Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2013, Article ID 304601, 14 pages
http://dx.doi.org/10.1155/2013/304601

Research Article

EAP-Based Group Authentication and Key Agreement Protocol for Machine-Type Communications

Rong Jiang,[1,2] Chengzhe Lai,[2,3] Jun Luo,[1] Xiaoping Wang,[1] and Hong Wang[4]

[1] *School of Computer, National University of Defense Technology, Changsha 410073, China*
[2] *Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada N2L 3G1*
[3] *State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China*
[4] *Xi'an Communication Institute, Xi'an 710106, China*

Correspondence should be addressed to Rong Jiang; jiangrong@nudt.edu.cn

Received 2 July 2013; Revised 23 August 2013; Accepted 23 August 2013

Academic Editor: Zhong Fan

Machine to machine (M2M) communications, also called machine-type communications (MTC), has widely been utilized in applications such as telemetry, industrial, automation, and SCADA systems. The group-based MTC, especially when MTC devices belong to non-3GPP network, will face new challenge of access authentication. In this paper, we propose a group authentication and key agreement protocol, called EG-AKA, for machine-type communications combining elliptic curve Diffie-Hellman (ECDH) based on EAP framework. Compared with conventional EAP-AKA, our protocol guarantees stronger security and provides better performance. Detailed security analysis has shown that the proposed EG-AKA protocol is secure in terms of user and group identity protection and resistance to several attacks. Furthermore, formal verification implemented in AVISPA proves that the proposed protocol is secure against various malicious attacks. Moreover, performance evaluation demonstrates its efficiency in terms of the signaling overhead, the bandwidth consumption, and the transmission cost.

1. Introduction

Machine to machine (M2M) communications [1], which is also defined as machine-type communications (MTC) [2] in release 10 of the 3rd Generation Partnership Project (3GPP), is one of the hottest issues not only in the standardization but also in the industrial circles. In M2M communications, both wireless and wired systems can communicate with other devices of the same ability. Thanks to MTC, many applications become possible [3, 4]. M2M communications uses a device, such as a sensor or meter, to capture an event (such as temperature and inventory level). Then this event is delivered through a wireless, wired, or hybrid network to an application (software program), which translates the captured event into meaningful information. For example, the event can be translated into what items need to be restocked [5]. Since MTC communications does not need direct human intervention, it is soon becoming a market-changing force for the next-generation intelligent real-time networked applications [6, 7].

Recently, most research on MTC has focused on congestion control, resource management, key management [8, 9], and so forth; however, there are few studies on security aspects. Lu et al. [10] point out that the existing challenges of M2M is energy efficiency (green), reliability, and security (GRS). Taleb and Kunz [11] present some potential challenges and solutions of MTC in 3GPP networks. Some security threats and corresponding solutions of 3GPP are discussed in [12]. Privacy preservation is also an important issue in M2M communications [13–15]. A new group message authentication protocol [16], which utilises only limited authenticated communication, combines short authenticated strings protocol with classical key agreement procedures. This SAS-based group authentication and key agreement protocol is secure against active attacks. If mobile terminals of non-3GPP short-distance wireless communication want to access the 3GPP core network, they must execute access authentication. Most access authentication protocols are based on Extensible Authentication Protocol (EAP), such as EAP-AKA [17], EAP-TTLS [18], EAP-PEAP [19], EAP-LEAP [20], and EAP-SPEKE

[21]. However, the existing access authentication protocols cannot provide enough security for MTC [22]; on the other hand, present standard has not considered the group-based access authentication. Recently, several standardization organizations start to present the concept and requirement of group authentication, but the mechanism and procedure have not yet been developed.

To the best of our knowledge, the existing network authentication systems are mainly designed for a single object, and they all need 3 or 4 rounds of interaction to realize the mutual authentication between a user and a server. In practical applications, however, there may be a large number of users with the same properties in a network, such as MTC, and user terminals can form a group when they are in the same region, or belong to the same application, or have the same behavior. In these applications, if substantial numbers of user terminals of a group access the network over a short period of time successively, the available authentication methods may suffer from network congestion by the increasing signal of the network. In order to prevent network from congesting and efficiently authenticate user terminals of a group, the concept of group authentication, which performs authentication for group units, is introduced. As a kind of network authentication technology, group authentication aims to authenticate multiple or all users over a shorter period of time. In this technology, the group is assigned a unique identifier, and user terminals are authenticated together as corporate entities. Group authentication can be fulfilled by utilizing the authentication agency or the gateway. After successful group authentication, user terminals and network side entities can share some keys.

In the current literature, a few authentication protocols of group communication have been proposed. An individual and group authentication model, which uses dynamic key cryptography and group key management for individual and group of users and services, is proposed for wireless network services [23]. Chen et al. propose G-AKA protocol for a group of mobile stations roaming from the same home network to a serving network [24]. Aboudagga et al. propose a group authentication protocol for mobile networks and design a new architecture for authentication management and an associated authentication protocol for mobile groups and individual nodes over heterogeneous domains [25]. However, there are still no appropriate group authentication methods for MTC in 3GPP. On the other hand, EAP-AKA [17] is an important authentication and key agreement protocol between 3G/LTE and non-3GPP, but EAP-AKA does not support group authentication mechanism and cannot be applied to group-based MTC. In addition, there are some vulnerabilities in EAP-AKA, such as disclosure of user identity, man-in-the-middle attack [26].

In this paper, in order to resolve group access authentication for MTC, we propose a novel group authentication and key agreement protocol based on Mun's protocol [26], named EG-AKA. Our protocol guarantees stronger security and provides better performance than the existing protocols. The main idea of our protocol is that the first MTC device of a group, which wants to access to 3GPP core network, performs a full AKA authentication procedure. In this process, the first MTC device obtains group authentication information and group temporary key (GTK) on behalf of other MTC devices of the same group. Then the authentication, authorization, and accounting server (AAA server) is enabled to carry out mutual authentication with remaining MTC devices of the group using obtained group authentication information and GTK without interacting with the home subscriber server (the HSS). The authentication delay can be decreased as a whole and the signaling overhead between the AAA server and the HSS is considerably reduced.

The remainder of this paper is organized as follows. In Section 2, we will introduce relevant background and knowledge. In Section 3, we propose our group authentication protocol. In Section 4, the authentication and other secrecy properties are verified by the model checking tools, and detailed performance evaluations are given in Section 5. Finally, we draw our conclusion and give the future work in Section 6.

2. Background

Before going to the details of the proposed protocol, we first recall the elliptic curve Diffie-Hellman technique [27], Mun's Protocol [26], which serves as the basis of the proposed EG-AKA protocol. Then, we present the abbreviations and network architecture used in this paper.

2.1. Elliptic Curve Diffie-Hellman. Elliptic curve cryptography (ECC), which is based on the algebraic structure of elliptic curves over finite fields, is a famous approach used in public-key cryptography. This cryptography was first proposed in 1985 independently by Koblitz [28] and Miller [29]. The primary advantage of ECC is that the key size is smaller while providing the same level of security, which can reduce storage and transmission requirements; that is, an elliptic curve group could provide the same level of security afforded by an RSA-based system with a large modulus and correspondingly larger key. For example, a 160 bit ECC public key should provide comparable security to a 1024 bit RSA public key. Elliptic curve Diffie-Hellman (ECDH) is an anonymous key agreement protocol that allows two parties, each having an elliptic curve public-private key pair, to establish a shared secret over an insecure channel [30]. This shared secret may be directly used as a key, or better yet, to derive another key which can then be used to encrypt subsequent communications using a symmetric key cipher. It is a variant of the Diffie-Hellman protocol using elliptic curve cryptography.

