(\sum_{k \in \{-1,1,2\}} (P_{i+k} + h_{i+k} Q_{i+k}),\ P_{\text{pub}}\right) \qquad (1)$$

If the above equation is satisfied, then U_i computes $D_i = e(a_i(P_{i+2} - P_{i-1}),\ P_{i+1})$ and broadcasts D_i to all others. Otherwise U_i stops.

2.2.3. *Key computation*

Each user U_i computes the session key, $K_i = e(a_i P_{i-1},\ P_{i+1})^n \cdot D_i^{n-1} \cdot D_{i+1}^{n-2} \cdots D_{i-2}$.

However the protocol above has the vulnerability that any malicious users can impersonate an entity to agree some session keys in a new group if these malicious users have the previous authentication transcripts of this entity. So, an active adversary can collude these malicious users to masquerade the victim without being detected in the MANET environment. More details can be found in Zhang and Chen (2004). An improved scheme using synchronous counter is shown in Du et al. (2003). In spite of such improvements, Shim (2007) showed the verification of all the messages from all participants is required to prevent the insider impersonation attack.

2.3. *Security requirements*

In this section, we describe security requirements for the AGKA protocols that consist of two parts. The first part is previous requirements, and the other is our additional requirements which enhance the security of the AGKA protocols.

2.3.1. *Previous requirements*

Following security requirements are shown in Shim (2007), Katz and Shin (2005), Pereira and Quisquater (2003).

2.3.1.1. *Implicit key authentication.* When a participant completed his role in a protocol session, each $M_i \in M$ is assured that no party $M_q| \in M$ can learn the key $S_n(M_i)$ (i.e. M_is view of the key) unless helped by a dishonest $M_j \in M$.

2.3.1.2. *Resistance to known-key attacks*. A protocol is said to be vulnerable to a known-key attack if compromise of past session keys allows either a passive adversary to compromise future session keys, or impersonation by an active adversary in the future.

The above three requirements are essential for the group key agreement protocol, which are similarly defined in Saeednia and Safavi-Naini (1998). Also, following requirements are also defined.

2.3.1.3. *Prevent insider impersonation attack*. The malicious insiders should not be able to impersonate other users in order to participate key agreement protocol.

2.3.1.4. *Prevent insider different key attack*. This type of attack is shown in Tang and Mitchell (2005). Any active malicious participants should not be able to manipulate the communication, who make any other participants compute the session key to be any value $K' \in G$.

Katz and Shin showed the formal model of the security against *insider impersonation attacks* in Katz and Shin (2005). And, Shim (2007) showed that each user should authenticate all participating entities. The security model in Katz and Shin (2005) is claimed to be impractical because the UC-compiler in their design requires additional round and $\mathbb{O}(n)$ signature verifications.

2.3.2. *Proposed new additional requirements*

In the practical implementation, even though the attack is failed, the malicious insiders might repeat the attack to obstruct the key agreement protocol. In this case, the participants may never complete the protocol with a common key. Therefore, it is necessary for the key agreement protocol to detect cheating and identify cheaters. The following is our proposed additional requirements for the key agreement protocol.

2.3.2.1. *Extended insider different key attack*. We extend the concept of insider different key attack to the key agreement protocol in which the attacker may deliberately contribute the incorrect value making all other participants compute the incorrect session key. The ultimate purpose of the attacker is making protocol not be completed.

2.3.2.2. *Detection of cheating*. If there is an attack to the key agreement protocol, it should be detected, regardless the source of the attack is from outside or inside.

2.3.2.3. *Identifying cheaters*. Whenever the attack is detected, the malicious insider, if exists, should be identified.

With these above additional requirements, the key agreement protocol is more secure and even able to prevent further attacks happened again. In order to detect and identify the malicious cheaters, we propose the trusted arbiter as the entity in the protocol. We define a trusted third party who only involves in the protocol only when the cheating has been occurred. We will describe more details in the following section.

3. How to detect and identify cheaters in AGKA protocols in MANET

We would like to propose new methods for how to detect and identity cheaters in the AGKA protocols in the MANET environment and introduce the new entity, the *trusted arbiter* (TA) that can be a trusted mobile operator or services providers in MANET.

