Wireless Netw
DOI 10.1007/s11276-014-0788-9

A group-based security protocol for machine-type communications in LTE-advanced

Daesung Choi · Hyoung-Kee Choi · Se-Young Lee

Abstract Machine-type communication (MTC) takes advantage of millions of devices being connected to each other in sensing our environment. A third-generation partnership project has been actively considering MTC as an enabler for ubiquitous computing and context-aware services. Until recently, we have not yet known how to productively manage the signaling traffic from these MTC devices because authentication requirements may impose such large signaling loads that they overwhelm the radio access of 4G cellular networks. This paper proposes the design of an efficient security protocol for MTC. This protocol is designed to be compatible with the incumbent system by being composed of only symmetric cryptography. Efficiency is attained by aggregating many authentication requests into a single one. The security and performance of the new design are evaluated via formal verification and theoretical analysis. Implementation of the proposed protocol in a real LTE-A network is provided through a feasibility analysis undertaken to prove the practicability of the protocol. Based on these evaluations, we contend that the proposed protocol is practical in terms of security and performance for MTC in LTE-Advanced.

Electronic supplementary material The online version of this article (doi:10.1007/s11276-014-0788-9) contains supplementary material, which is available to authorized users.

D. Choi · H.-K. Choi (✉) · S.-Y. Lee
School of Information and Communication Engineering, Sungkyunkwan University, Seoul, South Korea
e-mail: meosery@skku.edu

D. Choi
e-mail: dschoi@hit.skku.edu

S.-Y. Lee
e-mail: sylee@hit.skku.edu

Keywords Long-term evolution advanced (LTE-A) · Machine-type communications (MTC) · Authentication and key agreement (AKA)

1 Introduction

Machine-to-machine (M2M) communication makes it possible for tens of billions of machines in the world to talk with each other about their ambient surroundings through wired or wireless connections. Many promising applications in the categories of tracking, monitoring, maintenance, and security can be envisaged as emerging M2M communications. A third-generation partnership project (3GPP) has become regarded as such a promising solution to facilitate M2M communication that it has become known as machine-type communications (MTC) [1].

Incorporation of M2M communication into the 3GPP remains in its infancy because its many distinct features pose unprecedented challenges. The most common type of device in M2M communication will be, to the best of our knowledge, sensor nodes that are either mounted on a moving device to track it or is distributed in a field. In either case, the number of devices would be quite large and would tend to be deployed in a small area. Hence, they are in competition for the same access point to the network. We do not yet know how to manage these sensors in the most productive way. The challenges such distinct features pose to M2M communication lie in overcoming congestion and avoiding overloading of the radio access network and the core network.

The 3GPP specified a number of functions for overload control and also developed some extensions in an attempt to limit excessive signaling from the devices. A new priority level, called low access priority indication (LAPI) [2], was introduced into radio access control for the control plane signaling. The new priority is assigned to delay-tolerant applications, and when the radio access network is overloaded for some reason, the network can start to reject their attempts. This rejection is also another source of consuming radio resources, and therefore, the ability to bar low-priority devices from accessing radio resources was introduced with extended access barring (EAB) [2]. As for signaling reduction, access grant time interval (AGTI) is allocated so that each device can connect to the core network only at predefined times set by the network operator [3].

Although these functions and extensions may alleviate congestion in the radio access network, they cannot be applicable for delay-tolerant traffic. We found one such incident in the phase of authentication and key agreement (AKA) in a fourth-generation cellular network when a device registers with the core network. The AKA procedure is required whenever a device attaches itself to the network and changes one of near access points; this attachment and change can happen at any time. When a group of devices attempt to register simultaneously, masses of signaling traffic associated with the AKA would generate significant overloads on the authentication server and create bottlenecks in the link between a server and devices. To avoid this problem, we emphasize the need for the design of an efficient AKA procedure that reduces repetitive invocation of costly authentication signaling, especially in situations of group optimization.

Our goal in this paper is to take significant steps toward a new security mechanism that reduces the volume of signaling traffic in the AKA phase, even if the number of MTC devices is large and variable. We looked into the possible optimization of signaling by grouping the devices. A leader in the group would take responsibility for mutual authentication and secure key distribution and takes an advantage of the batch verification to save communication overhead and computational cost. In addition, we want the proposed method to outperform other existing protocols and still introduce lesser imperative amendments to the behavior of existing devices and network protocols. This requirement leads us to design new functions to be realized by only software updates in the current system. Furthermore, for prompt distribution of security contexts among devices in a group, we will capitalize on widely available Wi-Fi hotspots as a secondary channel mainly deployed for offloading data traffic in cellular networks. The leader collects authentication requests from MTC devices through this channel and compresses them into a single request. Last but not least, it should be pointed out that we want to accommodate two secure path layers; one path is between the MTC device and the core network, and the other is between MTC devices in the same group. The latter path is especially critical, considering the necessity of information sharing between devices in MTC.

The proposed protocol includes a few distinct ideas from previous studies. These are: (1) For efficiency and backward compatibility, symmetric cryptography is only used by the entire system when AKA are involved, (2) utilization of a secondary channel to distribute secret materials between MTC devices, and (3) any device in a group is able to share a secret key with any other device(s) in the same group. We evaluated the proposed security mechanism by formal verification, comprehensive security analysis, and performance evaluation. Further, to show the performance of the proposed protocol, we compared it with four other proposals that have been advanced.

The rest of the paper is organized as follows. In Sect. 2, we review related studies with an emphasis on grouping optimization. A 3GPP long-term evolution advanced (LTE-A) system is introduced in Sect. 3. In Sect. 4, we discuss the finer details and design decisions of the current authentication protocol. We provide various numerical analyses of security in Sect. 5 and performance evaluations and comparisons in Sect. 6. Section 7 concludes this paper.

2 Related work

At first glance, numerous seemingly natural approaches appear applicable to an objective of reducing costly authentication signaling. One tempting approach is to use delegation-based authentication [4]. In this, an authentication server authorizes a device to signal its own authentication that a serving network can then verify. The device needs no other access to the authentication server than to receive this authorization. However, this approach requires the introduction of a public key infrastructure, which is incompatible with the current secret key systems. It also is not practical in M2M communication because of the extra computational costs of signing and verifying the signatures.

An alternative might be grouping devices and having a leader of the group represent the group to serving networks. The leader authenticates itself to the network on behalf of all of the MTC devices. Once it is successful, the leader is entrusted with power over the end devices and authorized to authenticate the end devices locally without having each device access a remote authentication server. The Dynamic Group Based and Key Agreement (DGBAKA) [5] and the Group-based AKA (G-AKA) [6] are the two security protocols for authentication of a group of MTC devices in this roaming scenario. Because of the grouping model, these protocols can lessen communication costs on the network. However, overall complexity of the system can be built to up to large as the number of MTC devices increases.

EG-AKA (EAP-based Group Authentication) and SE-AKA (Secure and Efficient) proposed in [7, 8], respectively, are two group AKA protocols for LTE networks. Overall delay of the current AKA for a single user takes long because of a round-trip delay to the backend of authentication server in a core network. In order to improve this delay, EG-AKA and SE-AKA are designed to reduce the number of accessing times to the authentication server. The first member in the group is only required to handshake with the authentication server for authentication. The rest of the members are authenticated by a gateway located in the proximity. Because group members share a single group key a member can overhear private communications between other members. These protocols employ Elliptic Curve Diffie-Hellman (ECDH) to realize forward secrecy and backward secrecy. They also adopt an asymmetric key cryptosystem to protect devices' privacy. ECDH and asymmetric cryptography may not be suitable for resource-constraint MTC devices. EG-AKA [8] is for3GPP MTC devices to access the core network over non-3GPP air interfaces.

The authors in [9] proposed the ABAKA (A Batch Authenticated and Key Agreement) for vehicular networks. A strength of this approach lies in adopting the concept of batch verification to simultaneously authenticate multiple requests by one verification operation and to negotiate a session key with each vehicle by way of one broadcast message. Cao et al. in [10] proposed another authentication protocol for MTC environments based on an aggregate signature. An elected leader generates the aggregate signature and forwards it to the core network. The network can authenticate all the group members by verifying the aggregate signature and can establish distinct session keys with each member. However, in reality an asymmetric cryptographic algorithm is difficult to apply to the current cellular system because of compatibility issues.