Key establishment protocol of elliptic curve Diffie-Hellman is described briefly as follows. Suppose Alice wants to establish a shared key with Bob, but the channel available for them is not secure and may be eavesdropped by the others. Initially, the domain parameters (i.e., (p, a, b, G, n, h) in the prime case or $(m, f(x), a, b, G, n, h)$ in the binary case) must be agreed upon. Also, each party must have a key pair suitable for elliptic curve cryptography, consisting of a private key d (a randomly selected integer in the interval $[1, n-1]$) and a public key Q (where $Q = dG$, that is, the result of adding

G together d times). Let Alice's key pair be (d_A, Q_A) and Bob's key pair be (d_B, Q_B). Each party must have the other party's public key (an exchange must occur). Alice computes $(x_k, y_k) = d_A Q_B$. Bob computes $(x_k, y_k) = d_B Q_A$. The shared secret is x_k (the x coordinate of the point). Most standardized protocols based on ECDH derived a symmetric key from x_k using some hash-based key derivation function. The shared secret calculated by both parties is equal, because $d_A Q_B = d_A d_B G = d_B d_A G = d_B Q_A$. The only information about her private key that Alice initially exposes is her public key. So, no party other than Alice can determine Alice's private key, unless that party can solve the elliptic curve discrete logarithm problem. Bob's private key is similarly secure. No party other than Alice or Bob can compute the shared secret, unless that party can solve the elliptic curve Diffie-Hellman problem [27].

2.2. Mun's Protocol. Mun et al. [26] propose a new authentication and key agreement protocol based on EAP-AKA designed for 3G-WLAN interworking. This protocol combines elliptic curve Diffie-Hellman (ECDH) with symmetric key cryptosystem to overcome several vulnerabilities. In addition, their protocol provides perfect forward secrecy (PFS) to guarantee stronger security, mutual authentication, and resistance to replay attack. The major advantages of their protocol can be summarized as follows:

(1) providing strong user identity protection by encrypted IMSI using shared secret key between user equipment and HSS;

(2) using ECDH to provide perfect forward secrecy between the user equipment and the AAA server;

(3) resisting against three types of man-in-the middle attack.

Mun's protocol can guarantee stronger security; however, similar to EAP-AKA, the protocol is not suitable for group-based MTC due to lack of specific mechanism. We will modify Mun's protocol to design a novel security enhanced group authentication protocol for MTC.

2.3. Network Architecture. In order to avoid confusing, we list the abbreviations used throughout the rest of this paper in Table 1.

The network architecture mainly consists of four parts: machine-type communication devices, access point, the authentication, authorization, and accounting server, and the home subscriber server, as shown in Figure 1.

Machine-Type Communication (MTC) Devices. An MTC device, which communicates through a public land mobile network (PLMN), is a device equipped for machine-type communications.

Access Point (AP). AP is a device that allows wireless devices to connect to a wired network using Wi-Fi, Bluetooth, or other related standards.

The Authentication, Authorization, and Accounting (AAA) Server. In the LTE network, the authentication, authorization, and accounting (AAA) server provides access authentication services for MTC devices on behalf of the 3GPP core network.

The Home Subscriber Server (HSS). In the LTE network, the home subscriber server (HSS) locates in 3GPP core network and provides authentication and management services for MTC devices on behalf of 3GPP core network.

Table 1: Abbreviation used in the paper.

Abbreviation	Definition
3GPP	3rd generation partnership project
3G LTE	3G long term evolution
AAA Server	The authentication, authorization, and accounting server
AKA	Authentication and Key Agreement
AP	Access point
AVISPA	Automated validation of internet security protocols and applications
CN	Core network
EAP	Extensible authentication protocol
ECC	Elliptic curve cryptography
ECDH	Elliptic curve Diffie-Hellman
GAK	Group authentication and key agreement Protocol for MTC
GK	Group Key
GTK	Group temporary key
HSS	Home subscriber server
IMSI	International mobile subscriber identification number
M2M	Machine to machine
MAC	Message authentication code
MSK	Master session key
MTC	Machine-type communications
PFS	Perfect forward secrecy
PID	Permanent identity
PLMN	Public land mobile network

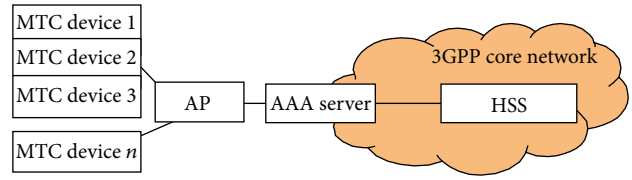

Figure 1: Network architecture of MTC.

3. The Proposed Group Authentication Protocol

In this section, we give the details of the group authentication and key agreement protocol for MTC (EG-AKA) to facilitate non-3GPP MTC devices to access to 3GPP core network (CN). In order to achieve this aim, there are three phases in the proposed protocol: group initialization, authentication

data distribution, and mutual authentication and key agreement.

3.1. Group Initialization. In the group initialization phase, each MTC device has a permanent ID (PID), such as international mobile subscriber identification number (IMSI). This PID is a long-term private identity that identifies MTC device and should be installed in the MTC device by the supplier in order to allow the MTC device to register in a 3GPP network. At the same time, we assume that each MTC device has preshared a secret key with 3GPP CN, and these MTC devices form several groups based on certain principles, and then the supplier provides a group key (GK) to each group for authentication. As shown in Table 2, we create an index table to manage information of MTC devices and group; the index table contains fields of group identity, MTC device identity (PID) for each MTC device, and initial values. Table 3 is the protocol notations used in this paper.

3.2. Authentication Data Distribution. Let $MTCD_{G1-1}$ be the first MTC device initiating authentication in group 1. We assume that a secure communication channel between the AAA server and the HSS has already been established and can provide security services to the transmitted data. The authentication data distribution processes as follows.

Step 1. $MTCD_{G1-1}$ sends an access request message to the AP.

Step 2. AP sends an EAP Request/Identity message to require the identity of $MTCD_{G1-1}$.

Step 3. Upon receiving the EAP Request/Identity message sent by AP, firstly, the $MTCD_{G1-1}$ computes

$$TID_{MTCD_{G1-1}} = f^1_{K_{MTCD_{G1-1}}-HSS}\left(PID_{MTCD_{G1-1}}\right), \quad (1)$$

$$TID_{G1-1} = f^1_{K_{MTCD_{G1-1}}-HSS}\left(PID_{G1}\right), \quad (2)$$

respectively, and then $MTCD_{G1-1}$ generates $AUTH_{G1}$ as follows:

$$AUTH_{G1} = \left(TID_{G1} \parallel TID_{MTCD_{G1-1}} \parallel R_{MTCD_{G1-1}} \parallel MAC_{MTCD_{G1-1}} \parallel ID_{HSS} \parallel ID_{AAA} \parallel ID_{AP}\right), \quad (3)$$

where $MAC_{MTCD_{G1-1}}$ is calculated as

$$MAC_{MTCD_{G1-1}} = f^2_{K_{MTCD_{G1-1}}-HSS}\left(R_{MTCD_{G1-1}} \parallel ID_{AAA} \parallel ID_{AP}\right). \quad (4)$$

Step 4. $MTCD_{G1-1}$ sends its $AUTH_{G1}$ to the AAA server through AP, and then the AAA server finds out corresponding HSS according ID_{HSS} and forwards $AUTH_{G1}$ and its own ID_{AAA} to the HSS by authentication data request message.

Table 2: Index table.

Group	Group ID	MTC device ID	Initial value
$G1$	PID_{G1}	PID_{G1-1}	IV_{G1-1}
		PID_{G1-2}	IV_{G1-2}
		⋮	
		PID_{G1-n}	IV_{G1-n}
$G2$	PID_{G2}	PID_{G2-1}	IV_{G2-1}
		⋮	

Table 3: Protocol notation.

Notation	Definition
$MTCD_{G1-1}$	The first MTC device initiating authentication in group 1
R_x	The random number generated by x
ID_x	The identity of x
PID_x	The permanent identity of x
TID_x	The temporary identity of x
K_{x-y}	The shared secret key between x and y
GK_{Gi}	The group authentication key of the ith group
GTK_{Gi}	The group temporary key of the ith group
f^1_k	Temporary identity generation function using k
f^2_k	MAC generation function using k
f^3_k	GTK generation function using k
f^4_k	Shared key generation function using k

Table 4: Temporary index table of $G1$.

