Mobile Information Systems 7 (2011) 241–269
DOI 10.3233/MIS-2011-0120
IOS Press

A handover security mechanism employing the Diffie-Hellman key exchange approach for the IEEE802.16e wireless networks

Yi-Fu Ciou[a], Fang-Yie Leu[a,*], Yi-Li Huang[a] and Kangbin Yim[b]
[a]*Department of Computer Science, Tunghai University, Tunghai, Taiwan*
[b]*Soonchunhyang University, Asan, Republic of Korea*

Abstract. In this paper, we propose a handover authentication mechanism, called the handover key management and authentication scheme (HaKMA for short), which as a three-layer authentication architecture is a new version of our previous work, the Diffie-Hellman-PKDS-based authentication method (DiHam for short) improving its key generation flow and adding a handover authentication scheme to respectively speed up the handover process and increase the security level for mobile stations (MSs). AAA server supported authentication is also enhanced by invoking an improved extensible authentication protocol (EAP). According to the analyses of this study the HaKMA can effectively and efficiently provide user authentication and balance data security and system performance during handover.

Keywords: HaKMA, DiHam, PKM, WiMax, IEEE802.16, wireless security

1. Introduction

Recently, due to their popularity and the characteristics of convenience and high access speed wireless networks have been a part of our everyday life. Through wireless systems, people can surf web contents, send emails and watch video program outdoors anytime anywhere. To satisfy the requirements of high-speed mobile wireless networks the IEEE 802.16 Working Group in 2005 developed the IEEE 802.16e standard [1] known as the WiMax system which is an extended version of IEEE 802.16 by adding mobility management and a handover scheme so as to provide users with mobile broadband wireless services

To prevent malicious attacks, the IEEE802.16 standard employs a key management and authorization mechanism called privacy key management (PKM) to authenticate users and wireless facilities [9, 26]. However, several problems have been found [17] on this mechanism, such as lack of mutual authentication, and having authorization vulnerabilities and key management failures Also, the high complexity of its authentication mechanism and the presence of design errors [17] make the PKM fail to effectively protect the wireless system. To solve these problems, the IEEE Network Group proposed the PKMv2 in 2005 to fix the defects of the PKMv1 by adding mutual authentication and EAP support. But this enhancement also makes the PKMv2 more complicated and difficult to fix than the PKMv1 if

*Corresponding author: Fang-Yie Leu, Department of Computer Science, Tunghai University – No. 181, Section 3, Taichung Port Road, Taichung City 40704, Taiwan. Tel.: +886 4 2359 0121#33815; E-mail: leufy@thu.edu.tw.

someday new shortcomings are found. On the other hand, Leu et al. [22] proposed a Diffie-Hellman-PKDS-based authentication method (DiHam for short) to improve some of the defects. However, the scheme does not guarantee full security since it only considers the initial network entry without providing handover and user authentication.

Therefore, in this paper we propose a handover authentication mechanism, called the handover key management and authentication system (HaKMA for short), which as a three layer authentication architecture is an extended version of the DiHam, one of our previous works by improving the key generation flow and adding a handover authentication scheme to respectively speed up the handover process and increase the security level for mobile stations (MSs). It also enhances the AAA server authentication by employing an improved version of the extensible authentication protocol (EAP). To meet different security levels of wireless communication, two levels of handover authentication are proposed. The analytical results show that the HaKMA is more secure than both the DiHam and the PKMv2.

The key contributions of this study are as follows.

(1) According to the security analyses and performance analyses of this study, the security level of the proposed the HaKMA is higher than those of the DiHam, and PKMv2. The authentication cost of the HaKMA is between the PKMv2 and DiHam.
(2) We apply a fast encryption and decryption scheme to protect wireless EAP messages, but keep the advantages of the PKMv2, such as mutual authentication and EAP support.
(3) The HaKMA provides fast and low cost handover authentication instead of skipping the handover authentication process.

The rest of this paper is organized as follows. Section 2 introduces the background and related work of this study. Sections 3 and Section 4, respectively describe the proposed handoff approaches for the DiHam and HaKMA Security analyses are presented in Section 5. Section 6 describes system simulation and gives discussion. Section 7 concludes this paper and outlines our future research.

2. Background and related work

2.1. The WiMax network architecture

Figure 1 shows a modern multi-layer wireless network configuration. The access service network gateway (ASN-GW) is connected to a network service provider (NSP) backbone network, and BSs are directly linked to their ASN-GWs. An ASN-GW can not only communicate with other ASN-GWs via the backbone network through R3 reference points, but also directly communicate with other ASN-GWs with direct links via R4 reference points [7,11]. An NSP may provide many ASN-GWs to serve users. An MS may currently link to a BS, or hand over between two BSs under the same ASN-GW, called the Intra-ASN-GW handover, or under different ASN-GWs, called the Inter-ASN-GW handover.

2.2. Privacy key management protocol

The PKM protocol first specified by the IEEE 802.16-2004 provides device authentication also known as facility authentication, and the PKMv2 proposed in the IEEE 802.16e-2005 is a new version of the PKM protocol which corrects design errors for security found in the PKMv1 [11] and supports user authentication.

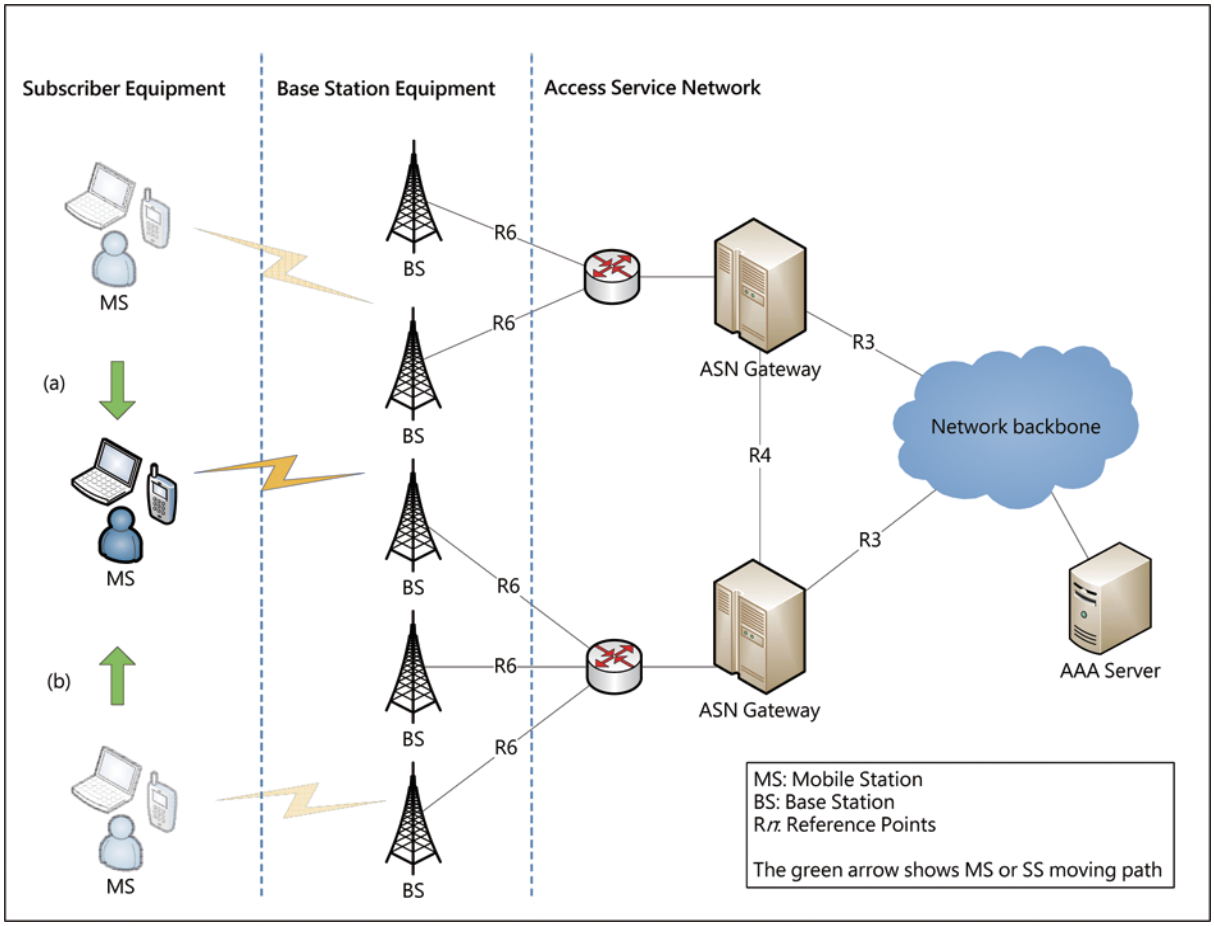

Fig. 1. The WiMax network configuration in which MS may perform an Inter-ASN-GW handover and an Intra-ASN-GW handover.

2.2.1. PKMv2

The PKMv2 uses X.509 digital certificates [15] together with either an RSA public-key encryption algorithm or a sequence of RSA device authentication to further authenticate communication facilities. After that, an EAP method is employed to authenticate users [18]. The encryption algorithms used by PKMv2 for key exchange between MS and the BS are more secure than those used by the PKMv1 [1].

The PKMv2 allowing mutual authentication or unilateral authentication, also supports periodic re-authentication/re-authorization and key renew. All PKMv2 key derivations are performed based on the Dot16KDF algorithm defined in IEEE802.16e standard [1].

Figure 2 illustrates the initial network entry procedure of the PKMv2 which consists of 9 steps [1,2, 20,21]:

(1) **Initiation of network entry**: In this step, MS performs the physical layer initialization process including initial ranging and network entry. After the wireless link between the BS and MS is established, Authenticator starts the EAP exchange step.

(2) **EAP exchange**: In this step, MS and the Authenticator (which could be in the BS or ASN-GW) negotiate with each other to choose a suitable EAP method through EAP Request/Identity and

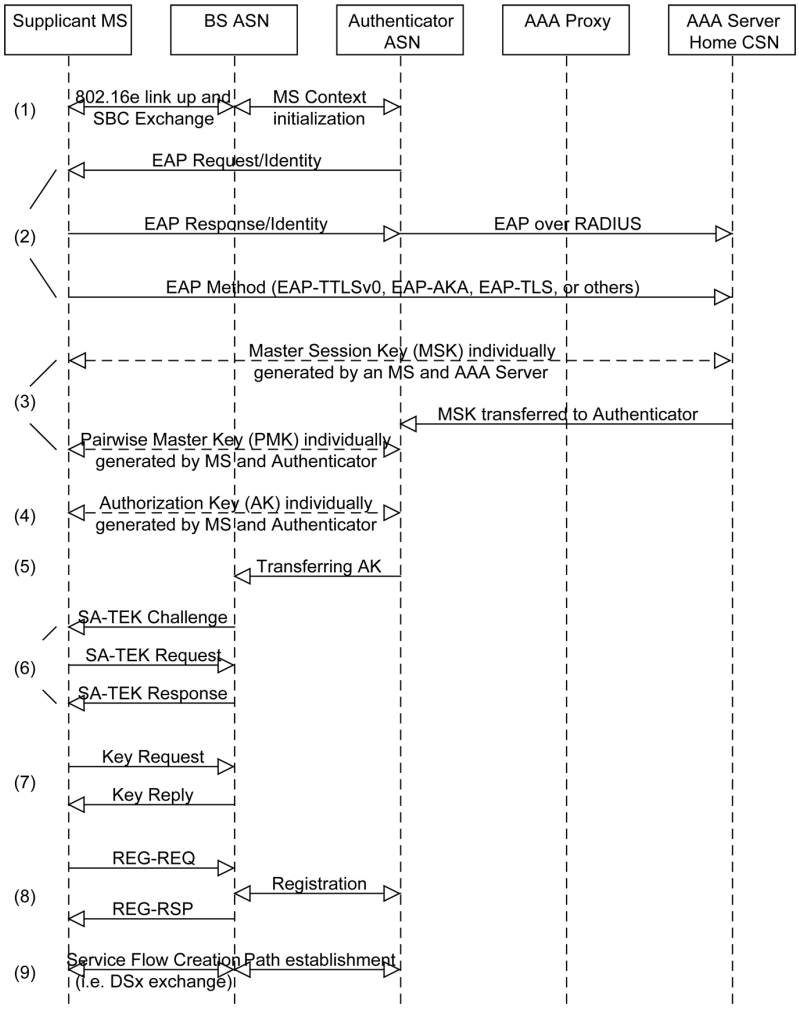

Fig. 2. The PKMv2 authentication procedure [2], where SBC stands for MS basic capability.