The Logical Key Hierarchy (LKH) [11] and its clones [12, 13] are a centralized key management scheme. A group maintains a structure of a binary tree for efficient management of many secret keys in a group. The LKH maps all members in the group as leaves of the binary tree. Every node in the tree has its secret value K_i and every member in the group is assigned to a leaf node in the tree. The root node's secret value is the group key and the other secret values are used for efficient updates of the group key. When a member joins or leaves the group the group key must be updated to the value which not known to a new member or a departing member. An efficient key management scheme provides a prompt update of the group key to remaining members in a group.

3 Overview of 3GPP MTC

The 3GPP has announced LTE-A as a radio access technology for fourth-generation mobile networks. Figure 1 illustrates the MTC architecture in the LTE-A network, which is composed of groups of devices, the core network, and the radio area network.

3.1 System architecture

The term "LTE" encompasses the Evolved Packet System (EPS), which itself comprises both a new form of radio access—Evolved Universal Terrestrial Radio Access (E-UTRAN)—and SAE, the current end product of the evolution of non-radio access. At a high level, SAE includes a number of logical entities to manage MTC devices and bearers in the Evolved Packet Core (EPC), while LTE is made up of essentially just one node, the evolved NodeB (eNodeB), which connects MTC devices to the EPC.

In an LTE-A core network, as shown in Fig. 1, the mobile management equipment (MME) and serving gateway (S-GW) are responsible, respectively, for transmitting signaling traffic and user data traffic. The home subscriber server (HSS) contains the subscription information of MTC devices and assists the MME to authenticate MTC devices by providing a set of authentication vectors.

The permanent subscriber identity in the LTE-A is the International Mobile Subscriber Identity (IMSI). At the time of subscription the MTC device of subscriber is given to a unique IMSI, and a master secret key K shared with the authentication server (i.e., HSS). The IMSI and K are stored in the Universal Subscriber Identity Module (USIM) in the device. The subscriber would also be assigned the Global Unique Temporary Identity (GUTI) by the MME in a secure manner after the successful user authentication. The purpose of the GUTI is to provide a temporary identification of MTC device in the serving network that does not reveal the user's permanent identity in the mobile networks.

Cellular carriers are looking for solutions to offload data traffic from their cellular networks. Offloading data to Wi-Fi hotspots is an economically attractive alternative because many carriers already operate a substantial number of hotspots. Since Release 11, the 3GPP specification has supported WLAN internetworking including seamless mobility. Today, as part of Release 12, 3GPP is investigating a tight integration between LTE and WLAN at the radio access level [14].

3.2 Communication scenarios

At least two communication scenarios can be envisioned for the use of MTC. In one, an MTC server communicates with MTC devices over the core network. In this communication, MTC devices collect sensing data and transmit this data to the server on a frequency determined by the MTC user. This scenario has been adopted by many MTC

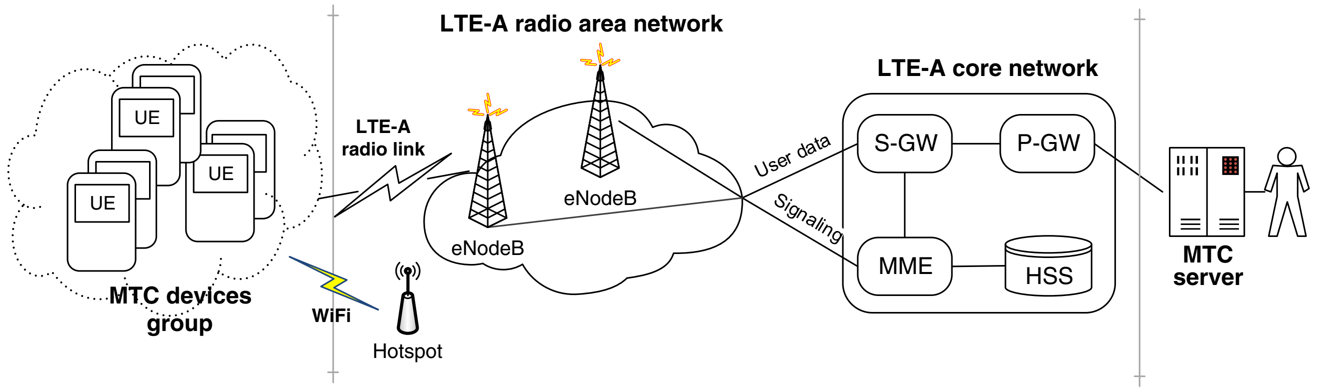


Fig. 1 The MTC architecture composed of groups of devices, the core network, and the radio area network

applications, including health monitoring and smart metering. The second scenario is direct communication among MTC devices to share sensing data and to distribute operational instructions. Sharing and distribution may be allowed by all group members or restricted to only those in a small subgroup or individual devices. In any case, one MTC device should be able to deliver data to designated receivers by broadcasting, multicasting and unicasting in a secure way.

The direct communication of MTC devices can be operated either (1) in an infrastructure mode in which packets are delivered via eNodeB in the radio access network or (2) in an ad hoc mode in which two devices in proximity can exchange data in the absence of infrastructure assistance. The 3GPP has proposed device-to-device (D2D) communication as a means of offloading cellular traffic, shortening delays, and increasing spectral efficiency [15]. D2D communication operates in the ad hoc mode and allows devices to communicate with each other in a direct link. Direct linkage can occur in the licensed cellular spectrum or in the unlicensed spectrum of such non-cellular technologies as Bluetooth or Wi-Fi Direct.

We exclude consideration of the D2D communication in this paper as the means of devices' direct communications. Instead, MTC devices in our service usage communicate with their peers by using Wi-Fi hotspots in the infrastructure mode. These decision were made after taking into consideration both practical implementation issues and performance issues. Firstly, the direct D2D communication is yet at an early stage in technical development and hence it is quite impractical to evaluate and compare performance measures in actual implementation. Secondly, the infrastructure mode of direct communication for MTC devices is generally preferred over the ad hoc mode of operation. This preference rests on the security of both direct communication and client/server communication because of the robust connections to and from MTC devices through the eNodeBs in the 3GPP or the base station in the Wi-Fi. Further, we insist on the Wi-Fi hotspot as this technology is expected to be promising in near future.

3.3 Extended authentication and key agreement (EPS-AKA)

The EPS-AKA is the security mechanism designed by the 3GPP for secure mutual authentication and sharing of the cryptography key [16, 17]. The EPS-AKA is composed of seven messages and is illustrated in Fig. 2. The MTC device sends its permanent identity, namely its *IMSI* in clear text. The MME passes this first message to the HSS with the serving network identity (*SN_ID*). If the *IMSI* is valid, the HSS generates and sends an array of several authentication vectors to the MME. A derived key (K_{ASME}) included in the authentication vector is a local root key derived from K, which is a master secret key shared between the MTC device and the HSS. The MME selects one authentication vector in the array and sends *RAND[i]* and *AUTN[i]* to the MTC device to challenge device authentication. The device authenticates the MME by verifying the message authentication code. It then derives CK, AK, IK and K_{ASME} from K to prepare its

Two and four secure keys are further derived from K_{ASME} to protect, respectively, the Access Stratum (AS) and Non-Access Stratum (NAS) layers [18]. The local master key, K_{ASME}, is valid for a maximum interval determined by the timing of the next EPS-AKA procedure. The device can choose to invoke the EPS-AKA protocol whenever the serving MME changes, because of roaming, to another serving network. In the same situation, the device also can choose to transfer the security context between the old and new MMEs in an effort to lower the overhead of the full EPS-AKA. The device may, of course, also need to periodically invoke the fresh EPS-AKA. Hence, the frequency of EPS-AKA runs is either random or can be configured by a network operator [18].

