A PROVABLY GROUP AUTHENTICATION PROTOCOL FOR VARIOUS LTE NETWORKS

Boriphat Kijjabuncha

A Dissertation Submitted in Partial
Fulfillment of the Requirements for the Degree of
Doctor of Philosophy (Computer Science and Information Systems)
School of Applied Statistics
National Institute of Development Administration
2018

A PROVABLY GROUP AUTHENTICATION PROTOCOL FOR VARIOUS LTE NETWORKS

Boriphat Kijjabuncha

School of Applied Statistics

Associate Professor ..... Major Advisor

(Pipat Hiranvanichakorn, D.E.)

The Examining Committee Approved This Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy (Computer Science and Information Systems)

Associate Professor ..... Committee Chairperson

(Ohm Sornil, Ph.D.)

Associate Professor ..... Committee

(Panjai Tantatsanawong, Ph.D.)

Assistant Professor ..... Committee

(Nithinant Thamakornnonta, Ph.D.)

Associate Professor ..... Committee

(Pipat Hiranvanichakorn, D.E.)

Assistant Professor ..... Dean

(Pramote Luenam, Ph.D.)

December 2018

ABSTRACT

Title of Dissertation	A Provably Group Authentication Protocol for Various LTE Networks
Author	Mr. Boriphat Kijjabuncha
Degree	Doctor of Philosophy (Computer Science and Information Systems)
Year	2018

Group authentication is beneficial for group work in the Long Term Evolution (LTE) networks because it reduces the traffic of networks. For practical use, members of a group should be able to come from different network providers. In addition, while some group members use a network service, others may use other network services. Although the group members are on different networks, they should be able to work together. To fulfill these needs, we propose a secure group authentication protocol (SE-GA) in which each group member uses his/her long-term private key and public key to create shared secret (keys) with network devices, such as Home and mobile management entity (MME). These shared keys are computed by using the Diffie-Hellman key exchange and are utilized in the authentication process. By using this technique instead of pre-shared keys between mobile devices and network devices, SE-GA is flexible and scalable. In SE-GA, only the first member in an MME's area has to authenticate himself/herself with the Home, while the remaining members in the area can authenticate directly with the MME. Thus the protocol reduces the amount of network usage.

In this research, authentication proof is also given using the well-known BAN logic. Security analysis of the proposed protocol is also given and a comparison of our protocol with SE-AKA and GLARM was demonstrated. According to the comparison, we can see that the proposed protocol outperforms the former ones.

ACKNOWLEDGEMENTS

I would first like to thank you my supervisor, Associate Prof. Pipat Hiranvanichakorn for his encouragement, constructive comments over the years. I greatly appreciate and respect his patience and guidance, and I fully intend to follow his teaching guidelines.

I would also like to thank Assistant Prof. Nithinant Thammakoranonta who guides me to write this thesis in the field of information systems and made many useful comments. My sincere thanks also go to my thesis committee, Associate Prof. Ohm Sornil and Associate Prof. Panjai Tantatsanawong, for their encouragement, insightful comments, and challenging questions.

I must express my very profound gratitude to my parents and to my fellows and the faculty staff at Silpakorn Univerisity (SU) and National Institute of Development Administration (NIDA), for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing the thesis. This accomplishment would not have been possible without them. Thank you

Finally, I would like to thank the Faculty of Information and Communication Technology, Silpakorn University for the scholarship and the opportunity to study.

Boriphat Kijjabuncha

December 2018

TABLE OF CONTENTS

LIST OF TABLES

Tables **Page**

LIST OF FIGURES

Figures **Page**

ABBREVIATIONS

Abbreviations	Equivalence
3GPP	The 3rd Generation Partner Project
BS	Base Station
HFS	Home Facilitator Server
HN	Home Network
HSS	Home Subscriber Server
LTE	Long Term Evolution
ME	Mobile Equipment
MME	Mobile Management Entity
MTC	Machine Type Communication
S-GW	Serving Gateway
SN	Serving Network

CHAPTER 1

INTRODUCTION

1.1 Background

Internet network communication is useful in many areas such as online trading, information exchange, group working due to its speed in data transmission with lower cost. Although data transmission across the Internet is quick and easy, it cannot be guaranteed that data is secured during communication. Therefore, the importance of data security on the Internet is paramount.

In this research, we introduce a group model that helps users to work with their group even though they live in different LTE networks. However, group communication needs security management to control the risks occurred in the system and protect against unauthorized users causing a system failure. Thus, network applications need privacy, confidentiality, integrity, authentication methods to protect their information from unauthorized access.

For the mobile environment, in order to use services of a network, mobile equipment such as smart phones, smart watches, laptops have to authenticate themselves with their home networks (HNs). However, if several mobile equipment in the same group authenticate with their HNs at the same time the traffic of the network will be crowded. This can reduce the stability of the system, and the performance of the network decreases. Therefore, an efficient group authentication protocol is needed in the group model.

Till now many works have been studied in the group communication and authentication (Cao, Ma and Li, 2012; Chen, Wang, Chi and Tseng, 2012; Lai, Li, Lu and Shen, 2013; Wang, Chang and Chou, 2015; Lai, Lu, Zheng, Li and Shen, 2016).

In 2010, a cocktail protocol with authentication and key agreement (Cocktail-AKA) on the Universal Mobile Telecommunications System (UMTS) is proposed by

Ou, Hwang and Jan (2010). The protocol allows a service network (SN) to calculate the medicated authentication vectors (MAV) in advance. MAV is calculated only once and can be reused. The MAV is used with prescription authentication vector (PAV) to produce many effective authentication vectors (AVs) for mutual authentication with the mobile stations (MSs). PAV is calculated from Home Environment (HE). Even though the protocol can reduce computational overhead on the HE and communication overhead for delivering the AVs, the protocol has some weaknesses which cannot resist denial-of-service attack (DoS attack) as described by Wu, Zhu and Pu (2010). After two years, Cao, Ma and Li (2012) proposed a group-based authentication scheme and key agreement for Machine Type Communication (MTC) in LTE network. In the protocol, the traffic of authentication is crowded and the cost of cryptographic computing is high because MTC devices may be simultaneously authenticated by the network. As a result, this protocol may not be suitable for mobile devices as discussed by Lai, Li, Lu and Shen (2013). In the same year, Chen, Wang, Chi and Tseng (2012) proposed a group-based authentication and key agreement (G-AKA) protocol for mobile stations (MSs) roaming from the same home network to a serving network. However, the protocol has some vulnerabilities such as man-in-the-middle attack as discussed by Lai, Lu, Zheng, Li and Shen (2016). In 2013, a secure and efficient group authentication and key agreement protocol (SE-AKA) is proposed by Lai et al. (2013). This protocol was supposed to be more secure than the evolved packet system authentication and key agreement (EPS-AKA) protocol proposed in the LTE project. In the protocol, the first mobile equipment (ME) uses its secret key to authenticate itself with its Home. Each remaining ME uses a group key and a synchronization value (SV) to authenticate itself with the service MME. However, this protocol has some weaknesses because a group member can be disguised by other members in the group. In 2016, the group-based lightweight authentication scheme for resource-constrained machine to machine communication (GLARM) is proposed by Lai et al. (2016). The protocol can reduce the MME overhead because the group leader collects all authentication messages from the group's members and communicates with the MME. However, as the protocol needs a group leader to send and response messages with the MME, if the group leader has some problems then the authentication process fails. Furthermore, the scope of this work is limited that all members of the group need to be

in the same service network. In real work, there may be some situation that some members of the group are in different service networks.

As mentioned above, we propose a secure group authentication protocol (SE-GA) which makes use of users' long-term public and private keys to create secret keys with network nodes such as Home and MME. The shared keys are computed by using the Diffie-Hellman key exchange protocol based on Elliptic Curve Cryptography (ECC). By this way, the authentication process is flexible and scalable, and it makes group authentication easy even though group members are on different networks. The SE-GA protocol is just one of many ways to protect the information system. The protocol is used to maintain integrity, availability and confidentiality of information system resources. In this research, we used authentication to access information, which is only part of the protection of information systems. The data integrity is obtained during the authentication process. After the authentication processes, the information exchange is confidential. In addition, the unauthorized users cannot access the information or disguise as other group members.

In the process of protocol, only the first member in an MME's area has to authenticate himself/herself with the Home, while the remaining members in the area can authenticate directly with the MME. Thus, SE-GA protocol can reduce network traffic. In addition, we introduce a proof for group authentication by using the well-known BAN authentication logic reported by Burrows, Abadi and Needham (1990). We have also analyzed the security of SE-GA and compared the features of the protocol with other works. From the analysis, we found the SE-GA outperforms the previous ones.

1.2 The Research Objectives

The research objectives are:

1) To develop the protocol names SE-GA.
2) The members of the group can authenticate with the serving network independently.
3) The protocol allows the group in which members can come from different home networks and they can work on different networks at the same time.

4) Each member cannot impersonate another member within the group.

5) The protocol must be able to protect the data sent between the parties.

6) The identity verification should be secured to ensure accuracy and to minimize interaction time.

CHAPTER 2

LITERATURE REVIEW

In this chapter, we begin with a brief review of some important concepts used in this study. They are Long Term Evolution (LTE) network, BAN Authentication Logic, Elliptic Curve Cryptography. Then some related group authentication protocol such as SE-AKA protocol and GLARM protocol are reported and their security analysis is discussed.

2.1 The Long Term Evolution (LTE) Network

The LTE network architecture can be classified into three domains, including radio access network (RAN) domain, core network (CN) domain, and home network (HN) domain, respectively. As demonstrated in Figure 2.1, the network includes entities as shown in Table 2.1. The network is described according to 3GPP (Third Generation Partnership Project) standard (Nohrborg, 2018) as follows.

1) Radio Access Network (RAN) domain includes mobile equipment (MEs), base stations (BSs) (i.e. eNodeB for outdoor, HeNodeB for indoor) where MEs are mobile equipment of 3GPP standard mobile devices and BSs forward messages from MEs to the serving network domain.

2) Core Network (CN) domain includes mobile management entities (MMEs) or serving gateways (S-GWs). An MME prepares services for the MEs's requests and S-GW forwards messages to another machines.

3) Home Network (HN) domain includes the Home facilitator server (HFS) which provides services for authentication process with MEs.

Table 2.1 The Notations of Entities in the Network Architecture

Notations	Definition
eNB	Type of base station (BS) called evolved Node B (eNodeB)
HeNB	Type of base station (BS) called Home evolved Node B (HeNodeB)
HFS	Home Facilitator Server
ME	Mobile Equipment (machine)
MME	Mobile Management Entity
S-GW	Serving Gateway

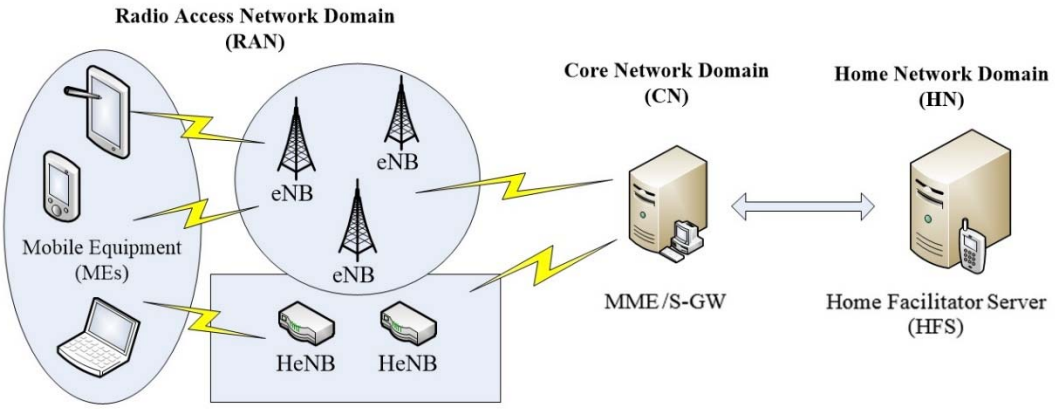

Figure 2.1 LTE Network Architecture

2.2 BAN Authentication Logic

Burrows, Abadi and Needham (1990) introduce the logic of authentication to prove the authentication protocol. In this research, we use this method to prove the authenticity of the authentication protocol. The syntax and semantics of logic, rules and the transformation are explained in 2.2.1 and 2.2.2. The examples of protocol analysis is shown in 2.2.3

2.2.1 Basic Notations

The explanation of the basic notations, borrowed from Burrows et al. (1990) page 20-21.

we distinguish several sorts of objects: principals, encryption keys, and formulas. They identify messages with statements in the logic. Typically, the symbols *A*, *B* and *S* denote specific principals; K_{ab}, K_{as} and K_{bs} denote specific shared keys; K_a, K_b and K_s denote specific public keys, and K_a^{-1}, K_b^{-1} and K_s^{-1} denote the corresponding secret keys; and N_a, N_b and N_c denote specific statements. The symbols *P*, *Q* and *R* range over principals; *X* and *Y* range over statement; and *K* ranges over encryption keys.