Group	Group ID	MTC device ID	Initial value
$G1$	TID_{G1-1}	$TID_{MTCD_{G1-1}}$	IV_{G1-1}
		$TID_{MTCD_{G1-2}}$	IV_{G1-2}
		⋮	
	TID_{G1-n}	$TID_{MTCD_{G1-n}}$	IV_{G1-n}

Step 5. When the HSS receives authentication data request message containing $MTCD_{G1-1}$'s $AUTH_{G1}$ and ID_{AAA}, it verifies the received $MAC_{MTCD_{G1-1}}$ in $AUTH_{G1}$.

If verification passes, the HSS derives $PID_{MTCD_{G1-1}}$ and PID_{G1} from $TID_{MTCD_{G1-1}}$ and TID_{G1} using $K_{MTCD_{G1-1}}$, respectively. Then HSS retrieves the corresponding group key GK_{G1} to generate a group temporary key $GTK_{G1} = f^3_{GK_{G1}}(R_{HSS} \parallel ID_{HSS})$.

Step 6. At the moment, the HSS also computes all temporary identities of the devices in group 1 and generates a temporary index table (as shown in Table 4) of group 1; then the HSS sends ID_{AP}, GTK_{G1}, R_{HSS}, and temporary index table to the AAA server by a preestablish security tunnel.

Step 7. The AAA server receives and stores ID_{AP}, GTK_{G1}, R_{HSS}, and temporary index table for future use.

3.3. Mutual Authentication and Key Agreement

Step 8. The AAA server generates R_{AAA} and computes MAC_{AAA} as follows:

$$MAC_{AAA} = f^2_{GTK_{G1}}\left(R_{AAA} \parallel R_{HSS} \parallel R_{MTCD_{G1-1}} \parallel IV_{G1-1} + i\right), \quad (5)$$

where i represents the ith run of mutual authentication with $MTCD_{G1-1}$. After that, the AAA server selects random number a and computes aP on E.

Step 9. The AAA server generates

$$AUTH_{AAA} = \left(MAC_{AAA} \parallel R_{AAA} \parallel R_{HSS}\right) \quad (6)$$

and sends $AUTH_{AAA}$ and aP to $MTCD_{G1-1}$.

Step 10. After receiving $AUTH_{AAA}$, $MTCD_{G1-1}$ verifies the received MAC_{AAA} in $AUTH_{AAA}$ as follows.

(1) Firstly, $MTCD_{G1-1}$ computes

$$GTK_{G1} = f^3_{GK_{G1}}\left(R_{HSS} \parallel ID_{HSS}\right); \quad (7)$$

(2) then, $MTCD_{G1-1}$ computes

$$MAC'_{AAA} = f^2_{GTK_{G1}}\left(R_{AAA} \parallel R_{HSS} \parallel R_{MTCD_{G1-1}} \parallel IV_{G1-1} + i\right); \quad (8)$$

(3) $MTCD_{G1-1}$ verifies whether MAC'_{AAA} equals MAC_{AAA} or not. If MAC'_{AAA} is not the same as MAC_{AAA}, the HSS or the AAA server is not valid. Therefore, the $MTCD_{G1-1}$ terminates the procedure.

Step 11. If verification is successful, $MTCD_{G1-1}$ computes bP, $K_{MTCD_{G1-1}-AAA} = f^4_{GTK_{G1}}(abP)$ and $MAC_{MTCD_{G1-1}-AAA} = f^1_{K_{MTCD_{G1-1}-AAA}}(R_{AAA} \parallel bP)$.

Step 12. $MTCD_{G1-1}$ sends $MAC_{MTCD_{G1-1}-AAA}$ and bP to the AAA server by authentication response message, at the same time, $MTCD_{G1-1}$ also calculates the MSK as EAP-AKA.

Step 13. When the AAA server receives $MAC_{MTCD_{G1-1}-AAA}$ and bP, it also computes $K_{MTCD_{G1-1}-AAA} = f^4_{GTK_{G1}}(abP)$ using bP and verifies $MAC_{MTCD_{G1-1}-AAA}$. If verification passes, AAA server also calculates the MSK as EAP-AKA.

Step 14. The AAA server sends $ID_{AP} \parallel MSK$ with EAP Success message to the AP.

Step 15. The AP verifies whether received ID_{AP} equals its own ID or not. If the result is incorrect, the AP drops the MSK and then terminates the execution. Otherwise the AP stores the MSK. Then AP encrypts ID_{AP} using the MSK and sends it with EAP Success message to $MTCD_{G1-1}$.

Step 16. Through decryption, $MTCD_{G1-1}$ recovers ID_{AP} and verifies whether or not the ID_{AP} received from the AP in Step 15 equals to the ID_{AP} used in Step 4. If the result is correct, the procedure of authentication and key agreement is successful. Consequently, $MTCD_{G1-1}$ can securely access to 3GPP CN using the MSK.

At this point, the full authentication and key agreement procedure for one MTC device is completed. The procedure is shown in Figure 2.

When other MTC device in the same group want to access the 3GPP CN, the AAA server performs mutual authentication and key agreement with $MTCD_{G1-2}$ locally using the existing GTK_{G1}. Taking the MTC device $MTCD_{G1-2}$ in the same group as an example, the full authentication and key agreement procedure for it is described as follows.

Steps 1 and 2 are similar to $MTCD_{G1-1}$s.

*Step 3**. Upon receiving EAP request/identity message by AP, similarly, the $MTCD_{G1-2}$ computes $TID_{MTCD_{G1-2}} = f^1_{K_{MTCD_{G1-2}}-HSS}(PID_{MTCD_{G1-2}})$ and $TID_{G1-2} = f^1_{K_{MTCD_{G1-2}}-HSS}(PID_{G1})$, respectively, and then $MTCD_{G1-2}$ generates $AUTH_{G2}$ as follows:

$$AUTH_{G1} = \left(TID_{G1} \parallel TID_{MTCD_{G1-1}} \parallel R_{MTCD_{G1-1}} \parallel MAC_{MTCD_{G1-1}} \parallel ID_{AAA}\right), \quad (9)$$

where $MAC_{MTCD_{G1-2}}$ is calculated as

$$MAC_{MTCD_{G1-2}} = f^2_{K_{MTCD_{G1-2}}-HSS}\left(R_{MTCD_{G1-2}} \parallel ID_{AAA} \parallel ID_{AP}\right). \quad (10)$$

*Step 4**. $MTCD_{G1-2}$ sends its $AUTH_{G1}$ to the AAA server through AP. Note that, the AAA server does not need to authenticate the group (G1) which $MTCD_{G1-2}$ belongs to by the HSSs assistance.

*Step 5**. The AAA server begins to perform mutual authentication with $MTCD_{G1-2}$ using the temporary index table (Table 4) and GTK_{G1} received in Step 6.

The remaining steps are similar to $MTCD_{G1-1}$s.

The other MTC devices perform the authentication and key agreement procedures similar to $MTCD_{G1-2}$s until all devices complete the authentication.

4. Security Analysis

In this section, both security analysis and formal verification implemented by the AVISPA tool are conducted to show that the proposed protocol can work correctly to achieve security properties.