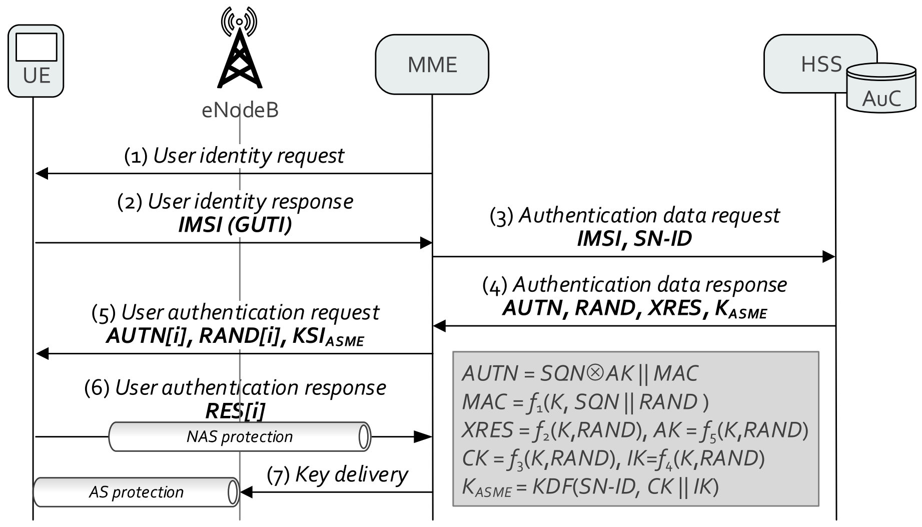

UE
eNodeB
MME
HSS
AuC
(1) User identity request
(2) User identity response IMSI (GUTI)
(3) Authentication data request IMSI, SN-ID
(4) Authentication data response AUTN, RAND, XRES, K_ASME
(5) User authentication request AUTN[i], RAND[i], KSI_ASME
(6) User authentication response RES[i]
NAS protection
(7) Key delivery
AS protection
AUTN = SQN⊗AK || MAC
MAC = f1(K, SQN || RAND)
XRES = f2(K,RAND), AK = f5(K,RAND)
CK = f3(K,RAND), IK=f4(K,RAND)
K_ASME = KDF(SN-ID, CK || IK)

Fig. 2 Message exchanges in EPS-AKA

3.4 Security requirements

The proposed security mechanism is designed mainly to accommodate performance optimization that is better than with the incumbent mechanism, the EPS-AKA, for the MTC environment. With regard to security needs, a basic model of the proposed mechanism is aligned with the general security requirements of the EPS-AKA and with those of most wireless access networks. We also want to address unacquainted threats and attacks that are introduced typically by the MTC environment and suggest appropriate countermeasures.

The general security requirements are specifically the *confidentiality* and *integrity* of data and *mutual authentication* of entities. An MTC device would not succumb to attachment to rogue access points (eNodeBs in our subject) if the attachment point were authenticated in advance by the MTC device. False eNodeBs or MTC servers could send forged messages to MTC devices to waste the resources of victims. This threat can be alleviated if mutual authentication is a prerequisite for any other operation in an MTC device. Illegal copying and modification of messages would not be easy once messages are authenticated and encrypted.

A group can be further divided into subgroups for efficient management of membership or different levels of access to secret information. An MTC device should be able to share a pairwise secret key with each entity or share group keys for one large group and its subgroups. This means every MTC device must have as many keys as there are groups that it belongs to plus keys for however many members are in these groups. The risk of handling this number of keys is that it will impose massive operational overhead as a prerequisite to executing the security functions necessary for resource-constrained MTC devices. Our design recognizes this risk and has scalable and efficient key management as one of its goals. Moreover, we want all secret keys to be capable of forward and backward secrecy. Otherwise, an adversary could derive past and future keys from the current secret key.

MTC device's secret materials are stored in the Universal Integrated Circuit Card (UICC). It is generally accepted that the UICC is tamper proof and the master key in the UICC is fairly resistant to any types of physical and network attacks to disclose secret keys in clear. In the continuing sense, we must assume that the master secret key in the core network, particularly in the HSS, is also secure and immune to any physical and network attacks.

4 Proposed protocol

We propose a new authentication and key management protocol for E-UTRAN. This protocol comprises four phases: system initialization, mutual authentication, membership update, and session key agreement.

4.1 System initialization phase

MTC devices operated by the same MTC service or sited in the same area can be grouped for management, control, and charging [17]. The network service provider can group MTC devices together. Group membership may be changed at any time when the service provider adds MTC devices to groups or retires MTC devices. The service provider is responsible for determining and maintaining group memberships.

The HSS and the MTC devices maintain the same structure of a binary tree for a group, as shown in Fig. 3. Every node ($NODE_k$) in the tree is associated with its

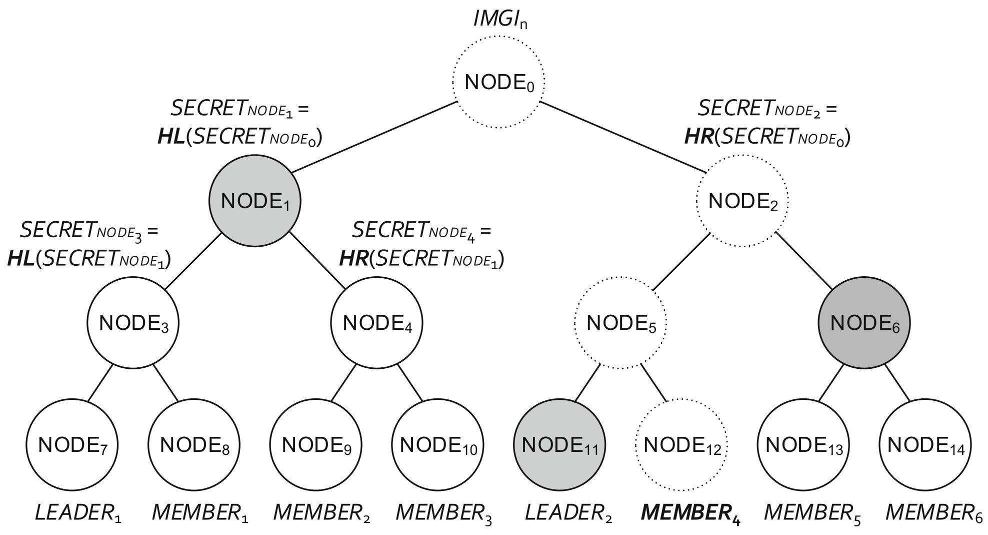


Fig. 3 Group management of MTC devices in binary tree

secret value, that is $SECRET_{NODE_k}$. A node's secret value is derived from the secret value of its parent node by using two hash functions, $HL(\cdot)$ and $HR(\cdot)$for left and right children, respectively. The MTC device is assigned to only a leaf node in the tree and given a set of secret values. These values contain the secret values of all nodes in the tree, with the exception of the restricted secret (RS) value. The RS value is referred to as a secret value of itself and secret values of its all parent nodes located on the path toward the root. Because some secret values are restricted, an MTC device is unable to derive the secret value of the node where this device is attached. For instance, the restricted values of $member_4$ in Fig. 3 are $SECRET_{NODE_{12}}$, $SECRET_{NODE_5}$, $SECRET_{NODE_2}$ and $SECRET_{NODE_0}$ (nodes with dotted lines in Fig. 3). In this way, $member_4$ has no way of knowing $Node_{12}$'s secret value.

At the time the tree is created, the HSS generates group key GK for the group. Further, the service provider stores a set of parameters in the secure storage of the MTC device at the time of registration. These parameters are $IMGI$, GK, PN_i, SEK_i along with the three hash functions of $HL(\ )$, $HR(\ )$ and $H(\ )$, and the secret values of the device. The group is identified by the $IMGI$ (International Mobile Group Identity). PN_i is a prime number created by the HSS, and SEK_i is a secret created by the HSS and shared between the HSS and $device_i$.

A set of leaders is included among the MTC devices to represent the group to the core network. These leaders are registered in the HSS and identified by their IMSIs. The leader itself is also an MTC device and hence needs to be in the binary tree and to store the same parameters as a member. However, the leader has computing power for various security operations and safe storage for security contexts.

4.2 Mutual authentication phase

Authentication and session key agreement first occurs between a selected leader and the core network. As a result, a secure link is established in the E-UTRAN. Then, all group members authenticate with the core network over the secure link via the leader and the Wi-Fi secondary link.