Our TA works as follows. TA involves in the protocol only when the cheating is found. In this situation, TA needs to collect the broadcasted messages from every participant during the communication.

We let TA act as the judge or the key escrow agent. For instance, in case of TA whose role as the judge, if the cheating is found, every participant is required to send their secret parameters used during the key agreement protocol to TA. On the other hand, if TA's role as the key escrow agent, participants send the secret parameters when the protocol begins. With the secret parameters, TA is able to identify the malicious insiders. Practical application of these two cases are described below:

3.1. *TA as a judge*

We assume that a business group who wishes to have a secure tele-conference with each other. Some malicious participants deliberately sabotage the tele-conference. Consequently, other participants fail to make such conference. In such conflict, they detect the failure and request TA for help. With the submitted data provided by all participants, TA is able to identify the malicious participants regardless of their numbers.

3.2. *TA as a key escrow agent*

Consider several government offices join to work on some serious project with the supervision of an arbiter. Sometimes, malicious officers deliberately sabotage the key agreement process for some purposes. Due to confidential requirement in the government offices, the secret data during the key agreement process may not be given to any outside. Therefore, TA as a key escrow agent, if exists, easily detects and identifies malicious officers regardless of their numbers.

We apply our approaches to improve (Burmester and Desmedt) and (Choi et al., 2004) for detecting and identifying the malicious insiders in the MANET environment.

3.3. *Trusted arbiter as a judge*

When TA acts as the judge, he involves only when the cheating is reported in MANET. We assume that, in the key agreement protocol, each user U_i has their own identity UID_i, public key pairs (pk_i, sk_i), where pk_i is U_i's public key, which is known to public including participants in G and sk_i is U_i's private key, kept in secret. With sk_i, U_i can generate a signature using any secure signature schemes.

3.3.1. *Enhancement of Brumester–Desmedt protocol*

We assume a group of $t \le n$ users (typically $t \ll n$), derive a common conference key K in MANET. Without loss of generality, the users are labeled from U_0 through U_{t-1}, and all indices i indicating users are taken modulo t. We denote the timestamp of user U_i as TS_i, the signature generated by each user as sig_i. Let G be a group of participated users, where $G = \{U_0, \ldots, U_{t-1}\}$.

3.3.1.1. *One-time setup*. An appropriate prime p and a generator α of $\mathbb{Z}_p^*$ are selected, and published to all group of users.

3.3.1.2. *Group setup*. Each user U_i agrees to generate the group key, and shares the group information G, which $G = \{U_0, \ldots, U_{t-1}|Session\}$. *Session* defines the times start, time expiration, for each run of the key agreement process. We use the timestamp for the freshness checking.

After the G is known to participants U_i, $0 \le i < t$, proceed followings.

3.3.1.3. *Round 1 broadcasts* z_i. Each user U_i generates a random integer r_i and computes $z_i = \alpha^{r_i}$. U_i generates $sig_{z_i} = sign_i(UID_i|z_i|G|TS_i)$ and broadcasts $\{Z_i, sig_{z_i}\}$.

3.3.1.4. *Round 2 broadcasts* X_i. Each user U_i receives $\{Z_{i-1}, sig_{z_{i-1}}\}$ and $\{Z_{i+1}, sig_{z_{i+1}}\}$. And then, U_i verifies sig_{i-1} and sig_{i+1}. If the verification is correct, U_i generates $X_i = (z_{i+1}/z_{i-1})^{r_i} \bmod p$ and the signature $sig_{X_i} = sign_i(UID_i|X_i|G|TS_i)$. U_i broadcasts $\{X_i, sig_{X_i}\}$.

3.3.1.5. *Round 3 compute the group key*. After receiving $\{X_i, sig_{X_j}\}$, U_i verifies sig_{X_j} and $0 \le j < t$ excluding $j = i$, U_i computes $K = K_i$ as $K_i = (z_{i-1})^{tr_i} \cdot X_i^{t-1} \cdot X_{i+1}^{t-2} \ldots X_{i+(t-3)}^2 \cdot X_{i+(t-2)}^1 \cdot \bmod p$.