4.1. Security Property. In Table 5, we compare our proposed EG-AKA protocol with the other main AKA protocols: Mun's protocol [26], EAP-AKA [17], EAP-TTLS [18], EAP-PEAP

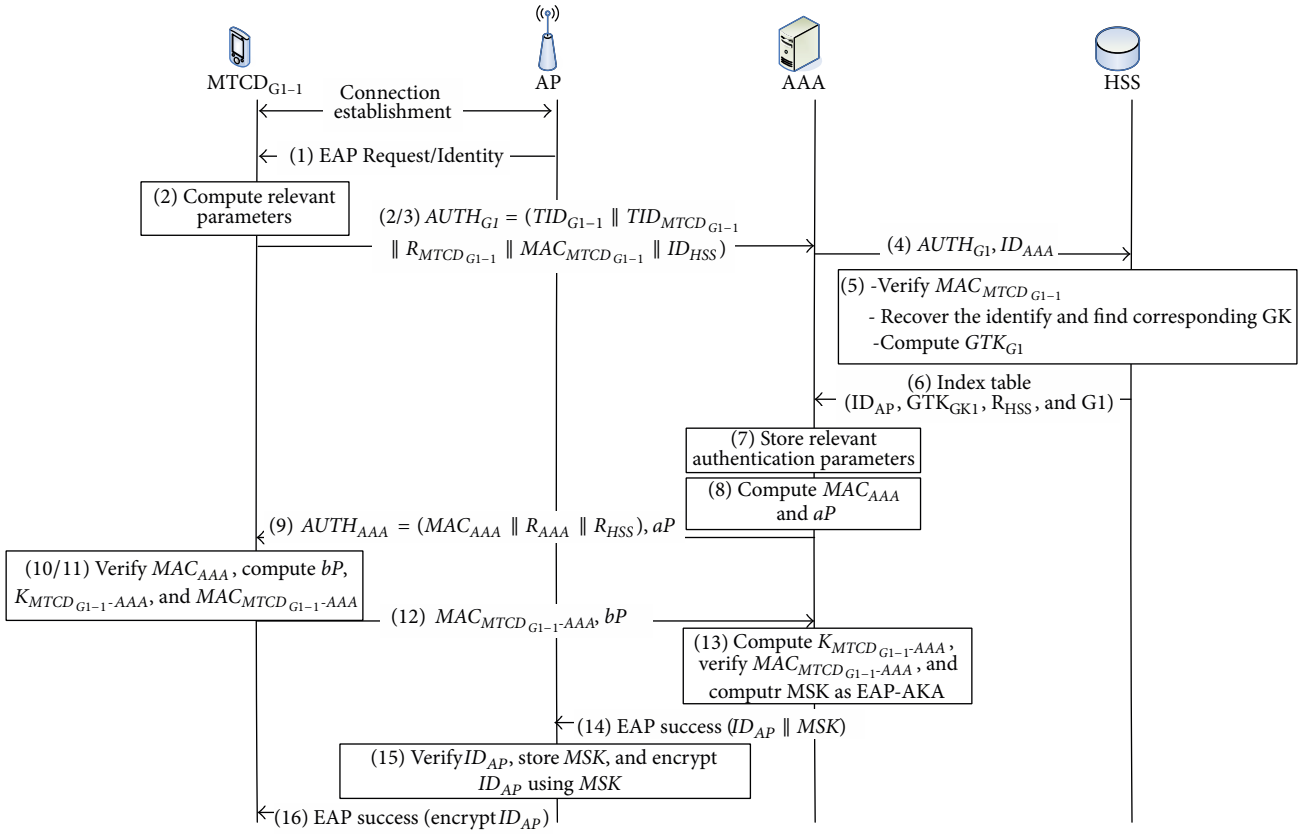

Figure 2: Authentication procedure of the first MTC device in our proposed protocol.

Table 5: Comparisons of properties among the EAP-based AKA protocols.

	Our proposed EG-AKA	Mun's protocol [26]	EAP-AKA [17]	EAP-TTLS [18]	EAP-PEAP [19]	EAP-LEAP [20]	EAP-SPEKE [21]
Type of cryptosystem	Symmetric and ECDH	Symmetric and ECDH	Symmetric	Public	Public	Public	Public
Computational overhead	Smaller	Smaller	Smallest	Large	Large	Large	Large
Protection of user identity	Yes	Yes	No	Yes	Yes	No	No
Heterogeneous network access	Yes	Yes	Yes	No	No	No	No
Secure against man-in-the middle attack	Yes	Yes	No	Yes	Yes	Yes	Yes
Secure against replay attack	Yes	Yes	Yes	Yes	Yes	Yes	Yes
PFS	Yes	Yes	Yes	Yes	No	No	No
Support group authentication	Yes	No	No	No	No	No	No

[19], EAP-LEAP [20], and EAP-SPEKE [21]. The comparison results demonstrate that our protocol can provide the most comprehensive security performance compared to the other AKA protocols. Providing group access authentication and heterogeneous network access are the two main advantages of our protocol. In particularly, our proposed protocol meets the following security properties.

Protect User and Group Permanent Identity. In our protocol, PID cannot be got by attackers. The reason is that the MTC device generates the TID by using the $K_{MTCD_{Gi-j}}$ and then sends TID to the HSS. Therefore, the MTC device and the HSS can only retrieve user and group permanent identity included in TID through using $K_{MTCD_{Gi-j}}$. Thus, our protocol provides strong user and group identity protection.

Secure against Man-in-the Middle Attack. In our proposed protocol, only the MTC devices and HSS can obtain real ID information of the devices and the group from encrypted temporary ID information. An attacker cannot derive and modify this information. The AP receives the EAP Success message with $ID_{AP} \| MSK$ sent by the AAA server. After

that, the AP can verify whether its own ID equal to the received ID or not. If not the procedure of authentication and key agreement will fail. Furthermore, the AP will send ID_{AP} encrypted by MSK to the MTC device. The MTC device can verify whether it has accessed this AP or not. The MTC device can verity the legality of HSS by MAC_{AAA} as well. Thus, our protocol can resist against several types of man-in-the middle attack.

Secure against Replay Attack. In our protocol, random numbers $R_{MTCD_{Gi-j}}$ generated by $MTCD_{Gi-j}$, R_{HSS} generated by the HSS and R_{AAA} generated by the AAA server are temporarily used in generating challenge messages toward the opposite side, respectively. Since these random numbers used in each authentication procedure are different, even if an attacker acquires a random number in a authentication procedure, he still cannot fake challenge messages by reusing the random number in a new authentication procedure. Meanwhile, these two sites maintain an identical initial value IV_{Gi-j} to keep themselves synchronized throughout AKA processing. An out-of-sync initialization value will lead to authentication failure. Thus a node without the required random numbers and initial value cannot perform a replay attack on our system.

Resistance to Impersonate Attack. Note that, in our protocol, all the MTC devices of a group share a common GTK. If an MTC device, without loss of generality, suppose that $MTCD_{G1-1}$ intends to impersonate another MTC device in the same group, for example, $MTCD_{G1-j}$. $MTCD_{G1-1}$ may eavesdrop traffic between $MTCD_{G1-j}$ and the HSS, but $MTCD_{G1-1}$ cannot generate unique R_{G1-j} and IV_{G1-j}. Therefore, $MTCD_{G1-1}$ cannot generate a correct $MAC_{MTCD_{G1-j}-AAA}$ to impersonate $MTCD_{G1-j}$ to perform a successful authentication with the HSS. Similarly, $MTCD_{G1-1}$ cannot get the $K_{MTCD_{G1-j}-AAA}$ between $MTCD_{G1-j}$ and the AAA server. Therefore, it cannot decrypt traffic between $MTCD_{G1-j}$ and the AAA server. In summary, the 3GPP CN can easily distinguish one MTC device from another even though all MTC devices use the same GTK; at the same time, one MTC device cannot decrypt traffic between any other MTC device and the 3GPP CN.

Perfect Forward Secrecy (PFS). Our protocol utilizes ECDH to provide PFS between the MTC device and the AAA server. While generating $K_{MTCD_{Gi-j}-AAA}$, our protocol uses aP and bP that are not related with $K_{MTCD_{Gi-j}-HSS}$. Therefore, if disclosure of $K_{MTCD_{Gi-j}-HSS}$ occurs, attackers cannot get $K_{MTCD_{Gi-j}-AAA}$. In other words, guessing $K_{MTCD_{Gi-j}-AAA}$ is a computationally difficult problem.

Provide Mutual Authentication and Key Agreement. We can verify that the proposed protocol can provide a successful mutual authentication between MTC devices and the 3GPP CN by formal verification described in the Section 4.2. Key agreement includes two parts: (a) between the MTC device and the AAA server: the key agreement between the MTC device and the AAA server can achieve through ECDH with symmetric key, and the MTC device and the AAA server can share a secret key $K_{MTCD_{Gi-j}-AAA}$ by Steps 11–13; (b) between the MTC device and the AP: the key agreement between the MTC device and the AP is the same as EAP-AKA [7], and the MTC device and AP can securely communicate with other by the MSK.

```
goal
secrecy_of kma
authentication_on mtcd_aaa
authentication_on aaa_mtcd
end goal
```

Figure 3: Analysis goals of the model.