To avoid collisions because of multiple simultaneous initiations, leaders are instructed to wait for a random amount of time before they can send the first message. If any leader hears the first message sent by another leader, the listener postpones sending until the current AKA procedure is complete. Figure 4 illustrates the proposed AKA composed of nine messages. In the paper we use $[A_i]_n$ in the equation to denote a concatenation of A_i for n vectors.

M.1: The MME asks the MTC devices identify the leader.

M.2: The leader responds to the MME with the identities of all members, including its own ($[IMSI_i]_n$) and the group identity ($IMGI$).

M.3: The MME adds its own identity SN_ID to the second message and sends the authentication request to the HSS through a secure channel. We assume that this channel is safe because of IP security. The HSS computes GTK from GK, $RAND$, and SN_ID by using Eq. (1). The parameter $RAND$ is a random number generated by the HSS. The HSS recognizes the leader's identification and confirms the group members associated with $IMGI$.

$$GTK = f_1(GK, RAND, SN\_ID) \tag{1}$$

M.4: The HSS confirms the legitimacy of the inquiring MME by checking the validity of SN_ID. The HSS then computes three parameters; AV, LMK, and G_{info}, and sends them to the MME. The local master key (LMK) is derived

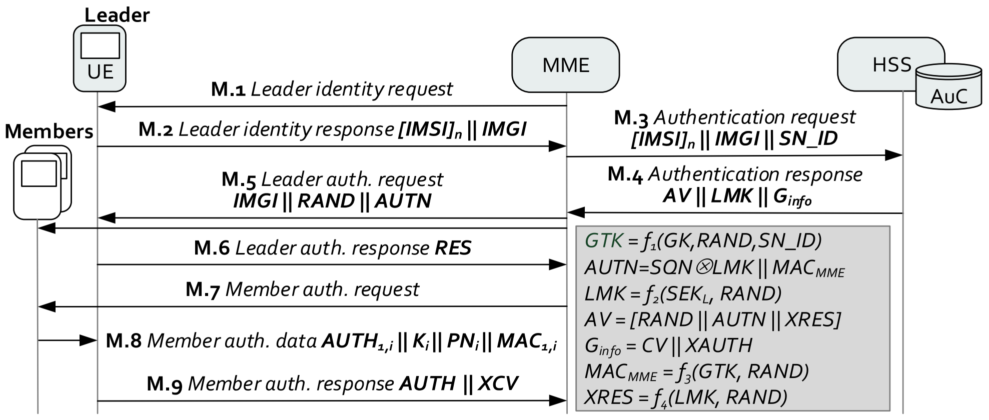

Leader
UE
MME
HSS
AuC
Members
M.1 Leader identity request
M.2 Leader identity response [IMSI]$_n$ || IMGI
M.3 Authentication request [IMSI]$_n$ || IMGI || SN_ID
M.4 Authentication response AV || LMK || G_{info}
M.5 Leader auth. request IMGI || RAND || AUTN
M.6 Leader auth. response RES
M.7 Member auth. request
M.8 Member auth. data $AUTH_{1,i}$ || K_i || PN_i || $MAC_{1,i}$
M.9 Member auth. response AUTH || XCV
GTK = f_1(GK,RAND,SN_ID)
AUTN=SQN⊗LMK || MAC_{MME}
LMK = f_2(SEK_L, RAND)
AV = [RAND || AUTN || XRES]
G_{info} = CV || XAUTH
MAC_{MME} = f_3(GTK, RAND)
XRES = f_4(LMK, RAND)

Fig. 4 Authentication and key agreement between a leader and the core network

from the secret key (SEK_L) shared between the leader and the HSS as shown in Eq. (2).

$$LMK = f_2(SEK_L, RAND) \tag{2}$$

LMK is equivalent to the local master key (K_{ASME}) in the EPS-AKA and can be used to derive NAS and AS keys. The authentication vector (*AV*) is composed of three parameters, $RAND, XRES, AUTN$. The parameter *XRES* is the expected response from the device for authentication. The authentication token (*AUTN*) contains the message authentication codes (MAC_{MME}) and a sequence number (*SQN*) encrypted with *LMK*. This sequence number prevents an adversary from using an authentication vector repeatedly. The message authentication code is derived by using $MAC_{MME} = f_3(GTK, RAND)$. The group information (G_{info}) contains two parameters; one is a confirmation value (*CV*), and the other is *XAUTH*. The MME uses the confirmation value to verify the group membership of MTC device and uses *XAUTH* to authenticate an individual device.

M.5: The MME sends this message to the chosen leader via a broadcast channel in the LTE downlink. The basic nature of broadcasting means all members should be able to listen to this fifth message. The leader first checks if *IMGI* is equal to its group identity and then validates *AUTN* to authenticate the MME by verifying MAC_{MME}. The MTC devices do the same to verify the MME.

M.6: The chosen leader computes *LMK* and extracts *SQN* to detect any reuse of the authentication vector. Then the leader prepares response value *RES* and sends the leader's authentication response (**M.6**) to the MME. Hereafter, the NAS security secures communication between the leader and the MME.

M.7: The MME authenticates the leader by checking the equivalence of *RES* and *XRES*. The MME broadcasts the member authentication request (**M.7**) to all MTC devices in the group. The identity of the chosen leader is included in this message for identification purposes.

M.8: MTC device *i* computes its own $K_i$and $AUTH_{1,i}$, respectively, using Eqs. (3) and (4).

$$K_i = KDF(PN_i \oplus SEK_i) \tag{3}$$

$$AUTH_{1,i} = f_3(SEK_i, PN_i) \tag{4}$$

$$MAC_{1,i} = H(K_i, AUTH_{1,i}) \tag{5}$$

where SEK_i is a secret key shared between each device and the HSS and PN_i is a given prime number. The device sends four parameters K_i, PN_i, $AUTH_{1,i}$ and $MAC_{1,i}$ to the leader in cases in which the leader appears not to have these values. Because the leader saves each device's four parameters for all MTC devices, only those devices new to the group report this eighth message to the leader. In our design, these four parameters do not change with every round of authentication, which is our way of preventing a traffic bottleneck with the leader. However, if security is threatened, the HSS can change K_i, and $AUTH_{1,i}$ anytime by replacing PN_i in order to update these four parameters in the MTC device. The parameter $MAC_{1,i}$ computed by using Eq. (5) serves to authenticate **M.8** for the leader. The MTC device sends **M.8** to the leader in the secondary channel via a Wi-Fi hotspot after encrypting this message with the session key. An agreement on the session key is discussed in Sect. 4.4.

M.9: The leader collects **M.8** from the devices and applies the Chinese Remainder Theorem (CRT) over K_i and modulus PN_i to compute a challenge to the confirmation value (*XCV*) as shown in Eq. (6).

$$XCV' = K_1 \bmod PN_1 = K_2 \bmod PN_2 = \cdots = K_k \bmod PN_n \tag{6}$$

$$XCV = XCV' \oplus SQN$$

The leader also uses Eq. (7) to prepare $AUTH$ for each member's authentication token.

$$AUTH = H\left(AUTH_{1,1} \oplus AUTH_{1,2} \oplus \cdots \oplus AUTH_{1,n} \oplus SQN\right) \tag{7}$$

The leader sends the member authentication response to the MME. This message is composed of XCV and $AUTH$. To validate that these members belong to the same group, the MME compares the CV received from the HSS and the XCV received from the leader. The MME authenticates each MTC device by comparing $AUTH$ with $XAUTH$. The AKA is successful at this point and the MME is ready join the group as a regular member.

Theorem (Chinese Remainder Theorem) *Suppose* $m_1, \cdots, m_n$ *are pairwise relatively prime, that is,* $(n_i, m_j) = 1$ *for* $i \neq j$ *and* $M = \prod_{i=1}^{n} m_i$*. Then the system of the following congruencies*

$$A = a_1 (\bmod\ m_1), A = a_2 (\bmod\ m_2), \cdots, A = a_n (\bmod\ m_n)$$

has a unique solution for mod M*. As a result, we can represent any integer of* A *by a n-tuple of integers* a_i *by using the following correspondence:*

$$A \leftrightarrow (a_1, a_2, \cdots, a_n)$$

where A *is less than* M *and* a_i *is less than* m_i *for all* n*.*

4.3 Membership update phase

4.3.1 *Joining a group*

Although the MME joins a group soon after successful completion of mutual authentication, MTC devices become a group member at the time of offline registration with the service provider. At the outset of joining, the HSS assigns the MME to an empty leaf node in the tree and after encryption sends it an information packet containing the secret key they will share. The package includes a new group key (GK'), the new member's secret values, identity, and location in the binary tree.