3.3.1.6. *Key confirmation*. U_i broadcasts $\sigma_i = h(UID_i||K_i||G)$ and $sign_i(\sigma_i)$, where || denotes the concatenation. Each user U_i compares $h(UID_j||K_i||G)$ with σ_j, where $0 \le j < t, j \ne i$. If the result is correct, then the group succeeds in the group key generation, and shares the key K. Otherwise, the key confirmation is failed. Fig. 1 depicts this process. If the key confirmation is failed, each user U_i sends his/her r_i and the signature $sig_i(r_i)$ to TA, who can identify the cheaters as shown in Fig. 2. We describe more details in Section 3.3.

3.3.2. *Enhancement of Choi et al.'s protocol*

In this section, we improve Choi et al.'s scheme. At first, we proceed following steps.

- **Setup:** The Key Generation Center (KGC) chooses a random number $s \in \mathbb{Z}_q^*$ and sets $P_{pub} = sP$. KGC keeps s as the master key, which is known only by itself.
- **Extraction:** A user submits his identity information $ID \in \{0,1\}^*$ to KGC. KGC computes the user's private key $S_{ID} = sQ_{ID}$ and sends it to the user via a secure channel, here $Q_{ID} = H_1(ID)$.

Let $\{U_i|i = 0, 1, \ldots, t-1\}$ be a set of t users who would like to share a session key. Suppose ID_i denotes the identity information of the user U_i. The indices are subject to modulo t. Let U_i's long-term public key and private key be $Q_i = H_1(ID_i)$ and $S_i = sQ_i$, respectively. We denote the timestamp of user as TS_i, the signature of each user as sig_i. Let G be a group of participated users, where $G = \{U_0, \ldots, U_{t-1}\}$.

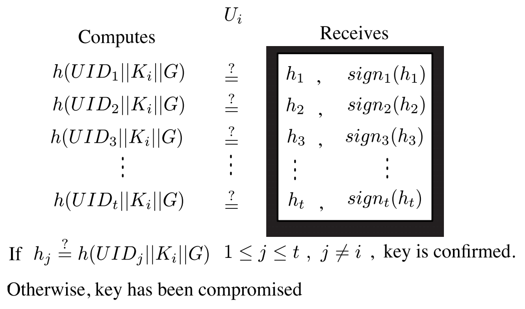
3.3.2.1. *Round 1*. Each user U_i picks a random integer $a_i \in \mathbb{Z}_q^*$ and computes $P_i = a_iP$, $h_i = H(P_i||G||TS_i)$ and $T_i = a_iP_{pub} + h_iS_i$. Each user U_i broadcasts $\{UID_i, G, TS_i, \langle P_i, T_i\rangle\}$ to all others and keeps a_i secret.

3.3.2.2. *Round 2*. Upon the receipt of $\{UID_{i-1}, G, TS_{i-1}, \langle P_{i-1}, T_{i-1}\rangle\}$, $\{UID_{i+1}, G, TS_{i+1}, \langle P_{i+1}, T_{i+1}\rangle\}$ and $\{UID_{i+2}, G, TS_{i+2}, \langle P_{i+2}, T_{i+2}\rangle\}$, each user U_i verifies as Eq. (1), where $h_{i+k} = H(P_{i+k}||G||\ TS_{i+k})$.

If the above equation is satisfied, then U_i computes $D_i = e(a_i(P_{i+2} - P_{i-1}), \ P_{i+1})$, $sig_{D_i} = sign_{S_i}(UID_i|D_i|G|TS_i')$ and broadcasts $\{D_i, UID_i, G, TS_i', sig_{D_i}\}$ to all others, where TS_j' is a new timestamp that is generated by U_i. Otherwise U_i stops.

3.3.2.3. *Key computation*. Each user U_i verifies D_j with UID_j, G, TS'_j, sig_{D_j}, $0 \le j \le t-1$ where $j \ne i$ and computes the session key as follows.

$$K_i = e(a_iP_{i-1}, \ P_{i+1})^n D_i^{n-1} D_{i+1}^{n-2} \ldots D_{i-2}$$



U_i
Computes
Receives
$h(UID_1||K_i||G)$ $\overset{?}{=}$ h_1 , $sign_1(h_1)$
$h(UID_2||K_i||G)$ $\overset{?}{=}$ h_2 , $sign_2(h_2)$
$h(UID_3||K_i||G)$ $\overset{?}{=}$ h_3 , $sign_3(h_3)$
$h(UID_t||K_i||G)$ $\overset{?}{=}$ h_t , $sign_t(h_t)$
If $h_j \overset{?}{=} h(UID_j||K_i||G)$ $1 \le j \le t$, $j \ne i$, key is confirmed.
Otherwise, key has been compromised

Fig. 1 – Each U_i confirms the generated key.