4.2. Formal Verification. The primary goal of our proposed protocol is to provide mutual authentication and key agreement services between MTC devices and the 3GPP CN. We tested our protocol using formal security verification tool known as the "Automated Validation of Internet Security Protocols and Applications" (AVISPA) [31]. The AVISPA project aims at developing a push-button, industrial-strength technology for the analysis of large-scale Internet security-sensitive protocols and applications. This technology will speed up the development of the next generation of network protocols, improve their security, and therefore increase the public acceptance of advanced, distributed IT applications based on them. AVISPA will achieve this by advancing specification and deduction technology to the point where industry protocols can be specified and automatically analyzed. A central aim of the project is then to integrate this technology into a robust automated tool, tuned on practical, large-scale problems, and migrated to standardization bodies, whose protocol designers are in dire need of such tools. In the AVISPA tool, protocols are specified using the High Level Protocol Specification Language (HLPSL for short). Then, the HLPSL specification is translated into an Intermediate Format which is used by the various verification tools embedded in AVISPA. We use On-the-fly-Model-Checker (OFMC) and SAT-based model checker (SATMC) to text our EG-AKA protocol. The authentication goals that we need to verify are shown in Figure 3. The output of the model checking results are shown in Figures 4 and 5. We can conclude that the proposed protocol can accomplish the goal of mutual authentication, and it can resist those malicious attacks such as replay attacks, MitM attacks, and secrecy attacks under the test of AVISPA using the OFMC back-end and SATMC back-end.

5. Performance Evaluation

In this section, we give a detailed performance evaluation of the proposed protocol from the signaling overhead and the transmission cost point of view.

5.1. Signaling Overhead. In order to evaluate the signaling overhead, we consider the following scenario: the number

```
% OFMC
SUMMARY
SAFE
DETAILS
BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
D:\SPAN\testsuite\results\EG-AKA.if
GOAL
as_specified
BACKEND
OFMC
COMMENTS
STATISTICS
parseTime: 0.00 s
searchTime: 0.01 s
visitedNodes: 24 nodes
depth: 4 plies
```

Figure 4: Results reported by the OFMC back-end.

```
SUMMARY
SAFE
DETAILS
STRONGLY_TYPED_MODEL
BOUNDED_NUMBER_OF_SESSIONS
BOUNDED_SEARCH_DEPTH
BOUNDED_MESSAGE_DEPTH
GOAL
as_specified
BACKEND
  SATMC

COMMENTS

STATISTICS
  attackFound                  false      boolean
  upperBoundReached            true       boolean
  graphLeveledOff              3          steps
  satSolver                    zchaff     solver
  maxStepsNumber               11         steps
  stepsNumber                  3          steps
  atomsNumber                  0          atoms
  clausesNumber                0          clauses
  encodingTime                 0.09       seconds
  solvingTime                  0          seconds
  if2sateCompilationTime       0.66       seconds

ATTACK TRACE
no attacks have been found..
```

Figure 5: Results reported by the SATMC back-end.

of MTC device is n, and the number of group is m. Suppose that each MTC device launches x (re)authentications. For EAP-AKA, authentication procedures performed by an MTC device require the total number of signaling messages which grows linearly with x. In EAP-AKA protocol, there are 12 signaling messages for one complete authentication procedure. Thus, the number of signaling message of a MTC device is $12x$ and the total number of signal message is $12nx$. In Mun's protocol, the MTC device runs a full authentication using 8 messages at one time, a total of $8x$ messages is required. Similarly, when n MTC devices belonging to m group perform authentication, there are a total of $12nx$ messages for EAP-AKA, and a total of $8nx$ messages for Mun's protocol. In the proposed protocol, the first MTC device initiating authentication in the group complete the whole procedure of authentication and the number of signaling

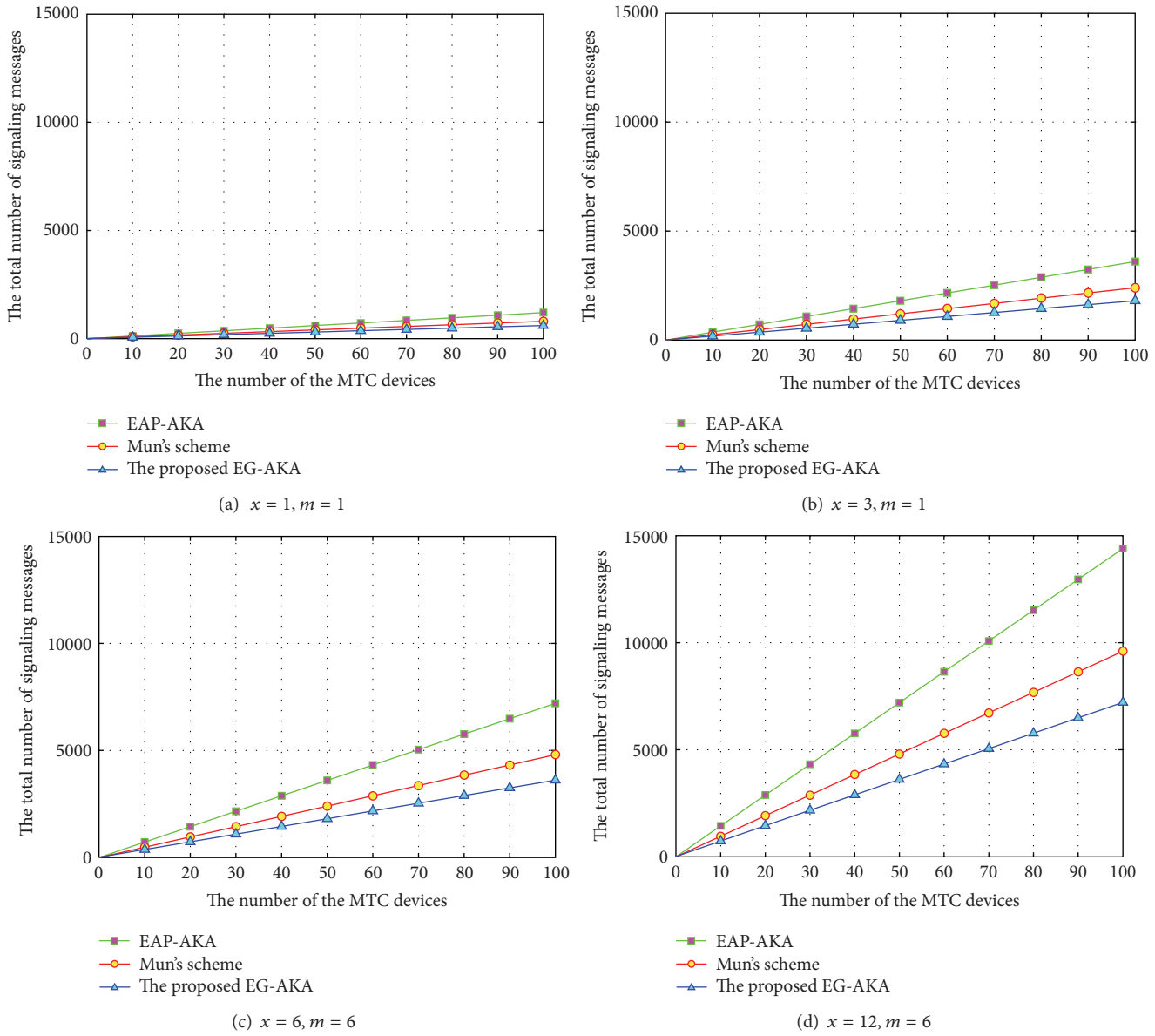

Figure 6: Comparison of the number of signaling messages of several EAP-based protocols.

message is 8. The rest devices of the group only need 6 signaling messages. In this scenario, the number of the rest devices is $n - m$ and the total number of signaling message is $8m + 6(n - m)$. If each device executes another $x - 1$ re-authentications, then the total number of signaling message is $8m + 6(n - m) + 6n(x - 1)$. Figure 6 illustrates the number of signaling messages of the proposed procedure over the existing authentication protocols for several different cases. It can be seen that signaling messages of several AKA protocols are increasing as the number of MTC devices increases. Among three AKA protocols, our EG-AKA outperforms other protocols. This is because our protocol shifts the impact of the number of MTC devices on network to the impact of that of the number of MTC device groups on network; our EG-AKA can reduce both authentication delay and signaling overhead within the core network.