At this time, to maintain backward secrecy, all members in the group and the HSS use Eq. (8) to restructure the binary tree to reflect the new arrival and update the group key to the new one, GK',

$$GK' = H(GK \oplus SECRET_{NODE_m}) \tag{8}$$

where $SECRET_{NODE_m}$ is the secret value of the joining node. The joining member is not able to access the old group key because of the one-way hash function.

It would be preferable to locate a rejoining member in the same position as before in the binary tree because we want to maintain forward and backward secrecy. Hence, we would like to assign the MME at the same position in a tree. However, if the HSS finds that the returning device's position is already occupied by another member, the HSS appends two children to the original node and moves the new member to the left child and the rejoining member to the right. The binary tree may become severely unbalanced because of our placement policy. If the HSS find the binary tree is too skewed, then it can create a new balanced binary tree for this group from scratch.

4.3.2 *Leaving a group*

Devices may leave a group once their missions are completed or they exhaust their power sources. The serving MME can be changed at any time as a result of a member's mobility. Leaving procedures for devices and for the MME are the same.

When a member m_k leaves a group, all remaining members in the group update a group key, GK to GK'' according to Eq. (9)

$$GK'' = GK \oplus SECRET_{m_k} \tag{9}$$

Note that unlike Eq. (8) the hash operation is unnecessary in the departure process. This is because a single hash operation is sufficient to prevent reversal of the group key to old group keys. The HSS notifies remaining members of the departure by broadcasting the identity, m_k. The departing member m_k cannot update the group key because $SECRET_{m_k}$ is the RS value of m_k. For instance, when $Member_5$ in Fig. 5 leaves the group, the remaining group members update the group key by XORing the current group key with $SECRET_{NODE_{12}}$. However, $Member_5$ cannot update the group key because it cannot derive $SECRET_{NODE_{12}}$. This inability to update the group key guarantees forward secrecy.

4.4 Session key agreement phase

When an MME's membership is complete, the MTC devices proceed to the next stage to establish session keys between the MME and the individual devices in the group. The MME and device m_k find the common values between their secret values. In case two common values are children of a parent, then this parent is chosen instead to reduce the number of common values. The MME and the device execute the XOR operation over the common secret values. Then, the hashed result of the output is determined to be the session key. For example, as shown in Fig. 5, let us assume that the MME and the member are located at leaf nodes of $Node_7$ and $Node_{12}$, respectively. The common secret values of these two nodes are $Node_8$, $Node_4$, $Node_{11}$ and $Node_6$, and the session key $SK_{7,12}$ is computed using Eq. (10).

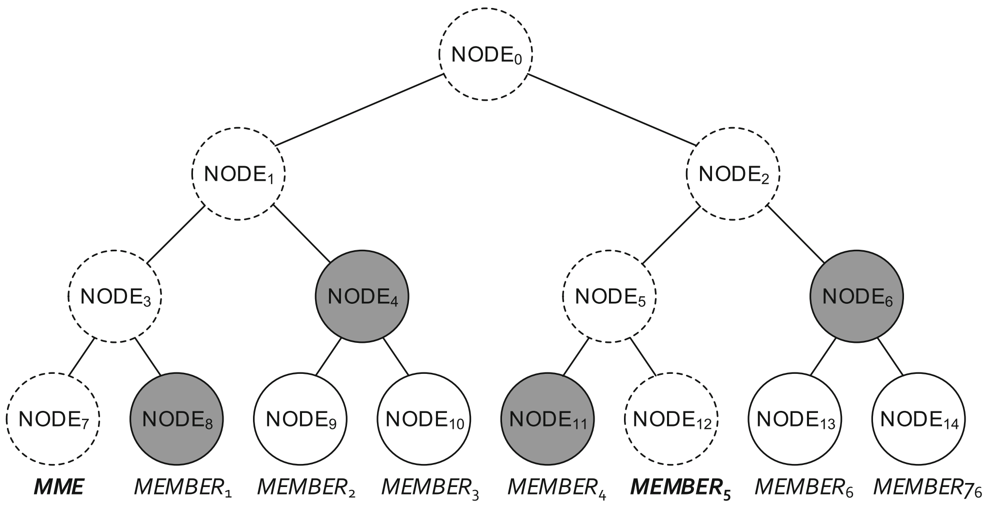


Fig. 5 Illustration of session key agreement between any members

$$SK_{7,12} = H((SECRET_{NODE_4} \oplus SECRET_{NODE_6} \oplus SECRET_{NODE_8} \oplus SECRET_{NODE_{11}}) \parallel RAND) \tag{10}$$

where *RAND* is a random value received from **M.5**. The session key is unique and hence secure in the sense that no leaf nodes other than $Node_7$ and $Node_{12}$ know these four common secret values.

This agreement on a session key between the MME and the device is closely analogous to the one between devices. The session key between the MME and the device is used to encrypt traffic flowing from and to the core network. The session key between MTC devices is used to distribute information among devices in such D2D communication or to encrypt **M.8** for secure communication between the leader and the device.

5 Security analysis

We subjected our proposed protocol to a security analysis. We contend that our protocol supports all security requirements demanded for MTC.

5.1 Formal verification

We used ProVerif [19] to assess the security properties of the proposed protocol. This tool is an automatic cryptographic verifier in the formal model. It has been used widely and successfully to model and analyze several security protocols. This tool is known for its capability to provide security properties verification for the authentication, confidentiality, and secrecy of a session key. Consequently, the proposed protocol was verified through ProVerif as achieving mutual authentication between a leader or an MTC device and MME, and as ensuring the secrecy of session keys shared by two authentication entities. The appendix is provided in the extended version of this paper to confirm verification results by ProVerif.

5.2 Mutual authentication

The message authentication code $MAC_{MME} = f_3(GTK, RAND)$ sent in **M.5** (see Fig. 4) is created with the group temporary key, *GTK*. Because *GTK* is shared only between the HSS and group members, the members, including a leader, can authenticate the MME if MAC_{MME} proves authentic.

For authentication in the other direction, the MME on behalf of the HHS authenticates the leader by comparing the value of *RES* with *XRES*. The leader can prove its knowledge of the local master key *LMK* and further *GTK* by presenting the correct value of *RES*. The MME is able to authenticate each MTC device if *AUTH* as computed in Eq. (7) matches *XAUTH*. In order to validate that these members belong to the same group, the MME compares *CV* received from the HSS and *XCV* received in **M.9** from the leader. According to the CRT, *XCV* is a unique solution for the combination of PN_i with all members.

5.3 Confidentiality and integrity

Once the proposed AKA is completed, then the MME joins a tree for group management and is able to share a session key [$SK_{i,j}$ in Eq. (10)] with individual devices in the tree. Several NAS and AS keys are derived from this session key to ensure the confidentiality and integrity of links between

Table 1 Length of parameters used in measurement of communication costs

Parameters	Length (bits)
AMF	16
Timestamp	32
MAC	48
Hash value/RES	128
RAND, ID, IMSI, IMGI	128
Asymmetric/symmetric keys	128
PN, SQN, AUTH, SECRET	128
Modulus value, IV	128
Multiplication/additive value over elliptic curve	160
LMK, encrypted message	256
Pseudo ID	320
ECDSA signature	448

the member and the MME and between the member and eNodeB.

5.4 Forward secrecy and backward secrecy

When a secret key is shared in a group, great care is required to restrict the key to members only. For instance, when a member leaves the group, its knowledge of the group key must be ended by updating the group key among the remaining group members. Furthermore, a new member should not be able to access the group keys used before it joined. The former requirement denotes forward secrecy and the latter denotes backward secrecy.

We contend that the proposed protocol could accommodate both the forward and backward secrecy demanded for group key management. The group key is updated whenever group membership changes. When a new member arrives, the group key is updated with this member's secret value by using Eq. (8). Because this new member does not know its secret value, reversion to the old group keys is prevented. Similarly, a departure also triggers updating of the group key by using Eq. (9); this new key cannot be accessed by the departing member for the same reasons that departures also trigger updates.