EAP Response/Identity messages. With the chosen method they mutually exchange certification messages so that the AAA server can authenticate the user.

(3) **MSK establishment**: In step 3, a master session key (MSK) is established between MS and the AAA server. The AAA server transfers the MSK to the Authenticator through a secure path [2,11] using the Radius protocol With the MSK, both the BS and Authenticator selfgenerate a pairwise master key (PMK).

(4) **Authorization Key (AK) generation**: Authenticator and MS individually generate the AK by using the PMK [1]. If the RSA-based authorization process is used, a pre-primary authentication key (pre-PAK) will be produced. The AK is derived from the pre-PAK. If RSA-based and EAP-based authorization is employed, a MSK and a pre-PAK will be generated. The AK is then calculated by using the two keys. If only EAP-based authorization is involved, only the MSK is generated, and the AK is derived from the MSK.

(5) **AK transfer**: Authenticator delivers the AK other keys (e.g., pre-AK and the MSK generated by itself) and parameters (e.g., MSK lifetime) to the serving BS. The BS caches them for the

following steps.

(6) **AK liveliness establishment and SA transfer**: This step consists of three phases. In the first the BS transmits a PKMv2_SA_TEK_Challenge message which includes a BS_Random challenge and identity of the AK to MS to establish Security Association (SA) between MS and BS In the second phase, MS responds with a PKMv2_SA_TEK_Request message which contains a random number *MS_Random* and other security attributes In the third phase, the BS transmits a PKMv2_SA_TEK_Response message containing the response of requested properties listed in the PKMv2_SA_TEK_Request message to MS to complete the SA.

(7) **TEK generation and transfer**: When MS would like to deliver data messages, it sends a Key-Request message to the BS. The BS then generates Traffic Encryption Keys (TEKs), encrypts them by using the Key Encryption Key (KEK) individually generated by MS and the BS in this step, and transfers the encrypted TEKs to the MS through a Key-Reply message. For each SA, the MS needs two TEKs to download streams and upload streams. This step repeats if multiple SAs are involved. Note that Leu et al. [22] challenged the TEK transfer since hackers could easily intercept the message and then decrypt the TEKs.

(8) **Network registration**: In this step, MS sends a REG_REQ message to the BS providing Access Service Network (ASN), and the BS responds with a REG_RSP message to negotiate network parameters such as Connection ID (CID), IP version, and so on. The REG process makes MS known to Authenticator and ASN-GW, and triggers the service or data transfer process.

(9) **Service flow creation**: In this step the BS uses DSA-REQ/RSP/ACK MAC management messages to create a new service flow and map an SA to that service flow thereby associating the corresponding TEKs with it [2].

2.2.2. PKMv2 handover procedure

Based on the IEEE 802.16 standard [1] and other related documents [2], the PKMv2 handover procedure shall be optimized within the same mobility domain. Sharing TEKs among BSs inside a trustable mobility domain is possible if the handover procedure can transfer TEK context information from BSs to neighbor BSs and these BSs are treated as entities of the same security levels.

In summary the PKMv2 protocol controls security key derivation, exchange and renew, and manages user authorization states to ensure the security of the user's data communication.

2.3. Diffie-Hellman PKDS-based authentication

The DiHam [22] was developed based on the PKMv1 by improving the key exchange flow and providing different data security levels. Basically, its key exchange process as shown in Fig. 3 consists of two phases the authentication phase and TEK exchange phase.

In the authentication phase, the AK is individually generated by the BS and MS after the delivery of the AuthenticationRequest message and AuthenticationReply message [22].

At first, MS generates three random numbers and the corresponding public keys, and sends an AuthenticationRequest message to the BS. The BS on receiving the message checks the correctness of the message generates three random numbers, the corresponding public keys, and common secret keys (CSK), and sends an AuthenticationReply message that contains a challenge field and information of the encrypted keys to the MS so that MS can authenticate the BS, and derive the CSKs. The formats of the authentication messages are shown in Fig. 4.

In the TEK exchange phase, three security levels of TEK generation processes are proposed to meet different user security requirements. This phase starts when MS sends a TEK-Exchange-Request message to the BS, and ends when the BS replies with a TEK-Exchange-Reply message.

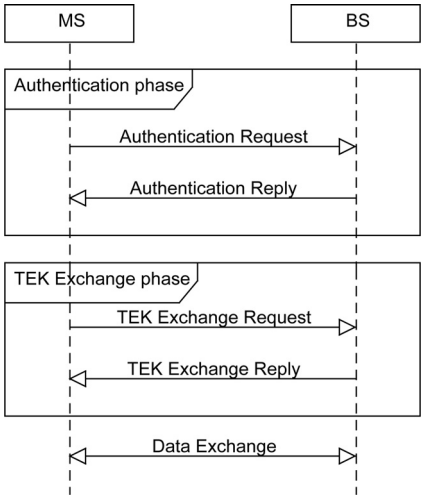

Fig. 3. The DiHam authentication process.

Authentication-Request (MS to BS):
OP_Code|Cert(MS)|P_{RM1}|P_{RM2}|P_{RM3}
Authentication-Reply (BS to MS):
OP_Code|Cert(MS)|P_{RB1}|P_{RB2}|P_{RB3}|Certfun(CSK1, CSK2)|EXOR(CSK2, pre_AK2)|
EXOR(CSK3, pre_AK3)

Fig. 4. The DiHam authentication messages deployed in the authentication phase.

Level-1 TEK-Exchange-Request (MS to BS):
OP_Code|Certfun(CSK1, pre_AK1)|*Security-capabilities*
Level-1 TEK-Exchange-Reply (BS to MS):
OP_Code|Certfun(CSK2, pre_AK2)|EXOR(CSK2, old_TEK)|EXOR(CSK3, new_TEK)|
TEK lifetime

Fig. 5. Formats of the exchanged messages for Level1 TEK.

2.3.1. Level1 TEK exchange

Level-1 TEK is designed for applications of low-security level such as web surfing. The BS is responsible for creating TEKs, encrypting TEKs with CSKs and transferring the TEKs to MS. Figure 5 shows the format of the exchanged messages

2.3.2. Level2 TEK exchange

In Level 2, MS generates a pre_TEK, and transfers the encrypted pre_TEK to the BS through a TEKExchangeRequest message. The BS then randomly chooses one of five generated TEKs and sends the TEK sequence number to MS telling MS which TEK is chosen. No keys actually used by MS and the BS are directly delivered through wireless channels. Only those deployed to generate keys are transferred. The security level of Level2 TEK is then higher than that of Level1 TEK. Level-2 TEK is suitable for voice phone calls and personal message communication. Figure 6 shows the formats of the two messages for Level-2 TEK.

Level-2 TEK-Exchange-Request (MS to BS):
OP_Code|Certfun(CSK1, pre_AK1)|EXOR(CSK2, pre_TEK)|$Security\text{-}capabilities$
Level-2 TEK-Exchange-Reply (BS to MS):
OP_Code|Certfun(CSK2, pre_AK2)|$TEK\ seq\ num$|$TEK\ lifetime$

Fig. 6. Formats of the exchanged messages for Level2 TEK.

Level-3 TEK-Exchange-Request (MS to BS):
OP_Code|Certfun(CSK1, pre_AK1)|$Security\text{-}capabilities$
Level-3 TEK-Exchange-Reply (BS to MS):
OP_Code|Certfun(CSK2, pre_AK2)|$TEK\ seq\ num$|$TEK\ lifetime$

Fig. 7. Formats of the exchanged messages for Level3 TEK.

2.3.3. Level3 TEK exchange

If a wireless communication needs a higher security level than that of Level2 TEK, Level3 TEK is then employed. In this level, the BS and MS individually generate 15 pre_TEKs and 75 TEKs. After that, the BS sends a TEKExchangeReply message that contains the sequence number of a chosen TEK for later data transfer to MS. Figure 7 shows the formats of the exchanged messages.

From the computation complexity viewpoint, a Level3 TEK exchange process consumes many more computation resources and has longer key generation delays than those of levels 1 and 2. But its security level is the highest among them.

However, the DiHam as stated above only considered initial network entry, without dealing with handover network re-entry. If it involves a handover procedure, the whole process needs to be fully performed on each handover, consequently causing serious service disruption time (SDT). Several fast handover schemes have been proposed. Mobility prediction proposed by Fülöp et al. [13] and the Client-based Mobility Frame System also introduced by Fülöp et al. [14] are two examples. Yang et al. [25] considered self-similarity in data traffic, handover, and frequency reuse to estimate the spectrum requirements of mobile networks so as to speed up communication and shorten SDT. However, the authors of these three papers did not deal with the security layer handover support. Otherwise, their re-entry delay would be long.

3. Handover involving the DiHam

A handover node, e.g., a node newly entering the system needs to perform initial ranging and authentication [1,21] before it can securely exchange data messages with a target BS. But this may interrupt the communication between MS and ASN-GW. To shorten the disruption time, a method to reduce the network re-entry delay is required. Reusing existing authentication keys and pre-calculating authentication keys are the possible methods.

If we temporarily ignore the difference between the three levels of TEKs proposed by the DiHam, and the DiHam is applied to the handover process, the target BS has to know MS's random numbers $RS_i, i =$ 1,2,3, before it can generate TEK. However, if the serving BS can deliver Cert(MS) and MS's RS_i to the target BS (we call the message delivered the Inter-BS message) before MS enters the overlapped area of the target BS's and serving BS's communication areas the generation of new authentication parameters by the target BS and MS's handover network reentry to the target BS can be performed simultaneously. Therefore, the AuthenticationReply message can be sent to MS immediately by the target BS once

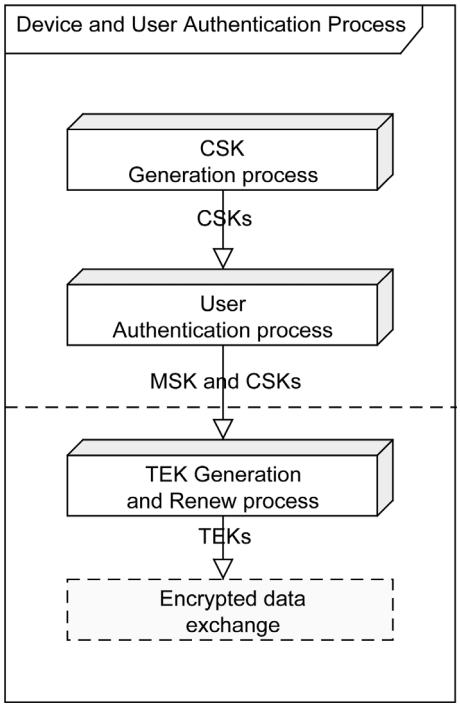

Fig. 8. Key hierarchy and sequence chart of the HaKMA authentication scheme. Processes above the dashed line are invoked by MS and Authenticator, and those beneath the dashed line are employed by MS and the BS.

the handover network reentry is completed. This change can effectively reduce handover delay, and MS's AuthenticationRequest message is no longer required since it is replaced by the Inter-BS message. Moreover, MS can also use MS's original private key RS_i and public key P_{RSi} $i = 1$–3 as well as the target BS's private keys *RBs* conveyed in the AuthenticationReply message to generate new *CSKs*, *pre_TEKs*, and then new *TEKs* to encrypt data messages.