The only propositional connective is conjunction, denoted by a comma. They treat conjunctions as set and take for granted properties such as associativity and commutativity. In this research, the conjunctions use the following constructs:

1) *P* believes *X* : *P believes X*, or *P* would be entitled to believe *X*. In particular, the principal *P* may act as though *X* is true. This construct is central to the logic.

2) *P* sees *X*: *P sees X*. Someone has sent a message containing *X* to *P*, who can read and repeat *X* (possibly after doing some decryption).

3) *P* said *X*: *P once said X*. The principal *P* at some time sent a message including the statement *X*. It is not known whether the message was sent long ago or during the current run of the protocol, but it is known that *P* believed *X* then.

4) *P* controls *X*: *P* has *jurisdiction* over *X*. The principal *P* is an authority on *X* and should be trusted on this matter.

5) fresh (*X*): The formula *X* is fresh; that is, *X* has not been sent in a message at any time before the current run of the protocol. This is usually true for *nonces*. That is, expressions invented for the purpose of being fresh. Nonces commonly include a timestamp or a number that is used only once.

6) $P \overset{K}{\leftrightarrow} Q$: *P* and *Q* may use a *shared key K* to communicate. The key *K* is good, in that it will never be discovered by any principal except *P* or *Q*, or a principal trusted by either *P* or *Q*.

7) $\{X\}_K$: This represents the formula X encrypted under the key K. Formally, $\{X\}_K$ is a convenient abbreviation for an expression of the form $\{X\}_K$ from P. The realistic assumption is made that each principle can recognize and ignore the message itself; the originator of each message is discussed for this purpose.

8) $<X>_Y$: This represents X combined with the formula Y; it is intended that Y be a secret and that its presence prove the identity of whoever utters $<X>_Y$. In implementations, X is simply concatenated with the password Y. This notation highlights that Y plays a special role, as proof of origin for X, in much the same way as an encryption key.

2.2.2 Logical Postulates

The explanation of the logical postulates, borrowed from Burrows et al. (1990) page 21-22.

1) The *message-meaning rules* involve interpreting messages. Two-thirds involve the interpretation of the encrypted message, and the third involves the interpretation of the secret message. They all describe how to acquire the beliefs about the origin of the message.

$$\frac{P \text{ believes } Q \overset{K}{\leftrightarrow} P,\ P \text{ sees } \{X\}_K}{P \text{ believes } Q \text{ said } X}$$

It means that, if P believes that the key K is shared with Q and sees X encrypted under K, then P believes that Q once said X.

2) The *nonce-verification rule* indicates that the message is the latest message, and hence the sender still believes it.

$$\frac{P \text{ believes fresh } (X),\ P \text{ believes } Q \text{ said } X}{P \text{ believes } Q \text{ believes } X}$$

It means that, if P believes that X is spoken recently, and Q says X (both in the past or in the present), then P believes that Q believes X.

3) The *jurisdiction rule* states that if *P* believes that *Q* has jurisdiction over *X* then *P* trusts *Q* about the truth of *X*.

$$\frac{P \text{ believes } Q \text{ controls } X,\ P \text{ believes } Q \text{ believes } X}{P \text{ believes } X}$$

2.2.3 The Kerberos Protocol Analyzed

The explanation of kerberos protocol analyzed, borrowed from Burrows et al. (1990) page 25-28.

The Kerberos protocol establishes a shared key between two principals with help from an authentication server (Miller, Neuman, Schiller and Saltzer, 1988). It is based on the shared-key Needham-Schroeder protocol (Needham and Schroeder, 1978), but makes use of timestamps as nonces, both to remove security problems (Bauer, Berson and Feiertag, 1983), (Denning and Sacco, 1981) and to reduce the total number of messages required. Kerberos was developed as part of Project Athena at MIT and is also used elsewhere.

We give the protocol below, with *A* and *B* as the two principals, K_{as} and K_{bs} as their private keys, and *S* as the authentication server. *S* and *A* generate the time stamps T_s and T_a respectively, and *S* generates the lifetime *L*. The fourth message is used only if mutual authentication is required.

Message 1. $A \rightarrow S: A, B.$

Message 2. $S \rightarrow A: \{T_s, L, K_{ab}, B, \{T_s, L, K_{ab}, A\}_{K_{bs}}\}_{K_{as}}.$

Message 3. $A \rightarrow B: \{T_s, L, K_{ab}, A\}_{K_{bs}}, \{A, T_a\}_{K_{ab}}.$

Message 4. $B \rightarrow A: \{T_a + 1\}_{K_{ab}}.$

The Figure 2.2 shows the message sequence. In the beginning, *A* sends a cleartext message to *S* to tell *S* that he wants to communicate with *B*. *S* responds with an encrypted message containing a timestamp, a lifetime, a session key for *A* and *B*, and a ticket that only *B* can read. This ticket also contains the timestamp, the lifetime, and the key. *A* forwards the ticket to *B* together with an authenticator (a timestamp

encrypted with the session key). On receiving, *B* decrypts the ticket and checks the timestamp and lifetime. If the ticket has been created recently enough, *B* uses the enclosed key to decrypt the authenticator. Then, if the authenticator's timestamp is recent, he uses the session key to return the timestamp, which *A* checks. Once the principals are satisfied, they can proceed to use the session key.

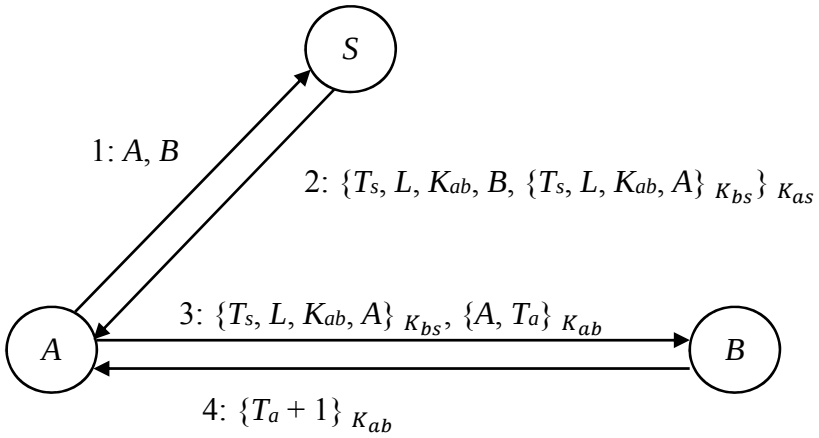

Figure 2.2 The Kerberos Protocol
Source: Burrows et al. (1990): 26

The idealize the protocol as follows:

Message 2. $S \rightarrow A$: $\{T_s, A \overset{K_{ab}}{\longleftrightarrow} B, \{T_S, A \overset{K_{ab}}{\longleftrightarrow} B\}_{K_{bs}}\}_{K_{as}}$.

Message 3. $A \rightarrow B$: $\{T_s, A \overset{K_{ab}}{\longleftrightarrow} B\}_{K_{bs}}$, $\{T_a, A \overset{K_{ab}}{\longleftrightarrow} B\}_{K_{ab}}$ from *A*.

Message 4. $B \rightarrow A$: $\{T_a, A \overset{K_{ab}}{\longleftrightarrow} B\}_{K_{ab}}$ from *B*.

The idealized messages correspond quite closely to the messages described in the published protocol. For simplicity, the lifetime *L* has been combined with the time stamp T_s, which is treated just like a nonce. The first message is omitted, since it does not contribute to the logical properties of the protocol.

A further difference can be seen in the idealized form of Message 2. The concrete protocol mentions the key K_{ab}, which in this sequence has been replaced by

the statement that A and B can use K_{ab} to communicate. This interpretation of the messages is possible only because we know how the information in the messages should be understood. Moreover, the idealized forms of the authenticator and of Message 4 contain the explicit statement that K_{ab} is a good session key, whereas this statement is only implicit in the use of K_{ab} in the concrete protocol. In fact, we could soundly add B believes A believes $A \overset{K_{ab}}{\longleftrightarrow} B$ to Message 4; we do not do so simply because the consequences of this addition seem of little importance for the subsequent use of the session key.

There is some potential for confusion between the second half of the third message and the last message. In the idealized protocol, we avoid this confusion by mentioning the originators explicitly. In the concrete protocol, either the mention of A in the third message or the addition in the fourth suffices to distinguish the two-Kerberos is slightly redundant in this respect.

At this point, the idealized protocol corresponds to the concrete one and that the guidelines for constructing idealized protocols are respected.

To analyze the Kerberos protocol, we give the following assumptions:

A believes $A \overset{K_{as}}{\longleftrightarrow} S$,	B believes $B \overset{K_{bs}}{\longleftrightarrow} S$,
S believes $A \overset{K_{as}}{\longleftrightarrow} S$,	S believes $B \overset{K_{bs}}{\longleftrightarrow} S$,
S believes $A \overset{K_{ab}}{\longleftrightarrow} B$,	B believes (S controls $A \overset{K}{\leftrightarrow} B$),
A believes (S controls $A \overset{K}{\leftrightarrow} B$),	B believes fresh (T_s),
A believes fresh (T_s),	B believes fresh (T_a).

The first group of four is about shared keys between the clients and the server. The fifth indicates that the server initially knows a key for communication between A and B. The next group of two indicates the trust that A and B have in the server to generate a good encryption key. The final three assumptions show that A and B believe that timestamps generated elsewhere are fresh; this indicates that the protocol relies heavily on the use of synchronized clocks.

We analyze the idealized version of Kerberos by applying our rules to the assumptions; the analysis is straightforward. In the interests of brevity, we give many of the formal details necessary for our machine-assisted proof only for Message 2, and they omit similar details later on. The main steps of the proof are as follows:

A receives Message 2. The annotation rules yield that

$$A \text{ sees } \{T_s, (A \overset{K_{ab}}{\longleftrightarrow} B), \{T_S, A \overset{K_{ab}}{\longleftrightarrow} B\}_{K_{bs}}\}_{K_{as}}$$

holds afterward. Since we have the hypothesis

$$A \text{ believes } A \overset{K_{as}}{\longleftrightarrow} S$$

the message-meaning rule for shared keys applies and yields the following:

$$A \text{ believes } S \text{ said } (T_s, (A \overset{K_{ab}}{\longleftrightarrow} B), \{T_S, A \overset{K_{ab}}{\longleftrightarrow} B\}_{K_{bs}})$$

One of our rules to break conjunctions (omitted here) then produces

$$A \text{ believes } S \text{ said } (T_s, (A \overset{K_{ab}}{\longleftrightarrow} B))$$

Moreover, we have the following hypothesis:

$$A \text{ believes fresh } (T_s)$$

The nonce-verification rule applies and yields

$$A \text{ believes } S \text{ believes } (T_s, A \overset{K_{ab}}{\longleftrightarrow} B)$$

Again, we break a conjunction, to obtain the following:

$$A \text{ believes } S \text{ believes } A \overset{K_{ab}}{\longleftrightarrow} B$$

Then, we instantiate K to K_{ab} in the hypothesis

$$A \text{ believes } S \text{ controls } A \overset{K}{\leftrightarrow} B$$

deriving the more concrete

$$A \text{ believes } S \text{ controls } A \overset{K_{ab}}{\longleftrightarrow} B$$

Finally, the jurisdiction rule applies, and yields the following:

$$A \text{ believes } A \overset{K_{ab}}{\longleftrightarrow} B$$

This concludes the analysis of Message 2.