5.2. Bandwidth Consumption. In order to analyze the bandwidth consumption, we assume that x AVs are transmitted every time the HSS successfully authenticates one ME, and there are n MTCDs forming m group. Without loss of generality, Table 6 shows the setting of parameters for evaluating bandwidth consumption.

The bandwidth consumption of AKA protocols are as follows, where bw_{first} represents the bandwidth consumption of the authentication of the first MTCD.

(1) Bandwidth analysis of EAP-AKA: the sizes of authentication messages are calculated as follows:

$$bw_{first} = \sum_{i=1}^{5} \left| Message_i \right| = 704 + 608x \text{ bits}. \quad (11)$$

Table 6: Setting of parameters.

Parameters	Value (bits)
ID/TID	128
GTK	128
MAC	64
Random number (RN)	128
ECDH key	192
Timestamp	32

The overall bandwidth consumption for n devices is calculated as $n \times (704 + 608x)$.

(2) Bandwidth analysis of Mun's scheme: the sizes of authentication messages are calculated as follows:

$$bw_{first} = \sum_{i=1}^{7} |Message_i| = 2432 \text{ bits.} \quad (12)$$

(i) $Message_1 = 2|ID| + |Timestamp| + |MAC| = 352$ bits.
(ii) $Message_2 = 2|ID| + |Timestamp| + |MAC| = 352$ bits.
(iii) $Message_3 = |RN| + |MAC| + |TK| + |ID| = 448$ bits.
(iv) $Message_4 = 2|RN| + |MAC| + |ECDH\ key| = 512$ bits.
(v) $Message_5 = |ID| + |ECDH\ key| = 256$ bits.
(vi) $Message_6 = |ID| + |MSK| = 256$ bits.
(vii) $Message_7 = |ID| + |MSK| = 256$ bits.

The overall bandwidth consumption for n devices is calculated as $n \times 2432$.

(3) Bandwidth analysis of EG-AKA: the sizes of authentication messages are calculated as follows:

$$bw_{first} = \sum_{i=1}^{6} |Message_i| = 2688 \text{ bits.} \quad (13)$$

(i) $Message_1 = 3|ID| + |RN| + |MAC| = 576$ bits.
(ii) $Message_2 = 4|ID| + |RN| + |MAC| = 704$ bits.
(iii) $Message_3 = |RN| + |GTK| + |ID| = 384$ bits.
(iv) $Message_4 = 2|RN| + |MAC| + |ECDH\ key| = 512$ bits.
(v) $Message_5 = |MAC| + |ECDH\ key| = 256$ bits.
(vi) $Message_6 = |ID| + |MSK| = 256$ bits.

Consider

$$bw_{remaining} = \sum_{i=1}^{3} |Message_i| = 1024 \text{ bits,} \quad (14)$$

where $bw_{remaining}$ represents the bandwidth consumption of authentication of each remaining ME.

(i) $Message_1 = 2|RN|+|MAC|+|ECDH\ key| = 512$ bits.
(ii) $Message_2 = |MAC| + |ECDH\ key| = 256$ bits.
(iii) $Message_3 = |ID| + |MSK| = 256$ bits.

The overall bandwidth consumption for n devices is calculated as $m * 2688 + (n - m) \times 1024$.

Figure 7 shows the bandwidth consumption of several AKA protocols, when the number of the MEs is different. From Figures 7(a) to 7(d), we can see that the bandwidth consumption of our EG-AKA protocol is much better than that of EPS-AKA and Mun's scheme. Meanwhile, our EG-AKA protocol can provide much better security compared to the other protocols.

5.3. Transmission Cost. In order to evaluate the transmission cost, assume that energy dissipated during 1-message transmission between MTC device and HSS is 1 unit, the energy dissipated during 1-message transmission between MTC device and AAA server is a unit ($a < 1$), and energy dissipated during 1-message transmission between AAA server and HSS is b unit ($b < 1$). Assume that the number of devices in a group is y.

Since the other EAP-AKA based protocols only enhance the security aspect and the procedure of signaling mode is the same as the traditional EAP-AKA protocol, we only compare our proposed protocol with the traditional EAP-AKA protocol. We consider the following two case as shown in Figure 2 in our proposed protocol:

(a) the AAA server has to fetch the fresh authentication vector form the HSS;
(b) the AAA server already has the fresh authentication vector.

In case (a), there are 4 messages between the MTC device and the AAA server, and there are 2 messages between the AAA server and HSS during one authentication procedure. The communication cost of our proposed protocol in this case is

$$C_{prop1} = (4a + 2b) + (y - 1) \times 4a. \quad (15)$$

In case (b), since the AAA server already has the fresh authentication vector, it does not need to communicate with the HSS anymore. Thus, the communication cost of our proposed protocol in this case is

$$C_{prop2} = 4ay. \quad (16)$$

Similarly, in the EAP-AKA protocol, there are 8 messages between the MTC device and the AAA server, and there are 2 messages between the AAA server and HSS during one authentication procedure. Therefore, the communication cost of the EAP-AKA protocol in case (a) is