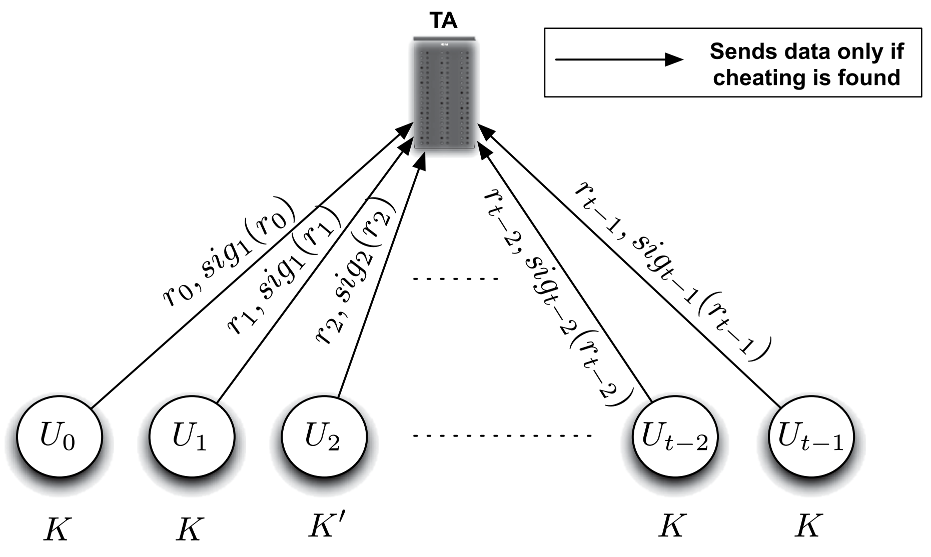


Fig. 2 – All participants send their r_i to TA to identify cheaters.

3.3.2.4. *Key confirmation.* U_i broadcasts $\sigma_i = h(\mathrm{UID}_i||K_i||\,G)$ and $\mathrm{sign}_i(h(\sigma_i))$. Every user compares $h(\mathrm{UID}_j||\,K_i||G)$ with σ_j, where $0 \le j < t, j \ne i$. If the result is correct, then the group succeeds in the group key generation, and shares the key K. Otherwise, the key confirmation is failed. Fig. 1 depicts this process. If the key confirmation is failed, each user U_i sends his/her a_i and the signature $\mathrm{sig}_i(a_i)$ to TA, who can identify the cheaters. We describe more details in Section 3.3.

3.4. *Trusted arbiter as a key escrow agent*

When TA acts as the key escrow agent, protocols are slightly different. Before the protocol begins, participants notify the group of users G to TA. At the **Round 1**, all participants send their random integer r_i (BD protocol) or a_i (Choi et al.'s protocol) to TA. With these secret information, TA can detect and identify the cheaters if the cheating happens. Remained steps follow the previous section.

3.5. *Detecting and identifying cheaters*

If no attacks have been occurred, every user will share the same group key K. However, if the attacks are found, users may have different keys, so the key confirmation is failed and the cheating is detected.

In order to identify the cheaters, as we mentioned in the previous section, TA involves in the protocol. Firstly, TA collects all the protocol transcripts. For instance, TA as the judge, the random integers used during the protocol will be sent to TA by every user after the key confirmation is failed. When acting as the key escrow agent, TA already has all the random integers at the beginning of the protocol session.

From the random integers, TA identifies the cheaters as follows. We assume that TA has the valid transcripts verified by all the users. TA re-generates the protocol messages with the received random integers and checks against the collected messages. If there is inconsistency between two sets of the protocol messages, TA easily identifies the malicious insiders.