A passes the ticket on to *B*, together with another message containing a timestamp. Initially, *B* can decrypt only the ticket:

$$B \text{ believes } A \overset{K_{ab}}{\longleftrightarrow} B$$

Logically, this result is obtained in the same way as that for Message 2, via the message-meaning, nonce-verification, and jurisdiction postulates. Knowledge of the new key allows *B* to decrypt the rest of Message 3. Through the message-meaning and the nonce-verification postulates, we deduce the following:

$$B \text{ believes } A \text{ believes } A \overset{K_{ab}}{\longleftrightarrow} B$$

The fourth message simply assures *A* that *B* believes in the key and has received *A*'s last message. After new applications of the message-meaning and nonce-verification postulates to the fourth message, the final result is as follows:

A believes $A \overset{K_{ab}}{\longleftrightarrow} B$ $\quad$ B believes $A \overset{K_{ab}}{\longleftrightarrow} B$

A believes B believes $A \overset{K_{ab}}{\longleftrightarrow} B$ $\quad$ B believes A believes $A \overset{K_{ab}}{\longleftrightarrow} B$

If only the first three messages are used, we do not obtain

A believes B believes $A \overset{K_{ab}}{\longleftrightarrow} B$

That is, the three-message protocol does not convince A of B's existence-A observes the same messages whether B is running or not. Although the result resembles that for the Needham-Schroeder protocol (Burrows et al., 1990), a major assumption in the Kerberos protocol is that the principal's clocks are synchronized with the server's clock. The effect of totally synchronized clocks can be obtained by synchronizing clocks to within a few minutes with a secure time server and then detecting replays within this interval. However, actual implementations do not always include this check and so provide only weaker guarantees.

A slight (but potentially expensive) peculiarity is that S double-encrypts the ticket in the second message. Looking back through the formal analysis, we see that this does not affect the properties of the protocol, since A forwards the ticket to B immediately afterward without further encryption. It has recently been proposed that future versions of Kerberos remove this unnecessary double encryption.

2.3 Elliptic Curve Cryptography

For the Elliptic Curve Cryptography (ECC), we describe the situation of Alice and Bob which they have a pair of key (public key and private key). The public key can send to other people for secret communication. When Alice and Bob want to agree upon a key which they will use to secure communication. In a finite field (F_q), an elliptic curve E is defined over F_q and P is a point on E ($P \in E$). For the beginning communication, Alice and Bob will generate a key by Alice chooses a random secret a in F_q ($a \in F_q$) and computes her public key aP on E ($aP \in E$) and sends to Bob. Bob does the same steps, Bob chooses a random secret b and calculates bP on E ($bP \in E$)

and sends to Alice. The secret common key between Alice and Bob is abP on E ($abP \in E$).

In addition operation in ECC, we need to find hard problems of ECC such as Elliptic Curve Discrete Logarithm Problem (ECDLP) and Elliptic Curve based Diffie-Hellman Problem (ECDHP). We described ECDLP and ECDHP as follows:

2.3.1 Elliptic Curve Discrete Logarithm Problem (ECDLP)

Consider an elliptic curve E defined over a finite field F_q. Let $P, Q \in E$ be a point having order n. The elliptic curve discrete logarithm problem is to find integer n, if it exists, such that $Q = nP$. In this problem, it is relatively easy to compute Q from n and P. Even though the attacker knows Q and P, but it takes exponential time to compute n which it is hard. Usually, the private and public keys are n and P respectively.

2.3.2 Elliptic Curve based Diffie-Hellman Problem (ECDHP)

Consider an elliptic curve E defined over a finite field F_q. Given P, aP, $bP \in E$. Let a and b are private keys and aP, bP are public keys of Alice and Bob, respectively. Find $Q \in E$ such that $Q = abP = baP$. In this problem, it is hard to compute Q, even when, aP and bP are known. Q is usually computed and used as a shared key between two parties.

2.4 The SE-AKA Protocol (A Secure and Efficient Group Authentication and Key Agreement Protocol)

The SE-AKA is proposed by Lai et al. (2013). The SE-AKA is used to facilitate the mobile entities (MEs) that have been subscribed in the home network (HN) to roam in a serving network (SN) which is far from home network (HN). The SE-AKA protocol can be divided into two protocols: 1) protocol execution for the first equipment, and 2) protocol execution for the remaining equipment of the same group. Due to the supplier provides a group key (GK_i) to every group for authentication then all MEs of each group can know the group key. The protocol designed for only SN and HN where all MEs

will stay in the same area to access into a SN. The notations are used in the SE-AKA protocol as shown in Table 2.2.

Table 2.2 Notations Used in the SE-AKA Protocol

Notations	Definition
R_{G1-j}	The random number generated by ME_j in group G1
R_{MME}	The random number generated by MME
ID_{G1}	The identity of group G1
ID_{MME}	The identity of MME
$TID_{ME_{G1-j}}$	The temporary identity of ME_j in group G1
MAC_{MME}	The message authentication code computes by MME
$MAC_{ME_{G1-j}}$	The message authentication code computes ME_j in group G1
AMF	Authentication management field
LAI	Location Area Identification
$KGK_{ME_{G1-j}}$	The key generation key between ME_{Gi-j} and MME
$f_{GTK_{G1}}$	A key generation function of group G1
aP, bP	A device' s public key
abP	A shared key between two parties
ME	Mobile Equipment
MME	Mobile Management Entity
HSS	Home Subscriber Server

In the first device authentication process as shown in Figure 2.3, the ME_1 uses a secret key which is known only between it and the Home to generate a message authentication code (MAC) to authenticate itself with the Home via MME. Home

verifies ME_1 by using the same secret key. If the verification is successful, the Home sends the group information management list (GIML), including group name, group ID, MEs′ IDs and synchronization values (SVs) to MME /SN.

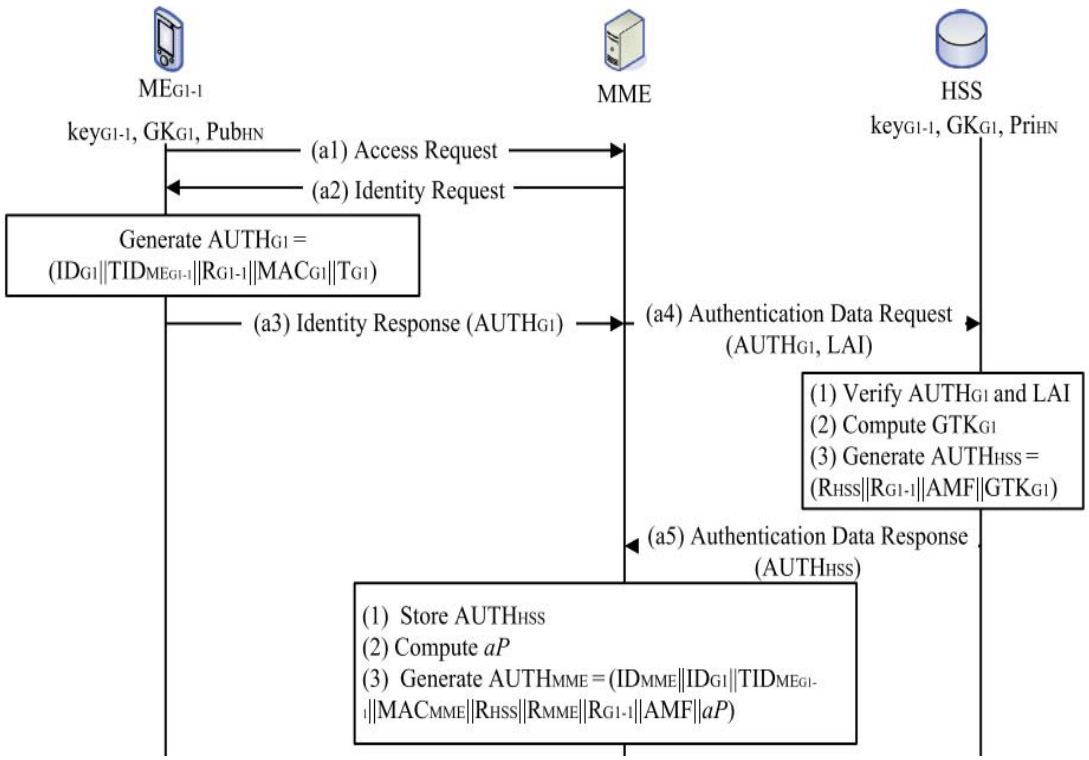

Figure 2.3 The SE-AKA Protocol for First ME

Source: Lai et al. (2013): 3497

In self-confirmation of each remaining ME of the group, the GK and SV are mainly utilized in the authentication process. For GK, every ME knows this value and SV is not a key, so the security of this verification is reduced, and the authentication process can be easily attacked. Then, a group member can impersonate other individuals who have not yet confirmed themselves.

As shown in Figure 2.4, an ME wants to disguise other person by sending the identity information ($AUTH_{ME_{G1-j}} = ID_{G1} || TID_{ME_{G1-j}} || R_{G1-j}$) of target member to the service MME. The MME uses a group temporary key (GTK) which is received from Home (HSS) to perform mutual authentication with the ME without HSS's assistance.

The GTK is generated from Home by using group key (GK). This key makes the MME to believe an ME .

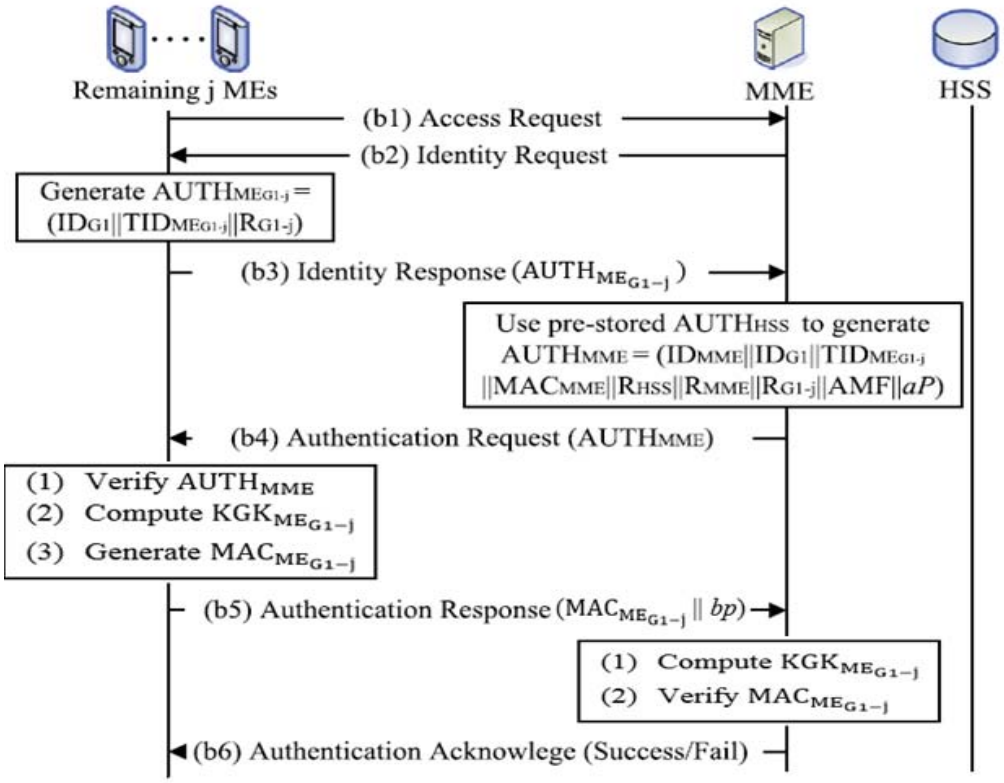

Figure 2.4 The Authentication Procedure of Remaining MEs
Source: Lai et al. (2013): 3498

In the protocol, the MME sends authentication request $\mathrm{AUTH_{MME}} = (\mathrm{ID_{MME}} || \mathrm{ID_{G1}} || \mathrm{TID_{ME_{G1-j}}} || \mathrm{MAC_{MME}} || \mathrm{R_{HSS}} || \mathrm{R_{MME}} || \mathrm{R_{G1-j}} || \mathrm{AMF} || aP)$ where $\mathrm{MAC_{MME}} = f_{\mathrm{GTK_{G1}}}(\mathrm{ID_{MME}} || \mathrm{ID_{G1}} || \mathrm{TID_{ME_{G1-j}}} || \mathrm{R_{HSS}} || \mathrm{R_{MME}} || \mathrm{R_{G1-j}} || \mathrm{AMF} || aP || \mathrm{SV_{G1-j}} + i)$ to the ME. The value i is the sequence of the mutual authentication with $\mathrm{ME_{G1-j}}$. If the fake ME could ever attack the synchronization value ($\mathrm{SV_{G1-j}}$), it selects a random number b and can computes bP, and computes $\mathrm{KGK_{ME_{G1-j}}} = f_{\mathrm{GTK_{G1}}}(\mathrm{ID_{MME}} || \mathrm{TID_{ME_{G1-j}}} || \mathrm{R_{MME}} || \mathrm{R_{G1-j}} || abP)$ and $\mathrm{MAC_{ME_{G1-j}}} = f_{\mathrm{KGK_{ME_{G1-j}}}}(\mathrm{ID_{MME}} || \mathrm{ID_{G1}} || \mathrm{TID_{ME_{G1-j}}} || \mathrm{R_{MME}} || \mathrm{LAI} || bp || abP || \mathrm{SV_{G1-j}} + i)$. It then sends $(\mathrm{MAC_{ME_{G1-j}}} || bp)$ to the MME.