$$C_{EAP-AKA1} = (8a + 2b)\, y \quad (17)$$

and in case (b) is

$$C_{EAP-AKA2} = 8ay. \quad (18)$$

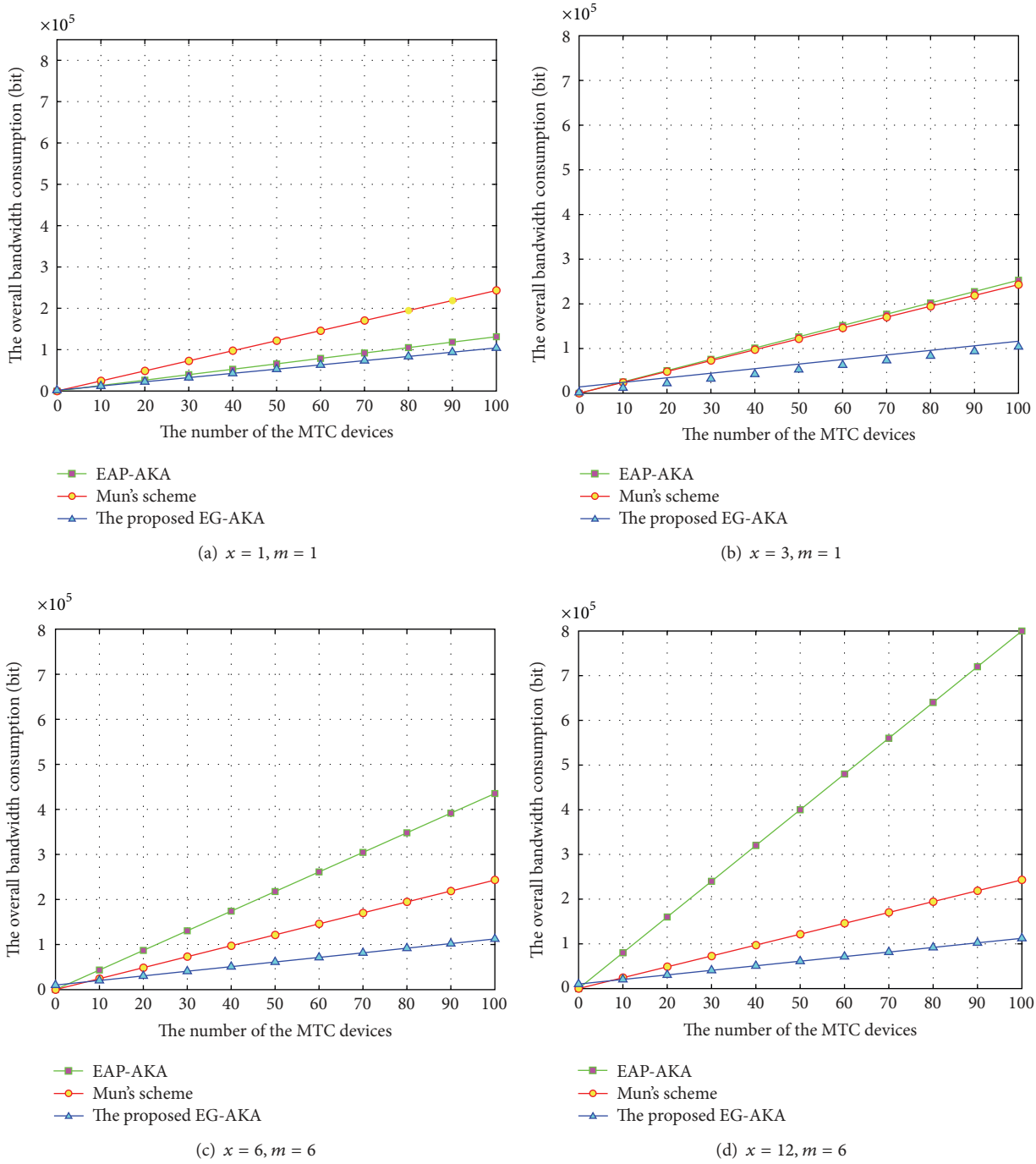

Figure 7: Comparison of the bandwidth consumption of several EAP-based protocols.

Suppose that the AAA server fetches n authentication vectors during the authentication procedure. The average communication cost of the proposed protocol is

$$C_{prop} = \frac{1}{n}C_{prop1} + \frac{n-1}{n}C_{prop2} = \frac{4any + 2b}{n}. \quad (19)$$

The average communication cost of the EAP-AKA protocol is

$$C_{EAP-AKA} = \frac{1}{n}C_{EAP-AKA1} + \frac{n-1}{n}C_{EAP-AKA2} = \frac{8any + 2by}{n}. \quad (20)$$

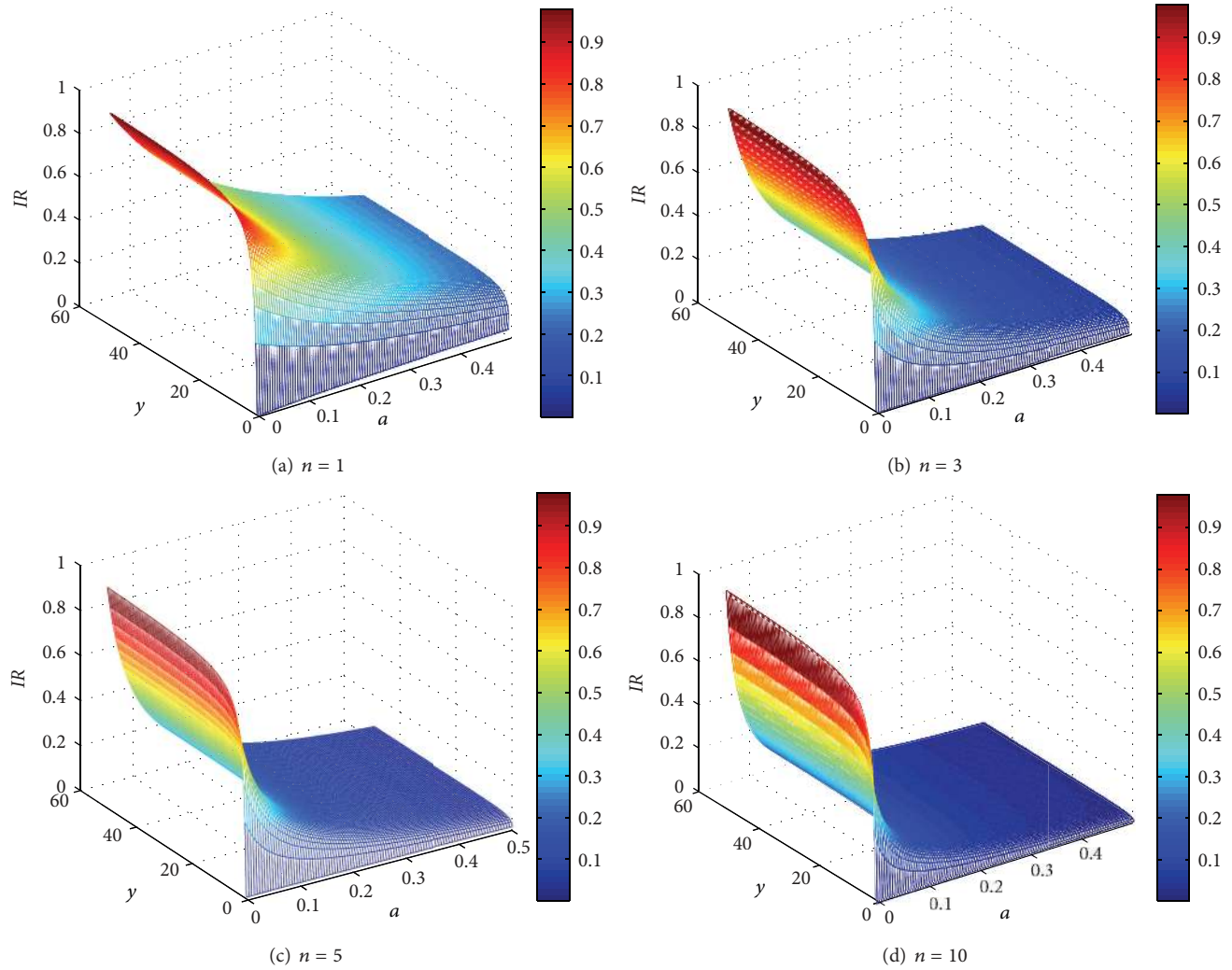

Figure 8: Comparison of the *IR*.

We define a improvement rate *IF* to evaluate the improvement of our proposed protocol compared to the EAP-AKA protocol. The definition of improvement rate *IR* is:

$$IR = \frac{C_{EAP-AKA} - C_{prop}}{C_{EAP-AKA}} = \frac{4any + 2by - 2b}{8any + 2by}. \quad (21)$$

From the definition of *IR*, we know that the bigger the *IR* is, the smaller the transmission cost of our proposed protocol is. Figure 8 plots the improvement rate *IR* varying with the number of devices, the number of fetched authentication vectors, and the energy dissipated during 1-message transmission between the MTC device and AAA server. From the figures, we can easily see that the more the number of MTC devices in the group is, the bigger the *IR* is. The reason is that in our proposed protocol we only need one communication between the AAA server and the HSS for the whole group authentication. While in the EAP-AKA protocol each MTC device has to execute a complete authentication. Furthermore, the more number of authentication vector the AAA server fetches from the HSS, the bigger the *IR* is. The reason is that our proposed protocol only needs one authentication vector for the whole group. The communication cost can be reduced dramatically.