Moreover, when TA acts as the key escrow agent and follows the protocol steps, the cheating can be detected earlier without waiting for the key confirmation. To do this, TA collects the broadcast information from each users during the protocol execution. Using the random integers sent by users before, TA can verify if those broadcast values are correct or not. If he finds something is going wrong, he can know the owner of the original data. Cheaters cannot deny since there exists signatures on data sent by them. By this way, whenever the cheating occurs, TA can take an action as soon as possible.

4. Security analysis

In this section, we conduct the security analysis for our enhanced design against various attacks such as implicit key authentication, forward security, replay attacks, insider impersonation attack, insider different key attack, and how to detect and identify cheaters in MANET.

4.1. *Implicit key authentication*

If there is no dishonest insider who leaks the key generation information, then outsiders of the group should be able to find at least one random integer r_i in the BD protocol, or a_i in the Choi et al.'s protocol to succeed to the attack. However, finding r_i given z_i or a_i given P_i is equivalent to solve discrete logarithm problem (DLP) (Koblitz, 1987; Odlyzko, 1984; Smart, 1999). Therefore, our enhanced design provides implicit key authentication due to the hardness of DLP.

4.2. *Forward and backward security*

In each group setup session, a participant picks a fresh random integer which differs from that in other session, which results in the shared group key is different in each time. Therefore, even a group key in a session is compromised, it does not provide any useful information to compute previous or future session keys. It also provides the resistance to known-key attacks.

4.3. *Insider impersonation attack*

The trial of impersonation may be happened twice; one in **Round 1**, the other in **Round 2**. We utilize the digital signature in both rounds to ensure the integrity of the protocol messages. If an adversary tries to impersonate U_i by sending forged z_i' and X_i', the adversary should be able to generate the signature for successfully passing the verification step performed by each participant. Thus, our enhanced BD protocol is secure against the insider impersonation attack.

Consider the impersonation attack on Choi et al.'s protocol shown in Zhang and Chen (2004). Two malicious insiders U_{i-1} and U_{i+2} can co-work to generate U_i's D_i from the known values P_i, P_{i+1} as follows:

$$\begin{aligned} D_i &= e(a_i(P_{i+2} - P_{i-1}), P_{i+1}) = e(P, P)^{a_i(a_{i+2}-a_{i-1})a_{i+1}} \\ &= e(a_iP,\ a_{i+1}P)^{-a_{i-1}+a_{i+2}} = e(P_i, P_{i+1})^{-a_{i-1}+a_{i+2}} \end{aligned}$$

The similar attack still happens as shown in Shim (2007) even though **Round 1** of Choi et al.'s protocol utilizes timestamp technique as suggested in Zhang and Chen (2004). Malicious users U_{i-2}, U_{i-1}, and U_{i+1} collude to impersonate U_i in the group G. In **Round 1**, colluding members broadcast $\langle P_i,\ T_i = R\rangle$ impersonating U_i.

In **Round 2**, U_j $(j \neq i)$ verifies $\langle P_{j-1},\ T_{j-1}\rangle$, $\langle P_{j+1},\ T_{j+1}\rangle$, and $\langle P_{j+2},\ T_{j+2}\rangle$. At that time nobody only knows the invalidity of $\langle P_i,\ T_i\rangle$ except U_{i-2}, U_{i-1}, U_{i+1}. In **Round 2**, when U_j $(j \neq i)$ broadcasts D_j, those colluding users broadcast $D_i = e(a_i(P_{i+2} - P_{i-1}), P_{i+1})$ with a_i is chosen by U_i previously and impersonate U_i successfully.

However, in our enhanced design, D_i is sent along with sig_{D_i}, the signature generated by U_i. If colluded users want to succeed in the above impersonation attack, they have to be able to generate sig_{D_i}. With the assumption that U_i uses a secure digital signature scheme, it is hard for the colluded users to generate the fake signature of U_i. Hence, our schemes can prevent such attacks (Zhang and Chen, 2004; Shim, 2007).