Upon receiving the response, MME verifies $MAC_{ME_{G1-j}}$ by using the received information to compute $MAC_{ME_{G1-j}}$ by itself. It then compares the computed $MAC_{ME_{G1-j}}$ with the received $MAC_{ME_{G1-j}}$. If they are the same then MME believes that ME.

2.5 The GLARM Protocol (Group-based Lightweight Authentication Scheme for Resource-Constrained Machine to Machine Communications Protocol)

The GLARM is proposed by Lai et al. (2016). The concept of this research is that it can use MAC of the group instead of the MAC of each device to authenticate itself of each device (Katz and Lindell, 2008). The protocol designed for resource-constrained M2M under 3GPP network architecture. In addition, the protocol enhances group identity authentication in the case of 3GPP and non-3GPP, as well as reduces the cost of identity overhead. In this protocol, each MTC device uses its own key to encrypt its own message which is used to authenticate. As a result, other MTC devices will not be able to read the message. This is different from SE-AKA protocol which uses group key for authentication in order to use the services because the group key is known in the group. If a person in the group can sniff message then he/she can impersonate another one.

The GLARM is composed of two protocols, (GLARM-1 and GLARM-2). Each protocol has two phases: initialization phase, group authentication and key agreement phase. In the initialization phase, the Machine-Type Communication (MTC) device has a private identity which is installed by supplier in order to register in the 3GPP network. The pre-shared key between home subscriber server (HSS) and MTC is different for each MTC. MTC gets a pre-shared key when MTC registers to HSS at the first time. After that, each MTC calculates the Temporary Identity (TID) by using pre-shared key and MTC's private identity. MTC sends TID to store at the HSS.

In the beginning, MTC devices which are in the same area are required to use the same application to create a group. The network supplier then provides a group key (GK_i) and group identity (ID_{Gi}) to the group using for authentication. Therefore, all

MTC devices in the group can know the GK_i and ID_{Gi}. In GLARM protocol, every MTC in the group must have the same Home and must work on the same network as shown in the Figure 2.5.

For this scheme, the protocol notations used in the GLARM are shown in Table 2.3.

Table 2.3 Notations Used in the GLARM Protocol

Notations	Definition
r_x	The random number generated by machine *x*
ID_x	The identity of machine *x*
TID_x	The temporary identity of machine *x*
$K_{G_{i-j}}$	The shared secret key between the j^{th} MTCD and HSS in the i^{th} group
GK_i	The group key of the i^{th} group
GTK_{Gi}	The group temporary key of the i^{th} group
IK	Integrity key
CK	Cipher key
K_{ASME}	Key for Access Security Management Entity
MSK	Master session key
AK	Authentication key
MAC_x	Message authentication code computed by machine *x*
MAC_{G_i}	Message authentication code compute by $MTCD_{leader}$ in the i^{th} group
$XRES_x$	Expected response computed by machine *x*
$XRES_{G_i}$	Expected response for Gi computed by HSS
RES_x	Authentication response computed by machine *x*
RES_{G_i}	Authentication response computed by $MTCD_{leader}$ in the i^{th} group
$AUTH_x$	The authentication token generated by machine *x*
LAI	Location area identification
AMF	Authentication management field
f^1, f^2, f^3, f^4, f^5	Authentication and key generation function

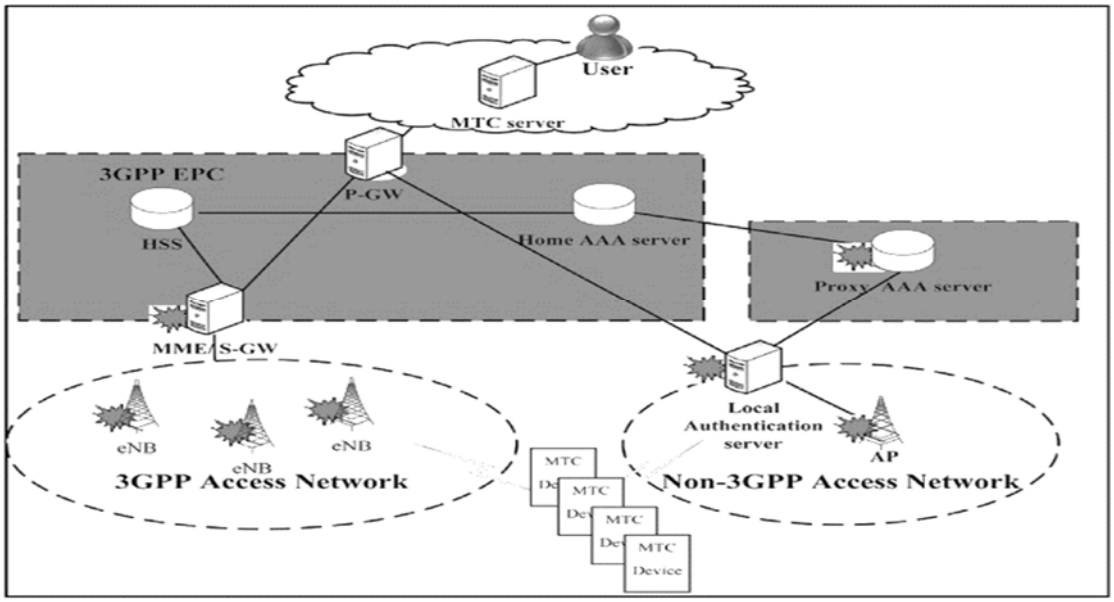

Figure 2.5 The Network Architecture

Source: Lai et al. (2016): 68

2.5.1 GLARM-1 Protocol

The GLARM-1 protocol is 3GPP access case. This protocol, the group leader ($\mathrm{MTCD_{leader}}$) represents a center of the group member to receive and send a message of group to the MME. The steps of protocol are shown in Figure 2.6. In the authentication phase, each MTCD calculates its MAC ($\mathrm{MAC_{MTCD_{G1-j}}} = f1_{\mathrm{K_{G1-j}}}$ ($\mathrm{ID_{G1}} \| \mathrm{ID_{G1-j}} \| \mathrm{r_{G1-j}}$) and generates its authentication message ($\mathrm{M_{MTCD_{G1-j}}} = \mathrm{ID_{G1}} \| \mathrm{ID_{G1-j}} \| \mathrm{r_{G1-j}}$). Then all MTCDs send $\mathrm{MAC_{MTCD_{G1-j}}}$ and $\mathrm{M_{MTCD_{G1-j}}}$ to $\mathrm{MTCD_{leader}}$.

On receiving the messages from all MTCDs, $\mathrm{MTCD_{leader}}$ calculates $\mathrm{MAC_{G1}}$ ($\mathrm{MAC_{G1}} = \mathrm{MAC_{MTCD_{G1-1}}} \oplus \mathrm{MAC_{MTCD_{G1-2}}} \oplus \ldots \oplus \mathrm{MAC_{MTCD_{G1-n}}} \oplus f1_{\mathrm{GK_1}}$ (LAI)). After that, $\mathrm{MTCD_{leader}}$ generates $\mathrm{AUTH_{G1}}$ ($\mathrm{AUTH_{G1}} = \mathrm{M_{MTCD_{G1-1}}} \| \ldots \| \mathrm{M_{MTCD_{G1-n}}} \| \mathrm{MAC_{G1}}$) and sends $\mathrm{AUTH_{G1}}$ to MME.

Upon receiving the message, MME forwards $\mathrm{AUTH_{G1}}$ together with LAI to HSS. When HSS receives $\mathrm{AUTH_{G1}}$, it computes $f1_{\mathrm{GK_1}}$(LAI) by using $\mathrm{GK_1}$ and verifies the $\mathrm{MAC_{G1}}$ in a message by using $\mathrm{K_{G1-j}}$. If verification passes, HSS generates $\mathrm{GTK_{G1}}$ ($\mathrm{GTK_{G1}} = f3_{\mathrm{GK_1}}((\mathrm{r_{HSS}}))$ and calculates $\mathrm{IK_j}$ ($\mathrm{IK_j} = f4_{\mathrm{K_{G1-j}}}(\mathrm{r_{HSS}})$), $\mathrm{CK_j}$ ($\mathrm{CK_j} = f5_{\mathrm{K_{G1-j}}}$

(r_{HSS})) for each MTC device. Then, HSS calculates $K_{ASME}^{MTCD_{G1-j}}$($K_{ASME}^{MTCD_{G1-j}}$ = KDF(CK_j || IK_j || ID_{MME} || ID_{G1-j})) by using a Key Derivation Function (KDF) (Nohrborg et al., 2018). After that, HSS creates key list (KL) of group including Group, Group ID, MTCD ID and K_{ASME}. HSS generates $AUTH_{HSS}$ ($AUTH_{HSS}$ = (r_{HSS} || AMF || MAC_{HSS}, where MAC_{HSS} = $f1_{GK_1}$(ID_{HSS} || r_{HSS} ||AMF)) and calculates $XRES_{G1}$ ($XRES_{G1}$ = $XRES_{MTCD_{G1-1}}$⊕….⊕ $XRES_{MTCD_{G1-n}}$, where $XRES_{MTCD_{G1-j}}$ =$f2_{K_{G1-j}}$ (ID_{G1} || ID_{G1-j} || r_{HSS}) and ⊕ represents XOR). Finally, HSS generates a group authentication vector (GAV = r_{HSS} || $XRES_{G1}$ || GTK_{G1} || $AUTH_{HSS}$) and sends KL, GAV to MME.

After the MME receives the message ($AUTH_{HSS}$) from HSS, MME generates $AUTH_{MME}$ ($AUTH_{MME}$ = (ID_{MME} || MAC_{MME} || MAC_{HSS} || r_{HSS} || r_{MME} || AMF), MAC_{MME} = $f1_{GTK_{G1}}$(ID_{MME} || MAC_{HSS} || r_{MME} || r_{HSS})) to perform mutual authentication with MTCDs and sends $AUTH_{MME}$ to the $MTCD_{leader}$.

After $MTCD_{leader}$ receives a message ($AUTH_{MME}$), $MTCD_{leader}$ then forwards $AUTH_{MME}$ to all MTCDs in the group. Each MTCD verifies MAC_{MME} in $AUTH_{MME}$ by computing GTK_{G1}, MAC'_{HSS}, MAC'_{MME} (GTK_{G1} = $f3_{GK_{G1}}$ (r_{HSS}), MAC'_{HSS} = $f1_{GK_1}$(ID_{HSS} || r_{HSS} ||AMF), MAC'_{MME} = $f1_{GTK_{G1}}$(ID_{MME} || MAC_{HSS} || r_{MME} || r_{HSS})) and then verifies MAC_{MME}. If verification passes, each MTCD computes $K_{ASME}^{MTCD_{G1-j}}$ ($K_{ASME}^{MTCD_{G1-j}}$ = KDF(CK_j || IK_j || ID_{MME} || ID_{G1-j})) and calculates $RES_{MTCD_{G1-j}}$(RES = $f2_{K_{G1-j}}$ (ID_{G1} || ID_{G1-j} || r_{HSS})). Finally, each MTCD sends $RES_{MTCD_{G1-j}}$ to $MTCD_{leader}$. $MTCD_{leader}$ calculates RES_{G1} (RES_{G1} = $RES_{MTCD_{G1-1}}$ ⊕ … ⊕ $RES_{MTCD_{G1-n}}$) and sends it to MME.