The benefits of pre-calculating keys in the handover procedure are that the serving BS only needs to deliver MS's RS_i and Cert(MS) to the target BS and the SDT only relies on the PHYlayer handover delay [21].

According to the IEEE802.16 standard, the optimized handover can skip the security sublayer operation and reuses old keys such as TEKs [1,16], or provide handover support through mobile IPv6 with other proposed schemes [6,12,19,24]. In the DiHam if the serving BS can transfer all required security keys including CSKs Cert(MS), and RS_i to the target BS, the target BS can then reuse the TEKs currently employed by the serving BS without the requirements of re-calculating authentication keys and deriving current TEKs resulting in the fact that the AuthenticationReply TEK-Exchange and TEK-Reply messages can be further omitted. After entering the target BS's communication range and finishing its handover network re-entry MS can directly communicate with the target BS.

4. The proposed security system

The HaKMA, besides retaining the advantages of DiHam's Key Exchange and AK Generation processes [22] also involves an enhanced version of an EAP method to authenticate users. A handover support that improves the security level of the HaKMA wireless environment is added as well.

Table 1
Terms and functions used in this study

Term or function	Explanation
P	A strong prime number
g	The primitive root of P
RS_i and RA_i,$i = 1, 2$	Private keys generated by MS and Authenticator
P_{RSi} and P_{RAi},$i = 1, 2$	Public keys generated by MS and Authenticator
$CSKi$, $i = 1,2$	Common secret keys
EXOR(x, y)	Exclusive OR function, i.e., $x \oplus y$
SEXOR(*key, data*)	Stream exclusive OR function, repeating key content to match the length of $data$, and performing exclusive OR bit by bit.
Certfun(a,b,. . .)	Modulus function, i.e., $g^{a+b+\cdots} \mathrm{mod} P$
Encrypt(*PubKey, message*)	The standard RSA-OAEP-Encryption function for encrypting *message* into *ciphertext* with given public key *PubKey*
Decrypt(*PrivKey, ciphertext*)	The standard RSA-OAEP-Decryption function that decrypts *ciphertext* into *plaintext* with given private key *PrivKey*
ADR(a,b)	A binary adder, but ignoring the carry of the greatest significant bit

As stated above, the HaKMA architecture consists of three isolated processes (see Fig. 8): The CSK Generation process in which ASN-GW and MS mutually authenticate each other, the User Authentication process in which the AAA server authenticates users, and the TEK Generation and Renew process in which TEKs are produced to encrypt data messages. The three processes together are called the HaKMA authentication scheme. As a layered architecture any change in one of the three processes does not affect the functions of others, consequently making it easier for us to develop a new authentication process for IEEE802.16 wireless networks when some functions in one of the three processes need to be modified. The outputs of the three processes are sequentially CSKs, MSK and CSKs and TEKs. In this study, we move Authenticator from the BS to ASN-GW to simplify the HaKMA architecture and its handover process.

4.1. Initial process of network entry

With the HaKMA, when an error occurs, the BS sends an error message to MS and Authenticator. If MS has successfully completed one or two previous processes, but fails in an underlying process, the failed process is resumed from its beginning, instead of re-initiating the CSK Generation process. Of course, if the CSK Generation process fails, the HaKMA authentication scheme should be restarted. Table 1 lists terms and functions used in this study.

4.1.1. CSK generation process

The main objective of the CSK Generation process is to perform mutual authentication between Authenticator and MS, and produce two CSKs. Figure 9 shows the sequence chart. It first establishes a secure communication channel between MS and Authenticator, and completes the following steps for initial network entry. MS and Authenticator first mutually check each other's X.509 certificate, and perform a DiHam-like process to generate CSKs which are only known to MS and Authenticator and with which both sides encrypt those messages exchanged in the User Authentication process and TEK Generation and renew process. The BS recognizes a message received by accessing its OP_Code, and relays messages for MS and Authenticator without providing any authentication functions.

This process can be further divided into two phases: Authenticator-CSK phase and MS-CSK phase.

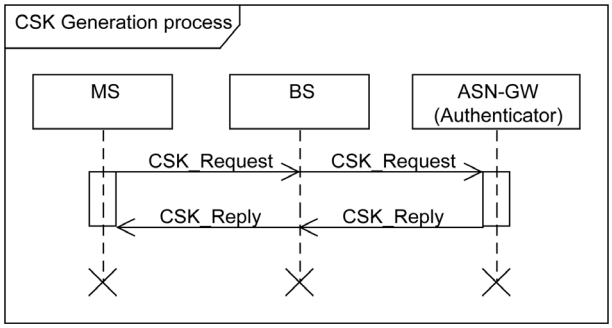

Fig. 9. Sequence chart of the CSK Generation process. BS only relays messages for MS and Authenticator.

4.1.1.1. Authenticator-CSK phase

In this phase, MS first sends a CSK_Request message the format of which is shown in Fig. 10, to Authenticator.

OP_Code| NS_{MS}| Cert(*MS*)| P_{RM1} | P_{RM2}|Capabilites|SAID|HMAC(*PubKey*(*MS*))

Fig. 10. Format of a CSK_Request message sent by MS to Authenticator.

In this message, RM_1 and RM_2, two random numbers, are private keys generated by MS, and P_{RM1} and P_{RM2} are two public keys where

$$P_{RMi} = g^{RM_i} \bmod P, 1 \leqslant i \leqslant 2 \tag{1}$$

NS_{MS}, the nonce, is a timestamp indicating when this message is created, the capabilities field lists the security configurations acceptable by MS, a SAID field contains MS's primary SAID that is currently filled with the basic CID, and the hash-based message authentication code (HMAC) function produces a message signature by inputting all fields of the message as its plaintext and *PubKey*(*MS*) as the encryption key.

Authenticator on receiving the message checks to see whether the message signature calculated by itself using *PubKey*(*MS*) retrieved from Cert(*MS*) and the HMAC(*PubKey*(*MS*)) sent by MS are equal or not. If not, implying the message has been altered, the Authenticator discards this message. If yes, it randomly selects two random numbers RA_1 and RA_2 as private keys to generate the corresponding public keys P_{RA1} and P_{RA2} where

$$P_{RAi} = g^{RA_i} \bmod P, 1 \leqslant i \leqslant 2, \tag{2}$$

It further produces two CSKs, i.e., *CSK1* and *CSK2*, where

$$CSKi = {P_{RMi}}^{RA_i} \bmod P, 1 \leqslant i \leqslant 2 \tag{3}$$

and calculates the certificate function Certfun(*PubKey*(*MS*)*, CSK1,CSK2*).

After that, Authenticator sends a CSK_Reply message of which the format is shown in Fig. 11, to MS. To ensure that the message is securely delivered, a HMAC is also added.

OP_Code|NS_{MS}| $NS_{Authenticator}$| Cert(*Authenticator*)|Encrypt(*PubKey*(*MS*), P_{RA1}|P_{RA2})|
Certfun (*PubKey*(*MS*)*,CSK1,CSK2*)|HMAC(*PubKey*(*Authenticator*))

Fig. 11. Format of a CSK_Reply message sent by Authenticator to MS.

4.1.1.2. MS-CSK phase

MS on receiving the CSK-Reply message checks to see whether the message has been maliciously modified or not by comparing not only the HMAC value calculated with the value retrieved from the CSK_Reply message received, but also NS_{MS} retrieved from the message with previous nonce involved in CSK_Request message. They should be individually equal. Otherwise, the message is discarded. MS further records $\mathrm{NS}_{Authenticator}$ for later authentication, and checks to see whether the Authenticator is trustable or not by comparing the authenticator's certificate Cert(*Authenticator*) with the certificate list provided by a trustable network provider and pre-installed in the MS. If yes, MS retrieves the public key P_{RA1} and P_{RA2} by performing RSA decryption function with its own private key, calculates CSKs, i.e., *CSK1* and *CSK2*, and the certificate function Certfun(*PubKey*(*MS*)*, CSK1,CSK2*), and then compares the calculated Certfun() value with the one conveyed on CSK_Reply message sent by Authenticator where

$$CSKi = P_{RAi}^{RM_i} \bmod P, 1 \leqslant i \leqslant 2 \tag{4}$$

If the two Certfun() values are equal, the CSK Generation process terminates. MS starts the User Authentication process. Otherwise, MS discards the message and the calculated CSKs, and waits for a valid CSK-Reply message for a predefined time period.

If MS cannot receive a reply from the Authenticator until timeout, it assumes the CSK Generation process fails and then restarts the process by re-sending a CSK-Request message to Authenticator

4.1.2. User authentication process

In this process, MS and Authenticator first negotiate with each other to choose an EAP method. After that, Authenticator communicates with the AAA server to check to see whether the user is authorized to access requested services or not.

However, EAP was originally designed for wired networks [23]. When it is applied to wireless networks, hackers may intercept and decrypt sensitive information. To solve this problem, in this study, CSKs are invoked to encrypt messages exchanged between MS and Authenticator. In fact, some EAP methods do not provide security mechanisms (e.g., EAP legacy methods). Employing our encryption scheme can ensure the security of the originally insecure EAP communication between MS and the BS so that hackers cannot easily decrypt sensitive information, even though an insecure EAP authentication method is used.

Further, we employ EAP-AKA [5] as an EAP example, and suggest employing EAPauthentication based AK generation flow [1] to balance handover performance and the security level of the system under consideration. Basically, our modular design strategy results in the fact that any authentication methods designed for wireless authentication [5] can substitute EAP-AKA to perform user authentication. A general EAP authentication process can be found in [1].

Before this process starts, MS first generates a 160-bit random number *RAND*, and derives the EAP encryption key

$$\text{EAP Encryption Key} = \text{ADR}(\text{EXOR}(\textit{CSK1,RAND}), \textit{CSK2}) \tag{5}$$

which is generated individually by MS and Authenticator and used to encrypt or decrypt EAP messages by employing a streamed Exclusive OR method/function SEXOR(*EAP Encrypt Key, EAP-messages*).

Figure 12 illustrates our User Authentication process in which MS sends a PKMv2-EAP-Start message the format of which is shown in Fig. 13, to notify. Authenticator of the start of the process. Authenticator on receiving the message extracts *RAND* by using its private key, derives the EAP encryption key, and replies with an EAP-Request/Identity message containing a list of available EAP methods that

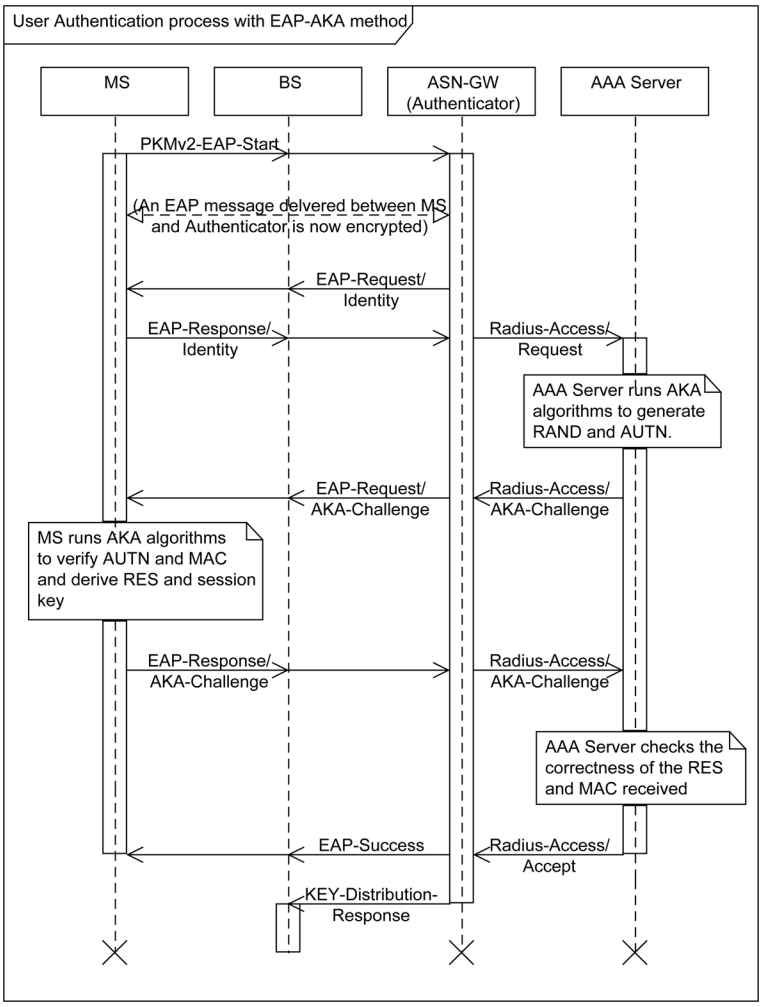

Fig. 12. The HaKMA's User Authentication process with the EAP-AKA method.