4.4. *Replay attacks*

In the replay attack, the attacker tries to participate to the group key agreement by re-sending the old message. The replay attack can happen in the same group or in other group of users. The attack is already shown in Nam et al. (2006). In our enhanced protocols, key generating parameters in each round are sent along with user's identity UID_i, group identity G, and a timestamp TS_i. G prevents the message's malicious usage in other group impersonating the U_i. Also, timestamp TS_i keeps the freshness of the message. Alternatively, we can use the sequence number or nonce instead of timestamp, if time synchronization is infeasible within the key exchange environments. With these techniques, our enhanced protocols are secure against the replay attack.

4.5. *Insider different key attack*

Insider different key attacks consist of two types. The first type is mentioned in Tang and Mitchell (2005). We will describe the second type later on.

The first attack is able to be applied to the original BD scheme. A malicious participant, say U_j $(1 \leq j < t)$, who can manipulate the communications in the network, is able to make any other participant, say U_i $(0 \leq i < t, i \neq j)$, compute the session key to be any value $K^* \in G$ chosen by U_j. To achieve this, in **Round 2**, U_j intercepts the message X_{i-t+2} and prevents it from reaching U_i. U_j then waits until all the other messages have been received and computes the session key K in the normal way. U_j now sends $X'_{i+t-2} = X_{i+t-2} \cdot K^*/K$ to U_i pretending that it comes from U_{i+t-2}. Finally, U_i generates the incorrect key K^*.

However, in our enhanced BD protocol, U_j has to generate the signature to convince U_i that X'_{i+t-2} is actually from U_{i+t-2}. Even U_j is U_{i+t-2} so that U_{i+t-2} can generate the incorrect X'_{i+t-2} along with the valid signature, U_i can detect the cheating in the key confirmation step by comparing other $h(UID_k||K_i||G)$, where $0 \leq k < t$, $k \neq i$. This type of attack is also applicable to Choi et al.'s original protocol, while our enhanced protocol overcomes this attack by the same principle.

As we mentioned *extended insider different key attack* in Section 2.3.2, we describe our attack as follows. In this type of attack, the malicious insider can make all other users in the group compute the different key from each other. Therefore the group user never agrees upon the common group key. Suppose that U_j wants to perform this type of attack. U_j first sends the initial information in the **Round 1**, however he sends the different information in the **Round 2**. For example, in the BD protocol, U_j generates z_j, where $z_j = \alpha^{r_j}$, but generates X'_j, where $X_j' = (z_{j+1}/z_{j-1})^{r_j'}$ for some $r' \neq r$. Then, every user including U_j generates all distinct keys. Similar attack can be applied to the Choi et al.'s protocol.

Whereas, in our enhanced protocols, any user U_i will detect this attack with *key confirmation* process. Once the attack is detected, with the help from TA, the malicious insiders can be identified and may be excluded from participating group key agreement process.

4.6. *Analysis for detecting and identifying cheaters in MANET*

As mentioned in Section 2.3.2, a secure group key agreement protocol should be not only secure against various attacks, but also able to detect if there is a cheating and identify all cheaters. Supported by these functionalities, the group key agreement can stop attacks from outside as well as cheating from inside in the MANET environment. Moreover, by identifying cheaters, the protocol prevents malicious users from sabotage group key agreement process. Our approach using TA enables the group key agreement protocol to provide detecting and identifying cheaters features. We analyze these features in our enhancement versions of BD and Choi's protocols. In order to detect cheating, we use key confirmation process.

From receiving the hash value $h(UID_j||K_i||G)$ and the accompanied signature sign_i $(h(UID_j||K_i||G))$ from other user, where $0 \leq j < t, j \neq i$, U_i can check whether his computed key is identical to that of others. If the incorrect group key is detected, the cheating has been occurred in MANET. Then, the process of identifying cheaters should be carried out with involvement of TA. TA should have all r_i (or a_i) which are used in the protocol session sent by all participants in the MANET environment.

TA identifies the cheaters as follows: We describe in BD scheme as an example. TA re-generates the protocol messages with r_i, where $0 \le i < t$, and compares the re-generated messages with the collected messages from **Round 1** and **Round 2**. If there is discrepancy between two sets of the messages in each round, TA can easily identify the malicious insiders. For example, in the enhanced BD protocol, TA computes $z_i^* = \alpha^{r_i}$ and $X_i^* = (z_{i+1}/z_{i-1})^{r_i}$ with r_i, and compares them with the collected z_i and X_i. If TA finds the discrepancy between z_i and z_i^*, or X_i and X_i^*, then U_i must be the cheater who contributed the incorrect values. With the signatures on z_i and X_i, the malicious participants cannot deny their cheating in MANET.