On receiving the message from $MTCD_{leader}$, MME compares $XRES_{G1}$ ($XRES_{G1}$ is received from HSS which MME believes HSS) with RES_{G1}. If verification passes, the MME believes group G1 and each MTCD. The concept of the protocol is RES_{G1} = $RES_{MTCD_{G1-j}}$ ⊕ … ⊕ $RES_{MTCD_{G1-n}}$ and $XRES_{G1}$ = $XRES_{MTCD_{G1-j}}$ ⊕….⊕ $XRES_{MTCD_{G1-n}}$ which means $RES_{MTCD_{G1-1}}$ and $XRES_{MTCD_{G1-1}}$ should be also equal (Katz and Lindell, 2008). As a result, MME believes each MTCD. On the other hand, MTCD believes MME in case of MTCD finds that MAC_{MME} equals MAC'_{MME}. After

the MME believes group G1, MME enables the KL to secure communication and sends acknowledge to all MTCDs.

After the successful authentication, both $\mathrm{MTCD}_{\mathrm{G1}-j}$ and the MME share a $\mathrm{K}_{\mathrm{ASME}}^{\mathrm{MTCD}_{\mathrm{G1}-j}}$ as essential material for subsequent key derivations (Lai et al., 2016).

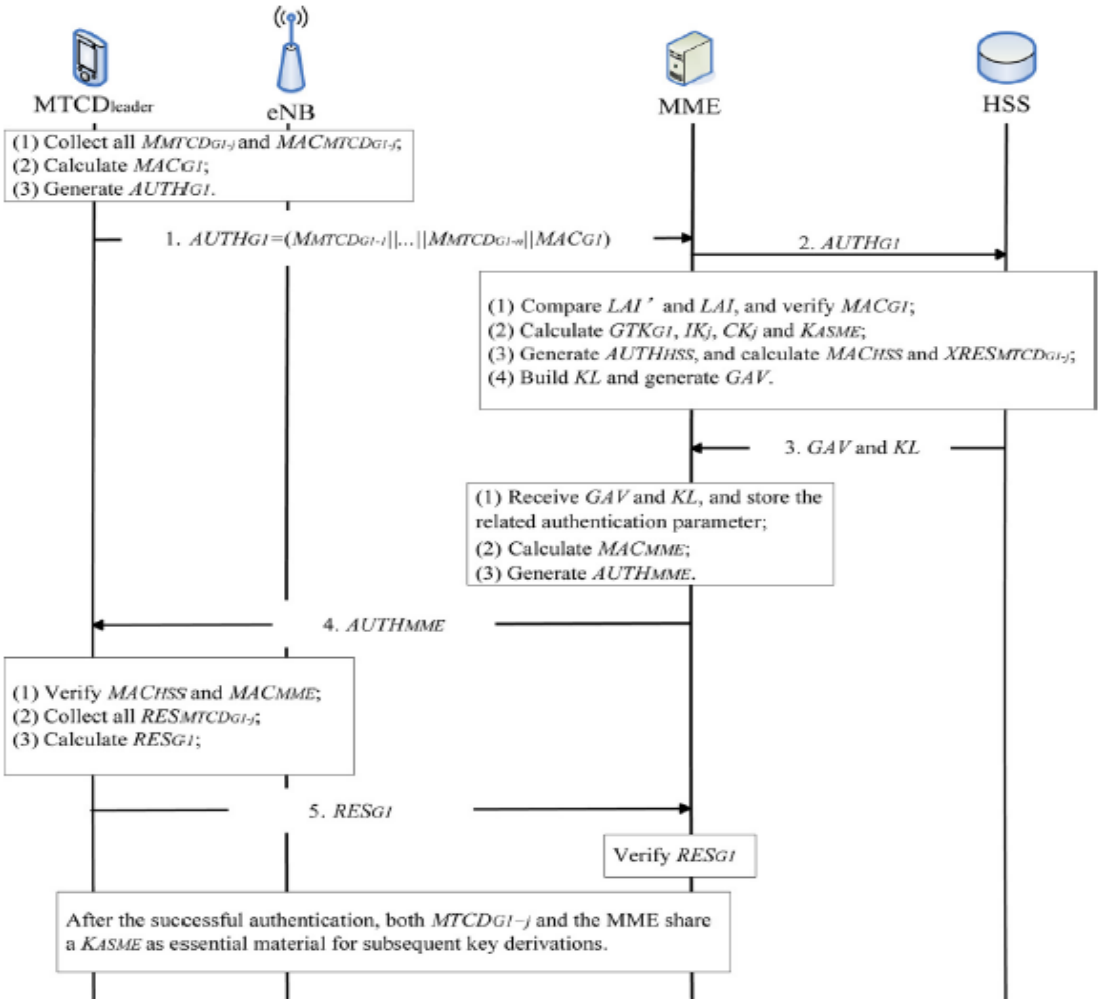

Figure 2.6 The GLARM-1 Protocol

Source: Lai et al. (2016):70

2.5.2 GLARM-2 protocol

The GLARM-2 protocol is non-3GPP access case which supports non-3GPP MTC devices to access the use of services. The steps of protocol are shown in Figure 2.7.

The authentication process begins with the following steps: each MTCD calculates two Temporary Identity (TID) because it is non-3GPP devices ($TID^1_{G1\text{-}j} = E_{K_{G1-j}}(ID_{G1\text{-}j})$, $TID^2_{G1\text{-}j} = E_{K_{G1-j}}(ID_{G1\text{-}j} \parallel r_{G1\text{-}j})$), compute own $MAC_{MTCD_{G1-j}}$ ($MAC_{MTCD_{G1-j}} = f1_{K_{G1-j}}(ID_{G1} \parallel ID_{G1\text{-}j} \parallel r_{G1\text{-}j})$), generate authentication message $M_{MTCD_{G1-j}}$ ($M_{MTCD_{G1-j}} = ID_{G1} \parallel TID_{G1\text{-}j}$) and sends $M_{MTCD_{G1-j}}$, $MAC_{MTCD_{G1-j}}$ to $MTCD_{leader}$.

Upon receiving all $M_{MTCD_{G1-j}}$ and $MAC_{MTCD_{G1-j}}$ from the group members, the $MTCD_{leader}$ calculates MAC_{G1} ($MAC_{G1} = MAC_{MTCD_{G1-1}} \oplus MAC_{MTCD_{G1-2}} \oplus \ldots \oplus MAC_{MTCD_{G1-n}}$). After that, $MTCD_{leader}$ generates $AUTH_{G1}$ ($AUTH_{G1} = M_{MTCD_{G1-1}} \parallel \ldots \parallel M_{MTCD_{G1-n}} \parallel MAC_{G1}$) and sends $AUTH_{G1}$ to the AP (This point forward is different from the GLARM-1 Protocol).

The AP forwards $AUTH_{G1}$ to the HSS through the Local Authentication Server (LAS) and PAAA/HAAA. When HSS is received $AUTH_{G1}$, HSS finds $ID_{G1\text{-}j}$ in its database and extracts the correct encryption key. If $K = TID^1_{G1\text{-}j}$ and $ID_{G1\text{-}j}$ is a prefix of $D_{K_{G1-j}}(TID^2_{G1\text{-}j})$, the HSS retrieves the suffix of $D_{K_{G1-j}}(TID^2_{G1\text{-}j})$ as $r_{G1\text{-}j}$. Next step, HSS computes $MAC'_{MTCD_{G1-j}}$ ($MAC'_{MTCD_{G1-j}} = f1_{K_{G1-j}}(ID_{G1} \parallel D_{K_{G1-j}}(TID^2_{G1\text{-}j}))$ and generates MAC'_{G1}. After that, HSS verifies MAC_{G1} with MAC'_{G1}. If verification passes, HSS accepts the validity of G1.

As this point, HSS generates a group temporary key GTK_{G1} ($GTK_{G1} = f3_{GK_{G1}}(r_{HSS})$), calculates IK_j ($IK_j = f4_{K_{G1-j}}(r_{HSS})$) and CK_j ($CK_j = f5_{K_{G1-j}}(r_{HSS})$) of each MTCD. In addition, HSS calculates a Master Session key ($MSK_{MTCD_{G1-j}} = hash(CK_j \parallel IK_j \parallel ID_{LAS} \parallel ID_{G1\text{-}j})$ for each MTCD and local authentication server (LAS) by using a hash function. Then, the HSS creates key list (KL) for all MTCDs. The KL is compounded Group, Group ID, MTCD ID and MSK. After that, HSS generates $AUTH_{HSS}$ ($AUTH_{HSS} = r_{HSS} \parallel ID_{HSS} \parallel MAC_{HSS}$, $MAC_{HSS} = f1_{GK_1}(ID_{HSS} \parallel r_{HSS})$) and calculates $XRES_{G1}$ ($XRES_{G1} = XRES_{MTCD_{G1-1}} \oplus \ldots \oplus XRES_{MTCD_{G1-n}}$, $XRES_{MTCD_{G1-j}} = f2_{K_{G1-j}}(ID_{G1} \parallel ID_{G1\text{-}j} \parallel r_{HSS} \parallel r_{G1\text{-}j})$). Finally, HSS generates a group authentication vector ($GAV = r_{HSS} \parallel XRES_{G1} \parallel GTK_{G1} \parallel AUTH_{HSS}$) and sends $AUTH_{HSS}$, KL, GAV to the LAS.

On receiving the message from HSS, LAS performs mutual authentication with $MTCD_{G1-j}$ by generating $AUTH_{LAS}$ ($AUTH_{LAS}$ = (ID_{LAS} || MAC_{LAS} || MAC_{HSS} || r_{HSS} || r_{LAS}, $MAC_{LAS} = f1_{GTK_{G1}}(ID_{LAS} || MAC_{HSS} || r_{LAS} || r_{HSS})$) and sends to $MTCD_{leader}$. $MTCD_{leader}$ forwards $AUTH_{LAS}$ to all MTCDs.

After each MTCD is received $AUTH_{LAS}$, it verifies MAC_{LAS} in $AUTH_{LAS}$ by computing $GTK_{G1} = f3_{GK_{G1}}(r_{HSS})$, $MAC'_{HSS} = f1_{GK_1}(ID_{HSS} || r_{HSS})$, $MAC'_{LAS} = f1_{GTK_{G1}}(ID_{LAS} || MAC_{HSS} || r_{LAS} || r_{HSS})$. Each MTCD verifies MAC'_{HSS} with MAC_{HSS} and MAC'_{LAS} with MAC_{LAS}. If verification passes, each MTCD computes $MSK_{MTCD_{G1-j}}$ and calculates $RES_{MTCD_{G1-j}} = f2_{K_{G1-j}}(ID_{G1} || ID_{G1\text{-}j} || r_{HSS} || r_{G1\text{-}j})$ and sends $RES_{MTCD_{G1-j}}$ to $MTCD_{leader}$. Then, $MTCD_{leader}$ calculates RES_{G1} ($RES_{G1} = RES_{MTCD_{G1-j}} \oplus \ldots \oplus RES_{MTCD_{G1-n}}$) and sends it to LAS.

On receiving the message from $MTCD_{leader}$, LAS compares $XRES_{G1}$ ($XRES_{G1}$ is received from HSS which LAS believes HSS) with RES_{G1}. If verification passes, LAS believes group G1 and each MTCD. On the other hand, MTCD believes LAS in case of MTCD finds that MAC_{LAS} equals MAC'_{LAS}. After the LAS believes group G1, LAS enables the KL for secure communication and sends acknowledge to all MTCDs. Now, the full authentication and key agreement are completed.

After the successful authentication, both $MTCD_{G1-j}$ and the LAS share an $MSK_{MTCD_{G1-j}}$ as essential material for subsequent key derivations. In addition, the LAS and MTC device derive an Authorization Key (AK) from MSK, and AK is used to derive lower level keys to secure the communications between MTC device and AP (Lai et al., 2016).