Authenticator supports to MS to start security negotiation. MS selects a suitable EAP method and sends an EAP-Response/Identity message to Authenticator. Authenticator then sends a Radius-Access request message which contains the Network Access Identifier (NAI) and other attributes conveyed in the EAP-Response/Identity message received to the AAA server via the Radius protocol. The AAA server invokes its AKA algorithms to generate a random number RAND and an AKA parameter AUTN, and then sends them to MS where AUTN is an authentication value produced by the AuC for authenticating the AAA server in the future and AuC is a mobile network element used to authenticate MS [5].

After that, the AAA server sends an AKA-Challenge message which contains AUTN to MS via Authenticator. MS on receiving the message runs its AKA algorithms to verify AUTN and MAC derives RES (an authentication result generated by MS) and the session key, and then sends an EAP-Response/AKA-Challenge message to the AAA server. After checking the correctness of the MAC and the RES received, the AAA server sends a Radius-Access/Accept message which contains an MSK 512 bits in length to Authenticator. Authenticator delivers an EAP-Success message to MS [5], indicating the user is authenticated.

OP_Code| $NS_{Authenticator}$| NS_{MS}| Encrypt(*PubKey*(*Authenticator*), *RAND*)|
HMAC(*PubKey*(*Authenticator*))

Fig. 13. Format of a PKMv2-EAP-Start message sent by MS to Authenticator.

OP_Code| BSID|SAID| CSK1| CSK2| MSK|BS-Random|TEK_Lifetime|TEK_Count| TEK1| TEK2

Fig. 14. Format of a KEY-Distribution-Response message sent by Authenticator to the BS.

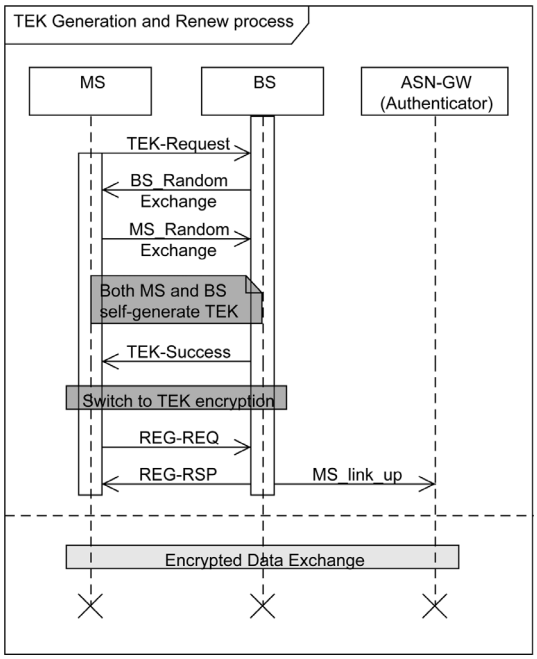

Fig. 15. Sequence chart of the TEK Generation and Renew process.

OP_Code|NS_{MS}|SAID|HMAC(ADR(CSK1,CSK2))

Fig. 16. Format of the TEK-Request message sent by MS to the BS.

4.1.3. Key distribution in the initial network entry

In the HaKMA, we design a key distribution message, prefixed by KEY-Distribution to deliver keys among the serving BS, target BS and Authenticator and from a HaKMA sub-process to the next sub-process. Authenticator sends a KEY-Distribution-Response message which contains the MSK and CSKs to the BS (see Fig. 12). Figure 14 shows the format of this message. Note that both TEK_Lifetime and TEK_Count are zero since TEKs have not been generated. Their usage will be described later.

BS on receiving this message extracts the MSK and CSKs from this message, and starts its TEK Generation and Renew process Currently, both the BS and MS possess the MSK and CSKs.

4.1.4. TEK generation and renew process

Like that in the DiHam, TEKs are individually generated by MS and the BS by employing required key parameters supplied by Authenticator. Figure 15 shows the process which starts when MS sends a TEK-Request message, the format of which is shown in Fig. 16 to the BS.

Both MS and the BS invoke the Dot16KDF algorithm which accesses the first 160 bits of MSK [1], to

individually derive AK, where

$$\text{AK} = \text{Dot16KDF}(\text{“AK”},2,\text{TRUNCATE(MSK,160)},160) \tag{6}$$

The BS (MS) self-generates a random number called *BS_Random* (*MS_Random*) which is also 160-bits in length. *BS_Fingerprint* (*MS_Fingerprint*) is then generated by encrypting *BS_Random* (*MS_Random*) with CSKs and AK, and delivered to MS(BS) through wireless channels where

$$\textit{BS\_Fingerprint} = \text{ADR}(\text{EXOR}(\textit{BS\_Random,CSK1}), \textit{AK}) \tag{7}$$

and

$$\textit{MS\_Fingerprint} = \text{ADR}(\text{EXOR}(\textit{MS\_Random,CSK}2), \textit{AK}) \tag{8}$$

The purpose is to protect the random number from being intercepted by a hacker. After that, a BS_Random_Exchange message the format of which is shown in Fig. 17 and which carries *BS_Fingerprint*, OP_Code, and lifetime of TEKs, denoted by Key-lifetime, is sent by the BS to MS.

OP_Code|NS_{MS}|NS_{BS}|BS_Fingerprint|Key-lifetime|HMAC(ADR(CSK1,CSK2))

Fig. 17. Format of the BS_Random_Exchange message sent by the BS to MS.

MS on receiving the message decrypts the *BS_Random* by using one of the following formulas:

$$\textit{BS\_Random} = \begin{cases} (\textit{BS\_Fingerprint} - AK) \oplus \textit{CSK1}, & \text{if } \textit{BS\_Fingerprin} \geqslant AK \\ (\textit{BS\_Fingerprint} + \bar{AK} + 1) \oplus \textit{CSK1}, & \text{if } \textit{BS\_Fingerprint} < AK \end{cases} \tag{9}$$

After that, MS sends an MS_Random_Exchange message the format of which is shown in Fig. 18 to the BS.

OP_Code|NS_{BS}|NS_{MS}|MS_Fingerprint|HMAC(ADR(CSK1,CSK2))

Fig. 18. Format of the MS_Random_Exchange message sent by MS to the BS. Note that Key-lifetime is not involved since it is determined by the BS.

Following that, MS generates TEKs where

$$\textit{TEK}_i = \text{EXOR}(\text{ADR}(\text{EXOR}(\textit{MS\_Random,AK}),\textit{BS\_Random}), \textit{CSK}_i), 1 \leqslant i \leqslant 2 \tag{10}$$

The BS on receiving the MS_Random_Exchange message retrieves the *MS_Random* and generates TEKs with the same formula where *MS_Random* is calculated by using one of the following formulas:

$$\textit{MS\_Random} = \begin{cases} (\textit{MS\_Fingerprint} - AK) \oplus \textit{CSK2}, & \text{if } \textit{MS\_Fingerprint} \geqslant AK \\ (\textit{MS\_Fingerprint} + \bar{AK} + 1) \oplus \textit{CSK2}, & \text{if } \textit{MS\_Fingerprint} < AK \end{cases} \tag{11}$$

Now both sides possess the TEKs. A TEK-Success message shown in Fig. 19 is sent by the BS to inform MS of the success of the TEK generation. MS registers its terminal device with the BS by sending a REG-REQ message, and the BS replies with a REG-RSP message which are both defined in the IEEE802.16 standard to finish this process. The data exchange can now be started.

OP_Code|NS_{BS}|NS_{MS}|HMAC(ADR(TEK1,TEK2))

Fig. 19. Format of the TEK-Success message sent by the BS to MS.

4.2. Handover process of network Re-entry

The main objectives of a handover process include minimizing the handover delay by key reuse and pre-distribution in the employed security scheme. If a user is authenticated in the initial network entry as stated above, we assume that he/she is still authenticated after handover. This means we can reuse the MSK and CSKs generated in previous processes to avoid the re-authentication delay. For security reasons, we can also renew TEKs on each handover. That means each time when MS moves to a target BS, new TEKs are required to substitute for the two TEKs used by the serving BS, called previous TEKs (prev_TEKs for short to avoid confusing with the term pre_TEKs used in PKMv2).

In this study, two security levels of handover are proposed to meet different security requirements. With Level1 handover, before prev_TEKs expire, the target BS after MS's handover reuses the same TEKs to encrypt data messages so as to shorten communication disruption time. With Level2 on each handover, the target BS temporarily reuses prev_TEKs to communicate with MS, generates new TEKs, and encrypts data messages with the new TEKs.

4.2.1. Key distribution in the network Re-entry

To deliver key information between MS and Authenticator, the KEY-Distribution-HOInfo message shown in Fig. 20 is designed to provide MS with handover support. In this message, the TEK_Count indicates the number of generated TEK pairs, and the TEK_Lifetime shows TEKs' remaining life time in minutes.

We also provide a KEY-Distribution-Request message shown in Fig. 21 for the target BS to request MS's key information from Authenticator.

Before MS hands over to the target BS, the serving BS sends a KEY-Distribution-HOInfo message to its Authenticator. Authenticator stores the required keys in the MS's corresponding tuple in its authentication key table (AK Table for short), a table used to keep authentication keys including the MSK and CSKs for the Authenticator's subordinate MSs. The AK Table is indexed by SAID to identify which MS the keys being considered belong to. Figure 22 shows the fields of this table. Authenticator further checks to see whether or not the target BS that it should newly associate with is in its BS Table a table for recording the BSIDs of the Authenticator's subordinate BSs, including those of its own and those subordinated by all its successor Authenticators, implying Authenticators are organized as a hierarchy. The table has only one field BSID. If yes, a KEY-Distribution-Response message that carries CSKs, MSK, and prev_TEKs is then sent to the target BS. If not, Authenticator, e.g., X, needs to relay the KEY-Distribution-HOInfo message to another ASN-GW that subordinates the target BS.

OP_Code|Serving_BSID|Target_BSID| SAID|CSK1| CSK2| MSK|
TEK_Lifetime|TKE_Count|TEK1|TEK2

Fig. 20. Format of the KEY-Distribution-HOInfo message sent by a serving BS to Authenticator or Authenticator to another Authenticator.

OP_Code|BSID|SAID

Fig. 21. Format of the KEY-Distribution-Request message sent by the BS to Authenticator.

Basically, X can use the backbone routing scheme to deliver the message to target Authenticator Y, or check its own Neighbor BS Table, a table for recording the Authenticators that all the BSs directly neighbor to any one of X's subordinated BSs belong to, to identify the right ASN-GW U, and relay the

SAID	BSID	CSK1	CSK2	MSK	TEK_Lifetime	TEK_Count	TEKs

Fig. 22. The fields of an AK Table. TEKs field stores TEKs.

message to U where the scheme of the neighbor BS Table is shown in Fig. 23. Once U receives the message its Authenticator Y looks up its BS Table to see whether the BSID conveyed on the message is one of its subordinate BSs or not. If yes, Y stores the MS's keys in its AK Table and sends a KEY-Distribution-Response message to the target BS.