5. Conclusion

Dynamic, heterogeneous and distributed MANET environment will create new opportunities, through the convergence of communications technologies and creation of highly adaptive reconfigurable devices. Increased mobility results in interesting new security challenges.

Due to infrastructure-less nature of the MANET environment, Light weight asymmetric techniques such as ID-based crypto systems could provide intelligent facilities for securing MANET environments. ID-based systems require no explicit public key available and the key is constructed from public available information. It is an asymmetric system where unique name plays the role of the public key. These characteristics of ID-based techniques make it very suitable for the MANET security architecture and applications.

Thus, we propose a novel authenticated group key agreement protocol for end-to-end security in the MANET environment without any infrastructure that is based on Burmester and Desmedt group key agreement protocol (Burmester and Desmedt) and their variants (Choi et al., 2004) that is based on ID-based crypto system.

We also showed our new requirements such as the extended insider different key attack, detecting and identifying cheaters in the group key agreement in the MANET environment, and proposed the novel enhanced protocols with introducing the trusted arbiter for identifying cheaters in MANET.

With signing the communication of each user and confirming the key, our novel design provides not only the functionality of detecting malicious insiders, but also, with involving the trusted arbiter, identifying all the cheaters in the protocol regardless of the number of cheaters in the MANET environment.

REFERENCES

Burmester M, Desmedt Y. A secure and efficient conference key distribution system. Advances in Cryptology – EuroCrypt ’94, Lecture Notes in Computer Science 1995;950:275–86.

Choi KY, Hwang JY, Lee DH. Efficient id-based group key agreement with bilinear maps. Public Key Cryptography – PKC, Lecture Notes in Computer Science 2004;2947:130–44.

Du X, Wang Y, Ge J. An improved id-based authenticated group key agreement schemes. Cryptology ePrint Archive 2003;260. Report.

Just M, Vadenay S. Authenticated multi-party key agreement. Advances in Crytopology-AsiaCrypt ’96, Lecture Notes in Computer Science 1996;1163:36–49.

Katz J, Shin JS. Modeling insider attacks on group key-exchange protocols. Proceedings of the 12th ACM conference on computer and communications security; 2005. p. 180–9.

Koblitz N. Elliptic curve cryptosystems. Mathematics of Computation 1987;48:203–9.

Nam J, Kim S, Won D. Weakness in jung, et al.’s ID-based conference key distribution scheme. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences 2006;E89-A(1):213–7.

Nam J. Enhancing security of a group key exchange protocol for users with individual passwords. Cryptology ePrint Archive 2007;166. Report.

Odlyzko A. Discrete logarithms in finite fields and their cryptographic significance. In: Advances in cryptology – Eurocrypt ’84. Springer-Verlag; 1984. p. 224–314.

Pereira O, Quisquater J-J. Some attacks upon authenticated group key agreement protocols. Journal of Computer Security 2003; 11(4):555–80.

Saeednia S, Safavi-Naini R. Efficient identity-based conference key distribution protocols, ACISP’98. Lecture Notes in Computer 1998;1438:320–31.

Shim K-A. Further analysis of id-based authenticated group key agreement protocol from bilinear maps. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences 2007;E90-A(1):295–8.

Smart N. The discrete logarithm problem on elliptic curves of trace on. Journal of Cryptology 1999;12.

Steiner M, Tsudik G, Waidner M. Cliques: a new approach to group key agreement. Proceedings of the 18th International Conference on Distributed Computing Systems (ICDCS’98); 1998. p. 380–7. Available from: citeseer.ist.psu.edu/steiner98cliques.html.

Tang Q, Mitchell CJ. Security properties of two authenticated conference key agreement protocols. Cryptology ePrint Archive 2005;185. Report.

Zhang F, Chen X. Attack on an id-based authenticated group key agreement scheme from pkc 2004. Information Processing Letters 2004;91:191–2.