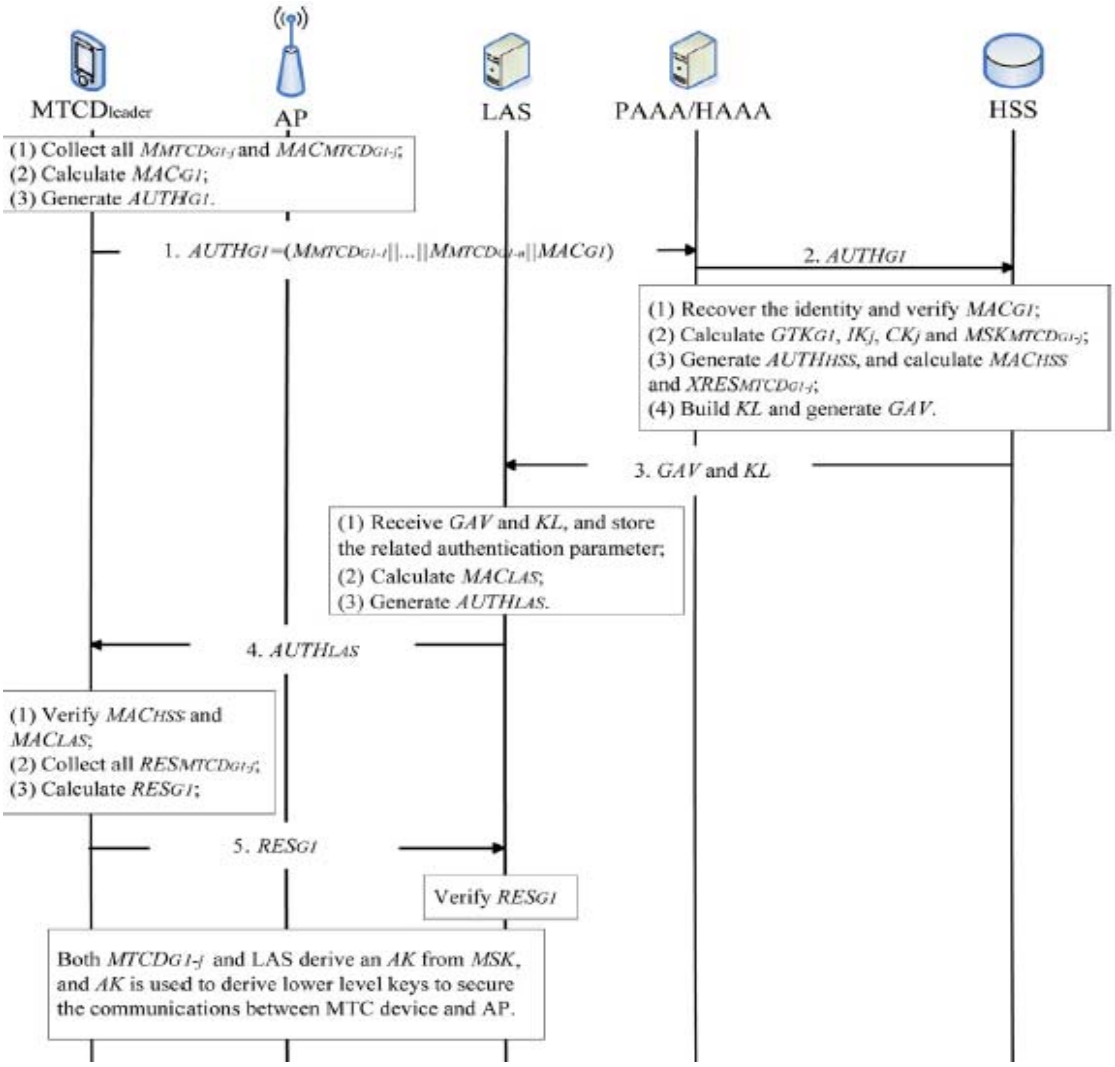

Figure 2.7 The GLARM-2 Protocol

Source: Lai et al. (2016): 72

CHAPTER 3

THE PROPOSED GROUP AUTHENTICATION PROTOCOL

In this chapter, SE-GA protocol for ME/MEs in a group is used to access into various serving network domains. The objectives of SE-GA protocol are: 1) Members of the group can authenticate with the serving network independently. 2) The protocol allows the group in which members can come from different home networks and they can work on different networks at the same time as show in Figure 3.1. 3) Each member cannot impersonate another member within the group, and 4) The protocol must be able to protect the data between the parties. In addition, identity verification should be secured to ensure accuracy and to minimize interaction time.

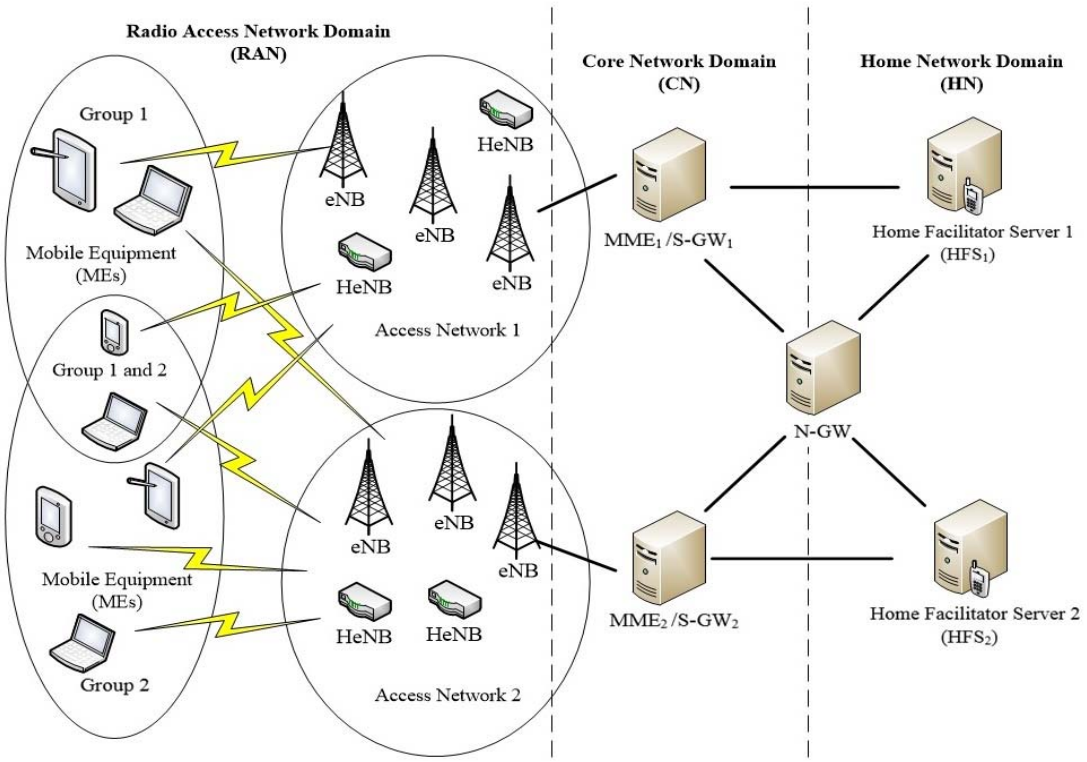

Figure 3.1 Network Architecture based on 3GPP standard in SE-GA Protocol

3.1 Initialization

In the initial stage, each ME creates a pair of long-term private key and public key, and ME sends the public key to its Home. Then the HN and ME can create a shared secret key by using a Diffie-Hellman key exchange. It is noted that a long-term public key of the Home is well-known. When several MEs form a group G_n, they create a session group key.

Each group member then sends the group's information, i.e. Group ID, number of members, Temporary Identity Numbers (TID) and all long-term public keys of the group members to his/her Home. This data is sent with integrity control by utilizing the shared key between the group member and the Home. The data does not need to be secret. However, if we need secrecy, the information can be covered by using the shared key. On receiving the messages, each Home keeps the group's information in GDL as shown in Table 3.1.

Table 3.1 Group Detail List (GDL)

Group Number	**Group ID**	**TID_{ME_i}**	**ID_{HFS_k}**	**Public Key_{ME_i}**
G_1	ID_{G_1}	TID_{ME_1}	ID_{HFS_1}	Pub_{ME_1}
G_1	ID_{G_1}	TID_{ME_2}	ID_{HFS_2}	Pub_{ME_2}
.	.	.	.	.
.	.	.	.	.
G_1	ID_{G_1}	TID_{ME_i}	ID_{HFS_k}	Pub_{ME_i}
G_2	ID_{G_2}	TID_{ME_1}	ID_{HFS_1}	Pub_{ME_1}
G_2	ID_{G_2}	TID_{ME_3}	ID_{HFS_2}	Pub_{ME_3}
.	.	.	.	.
.	.	.	.	.
G_2	ID_{G_2}	TID_{ME_m}	ID_{HFS_l}	Pub_{ME_m}

3.2 The SE-GA Protocol for Each ME_i in a Group G_n

When an ME_i connects to a wireless point, it authenticates itself with that network in order to use network services.

In the authentication process, an ME_i device in a group G_n, connects to the wireless point in any area mobile management entity (MME_j). The ME_i then sends an access request $AUTH_i$ to the MME_j. When the MME_j receives a request, it checks whether the ME_i is a member in the previously requested group by using HFS_k and ID_{G_n} in the $AUTH_i$ to determine if a Group Detail List (GDL) exists in the MME_j's database. If not, ME_i is the first machine in the group that requests the connection with MME_j. MME_j then performs the authentication process for the first ME device (i.e. using case 1) and gets a GDL from ME_i's Home. Otherwise, if there is the GDL of that ME_i, then MME_j performs an authentication process as if the ME_i is a remaining ME device (i.e. using case 2). Table 3.2 shows the notations used in the SE-GA protocol. The machine *x* or *y* can be an MME, HFS or ME. When *x* or *y* is represent by $G_n - i$, it means an ME_i of a group *n*.

Table 3.2 Notations Used in the SE-GA Protocol

Notations	Definition
R_x	The random number generated by machine *x*
TS_x	The Time stamp generated by machine *x*
ID_x	The identity of machine *x*
PID_x	The permanent identity of machine *x*
TID_x	The temporary identity of machine *x*
SK_{x-y}	The shared secret key between machine *x* and *y*
SSK_{x-y}	The shared session key between machine *x* and *y*
MAC_x	The message authentication code computed by machine *x*
LAI_x	Location Area Identification of machine *x*
$f1_{SK_{MME_j-ME_i}}$	MAC generating function using $SK_{MME_j-ME_i}$
$f2_{SK_{MME_j-ME_i}}$	SSK generating function using $SK_{MME_j-ME_i}$
$f3_{SK_{MME_j-HFS_k}}$	MAC generating function using $SK_{MME_j-HFS_k}$
$f4_{SK_{MME_j-HFS_k}}$	MAC generating function using $SK_{MME_j-HFS_k}$
$f5_{SK_{ME_i-HFS_k}}$	MAC generating function using $SK_{ME_i-HFS_k}$
aP, *bP*	A device' s public key
abP	A shared key between two parties
ME_i	The i^{th} Mobile Equipment (machine)
MME_j	The j^{th} Mobile Management Entity of network
N-GW	Networking Gateway
HFS_k	Home Facilitator Server of the k^{th} network

The steps of the SE-GA protocol are as the following.

3.2.1 Case 1: Authentication for the First ME

If ME_i is the first member of a group G_n that want to authenticate with MME_j, then MME_j does not have a GDL of the ME_i's group in MME_j's database. Therefore, MME_j looks for the ME_i's home network (HFS_k) in the authentication request and then forwards the authentication data request, local area identification of MME_j, identity of MME_j and MAC_{MME_j} (i.e. $AUTH_i$, LAI_{MME_j}, ID_{MME_j}, MAC_{MME_j}) to HFS_k of ME_i through N-GW. If the authentication data request passes the network gateway (N-GW),

the N-GW only forwards the authentication request to the destination (HFS_k). This case is composed of step (1) – step (5) as shown in Figure 3.2.