The AK table should be updated dynamically each time when MS performs a network entry or re-entry, MS is going to hand over, or an MS key's lifetime expires. When MS initially enters a network, the AK Table is updated on the completion of the User Authentication process. Authenticator saves MS's keys leaving the TEKs field empty. The field will be filled after Authenticator receives a KEY-Distribution-HOInfo message from its BS or another Authenticator (which will be described later) and stores them in its AK Table. Generally, when MS is going to hand over, Authenticator extracts MS's TEKs from the KEY-Distribution-HOInfo message received from the serving BS and stores them in the AK Table. The serving Authenticator on receiving this MS's MSHO_link_up message sent by the target Authenticator deletes this MS key record. Finally, if the TEK_Lifetime expires, the TEK Generation and Renew process should be reinitiated to reproduce TEKs. After that Authenticator replaces the TEKs with the new TEKs in its AK Table.

BSID	ASN-GW	MAC	Address

Fig. 23. The schema of a Neighbor-BS Table. This table is statically constructed, and only contains neighbor BSs that are subordinated by ASN-GWs other than the underlying Authenticator's ASN-GW.

4.2.2. TEK generation and renew process on handover

Both processes of the two handover security levels start when MS sends a HO_IND message to its serving BS (see Figs 24 and 25). The serving BS then sends a MSHO_link_down message to inform its ASN-GW to start transferring data messages received from MS's corresponding node (CN) to both the serving BS and the target BS, and delivers a KEY-Distribution-HOInfo message which contains MS security attributes such as CSKs, MSK and TEKs that the serving BS currently uses, to its Authenticator.

Authenticator stores the keys in its AK Table if the target BS is one of its subordinate BSs. Otherwise it sends the keys to another ASN-GW during MS handover. No matter which is the case, the target Authenticator delivers a KEY-Distribution-Response message to the target BS. The target BS on receiving the message retrieves security keys and saves them for future use. Now, data message transfer can be resumed before the TEK Generation and Renew process starts, i.e., the target BS can relay data messages to MS before a new random number exchange, i.e., exchanging new *MS_Random* and *BS_Random* is completed.

After the completion of the TEK Generation and Renew process, an MSHO_link_up message will be sent by target BS to its ASN-GW to terminate sending data messages to the serving BS, and the transmission of encrypted data messages can be continued.

4.2.3. Level-1 Intra-ASN-GW Handover: TEK reuse mode

Once MS chooses a Level-1 handover, the KEY-Distribution-Response message sent to the target BS by Authenticator includes TEKs used by the serving BS. The target BS then waits for MS to complete its network re-entry, and on receiving the TEK-Request message sent by MS as shown in Fig. 24 it

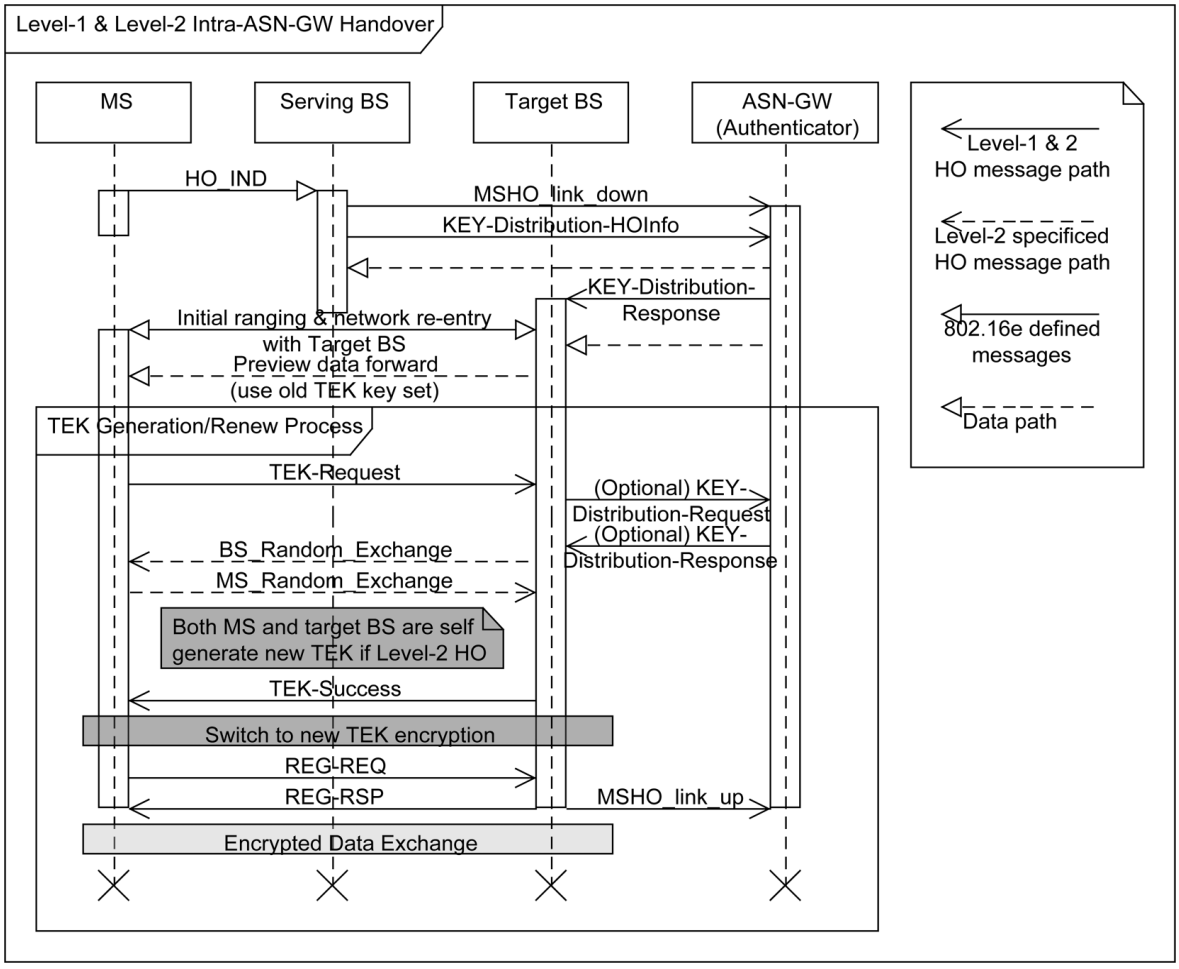

Fig. 24. The process of a Level-1 & Level-2 Intra-ASN-GW handover. The handover steps before the TEK Generation/Renew Process are the same in all Intra-ASN-GW handovers

delivers a TEK-Success message to MS, indicating the success of the TEK reuse mode. MS then sends a REG-RSP message to register itself with the BS. The BS replies with a REG-RSP message and sends an MSHO_link_up message to inform the ASN-GW of the termination of the handover service.

4.2.4. Level2 Intra-ASN-GW Handover: TEK regeneration mode

If MS selects a Level-2 handover the steps with which MS completes the network re-entry are mostly the same as those of a Level-1 Intra-ASN-GW handover. The following steps are a little different. The target BS on receiving a TEK-Request message from MS generates a new *BS-Random*, extracts CSKs and the MSK from the KEY-Distribution-Response message received from Authenticator uses the Dot16KDF algorithm to generate an AK, and then as shown in Fig. 24 sends a BS_Random_Exchange message containing a newly generated *BS-Fingerprint* (see Eq. (7)) to MS. MS then generates a new *MS_Random* and sends a MS_Random_Exchange message which contains a newly generated MS-Fingerprint (see Eq. (8)) to the target BS. The BS and MS individually generate new TEKs by using the new *BS-Random* (see Eq. (9)) and the *MS-Random* (see Eq. (11)). After that, MS and the target BS which is now MS's serving BS deliver data messages to each other by using the new TEKs. The following steps are the

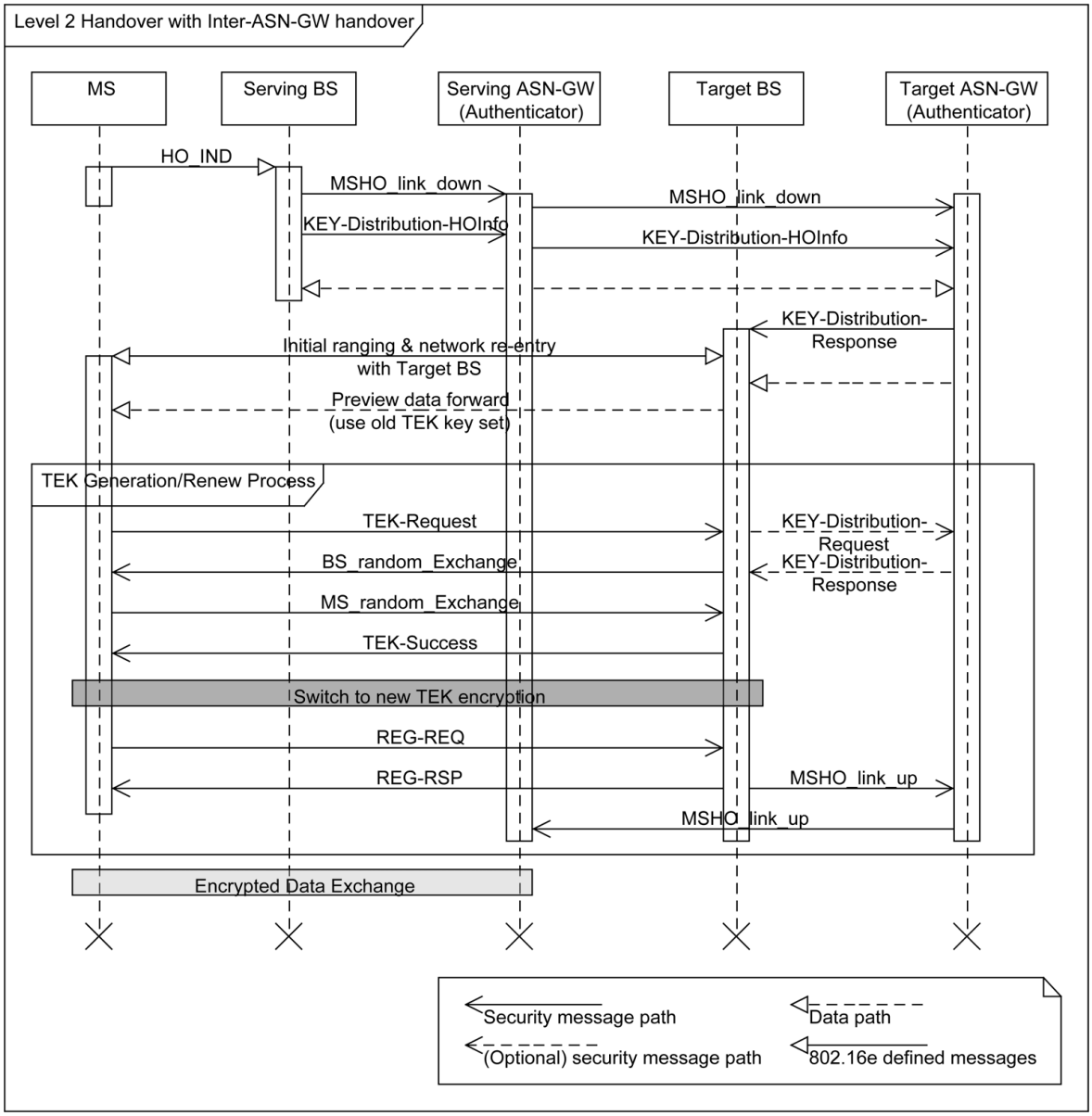

Fig. 25. The process of a Level-2 Inter-ASN-GW handover. The handover steps before the TEK Generation and Renew Process are the same as shown in Fig. 24

same as the corresponding steps of a Level-1 handover. Now the previous serving BSs can no longer communicate with MS since the prev_TEKs are out of date.