5.5 Replay

Wireless communication links between members and between the MTC device and the MME are subject to replay attacks. Because the leader and the MME agree on the session key after **M.6,** an adversary could overhear the three meaningful messages of **M.2**, **M.5,** and **M.6** as shown in Fig. 4. With possession of these three messages, an adversary might attempt authentication by pretending to be a leader and replaying **M.6** to the MME. However, this attempt would fail immediately because the value of RES recorded through eavesdropping differs from the one the MME expects because of a new value of $RAND$ in $RES = f_4(LMK, RAND)$.

In another situation, an external adversary might replay **M.8** $AUTH_{1,i} \parallel K_i \parallel PN_i \parallel MAC_{1,i}$ and **M.9** $AUTH \parallel XCV$ to penetrate a group. The MME should be able to reject the forged **M.9** immediately because the sequence number (SQN) in Eq. (6), $XCV = XCV' \oplus SQN$, has been updated, and consequently, the two confirmation values, XCV and CV, do not match. A replay of **M.8** is quite difficult to detect in the proposed AKA because this message changes infrequently. This replay does not jeopardize the system because any parameters are altered by the replayed message. We dispensed with a timestamp option to protect **M.8** from being replayed because time synchronization can be another burden for the MTC device.

5.6 Man-in-the-middle (MitM)

In a man-in-the-middle attack scenario, an adversary plays two roles by pretending to be a leader to the MME and pretending to be the MME to a leader and members. An adversary executing this attack disguised as a leader could send a forged **M.6** to the MME. This would fail because of the inability to compute RES without knowing the local master key. Similarly, an adversary in the guise of the MME could not easily deceive the leader. On hearing **M.2,** the adversary would send a forged **M.5** with an arbitrary sequence number (SQN') to the leader. In general, the correct sequence number should be in the range of the

Table 2 Comparison of communication costs per message when $n = 10$ and $t = 20$

(Kbits)	M.2	M.3	M.4	M.5	M.6	M.7	M.8	M.9	M.10	M.11	M.12	M.13	Total
Proposed	28.2	30.7	14.6	9.0	2.6	2.6	46.1	51.2	7.7				192.58
EPS-AKA	25.6	51.2	140.8	64.0	25.6								307.20
Cao-AKA	1.3	1.3	1.3	1.3	1.3	2.6	7.0	3.2	1.3	109.4	63.4	17.9	201.66
ABAKA	134.4	128.0											262.40
G-AKA	89.6	9.0	8.6	89.3	11.5								208.00
SE-AKA	78.4	11.8	84.8	198.4	41.6								415.04
EG-AKA	112	13.8	84.8	92.8	57.6	51.2	51.2						463.04

Table 3 Average elapsed time of nine cryptographic operations used in comparing computational delays

Operations	Symbol	Time (μs)
RAND-128	T_{rand}	45
HMAC-SHA-256	T_{hash}	67
XOR	T_{xor}	2
Modulus	T_{mod}	124
AES-256	T_{aes}	161
Multiplication over elliptic curve	T_{mul}	612
Addition over elliptic curve	T_{add}	125
MaptoPoint hash function	T_{mtp}	525
Pairing	T_{pair}	4514

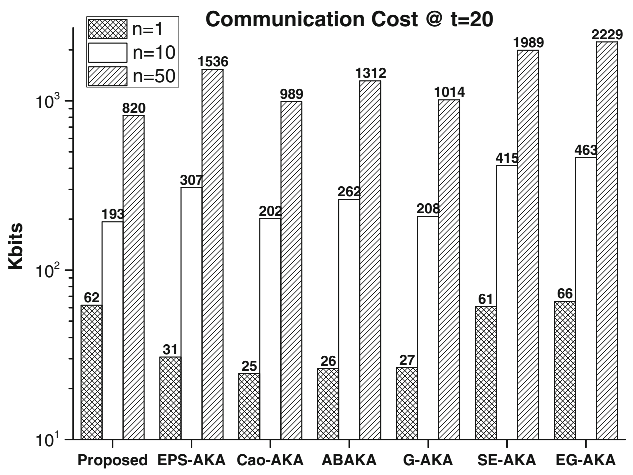


Fig. 6 Comparison of communication costs for seven AKAs when $n = 1$, 10, and 50

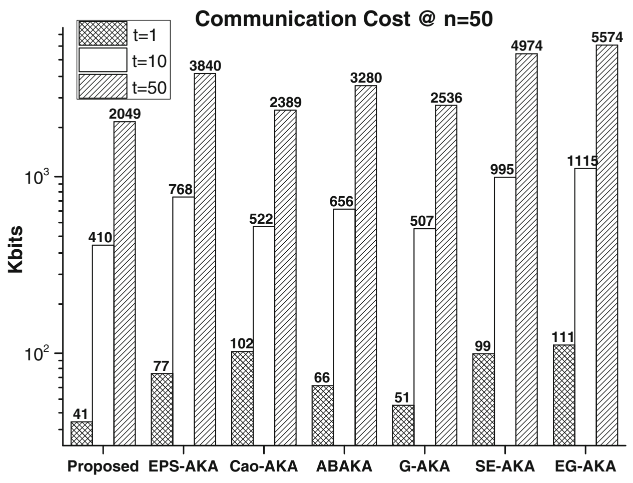


Fig. 7 Comparison of communication costs for seven AKAs when $t = 1$, 10, and 50

authentication vector indices. The leader extracts $SQN^{`}$ from $AUTN$ and examines the integrity of $SQN^{`}$ against MAC_{MME}. The integrity test should fail, and hence, the MITM attack would also fail.

5.7 Impersonation

A non-member impersonates a group member to try to receive the benefit of membership in illegitimate ways. An adversary might initiate an impersonation attack by sending **M.8** in Fig. 4 $AUTH_{1,i} \| K_i \| PN_i \| MAC_{1,i}$ to the leader. **M.8** is authenticated by the last parameter $MAC_{1,i}$ and is encrypted with a secret session key $SK_{i,j}$ shared by a member and a leader. Any members other than member i cannot derive the session key $SK_{i,j}$, and hence, the impersonation attack would fail.

6 Performance evaluation

Some design decisions were made in the course of simulating the system. Those decisions were made after taking into consideration both practical implementation issues and performance issues.

6.1 Communication costs

Communication cost is by definition the number of bits to complete the t times of the repetitive AKAs for the n number of MTC devices. It is a function of n, t, and a sum of the message length in a single AKA incident. For comparison purposes, we measured the communication cost of the proposed AKA and the communications costs of six other protocols that have been advanced. They are: (1) EPS-AKA [16], (2) Cao-AKA [10], (3) ABAKA [9], (4) G-AKA [6], (5) SE-AKA [8] and (6) EG-AKA [7]. The measurements are based on the length of the parameters in Table 1.

The MTC device exchanges five messages to complete the EPS-AKA. It required $n \times t \times 5$ message for n devices to complete t times of the EPS-AKA. The sum of message length for a single EPS-AKA is 1536 bits. Consequently, the communication cost of the EPS-AKA is $1536nt$. In the beginning, each MTC device in the Cao-AKA exchanges two messages with the key center and then executes the EPS-AKA. It takes $9n$ messages for n devices. In the rest period of $t - 1$ times of the AKA, a group leader accepts a single message from $n - 1$ devices and exchanges another two messages with the core network. As many as $9n + (n + 1)(t - 1)$ messages are required to complete t times of the AKA for n devices. This accounts for a total communication cost of $1120n + 928nt + 256t - 256$ (see Table 3).