6. Conclusion and Future Work

In this paper, we propose a group authentication and key agreement protocol for MTC device under the EAP framework, named EG-AKA. To the best of our knowledge, there is no protocol in the current literature that handles specific group access authentication for non-3GPP MTC. The proposed EG-AKA protocol not only enhances security on the basis of Mun's protocol, but also design specific group authentication mechanism for MTC. Formal verification and security analysis show that the proposed protocol is secure and fulfill its design goals. Detailed evaluations of performance illustrate that the proposed protocol achieves better performance in terms of transmission and signaling overhead compared with several existing protocols. In our future work, we will consider more practical group authentication protocol based on symmetric cryptography for resource-constrained devices in heterogeneous networks.

```
role mtcd (MTCD, AAA: agent, GK:
   symmetric_key, F1, F2, F3, F4: hash_func,
   SND, RCV: channel (dy))
played_by MTCD def =
local

      State: nat,
      Rmtcd, Rhss, Raaa, IDhss, Key: text,
      Kma: message

init
State := 0
transition
(1)   State = 0/\RCV (start) = |>
      State' := 2/\Rmtcd' := new ()
                  /\SND (Rmtcd')
(2)   State = 2/\RCV ({F2(Raaa'. Rhss'.
      Rmtcd)}_({F3(IDhss'. Rhss')}_GK).
      Raaa'. Rhss')
/\witness (MTCD, AAA, aaa_mtcd, Raaa', Rhss')
      = |>
      State' := 4/\Kma' := {F4(Key')}_({F3(
         IDhss'. Rhss')}_GK)/\secret (Kma',
         kma, {AAA, MTCD})
(3)   State = 4/\SND ({F1(Raaa. Key')}_Kma)/\
        wrequest (MTCD, AAA, mtcd_aaa, Raaa) = |>
State' := 6
end role % the role of MTCD

role aaa (MTCD, AAA: agent, GK:
      symmetric_key, F1, F2, F3, F4: hash_func,
      SND, RCV: channel (dy))
played_by AAA def =
local

      State: nat,
      Rmtcd, Rhss, Raaa, IDhss, Key: text,
      Kma: message

init
State := 1
transition
(1)   State = 1/\RCV (Rmtcd') = |>
      State' := 3/\Raaa' := new ()
      /\SND ({F2(Raaa'. Rhss'. Rmtcd')}_ ({
         F3(IDhss'. Rhss')}_GK). Raaa'.
         Rhss')
      /\wrequest (AAA, MTCD, aaa_mtcd, Raaa',
         Rhss')
(2) State = 3/\RCV ({F1(Raaa. Key')}_Kma')
      /\witness (AAA, MTCD, mtcd_aaa, Raaa)
      = |>
     State' := 5/\Kma' := {F4(Key')}_({F3(
         IDhss'. Rhss')}_GK)
      /\secret (Kma', kma, {AAA, MTCD})
end role % the role of AAA
```

Figure 9: The formal security verification program.

Appendix

For more details see Figure 9.

Acknowledgments

This work is supported by China Scholarship Council and the National Natural Science Foundation of China under Grant no. 61170261.

References

[1] T. Bourgeau, H. Chaouchi, and P. Kirci, "Machine-to-machine communications," in *Next-Generation Wireless Technologies*, pp. 221–241, Springer, New York, NY, USA, 2013.

[2] 3GPP TR 23. 888 V1. 4. 0. System Improvements for Machine-Type Communications, 2011.

[3] R. Deng, J. Chen, C. Yuen, P. Cheng, and Y. Sun, "Energy-efficient cooperative spectrum sensing by optimal scheduling in sensor-aided cognitive radio networks," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 2, pp. 716–725, 2012.

[4] P. Cheng, R. Deng, and J. Chen, "Energy-efficient cooperative spectrum sensing in sensor-aided cognitive radio networks," *IEEE Wireless Communications*, vol. 19, no. 6, pp. 100–105, 2012.

[5] B. Emmerson, "M2M: the Internet of 50 billion devices," *WinWin Magazine*, pp. 19–22, 2010.

[6] C. Lai, H. Li, X. Li, and J. Cao, "A novel group access authentication and key agreement protocol for machine-type communication," *Transactions on Emerging Telecommunications Technologies*. In press.

[7] C. Lai, H. Li, Y. Zhang, and J. Cao, "Security issues on machine to machine communications," *KSII Transaction on Internet and Information Systems*, vol. 6, no. 2, pp. 498–514, 2012.

[8] M. Wen, Y.-F. Zheng, W.-J. Ye, K.-F. Chen, and W.-D. Qiu, "A key management protocol with robust continuity for sensor networks," *Computer Standards and Interfaces*, vol. 31, no. 4, pp. 642–647, 2009.

[9] R. Jiang, J. Luo, F. Tu, and J. Zhong, "LEP: a lightweight key management scheme based on ebs and polynomial for wireless sensor networks," in *Proceedings of the IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC '11)*, pp. 1–5, Xi'an, China, September 2011.

[10] R. Lu, X. Li, X. Liang, X. Shen, and X. Lin, "GRS: the green, reliability, and security of emerging machine to machine communications," *IEEE Communications Magazine*, vol. 49, no. 4, pp. 28–35, 2011.

[11] T. Taleb and A. Kunz, "Machine type communications in 3GPP networks: potential, challenges, and solutions," *IEEE Communications Magazine*, vol. 50, no. 3, pp. 178–184, 2012.

[12] 3GPP TR 33. 868 V0. 5. 0, Security aspects of Machine-Type Communications, 2011.

[13] B. Wang, B. Li, and H. Li, "Knox: privacy-preserving auditing for shared data with large groups in the cloud," in *Applied Cryptography and Network Security*, pp. 507–525, Springer, New York, NY, USA, 2012.

[14] R. Jiang, J. Luo, and X. Wang, "An attack tree based risk assessment for location privacy in wireless sensor networks," in *Proceedings of the 8th IEEE International Conference on Wireless Communications, Networking and Mobile Computing*, pp. 1–4, 2012.

[15] B. Wang, B. Li, and H. Li, "Oruta: privacy-preserving public auditing for shared data in the cloud," in *Proceedings of the IEEE 5th International Conference on Cloud Computing (CLOUD '12)*, pp. 295–302, 2012.

[16] S. Laur and S. Pasini, "Sas-based group authentication and key agreement protocols," in *Public Key Cryptography-PKC*, pp. 197–213, Springer, 2008.

[17] J. Arkko and H. Haverinen, "Extensible authentication protocol method for 3rd generation authentication and key agreement (EAP-AKA)," 2006.

[18] P. Funk and S. Blake-Wilson, "EAP Tunneled TLS Authentication protocol version 1 (EAP-TTLS v1)," http://tools.ietf.org/html/draft-funk-eap-ttls-v1-00.

[19] Microsoft, "Securing Wireless LANs with PEAP and Passwords, Introduction: Choosing a Strategy for Wireless LAN Security," http://technet.microsoft.com/en-us/library/dd162271.aspx.

[20] O. George, "Ultimate wireless security guide: an introduction to LEAP authentication," Tech. Rep., 2007.

[21] D. P. Jablon, "Strong password-only authenticated key exchange," *ACM SIGCOMM Computer Communication Review*, vol. 26, no. 5, pp. 5–26, 1996.

[22] C. Lai, H. Li, R. Lu, X. Shen, and J. Cao, "A unified end-to-end security scheme for machine-type communication in lte networks," in *Proceedings of the 2nd IEEE/CIC International Conference on Communications in China (ICCC '13)*, pp. 1–6, 2013.

[23] H. H. Ngo, X. Wu, P. D. Le, and B. Srinivasan, "An individual and group authentication model for wireless network services," *Journal of Convergence Information Technology*, vol. 5, no. 1, pp. 82–94, 2010.

[24] Y.-W. Chen, J.-T. Wang, K.-H. Chi, and C.-C. Tseng, "Group-based authentication and key agreement," *Wireless Personal Communications*, vol. 62, no. 4, pp. 965–979, 2012.

[25] N. Aboudagga, J.-J. Quisquater, and M. Eltoweissy, "Group authentication protocol for mobile networks," in *Proceedings of the 3rd IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob '07)*, White Plains, NY, USA, October 2007.

[26] H. Mun, K. Han, and K. Kim, "3G-WLAN interworking: security analysis and new authentication and key agreement based on EAP-AKA," in *Proceedings of the Wireless Telecommunications Symposium (WTS '09)*, pp. 1–8, Prague, Czech Republic, April 2009.

[27] Mathcam, "PlanetMath-Elliptic Curve Diffie-Hellman key exchange," http://planetmath.org/DiffieHellmanKeyExchange.

[28] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, no. 177, pp. 203–209, 1987.

[29] V. S. Miller, "Use of elliptic curves in cryptography," in *Proceedings of the Advances in Cryptology (CRYPTO '85)*, pp. 417–426, Springer, 1986.

[30] E. B. Barker, D. Johnson, and M. E. Smid, "SP 800-56A. Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography (Revised)," 2007.

[31] T. A. Team, "AVISPA v1. 1 User Manual 2006," http://avispa-project.org/.