4.2.5. Inter ASN-GW handover

If MS hands over between two BSs which belong to different ASN-GWs, the serving Authenticator needs to transfer the KEY-Distribution-HOInfo message to the target Authenticator. As shown in Fig. 1, we assume any ASN-GW can communicate with other neighbor ASN-GWs via R4 reference points. Hence, the serving ASN-GW needs to know which ASN-GW the KEY-Distribution-HOInfo

message should be forwarded to. To solve this problem, each ASN-GW, e.g., G as state above, maintains a BS Table to collect all BSIDs of its subordinate BSs and a NeighborBS Table to gather all the BSs subordinated by other ASN-GW but neighbor to one of G's subordinate BSs. Hence, from the Target_BSID conveyed in the KEY-Distribution-HOInfo message the serving Authenticator can determine which ASN-GW (Authenticator) is the one it should forward the message to. During the handover if the serving Authenticator could not find the corresponding ASN-GW of the target BSID in its BS and NeighborBS Tables, the target Authenticator would not provide security keys to the target BS, implying MS should re-enter the system, i.e., performing the initial process of network entry described above. An ASN-GW on receiving a KEY-Distribution-HOInfo message from another ASN-GW passes this message to its Authenticator.

Once the KEY-Distribution-HOInfo message arrives at the neighbor Authenticator, the Authenticator forces a Level-2 Handover to renew TEKs. Figure 25 shows the Inter-ASN-GW handover.

5. Security analyses

The objective of security analyses is to confirm that our study is secure enough to meet wireless security requirements presented in the IEEE 802.16 standard and related research [23]. The security under different attacks is also analyzed.

5.1. Message integrity and replay attack avoidance

Message integrity ensures that a message M has not been changed during its delivery. In this study, the receiving end on receiving M uses the HMAC function to detect data tampering retrieves the nonce conveyed on M and saves it. The HMAC code conveyed on M can act as a verification code for the message itself. If at least one parameter has been changed, including the nonce, the HMAC code varies M will be discarded If the HMAC code passes the verification, we further verify the nonce.

The first time a message M is sent, the receiving end R records the nonce contained in M. If R receives the same or similar message (with the same OP_Code) again, it confirms that this is not a replay attack by comparing the nonce previously saved and the one retrieved from M. If the nonce received is smaller than or equal to the one saved, then M is considered as an illegal one and will be discarded. All messages delivered in the CSK Generation process and TEK Generation and Renew process are detected by this method.

The DiHam scheme provides key integrity, rather than message integrity, by comparing the keys calculated by using the authentication function Certfun$(a, b, \ldots)$ and by using the data carrier function EXOR(x, y) individually with the corresponding value retrieved from the received message (see Figs 4–7). All messages exchanged in the authentication phase and TEK generation phase could be maliciously altered, but the receiving end cannot discover the change. The DiHam also lacks the involvement of the nonce. Hence, it cannot discover replay attacks issued by resending an intercepted Authentication-reply message.

The PKMv2 uses a cipher-based message authentication code (CMAC) or HMAC to authenticate authentication messages, and detects replay attacks by employing CMAC_KEY_COUNT after the success of EAP authentication or reauthentication [1,12]. However, due to involving no nonce, it cannot avoid replay attacks during the EAP authentication session.

5.2. Confidentiality

Confidentiality ensures the security of sensitive data such as encryption keys. Hence, hackers cannot

directly acquire any unauthorized information from the intercepted messages. In our scheme, we analyze the confidentiality by checking to see whether exchanged information can be decrypted easily or not, and estimating the probability that a message being considered is cracked.

5.2.1. CSK confidentiality

The HaKMA uses the key exchange process of the DiHam to produce two CSKs. In this process, two public keys are exchanged between MS and Authenticator for each CSK and only the MS public keys P_{RM1} and P_{RM2} are transferred through wireless channels (see Fig. 10). The Authenticator public keys P_{RA1} and P_{RA2} are encrypted by using MS's certificate public key *PubKey*(*MS*) (see Fig. 11) which can only be decrypted by using MS's certificate private key *PrivKey*(*MS*). Therefore, hackers who only know P and P_{RMi} cannot easily derive *CSK1* and *CSK2* where

$$CSK_i = x \bmod P = {P_{RMi}}^y \bmod P, 1 \leqslant i \leqslant 2 \tag{12}$$

in which $x = {P_{RAi}}^{RM_i}$ (see Eq. (4)) and $y = RA_i$ (see Eq. (3)) are known and need to be determined, thus

$$x = {P_{RMi}}^y, 1 \leqslant i \leqslant 2 \tag{13}$$

The possible combinations of x and y pair are infinite. Due to the difficulty of determining the real values for x and y, hackers can only generate CSKs by other methods, e.g., the brute-force method.

Further, the number of possible 160-bit CSK values is $2^{160} \approx 1.4615 \times 10^{48}$. The probability of successfully guessing the CSK on one trial is $1/2^{160}$ which is approximately zero However, two CSKs are used in the HaKMA The probability will be $1/2^{320}$. Therefore, we can conclude that the CSK confidentiality is high.

5.2.2. EAP encryption key confidentiality

In this study we use the EAP encryption key to encrypt and decrypt the messages exchanged between MS and Authenticator. Since this encryption key is static and may be illegally decrypted, we involve the random number *RAND*, which is encrypted by Authenticator's public key (see Eq. (5) and Fig. 13) during its delivery to generate encryption keys. Hackers cannot directly access *RAND*. Hence, it is hard to derive the EAP encryption key. Furthermore each EAP encryption key is used only by one session i.e., each different session uses a different EAP encryption key, making it more difficult for hackers to collect EAP messages and then accordingly decrypt the key. Thus, our scheme has high EAP encryption key confidentiality.

5.2.3. TEK confidentiality

Since TEKs are used to encrypt data messages, we need to keep them secure. TEKs are self-generated by MS and the BS. Two random numbers *BS_Random* and *MS_Random* are also involved in the key generation process. To prevent hackers from collecting random numbers so as to derive TEKs, the two random numbers are encrypted to the *BS_Fingerprint* and *MS_Fingerprint* Since our TEK generation scheme involves the ADR function [22] (see Eq. (10)) which ignores the carry to calculate TEKs from *BS_Random* and *MS_Random* which in turn are respectively derived from *BS_Fingerprint* (see Eq. (7)) and *MS_Fingerprint* (see Eq. (8)), hackers have to face the four different mathematical equations in Eqs (9) and (11). Since each formula's possible outputs are up to $2^{160} \approx 1.4615 \times 10^{48}$, and all four equations involve *AK* and *CSK1* or *AK* and *CSK2* as parameters which are unknown to hackers the

number of possible parameter combinations for each equation is $2^{160\times3} = 2^{480} \approx 3.1217 \times 10^{144}$. Thus, we can conclude that the TEK confidentiality is high.

If hackers try to decrypt data messages, they must find the two correct TEKs for the uploading and downloading streams. If we assume that the time required to try a possible TEK is only one instruction, then it will take them about 1.4573×10^{29} years on a 159000 MIPS machine [4]. In other words, the HaKMA is a secure and safe system.

5.3. Mutual authentication

Mutual authentication between two nodes implies the two nodes authenticate each other before their communication starts. This can be securely achieved if the two nodes possess CSKs, or perform the public key infrastructure (PKI) public key exchange process. In this study, the mutual authentication focuses on device verification in the CSK Generation process. MS first sends its certification Cert(MS) through a CSK_Request message to Authenticator. Authenticator validates the correctness of a receiving certificate by running X.509 certificate signature algorithms, and/or it can also connect to CA to validate the effectiveness of the certificate through the NSP's backbone network. If the device certification is directly issued by the NSP or is already registered with the NSP, Authenticator can even validate the legitimacy of the device certification by contacting its authentication server [17].

Authenticator's certificate is contained in the CSK_Reply message (see Fig. 11). MS on receiving this message validates the correctness of the receiving certificate by running X.509 certificate signature algorithms too, and validates the legitimacy and effectiveness of the Authenticator's certificate by looking up the Authenticator blacklist and whitelist issued by the NSP.

The PKMv2 completes its mutual authentication by exchanging X.509 certificates. The authentication process is very similar to that described above. But the HaKMA enhances the process by involving MS's public key to encrypt two public keys P_{RA1} and P_{RA2}, implying that hackers can only intercept the two MS public keys, consequently increasing the difficulty of decrypting the two CSKs.

In the User Authentication process, the EAP message exchanged between MS and ASN-GW are encrypted by an EAP Encryption Key (see Eq. (5)) derived from the two CSKs. However, the two CSKs have never been transmitted through wireless channels, and they are only known to MS and Authenticator. Also, CSKs are produced at the end of the CSK Generation process. Only MS and Authenticator can start the User Authentication process with the valid EAP Encryption Key. Furthermore the TEK Generation and Renew process uses CSKs and MSK to generate TEKs (see Eqs (6)–(11)), implying the second and third processes of the HaKMA authentication scheme inherit mutual authentication from the first, i.e., the CSK Generation process. Hence, we can conclude that the HaKMA scheme provides mutual authentication in all three processes.

The DiHam process does not provide mutual authentication in the Authentication phase. It only uses CSKs generated in the Authentication phase to provide mutual authentication in the TEK Exchange phase.

5.4. User authentication

In the User Authentication process, an EAP method is involved. Since the PKMv2 also uses the EAP to perform user authentication, both the PKMv2 and HaKMA provide the same level of user authentication security But the HaKMA involves the EAP encryption key to encrypt EAP messages. Therefore, the HaKMA's user authentication security is higher than that of the PKMv2. Generally, the user identification security in the two schemes heavily relies on the selected EAP method [23]. The DiHam does not provide any user authentication. Its user identification is performed by recognizing MS's certification, or by using the certificate signed by the NSP [2].

5.5. Forward and backward secrecy on handover

Forward (backward) secrecy means the key K_n used in session n cannot be used in session $n+1$ (session $n-1$). In a Level-1 Intra-ASN-GW Handover, we reuse TEKs during and after the handover, implying a Level-1 Intra-ASN-GW Handover does not provide forward and backward secrecy. In a Level-2 Intra-ASN-GW Handover and Inter-ASN-GW Handover, we temporarily reuse TEKs to shorten the SDT, and generate new TEKs by involving the random numbers exchanged between MS and BS, i.e., the two handover processes provide forward and backward secrecy.

The PKMv2, due to considering performance optimization on Fast BS Switch (FBSS) and reusing all security attributes including TEKs, does not provide forward and backward secrecy.

Basically, the DiHam process has no forward and backward secrecy since it does not provide handover support. But if we apply the DiHam to the handover process, as described above, the BS and MS have to re-calculate TEKs for each handover, implying forward and backward secrecy. Note that if TEKs are reused after each handover, the DiHam's forward and backward secrecy will no longer exist.

5.6. Man-in-the-middle attack avoidance

A Maninthe-middle attack means hackers stay between valid MS and Authenticator to act as a legitimate Authenticator and MS. In the CSK Generation process and User Authentication process, MS and Authenticator exchange device certificate and determine whether the other side is legitimate or not But in the CSK Generation process, we use MS's and Authenticator's public keys i.e., *PubKey*(*Authenticator*) and *PubKey*(*MS*), to encrypt important keys such as P_{RA1} and P_{RA2} in the CSK_Reply message (see Fig. 11), and *RAND* in the PKMv2-EAP-Start message (see Fig. 13). The receiving end needs its own private key to decrypt those encrypted messages and keys. Now we assume that a hacker, H, is standing between a valid MS and Authenticator and wishes to steal EAP user passwords by eavesdropping EAP messages. Then H needs to act as an Authenticator so that it can acquire the valid CSK to continue the following User Authentication process since our EAP messages are all encrypted by using CSKs and other parameters like *RAND* (see Eq. (5) and Fig. 13). To complete the CSK Generation process besides relaying MS's and Authenticator's certificates H also needs to replace the Authenticator certificate with its own so that it can decrypt *RAND*. However, if H replaces the certificate with its own, this illegal certificate will not be recognized by MS and this session will be terminated On the other hand, if H continues using the real authenticator's certificate, it will not be able to decrypt the *RAND* carried on the next PKMv2-EAP-Start message sent by MS since *RAND* can only be decrypted by Authenticator's private key that H currently does not have, implying the User Authentication process is still secure because all EAP messages are encrypted by both the CSKs and *RAND*. As a result, our scheme can prevent man-in-the-middle attacks.