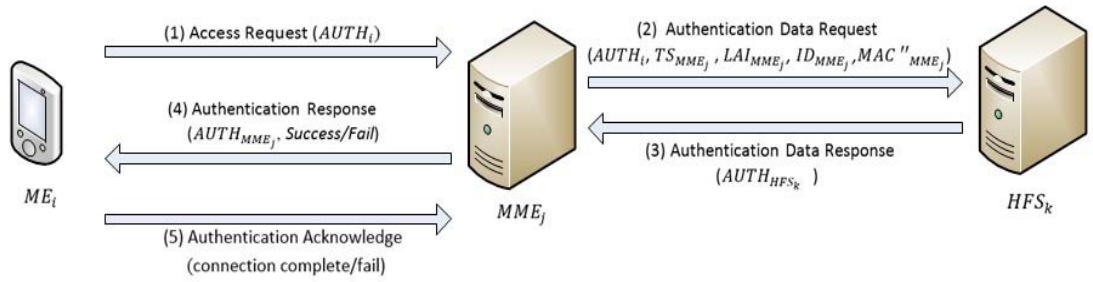

Figure 3.2 The SE-GA Protocol for the First ME

Step (1): ME_i → MME_j : **Access Request** ($AUTH_i$).

The ME_i generates $AUTH_i = (ID_{G_n} \| TID_{ME_i} \| R_{G_n-i} \| TS_{G_n-i} \| HFS_k \| LAI_{ME_i} \| bP \| MAC_q \| MAC_i)$ and sends it to MME_j. $MAC_q = f1_{SK_{MME_j-ME_i}}(ID_{G_n} \| TID_{ME_i} \| R_{G_n-i} \| TS_{G_n-i} \| HFS_k \| LAI_{ME_i} \| bP)$ and it is used by MME_j to verify whether it is the correct ME_i. While $MAC_i = f5_{SK_{ME_i-HFS_k}}(ID_{G_n} \| TID_{ME_i} \| R_{G_n-i} \| TS_{G_n-i} \| HFS_k \| LAI_{ME_i} \| bP)$ and it is used by HFS_k to verify whether it is the correct ME_i. The function $f1_{SK_{MME_j-ME_i}}$ and $f5_{SK_{ME_i-HFS_k}}$ are used for generating message authentication codes MAC_q and MAC_i respectively. $SK_{MME_j-ME_i}$ is a shared secret key between MME_j and ME_i, and is computed from ME_i's private key and MME_j's public key by using the Diffie-Hellman key exchange. It is noted that MME_j's public key is well-known on the internet. In part of $SK_{ME_i-HFS_k}$, it is a shared secret key between ME_i and its home network (HN) which is computed by performing the Diffie-Hellman key exchange in the initialization state. The value bP is a session public key of ME_i. It is created by selecting a random number b and computing bP on Elliptic Curve. TID_{ME_i} is a temporary identity of ME_i in HFS_k and is used for registration in 3GPP/LTE networks. The value is installed in ME_i by the supplier of ME_i.

Step (2): MME_j → HFS_k : **Authentication Data Request**

$(AUTH_i , TS_{MME_j} , LAI_{MME_j}, ID_{MME_j} , MAC''_{MME_j})$.

When the MME_j receives the authentication data request from ME_i, it uses HFS_k and ID_{G_n} in the $AUTH_i$ to find out whether this request is the first request of the group, by searching for ID_{G_n} in the Group Detail List (GDL) of MME_j's database. If it cannot find the information in MME_j's database, then MME_j forwards $AUTH_i$, TS_{MME_j}, ID_{MME_j}, LAI_{MME_j}, MAC''_{MME_j} to the HFS_k . The LAI_{MME_j} reports the location of the wireless point which ME_i connects to, and $MAC''_{MME_j} = f3_{SK_{MME_j - HFS_k}}(AUTH_i \| TS_{MME_j} \| ID_{MME_j} \| LAI_{MME_j})$. The long-term secret key ($SK_{MME_j - HFS_k}$) between MME_j and HFS_k is computed by using the HFS_k's public key and MME_j's private key in the Diffie-Hellman key exchange. It is noted that HFS_k's public key is well-known on the internet.

MME_j also keeps *bP* and MAC_q in order to use them afterward.

Step (3): HFS_k → MME_j : **Authentication Data Response**

$(AUTH_{HFS_k})$.

Upon receiving authentication data request ($AUTH_i$, TS_{MME_j} , LAI_{MME_j}, ID_{MME_j}, MAC''_{MME_j}) from MME_j, the HFS_k verifies MME_j by computing $MAC'''_{MME_j} = f3_{SK_{MME_j - HFS_k}}(AUTH_i \| TS_{MME_j} \| ID_{MME_j} \| LAI_{MME_j})$ and compares it with MAC''_{MME_j}. Here, $SK_{MME_j - HFS_k}$ is computed by using HFS_k's private key and MME_j's public key. If it is the same MAC value then HFS_k believes that the message is sent from MME_j.

Before HFS_k verifies MAC_i which is in $AUTH_i$, the HFS_k compares LAI_{MME_j} with LAI_{ME_i} to check whether they are the same. If they have the same value, HFS_k verifies MAC_i by computing $MAC'_i = f5_{SK_{ME_i - HFS_k}}(ID_{G_n} \| TID_{ME_i} \| R_{G_n - i} \| TS_{G_n - i} \| HFS_k \| LAI_{ME_i} \| bP)$ from data in $AUTH_i$. Then HFS_k compares MAC'_i with the MAC_i. If these values are the same, the HFS_k can believe that the message is sent from ME_i.

The HFS_k then generates $AUTH_{HFS_k} = (R_{G_n-i} \| ID_{HFS_k} \| HFS_k \| GDL \| TS_{HFS_k} \| MAC_{HFS_k})$, where $MAC_{HFS_k} = f4_{SK_{MME_j-HFS_k}}(R_{G_n-i} \| ID_{HFS_k} \| HFS_k \| GDL \| TS_{HFS_k})$ and it sends $AUTH_{HFS_k}$ to the MME_j. GDL is composed of group number, group identity, temporary identity of every ME_i, identity of HFS_k and public keys of all MEs in this group.

Step (4) : $MME_j \rightarrow ME_i$: **Authentication Response**

($AUTH_{MME_j}$, Success/Fail).

After MME_j receives $AUTH_{HFS_k}$ from HFS_k, MME_j computes $MAC'_{HFS_k} = f4_{SK_{MME_j-HFS_k}}(R_{G_n-i}||ID_{HFS_k}||HFS_k||GDL||TS_{HFS_k})$ to verify the message from HFS_k. If the verification passes, MME_j computes $MAC'_q = f1_{SK_{MME_j-ME_i}}(ID_{G_n}||TID_{ME_i}||R_{G_n-i}||TS_{G_n-i}||HFS_k||LAI_{ME_i}||bP)$ and compares it with MAC_q from step (1). The $SK_{MME_j-ME_i}$ is computed by MME_j's private key and ME_i's long-term public key got from GDL. If $MAC'_q = MAC_q$, MME_j installs GDL of G_n into MME_j's database. The GDL facilitates the MME_j to check the remaining ME_i's authentication information. Then, MME_j can trust the message $AUTH_i$ which is sent by ME_i, because MME_j got correct response from ME_i's Home.

MME_j then randomizes a number *a* to compute a session public key *aP* and a secret value *abP* on Elliptic Curve. Note that *bP* is obtained from step (1). MME_j also generates $AUTH_{MME_j} = (ID_{MME_j}||ID_{G_n}||TID_{ME_i}||R_{MME_j}||R_{G_n-i}||TS'_{MME_j}||aP \| MAC_{MME_j})$, where $MAC_{MME_j} = f1_{SK_{MME_j-ME_i}}(ID_{MME_j} \| ID_{G_n} \| TID_{ME_i} \| R_{MME_j} \| R_{G_n-i} \| TS'_{MME_j} \| aP)$. It then sends $AUTH_{MME_j}$ and a response 'success' to ME_i. MME_j can now compute session key between itself and ME_i by $SSK_{MME_j-ME_i} = f2_{SK_{MME_j-ME_i}}(ID_{MME_j} \| TID_{ME_i} \| R_{MME_j} \| R_{G_n-i} \| abP)$.

Step (5) : ME_i → MME_j : **Authentication Acknowledge**

(connection complete/fail).

When the ME_i gets the authentication data response from MME_j, it verifies MME_j by computing $MAC'_{MME_j} = f1_{SK_{MME_j-ME_i}}$ ($ID_{MME_j} \| ID_{G_n} \| TID_{ME_i} \| R_{MME_j} \| R_{G_n-i} \| TS'_{MME_j} \| aP$) and compares MAC_{MME_j} with MAC'_{MME_j}. The $SK_{MME_j-ME_i}$ is computed from ME_i's private key and MME_j's public key by using the Diffie-Hellman key exchange. MME_j's long-term public key is well-known on the internet.

If MAC_{MME_j} and MAC'_{MME_j} are the same then it is the correct MME_j. ME_i then computes abP by using aP from $AUTH_{MME_j}$ and creates a session key between ME_i and MME_j by $SSK_{MME_j-ME_i} = f2_{SK_{MME_j-ME_i}}$($ID_{MME_j} \| TID_{ME_i} \| R_{MME_j} \| R_{G_n-i} \| abP$). Finally, the ME_i has a shared session key $SSK_{MME_j-ME_i}$ with MME_j and sends connection complete to MME_j. Otherwise, ME_i sends a response, 'connection failure' to MME_j.

3.2.2 Case 2: Authentication for the Remaining MEs

If ME_i is a remaining member of the group G_n that has a member authenticated with MME_j, then MME_j has the Group Detail List (GDL) of group G_n in the MME_j's database. The MME_j can use the ME_i's public key in GDL to create a shared secret key ($SK_{MME_j-ME_i}$) between MME_j and ME_i. This case is composed of step (1) – step (3) as shown in Figure 3.3.

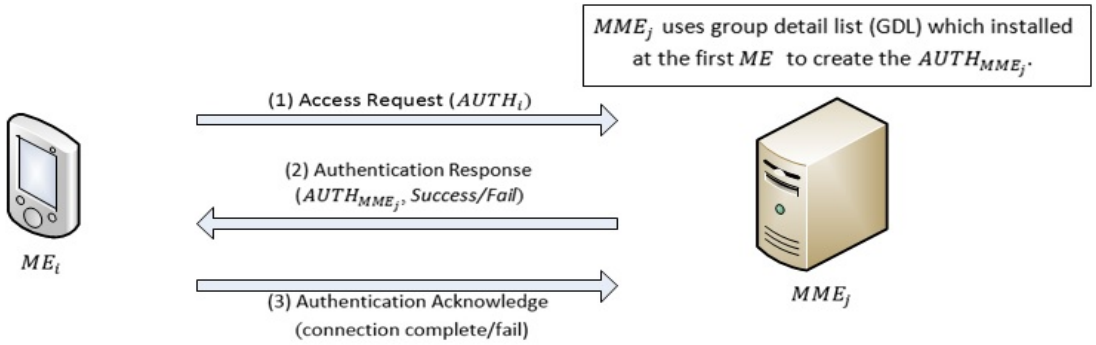

Figure 3.3 The SE-GA Protocol for Remaining ME Devices

Step (1) : ME_i → MME_j : **Access Request** ($AUTH_i$)

The ME_i generates $AUTH_i = (ID_{G_n} \| TID_{ME_i} \| R_{G_n-i} \| TS_{G_n-i} \| HFS_k \| LAI_{ME_i} \| bP \| MAC_q \| MAC_i)$, $MAC_q = f1_{SK_{MME_j-ME_i}}(ID_{G_n} \| TID_{ME_i} \| R_{G_n-i} \| TS_{G_n-i} \| HFS_k \| LAI_{ME_i} \| bP)$ and $MAC_i = f_{SK_{ME_i-HFS_k}}(ID_{G_n} \| TID_{ME_i} \| R_{G_n-i} \| TS_{G_n-i} \| HFS_k \| LAI_{ME_i} \| bP)$ and sends $AUTH_i$ to MME_j.

Step (2) : MME_j → ME_i : **Authentication Response** ($AUTH_{MME_j}$, Success/Fail).

When the MME_j receives an authentication data request from ME_i, it checks the request of ME_i by using HFS_k and ID_{G_n} in the $AUTH_i$ to find out whether this request is the first request of group by searching for ID_{G_n} in the Group Detail List (GDL) of MME_j's database. If it can find ID_{G_n}, then MME_j computes a long-term secret key ($SK_{MME_j-ME_i}$) between MME_j and ME_i by using ME_i's public key in GDL and MME_j's private key.