Table 4 Evaluations of the computational delays demanded by the device and the core network

Proposed	Device	$2t(n+1)T_{hash} + 2t(n+1)T_{xor} + ntT_{mod}$
	Core network	$(2nt+6t)T_{hash} + (2nt+3t)T_{xor} + tT_{rand} + ntT_{mod} + tT_{aes}$
EPS-AKA	Device	$6ntT_{hash} + ntT_{xor}$
	Core network	$6ntT_{hash} + ntT_{xor} + ntT_{rand}$
Cao-AKA	Device	$6nT_{hash} + nT_{xor} + n(t-1)T_{rand} + n(5t-3)T_{mul} + n(2t-1)T_{mtp} + 3n(t-1)T_{add}$
	Core network	$nT_{hash} + nT_{xor} + (3n+t-1)T_{rand} + 2(n+t+nt-1)T_{mul} + 2ntT_{mtp} + n(t+1)T_{add}$
ABAKA	Device	$ntT_{mul} + ntT_{mtp} + ntT_{add}$
	Core network	$t(5n+2)T_{mul} + 2ntT_{mtp} + 2ntT_{add} + tT_{rand}$
G-AKA	Device	$t(4n-3)T_{hash} + ntT_{rand}$
	Core network	$2ntT_{hash} + ntT_{rand}$
SE-AKA	Device	$t(4n+1)T_{hash} + 2ntT_{mul} + 2ntT_{rand}$
	Core network	$t(3n+2)T_{hash} + 2ntT_{mul} + t(2n+1)T_{rand}$
EG-AKA	Device	$9ntT_{hash} + 2ntT_{mul} + ntT_{rand} + ntT_{aes}$
	Core network	$2t(3n+1)T_{hash} + 2ntT_{mul} + t(2n+1)T_{rand} + ntT_{aes}$

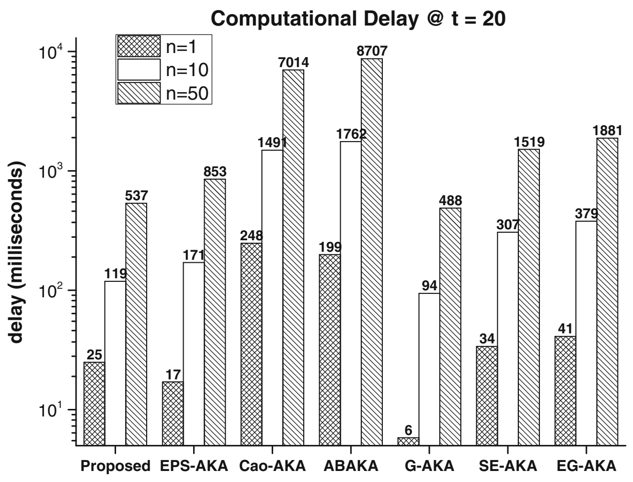


Fig. 8 Comparison of computational delays for seven AKAs when $n = 1$, 10 and 50

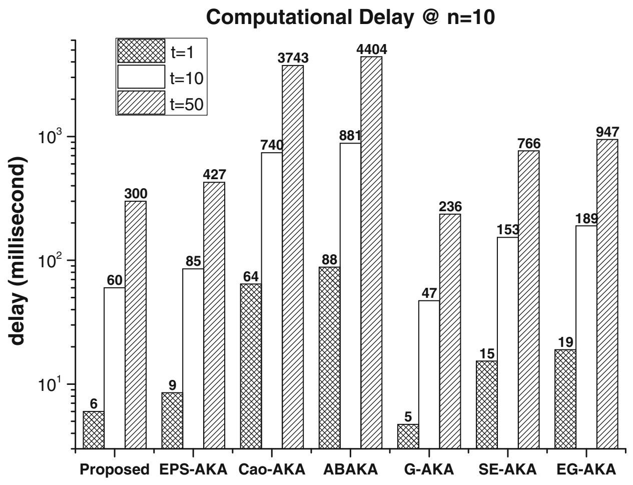


Fig. 9 Comparison of computational delays for seven AKAs when $t = 1$, 10, and 50

A group leader in the proposed AKA exchanges the same number of five messages[1] as the EPS-AKA. The leader collects and processes the response messages from $n-1$ devices and delivers them to the MME. When the MME can authenticate the leader and $n-1$ devices, the HSS creates an admission message and broadcasts it to the group members. The six messages necessary from the device to the MME and the three messages in the other direction amount to $n+9$ messages for completion of a single AKA. For the next $t-1$ times of the AKA, a single MTC device sends $6(t-1)$ messages to the MME and accepts $4(t-1)$ messages from the MME. The total number of messages adds up to $n+10t-1$.

Table 2 compares the communication costs per message when $n = 10$ and $t = 20$. The proposed AKA and the Cao-AKA require 10 and 13 messages, respectively, for a single AKA round. After 20 repetitions of the AKA, the communication costs of the proposed AKA and the Cao-AKA are 192.58 Kbits and 201.55 Kbits, respectively. In comparison, the proposed AKA and the Cao-AKA demand more messages than the EPS-AKA and the ABAKA, but these two protocols outperform other AKAs by using fewer bits for communication. Messages in the ABAKA tend to be long because of the signature and asymmetric encryption. The EPS-AKA, a lack of grouping optimization, increases the communication cost linearly with the number of devices involved.

Figure 6 compares the communication costs of five AKAs for $n = 1$, $n = 10$, and $n = 50$, where the number of repetition is fixed at 20 ($t = 20$). If the number of MTC devices is one of the communication costs, the proposed AKA is the most expensive. As the number of members increases, the proposed protocol demands less bandwidth

[1] Note that the first message (**M.1**) is not included in computation of the communication cost.

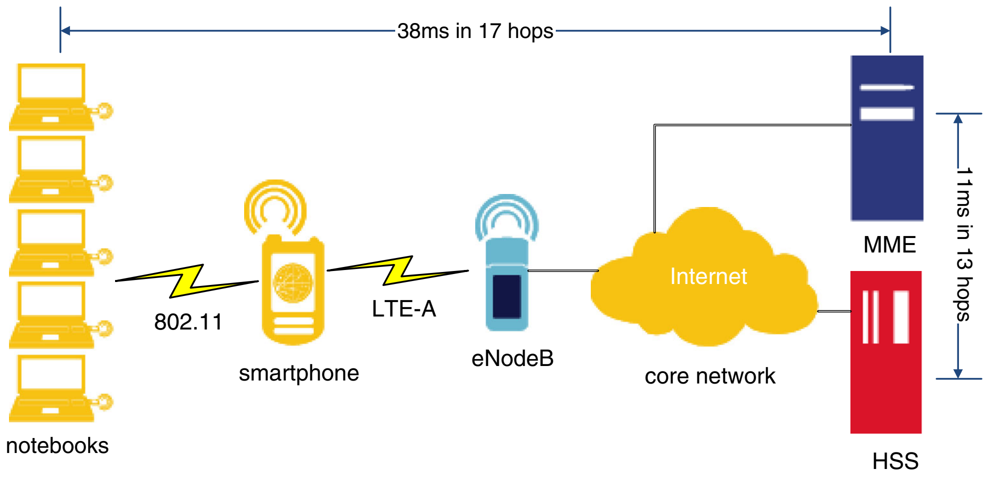


Fig. 10 Implementation of legacy and proposed protocols in the real LTE-A network for measurement of actual delay

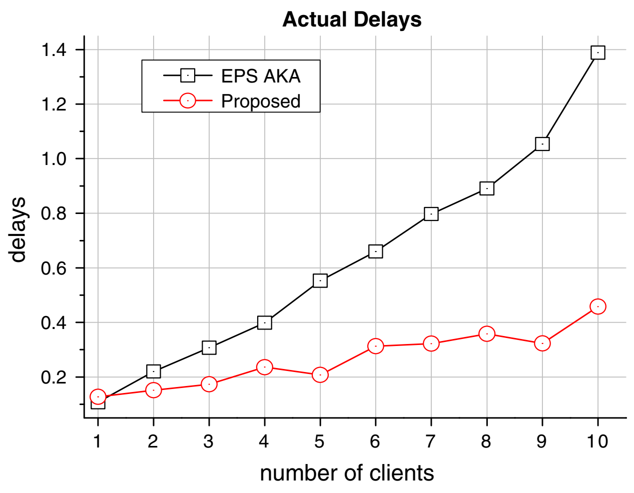


Fig. 11 Delay measurement for the proposed AKA and the EPS-AKA

by reaping the benefit of grouped requests. Figure 7 also compares the communication costs of seven AKAs for $t = 1$, $t = 10$, and $t = 50$, where the number of devices is fixed at 50 ($n = 50$). In this comparison, the proposed AKA is the most efficient. The communication cost of the proposed is improved by 25 percent over G-AKA at $t = 1$. Note that the communication cost of the Cao-AKA is greatest after the first round of the AKA. After 50 repetitions of the AKA, the performance of this Cao-AKA protocol is second to the proposed AKA.