6. System experiments and discussion

In this study, several analyses and experiments were performed to evaluate the performance of the HaKMA and the compared schemes, including the PKMv2 and DiHam.

6.1. Performance analysis on key generation algorithms

Generally, in a Diffie-Hellman based authentication method, exponential operations dominate decisive performance differences [22]. In this study, two CSKs were individually generated by MS and Authen-

Table 2
Modular operations for different security schemes

Security scheme	Exponential operations			
	CSK	EAP	TEK	Total
DiHam with level-1 TEK	7	–	1	8
DiHam with level-2 TEK	7	–	6	13
DiHam with level-3 TEK	7	–	76	83
PKMv2 with EAP-AKA	–	2	–	2
HaKMA with EAP-AKA	5	2	–	7

ticator, and only Diffie-Hellman based public keys, are transmitted through wireless channels Deriving CSKs from exchanged public keys needs to solve discrete logarithm problems [8,10,22].

In the DiHam, Diffie-Hellman style keys are widely used, e.g., the generation of the CSKs, AKs and TEKs which provide a very secure method to protect the communication system, but the costs of key calculation are high. It has at least 7 exponential operations in the CSK Generation phase, and 1–76 exponential operations in different levels of the TEK Exchange phase. In the HaKMA, only 5 exponential operations are performed in the CSK Generation process, 1–2 exponential operations are involved in the User Authentication process depending on what EAP method is selected, and no exponential operations are involved in the TEK Generation and Renew process. The PKMv2 requires 0–2 exponential operations for the Diffie-Hellman style key exchange in its EAP Authentication process with a specific EAP method.

Other important algorithms employed in the HaKMA and PKMv2 are HMAC, CMAC and Dot16KDF. The HaKMA uses the HMAC algorithm six times in the CSK Generation process and TEK Generation and Renew process. The Dot16KDF algorithm is invoked only once in the TEK Generation and Renew process for generating AK. The PKMv2 uses this algorithm to derive AK, and the CMAC algorithm five times before the REG-REQ message is sent to the BS by MS. Note that the Dot16KDF algorithm invokes the CMAC or SHA-1 algorithm many times depending on the length of the key produced. But we ignore the difference since it is small and the output key lengths in both the PKMv2 and HaKMA are the same. Also, the costs of those algorithms performing fast operations such as exclusive OR operation and shift operation, are much smaller than those of exponential operations and can thus be ignored. Table 2 summarizes the costs of the evaluated schemes. We can see that the cost of the HaKMA operations is between those of the PKMv2 with the EAP-AKA method and the DiHam with level-1 TEK.

6.2. Costs and service disruption time

To evaluate the performance of the HaKMA, we calculate the processing cost for each message. The cost consists of two parts, message computation cost T, and message transmission cost T'. Thus, the processing cost $\mathbb{C}$ can be expressed as

$$\mathbb{C}_{CSK} = T_{CSK} + T'_{CSK} \tag{14}$$

$$\mathbb{C}_{MSK} = T_{MSK} + T'_{MSK} \tag{15}$$

$$\mathbb{C}_{TEK1} = T_{TEK1} + T'_{TEK1} \tag{16}$$

$$\mathbb{C}_{TEK2} = T_{TEK2} + T'_{TEK2} \tag{17}$$

The items used to evaluate the cost of the HaKMA and their descriptions are listed in Table 3.

Table 3
The items used to evaluate the cost of the HaKMA and their descriptions

Item	Description
$\mathbb{C}_{CSK}$, $\mathbb{C}_{MSK}$, and $\mathbb{C}_{TEKn}$	Costs of CSK generation process, User Authentication process, and Level-n TEK Generation and Renew process, respectively, where $n = 1$ or 2
$\mathbb{C}_{Intra_HOn}$ and $\mathbb{C}_{Inter_HO}$	Costs of the Intra-ASN-GW Handover and Inter-ASN-GW Handover, respectively, where $n = 1$ or 2
$\mathbb{C}_{Initial}$	Costs of the HaKMA initialization entry
T_{Exp}	Computation cost of an exponential operation
T_{Hmac}	Computation cost of a HMAC operation
T_{Enc}	Computation cost of a RSA-OAEP-Encryption operation
T_{Dec}	Computation cost of a RSA-OAEP-Decryption operation
$T_{Dot16KDF}$	Computation cost of a Dot16KDF algorithm

The costs of the HaKMA during the initial network entry and handovers can then be expressed as

$$\mathbb{C}_{Initial} = \mathbb{C}_{CSK} + \mathbb{C}_{MSK} + \mathbb{C}_{TEK2} \tag{18}$$

$$\mathbb{C}_{Intra\_HO1} = \mathbb{C}_{TEK1} + T'_{IntraHO} \tag{19}$$

$$\mathbb{C}_{Intra\_HO2} = \mathbb{C}_{TEK2} + T'_{IntraHO} \tag{20}$$

$$\mathbb{C}_{Inter\_HO} = \mathbb{C}_{TEK2} + T'_{InterHO} \tag{21}$$

6.2.1. Message computation costs

In the HaKMA, each message computation cost consists of the costs of message generation and receiving message verification. For example, the generation cost of a CSK_Request message on MS (see Fig. 10) is

$$2T_{Rand} + 2T_{Exp} + T_{Hmac} \tag{22}$$

The verification cost on Authenticator is T_{Hmac}. The total cost of the verification and generation of a CSK_Reply message (see Fig. 11) on Authenticator is

$$4T_{Exp} + 2T_{Rand} + T_{Enc} + T_{Exp} + 2T_{Hmac} \tag{23}$$

where $4T_{Exp}$ is the cost of invoking the Diffie-Hellman algorithm, $2T_{Rand}$ is the cost of producing two random numbers, and T_{Enc}, T_{Exp} and $2T_{Hmac}$ are the costs of invoking the encryption function, modulus function and HMAC function, respectively.

MS after receiving the CSK-Reply message spends $T_{Hmac} + T_{Dec}$ for verifying the message and another $3T_{Exp}$ for generating CSKs and invoking a modulus function. The total verification and key generation cost is

$$T_{Hmac} + T_{Dec} + 3T_{Exp} \tag{24}$$

The cumulative computation cost of the CSK Generation process is then (See Eqs (22), (23) and (24))

$$T_{CSK} = 4T_{Hmac} + 4T_{Rand} + 10T_{Exp} + T_{Enc} + T_{Dec} \tag{25}$$

The computation costs for other sub-processes can be calculated by a similar method. Table 4 lists the summaries, in which the CSK Generation process has higher cost than that calculated in Table 2 since what Table 4 lists are the cumulative costs. That means the operations are performed one by one instead of in parallel.

Table 4
Cumulative computation Costs of the sub-processes in the HaKMA

HaKMA sub-process	Computation Cost
CSK Generation Process	$T_{CSK} = 4T_{Hmac} + 4T_{Rand} + 10T_{Exp} + T_{Enc} + T_{Dec}$
User Authentication Process	$T_{MSK} = 2T_{Hmac} + 1T_{Rand} + T_{Enc} + T_{Dec}$
Level-1 TEK Generation & Renew Process	$T_{TEK1} = 4T_{Hmac}$
Level-2 TEK Generation & Renew Process	$T_{TEK2} = 8T_{Hmac} + 2T_{Rand} + 2T_{Dot16KDF}$

Table 5
The configurations used in the experiments

Variable	Configuration value
Bandwidth	Upward link: 1.5125 Mbps Downward link: 3.2425 Mbps Measured on: WiMax network / ISP: Vee Telecom Multimedia Corp.
Network topology delay	Between MS and BS (wireless connection): $d_w = 80.5$ ms Between BS and Authenticator, and between Authenticators (wired infrastructure connection): $d_l = 9$ ms
AAA server delay	$d_{AAA} = 923.220$ ms Measured in the situation: MTU = 1500 bytes, avg. message processing time = 90 ms, and a total of 5 messages are exchanged (including the network topology delay)

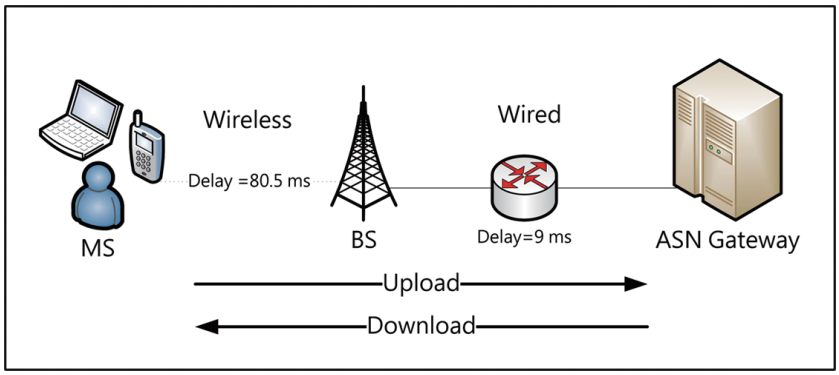

Fig. 26. The schematic diagram of the HaKMA performance evaluation.

6.2.2. *Message transmission cost*

The transmission cost of a message Msg in the HaKMA consists of delivery delay c and transmission delay d. Delivery delay c is the time required to deliver a message on a link where $c = Msg/bandwidth$, and transmission delay comprises the network topology delay including queuing delays on routers and the delays due to packet retransmission. In this study, we calculate the cumulative length of all messages generated, and compute the transmission cost under the network configuration shown in Fig. 26. Table 5 summarizes the measurements and specifications of the configuration, which are acquired on a real WiMax wireless network: Vee Telecom Multimedia Corporation, Taiwan [3].

Based on the configuration, if the message length is L in bytes, the upload delivery delay c_u in milliseconds from MS to Authenticator through the BS is

$$c_u(Msg) = \frac{8L}{1.5125 \times 2^{20}} \times 10^3 = \frac{L \times 10^3}{1.5125 \times 2^{17}} \tag{26}$$

Table 6
Maximum message lengths involved in the HaKMA

Message Msg	Length L (bytes)
CSK_Request	1104
CSK_Reply	1130
PKMv2-EAP-Start	66
KEY-Distribution-Response	182
TEK-Request	28
BS_Random_Exchange	48
MS_Random_Exchange	46
TEK-Success	26
KEY-Distribution-HOInfo	168
KEY-Distribution-Request	14

and the download delivery delay c_d in millisecond from Authenticator to MS is

$$c_d(Msg) = \frac{L \times 10^3}{3.2425 \times 2^{17}} \tag{27}$$

where all the values of Msg and the corresponding L are listed in Table 6.

Now, message transmission costs can be identified as:

$$T'_{CSK} = c_u(\text{CSK\_Request}) + c_d(\text{CSK\_Reply}) + 2(d_w + d_l) \tag{28}$$

$$\begin{aligned} T'_{MSK} = {} & c_u(\text{PKMv2-EAP-Start}) + c_d(\text{KEY-Distribution-Response}) \\ & + d_w + 2d_l + d_{AAA} \end{aligned} \tag{29}$$

$$T'_{TEK1} = c_u(\text{TEK-Request}) + c_d(\text{TEK-Success}) + 2d_w \tag{30}$$

$$\begin{aligned} T'_{TEK2} = {} & c_u(\text{TEK-Request}) + c_d(\text{BS\_Random\_Exchange}) \\ & + c_u(\textit{MS\_Random\_Exchange}) + c_d(\textit{TEK-Success}) + 4d_w \end{aligned} \tag{31}$$

Since the handover processes involve an extra KEY-Distribution-HOInfo message, the Intra-ASN-GW handover transmission and Inter-ASN-GW handover transmission costs for the message are

$$\begin{aligned} T'_{IntraHO} = {} & c_u(\text{KEY-Distribution-HOInfo}) \\ & + c_d(\text{KEY-Distribution-Response}) + 2d_l \end{aligned} \tag{32}$$

$$\begin{aligned} T'_{InterHO} = {} & 2c_u(\text{KEY-Distribution-HOInfo}) \\ & + c_d(\text{KEY-Distribution-Response}) + 3d_l \end{aligned} \tag{33}$$

6.2.3. System platform and experimental results

To evaluate the costs of the HaKMA to see whether it is feasible in practice or not, we implement various sub-processes and related algorithms used in the HaKMA. The specifications of the experimental system platform are listed in Table 7. The experimental results of all HaKMA subprocesses themselves and network entry/re-entry based on the configuration shown in Fig. 26 and those parameters listed in Table 5 are summarized in Tables 8 and 9, respectively.