6.2 Computational delays

We took advantage of the *Crypto* ++ Library [20] to measure the elapsed time of the cryptographic operations. The measurement ran on an Intel Core Duo 1.86 GHz and 2 gigabyte RAM under an Ubuntu 11.10 operating system. Table 3 demonstrates the average elapsed time of nine cryptographic operations.

The HSS and the MME in the core network and the devices execute many cryptographic operations in the process of message generation. We analyzed these cryptographic operations in each message and summed the elapsed time of the operations for all messages that consist of the AKA as a way to measure and compare the computational delays of the seven protocols. Table 4 displays equations of the computational delays demanded by the device and the core network. Based on these equations, we compared the computational delays of the seven protocols in Figs. 8 and 9 with different values of n and t.

Figure 8 compares the computational delays at $t = 20$ when the number of MTC devices equals 1, 10, and 50. The Cao-AKA and the ABAKA employ asymmetric elliptic curve cryptography (ECC) as a security primitive. The pairing operation in the ECC is quite expensive, revealing that these two AKAs perform poorly on cryptographic operations. The computational delay of the proposed AKA improves by 38 percent over the one by the EPS-AKA when the number of devices is 50. Figure 9 shows a similar trend as in Fig. 8, except for the difference in values.

The G-AKA performs better than our protocol in all three different values of n (see Fig. 8). This is attributed to the fact that the G-AKA adopts symmetric cryptography that demands only a five-message exchange. However, we have not yet confirmed the security of the G-AKA, which is subject to an MITM attack.

6.3 Actual delay measurement

We have implemented the legacy and proposed protocols in the real LTE-advanced network so as to measure actual delays. This measurement is especially important to developers and engineers in the mobile industry as a determinant of whether to deploy these protocols in the real environment.

In our implementation, as shown in Fig. 10, ten MTC devices were implemented on five notebooks to form a group connected to one another through a Wi-Fi network. A tethering smartphone serves as an access point to the Wi-Fi network through which the MTC devices can access the LTE-A and the core network. The MME's role and the HSS's role are implemented on two desktop PCs located far apart on the Internet. The distance between the entities as measured using *ping* show that the two MTC devices are separated on average by 15 ms. It takes 17 hops of 38 ms from the device to the MME and 13 hops of 11 ms from the MME to the HSS.

Figure 11 shows our experiment with the delay measurements for the proposed AKA and the EPS-AKA. The number of clients varies from 1 to 10. Each point in Fig. 11 is an average of 50 repeated authentication delays (i.e. $t = 50$) for each protocol. When a group has only one device, the average delays of the EPS-AKA and the proposed protocol, respectively, are 0.11 and 0.13 seconds. Shortly after the number of devices in a group becomes two or more, the proposed protocol outperforms the EPS-AKA as shown in Fig. 11.

7 Conclusion

The 3GPP sees MTC as a significant sector in the LTE-A network for fourth-generation mobile communications. This paper proposed an applicable security mechanism for AKA and its adaptation to the MTC. The proposed mechanism improves grouping optimization by aggregating authentication requests and uses Wi-Fi hotspots as a secondary channel to share data among devices. Extensive security analysis and formal verification by using ProVerif have shown that the proposed AKA is secure against diverse malicious attacks. Thorough analysis and comprehensive evaluations with respect to communication overhead, computational overhead and actual implementation confirm that the proposed AKA outperforms other existing AKA solutions that have been advanced.

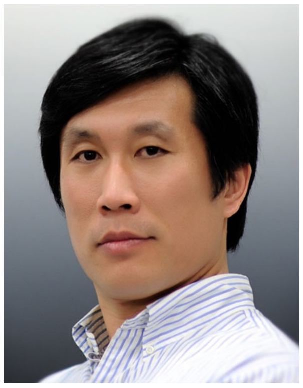
Finally, we believe that there are several aspects that still need be investigated in order to improve the delay response associated with AKA. We want to expand experiments of the study in diverse service usages to see how the system perform in different environments. Further, our proposed mechanism could be extended to make use of more advanced access networks, such as multi hops and D2D. We hope that the ideas presented here, and the discussion of the challenges that lie ahead, will motivate researchers to solve the security issues presented in M2M communications.

References

1. Lien, S.-Y., et al. (2011). Toward ubiquitous massive accesses in 3GPP machine-to-machine communications. *IEEE Communications Magazine, 49*(4), 66–74.
2. Jain, P., Hedman, P., & Zisimopoulos, H. (2012). Machine type communications in 3GPP systems. *IEEE Communications Magazine, 50*(11), 28–35.
3. Taleb, T., & Kunz, A. (2012). Machine type communications in 3GPP networks: Potential, challenges and solutions. *IEEE Communications Magazine, 50*(3), 178–184.
4. Lee, T., et al. (2009). Enhanced delegation-based authentication protocol for PCSs. *IEEE Transactions on Wireless Communications, 8*(5), 2166–2171.
5. Zhang, Y., et al. (2012). Dynamic group based authentication protocol for machine type communications. In *IEEE International Conference on Intelligent Networking and Collaborative Systems (InCoS).*
6. Chen, Y., et al. (2010). Group-based authentication and key agreement. *Springer Wireless Personal Communications, 62*(4), 965–979.
7. Jiang, R., et al. (2013). EAP-based group authentication and key agreement protocol for machine-type communications. *International Journal of Distributed Sensor Networks (Hindawi).*
8. Lai, C., et al. (2013). SE-AKA: A secure and efficient group authentication and key agreement protocol for LTE networks. *Computer Networks (Elsevier), 57*(17), 3492–3510.
9. Huang, J., et al. (2011). ABAKA: An anonymous batch authenticated and key agreement scheme for value-added services in vehicular ad hoc networks. *IEEE Transaction on Vehicular Technology, 60*(1), 248–262.
10. Cao, J., et al. (2012). A group-based authentication and key agreement for MTC in LTE networks. In *IEEE Global Communications Conference (Globecom).*
11. Wong, C., et al. (1998). Secure group communication using key graphs. In *ACM Conferences on Applications, Technologies, Architectures, and Protocols for Computer Communication (Sigcomm).*
12. Harney, H., et al. (1999). Logical key hierarchy protocol. IETF Internet Draft.
13. Pietro, R., et al. (2002). Efficient and secure keys management for wireless mobile communications. In *ACM International Workshop on Principles of Mobile Computing (POMC).*
14. Astely, D., et al. (2013). LTE release 12 and beyond. *IEEE Communications Magazine, 51*(7), 154–160.
15. Yang, M., et al. (2013). Solving the data overload: Device-to-device bearer control architecture for cellular data offloading. *IEEE Vehicular Technology Magazine, 8*(1), 31–39.
16. 3GPP TS 33.102 ver.11.5.1. (2013). 3G security: security architecture (release 11).
17. 3GPP TR 33.868 ver.12.0.0. (2014). Security aspects of machine-type communications (release 12).
18. Han, C., & Choi, H. (2014). Security analysis of handover key management in 4G LTE/SAE networks. *IEEE Transaction on Mobile Computing, 13*(2), 457–468.

19. Blanchet, B., Smyth, B., & Cheval. V. (2013). ProVerif 1.88: Automatic cryptographic protocol verifier, user manual and tutorial.
20. Crypto++. http://www.cryptopp.com/.

Daesung Choi received a M.S. degree (2014) in IT Convergence from Sungkyunkwan University in South Korea. He is a software engineer at Samsung Electronics. His research interests include security and privacy in mobile communications.



Hyoung-Kee Choi received a Ph.D. degree in electrical and computer engineering from Georgia Institute of Technology in 2001. He is an associate professor in Sungkyunkwan University, Korea. He joined Lancope in 2001 and remained until 2004, where he contributed to research in Internet security. His research interests span network security and reverse engineering.

Se-Yeong Lee is a M.S. student in IT Convergence at Sungkyunkwan University in South Korea. He received his B.S. degree (2014) in Computer Engineering from Sungkyunkwan University in South Korea. He has research interests in network security, especially authentication.