Table 7
Specification of the experimental system platform

Component	Authenticator/BS equipment	MS equipment
H/W Platform	Intel × 86	ARM11
CPU	Intel Core i5 750 2.67 GHz	Samsung S3C6410 667 MHz
RAM	8GB	256MB
OS	Windows 7 Enterprise x64	Linux kernel 2.6.27 / Android 1.6

Table 8
Experimental results of the HaKMA sub-processes

HaKMA sub-process	Algorithm Experimental Result (ms)	System Experimental Result (ms)
CSK Generation Process, $\mathbb{C}_{CSK}$	2776.914	2964.142
User Authentication Process, $\mathbb{C}_{MSK}$	14.601	1037.083
Level-1 TEK Generation & Renew Process, $\mathbb{C}_{TEK1}$	0.160	161.362
Level-2 TEK Generation & Renew Process, $\mathbb{C}_{TEK2}$	4.075	326.622

6.2.4. Comparison of initial network entry and network Re-entry

For the MS, each of $\mathbb{C}_{Intra\_HO1}$, $\mathbb{C}_{Intra\_HO2}$, and $\mathbb{C}_{Inter}$ (i.e., network re-entry cost) is much shorter than $\mathbb{C}_{Initial}$ (i.e., initial network entry cost) because only the TEK Generation and Renew process is performed during the handover. Several keys originally generated in the CSK Generation process, e.g., CSKs, and the User Authentication process, e.g., the MSK, are now reused and as shown in Figs 20 and Fig. 14 delivered by key distribution messages to avoid introducing the authentication delay in the CSK Generation process, and the AAA Server delay in the User Authentication process. The experimental result shows that in the HaKMA the network re-entry cost ranges between 4.17% and 8.22% of the network initial entry cost. In other words, the HaKMA is feasible in a wireless system and has very low service disruption time.

7. Conclusions and future work

Wireless networks have been a part of our everyday life. Due to the mobility of end devices, we expect that wireless networks could someday substitute for wired broadband networks to serve users. However how to protect sensitive information delivered through wireless channels in a highly secure communication environment is an important issue in recent research.

In this paper, the HaKMA security scheme which provides fast and secure key generation process, mutual authentication and EAP based user authentication is proposed. The three-layer architecture simplifies key generation flows compared to those proposed in the DiHam and PKMv2. It further provides a fast and secure key renew process for handover. We also introduce two levels of handover processes to minimize SDT give connections between MS and BS forward and backward secrecy and analyze the HaKMA's security and performance. Table 10 summarizes the comparison on important issues. From this we can conclude that the HaKMA provides low-cost and effective handover, and its authentication approach is more secure than those of the DiHam and PKMv2.

In the future, we would like to enhance the HaKMA by developing its error handling capability. When the HaKMA receives an invalid message, it currently drops the message and waits for valid messages before timeout. If we wish to raise reliability for the HaKMA on error handling, the side that finds an error could send an error message to inform the other site of the occurrence of the error so that the compensative operations can be triggered immediately without wasting time to wait for

Table 9
Experimental results of the HaKMA for initial entry and handover network re-entry

Process	Cost (ms)
Initial Entry, $\mathbb{C}_{Initial}$	4327.847
Intra-HO/Level-1, $\mathbb{C}_{Intra\_HO1}$	180.638
Intra-HO/Level-2, $\mathbb{C}_{Intra\_HO2}$	345.898
Inter-HO/Level-2, $\mathbb{C}_{Inter}$	355.745

Table 10
Comparison of different security schemes with proposed scheme

Security Scheme	DiHam	PKMv2	HaKMA
Security enhancement and practical issues			
Messages Integrity	No	After EAP success	Yes
Confidentiality	Yes	Yes	Yes
Mutual Authentication	After Auth. phase	Yes	Yes, enhanced
User Authentication	No, rely on cert.	Yes, by EAP	Yes, by EAP
Efficient on key generation	Low	High	Medium
Handover related issues			
Handover support	No/Yes(see Sec. 3)	Skip	Yes
Forward and Backward Secrecy	−/Yes w/ renew TEK	No	Yes
Efficient under Handover	−/No	Yes when reuse	Yes
Low Cost under Handover	−/No	Yes	Yes
Fault-tolerant under Handover	−/No	No	Yes
Against threats			
Replay Attack	No	After EAP success	Yes
Man-in-the-middle Attack	No	Possible	Yes

valid messages In the handover support, we will design a flexible MS keys' routing scheme to deliver keys between/among Authenticators, and develop behavior and reliability models so that users can predict the HaKMA's behavior and reliability before using it. The handover authentication between two heterogeneous networks such as IEEE 802.11 or 3GPP LTE will also be developed. Those constitute our future research.

References

[1] IEEE Standard for Local and Metropolitan Area Networks Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems Amendment 2: Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation in Licensed Bands and Corrigendum 1, *IEEE Std 802.16e-2005*, 2006.
[2] WiMAX Forum Network Architecture, *WMF-T32-003-R010v05*, 2009.
[3] Vee telecom network speed test, http://speed.vee.com.tw/, Accessed in 2011.
[4] Benchmark Results: SiSoftware Sandra 2011, http://www.tomshardware.com/reviews/core-i7-990x-extreme-edition-gulftown,2 874-6.html, Accessed in 2011.
[5] J. Arkko and H. Haverinen, Extensible Authentication Protocol Method for 3rd Generation Authentication and Key Agreement (EAP-AKA), *RFC 4187*, 2006.
[6] C.J. Bernardos, M. Gramaglia, L.M. Contreras, M. Calderon and I. Soto, Network-based Localized IP mobility Management: Proxy Mobile IPv6 and Current Trends in Standardization, *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications* **1**(2/3) (2010), 16–35.
[7] T.M. Bohnert, M. Castrucci, N. Ciulli, G. Landi, I. Marchetti, C. Nardini, B. Sousa, P. Neves and P. Simoes, QoS management and control for an all-IP WiMAX network architecture: Design, implementation and evaluation, *Mobile Information Systems* **4**(4) (2008), 253–271.
[8] W. Diffie and M. Hellman, New directions in cryptography, *Ieee Transactions on Information Theory* **22**(6) (1976), 644–654.

[9] C. Eklund, R. Marks, K. Stanwood and S. Wang, IEEE standard 802.16: a technical overview of the WirelessMAN air interface for broadband wireless access, *IEEE Communications Magazine* **40**(6) (2002), 98–107.
[10] T. Elgamal, A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms, *Ieee Transactions on Information Theory* **31**(4) (1985), 469–472.
[11] M. Ergen, *Mobile broadband including WiMAX and LTE*, Springer Science+Business Media, LLC, Boston, MA, 2009.
[12] K. Etemad and M. Lai, *WIMAX technology and network evolution*, Wiley, Hoboken, N.J., 2010.
[13] P. Fülöp, S. Imre, S. Szabó and T. Szálka, Accurate mobility modeling and location prediction based on pattern analysis of handover series in mobile networks, *Mobile Information Systems* **5**(3) (2009), 255–289.
[14] P. Fülöp, B. Kovács and S. Imre, Mobility management algorithms for the Client-driven Mobility Frame System – mobility from a brand new point of view, *Mobile Information Systems* **5**(4) (2009), 313–337.
[15] R. Housley, W. Polk, W. Ford and D. Solo, Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, *RFC 3280*, 2002.
[16] S.F. Hsu and Y.B. Lin, A Key Caching Mechanism for Reducing WiMAX Authentication Cost in Handoff, *IEEE Transactions on Vehicular Technology* **58**(8) (2009), 4507–4513.
[17] D. Johnston and J. Walker, Overview of IEEE 802.16 security, *IEEE Security & Privacy* **2**(3) (2004), 40–48.
[18] J. Jonsson and B. Kaliski, Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1, *RFC 3447* (2003).
[19] H. Kim and J.-H. Lee, Diffie-Hellman key based authentication in proxy mobile IPv6, *Mobile Information Systems* **6**(1) (2010), 107–121.
[20] L.S. Lee and K. Wang, A Network Assisted Fast Handover Scheme for IEEE 802.16E Networks, in: *Proceedings of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications* (2007), 1–5.
[21] L.S. Lee and K. Wang, Design and Analysis of a Network-Assisted Fast Handover Scheme for IEEE 802.16e Networks, *IEEE Transactions on Vehicular Technology* **59**(2) (2010), 869–883.
[22] F.Y. Leu, Y.F. Huang and C.H. Chiu, Improving Security Levels of IEEE802.16e Authentication by Involving Diffie-Hellman PKDS, in: *Proceedings of the International Conference on Complex, Intelligent and Software Intensive Systems* (2010), 391–397.
[23] D.Q. Liu and M. Coslow, Extensible authentication protocols for IEEE standards 802.11 and 802.16, in: *Proceedings of the ACM International Conference on Mobile Technology, Applications, and Systems* (2008), 1–9.
[24] Z. Yan, H. Zhou and I. You, N-NEMO: A Comprehensive Network Mobility Solution in Proxy Mobile IPv6 Network, *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications* **1**(2) (2010), 52–70.
[25] W.S. Yang, E.S. Yang, H.J. Kim and D.K. Kim, Estimation of spectrum requirements for mobile networks with self-similar traffic, handover, and frequency reuse, *Mobile Information Systems* **6**(4) (2010), 281–291.
[26] Y. Yang and R. Li, Toward Wimax Security, in: *Proceedings of the International Conference on Computational Intelligence and Software Engineering* (2009), 1–5.

Yi-Fu Ciou received the B.S degree in computer science engineering from the Tunghai University, Taiwan. He is currently a master student in Department of Computer Science at the Tunghai University, Taiwan. His research interests include wireless network standard, security, embedded systems, and cloud computing.

Fang-Yie Leu received his BS, master and Ph.D. degrees all from National Taiwan University of Science and Technology, Taiwan, in 1983, 1986 and 1991, respectively, and another master degree from Knowledge System Institute, USA, in 1990. His research interests include wireless communication, network security, Grid applications and Chinese natural language processing. He is currently a professor of Tunghai University, Taiwan, and director of database and network security laboratory of the University. He is also a member of IEEE Computer Society.

Yi-Li Huang received his master degrees from National Central University of Physics, Taiwan, in 1983. His research interests include security of network and wireless communication, solar active-tracking system, and grey theory. He is currently a senior instructor of Tunghai University, Taiwan, and director of information security and grey theory laboratory of the University.

Kangbin Yim received his B.S., M.S., and Ph.D. from Ajou University, Suwon, Korea in 1992, 1994 and 2001, respectively. He is currently associate professor as he has joined Dept. of Information Security Engineering, Soonchunhyang University since 2003. He has served as an executive board member for Korea Institute of Information Security and Cryptology, Korean Society for Internet Information and The Institute of Electronics Engineers of Korea. As he is currently an editorial board member of JoWUA and JISIS, he worked as the track chair of several workshops such as BWCCA and IMIS. His research interests include vulnerability assessment, code obfuscation, malware analysis, insider threats, access control, secure hardware, and systems security. Related to these topics, he has worked on more than forty projects and published more than ninety domestic and international research papers.