Before MME_j verifies MAC_q which is in $AUTH_i$, the MME_j compares LAI_{ME_i} with LAI_{MME_j} to check whether they are the same. If they have the same value, the MME_j computes $MAC'_q = f1_{SK_{MME_j-ME_i}}(ID_{G_n} \| TID_{ME_i} \| R_{G_n-i} \| TS_{G_n-i} \| HFS_k \| LAI_{ME_i} \| bP)$. It then compares MAC'_q with MAC_q from step (1). If $MAC'_q = MAC_q$ then MME_j trusts ME_i and messages are sent by ME_i.

MME_j then randomizes a number *a* to compute a session public key *aP* and a secret value *abP* on Elliptic Curve. Further, MME_j generates $AUTH_{MME_j} = ID_{MME_j} || ID_{G_n} || TID_{ME_i} || R_{MME_j} || R_{G_n-i} || TS'_{MME_j} || aP || MAC_{MME_j})$, where $MAC_{MME_j} = f1_{SK_{MME_j-ME_i}}(ID_{MME_j} || ID_{G_n} || TID_{ME_i} || R_{MME_j} || R_{G_n-i} || TS'_{MME_j} || aP)$. It then sends $AUTH_{MME_j}$ and a response, 'success' to ME_i.

At this point, MME_j can compute a session key between itself and ME_i by $SSK_{MME_j-ME_i} = f2_{SK_{MME_j-ME_i}}(ID_{MME_j} || TID_{ME_i} || R_{MME_j} || R_{G_n-i} || abP)$.

Step (3) : ME_i → MME_j : **Authentication Acknowledge** (connection complete/fail)

When the ME_i gets the authentication response from MME_j, it verifies the message by computing $MAC'_{MME_j} = f1_{SK_{MME_j-ME_i}}(ID_{MME_j} || ID_{G_n} || TID_{ME_i} || R_{MME_j} || R_{G_n-i} || TS'_{MME_j} || aP)$ and compares MAC_{MME_j} with MAC'_{MME_j}. The $SK_{MME_j-ME_i}$ is computed from ME_i's private key and MME_j's public key.

If MAC_{MME_j} and MAC'_{MME_j} have the same value then ME_i believes that the message is sent from MME_j. ME_i then computes *abP* by using *aP* from $AUTH_{MME_j}$ and creates session key between ME_i and MME_j by $SSK_{MME_j-ME_i} = f2_{SK_{MME_j-ME_i}}(ID_{MME_j} || TID_{ME_i} || R_{MME_j} || R_{G_n-i} || abP)$. Finally, the ME_i has a shared session key $SSK_{MME_j-ME_i}$ with MME_j and sends connection complete to MME_j. Otherwise, if MAC_{MME_j} and MAC'_{MME_j} are not the same then ME_i sends a response, 'connection failure' to MME_j.

CHAPTER 4

AUTHENTICATION PROOF BY USING BAN LOGIC

In this section, we give an authentication proof of the SE-GA protocol by using the well-known BAN Logic. The notations used in SE-GA protocol are shown in Table 4.1.

Table 4.1 Notations Used in the Proof

Notations	Description
bP	A session public key of $\mathrm{ME_i}$
aP	A session public key of $\mathrm{MME_j}$
$\mathrm{SK_{ME_i-HFS_k}}$	A long-term secret shared between $\mathrm{ME_i}$ and $\mathrm{HFS_k}$
$\mathrm{SK_{ME_i-MME_j}}$	A long-term secret shared between $\mathrm{ME_i}$ and $\mathrm{MME_j}$
$\mathrm{SK_{MME_j-HFS_k}}$	A long-term secret shared between $\mathrm{MME_j}$and $\mathrm{HFS_k}$
$\mathrm{SSK_{MME_j-ME_i}}$	A shared session key between $\mathrm{MME_j}$ and $\mathrm{ME_i}$

We will prove the authentication of the mobile equipment in both cases: the case of the first ME device and the case of the remaining ME devices which connect to an MME.

4.1 Authentication Proof for the First ME

The communicating messages used in the case of the first ME device are as follows:

(a) $ME_i \rightarrow MME_j$: $AUTH_i = (ID_{G_n}, TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i}, bP,$
$MAC_q((ID_{G_n}, TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i}, bP),$
$SK_{ME_i-MME_j}),$
$MAC_i((ID_{G_n}, TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i}, bP),$
$SK_{ME_i-HFS_k})).$

(b) $MME_j \rightarrow HFS_k$: $(AUTH_i, TS_{MME_j}, LAI_{MME_j}, ID_{MME_j},$
$MAC''_{MME_j}((AUTH_i, TS_{MME_j}, LAI_{MME_j}, ID_{MME_j}),$
$SK_{MME_j-HFS_k})).$

(c) $HFS_k \rightarrow MME_j$: $(R_{G_n-i}, ID_{HFS_k}, HFS_k, GDL, TS_{HFS_k}),$
$MAC_{HFS_k}((R_{G_n-i}, ID_{HFS_k}, HFS_k, GDL, TS_{HFS_k}), SK_{MME_j-HFS_k}).$

(d) $MME_j \rightarrow ME_i$: $(ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, TS'_{MME_j}, aP),$
$MAC_{MME_j}((ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j},$
$R_{G_n-i}, TS'_{MME_j}, aP), SK_{ME_i-MME_j}).$

The messages can be transformed into the idealized forms as

(a) $ME_i \rightarrow MME_j$: $AUTH_i = \langle ID_{G_n}, TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i},$
$bP \rangle_{SK_{ME_i-MME_j}},$
$\langle ID_{G_n}, TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i},$
$bP \rangle_{SK_{ME_i-HFS_k}}.$

(b) $MME_j \rightarrow HFS_k$: $\langle AUTH_i, LAI_{MME_j}, TS_{MME_j}, ID_{MME_j} \rangle_{SK_{MME_j-HFS_k}}.$

(c) $HFS_k \rightarrow MME_j$: $\langle R_{G_n-i}, ID_{HFS_k}, HFS_k, GDL, TS_{HFS_k} \rangle_{SK_{MME_j-HFS_k}}.$

(d) $MME_j \rightarrow ME_i$: $\langle ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, TS'_{MME_j}, aP \rangle_{SK_{ME_i-MME_j}}$.

In this form TS_{G_n-i}, TS_{MME_j}, TS'_{MME_j}, TS_{HFS_k} are nonces.

We need to prove that MME_j believes ME_i's long term public key in GDL which it has received from HFS_k and uses the key to compute a long-term secret key ($SK_{ME_i-MME_j}$) between MME_j and ME_i. MME_j uses $SK_{ME_i-MME_j}$ to verify ME_i's message. It then can believe ME_i's session public key, bP. Further, it needs to prove that ME_i can believe MME_j's session public key, aP. Both MME_j and ME_i can use aP and bP to compute a shared session secret, abP. To analyze this protocol, the following assumptions are made.

1) HFS_k believes $MME_j \overset{SK_{MME_j-HFS_k}}{\longleftrightarrow} HFS_k$.
2) HFS_k believes $ME_i \overset{SK_{ME_i-HFS_k}}{\longleftrightarrow} HFS_k$.
3) MME_j believes $HFS_k \overset{SK_{MME_j-HFS_k}}{\longleftrightarrow} MME_j$.
4) ME_i believes $MME_j \overset{SK_{ME_i-MME_j}}{\longleftrightarrow} ME_i$.
5) MME_j believes fresh (TS_{G_n-i}).
6) MME_j believes fresh (TS_{HFS_k}).
7) HFS_k believes fresh (TS_{G_n-i}).
8) HFS_k believes fresh (TS_{MME_j}).
9) ME_i believes fresh (TS'_{MME_j}).
10) HFS_k believes MME_j control ($AUTH_i$, TS_{MME_j}, LAI_{MME_j}, ID_{MME_j}).
11) HFS_k believes ME_i control (ID_{G_n}, TID_{ME_i}, R_{G_n-i}, HFS_k, LAI_{ME_i}, bP).
12) MME_j believes HFS_k controls (R_{G_n-i}, ID_{HFS_k}, HFS_k, GDL).
13) MME_j believes ME_i controls (ID_{G_n}, TID_{ME_i}, R_{G_n-i}, HFS_k, LAI_{ME_i}, bP).
14) ME_i believes MME_j controls (ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, aP).

The steps of the proof are as follows:

(a) HFS_k believes $MME_j \xleftrightarrow{SK_{MME_j - HFS_k}} HFS_k$ **and** HFS_k sees
$\langle AUTH_i , LAI_{MME_j} , TS_{MME_j} , ID_{MME_j} \rangle_{SK_{MME_j - HFS_k}}$,
then HFS_k believes MME_j said
$(AUTH_i , LAI_{MME_j}, TS_{MME_j}, ID_{MME_j})$.

(b) HFS_k believes fresh (TS_{MME_j}) **and** HFS_k believes MME_j said
$(AUTH_i , LAI_{MME_j} , TS_{MME_j} , ID_{MME_j})$,
then HFS_k believes MME_j believes
$(AUTH_i , LAI_{MME_j} , TS_{MME_j} , ID_{MME_j})$.

The conjunction can be broken and the result is HFS_k believes MME_j believes $(AUTH_i , LAI_{MME_j} , ID_{MME_j})$.

(c) HFS_k believes MME_j control $(AUTH_i , LAI_{MME_j} , ID_{MME_j})$
and HFS_k believes MME_j believes $(AUTH_i , LAI_{MME_j} , ID_{MME_j})$,
then HFS_k believes $(AUTH_i , LAI_{MME_j} , ID_{MME_j})$.

In steps *a) - c)*, HFS_k uses a long-term secret key between MME_j and HFS_k (i.e. $SK_{MME_j - HFS_k}$) to verify the message $(AUTH_i , LAI_{MME_j} , TS_{MME_j} , ID_{MME_j})$ received from MME_j. If the verification passes, HFS_k believes that the message is sent from MME_j.

After HFS_k believes the message is sent from MME_j, it verifies the authentication message ($\langle ID_{G_n}, TID_{ME_i}, R_{G_n - i}, TS_{G_n - i}, HFS_k, LAI_{ME_i}, bP \rangle_{SK_{ME_i - HFS_k}}$) which is in $AUTH_i$. If the verification passes, HFS_k believes that the message is from ME_i. The proof is as follows.

(d) HFS_k believes $ME_i \xleftrightarrow{SK_{ME_i-HFS_k}} HFS_k$ **and** HFS_k sees
$\langle ID_{G_n}, TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i}, bP \rangle_{SK_{ME_i-HFS_k}}$,
then HFS_k believes ME_i said
$(ID_{G_n}, TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i}, bP)$.

(e) HFS_k believes fresh (TS_{G_n-i}) **and** HFS_k believes ME_i said
$(ID_{G_n}, TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i}, bP)$,
then HFS_k believes ME_i believes
$(ID_{G_n}, TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i}, bP)$.

The conjunction can be broken and the result is HFS_k believes ME_i believes $(ID_{G_n}, TID_{ME_i}, R_{G_n-i}, HFS_k, LAI_{ME_i}, bP)$.

(f) HFS_k believes ME_i control $(ID_{G_n}, TID_{ME_i}, R_{G_n-i}, HFS_k, LAI_{ME_i}, bP)$
and HFS_k believes ME_i believes
$(ID_{G_n}, TID_{ME_i}, R_{G_n-i}, HFS_k, LAI_{ME_i}, bP)$,
then HFS_k believes
$(ID_{G_n}, TID_{ME_i}, R_{G_n-i}, HFS_k, LAI_{ME_i}, bP)$.

In steps *d) - f),* HFS_k verifies message MAC_i $(ID_{G_n}, TID_{ME_I}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i}, bP)$ by computing MAC'_i. The HFS_k then compares MAC'_i with the MAC_i. If the verification passes, it is the correct ME_i. Then HFS_k believes authentication message from ME_i.

After that, HFS_k sends the authentication message $(R_{G_n-i}, ID_{HFS_k}, HFS_k, GDL, TS_{HFS_k})$ to MME_j.

(g) MME_j believes $HFS_k \xleftrightarrow{SK_{MME_j - HFS_k}} MME_j$ **and** MME_j see

$\langle R_{G_n-i}, ID_{HFS_k}, HFS_k, GDL, TS_{HFS_k} \rangle_{SK_{MME_j-HFS_k}}$,

then MME_j believes HFS_k said

$(R_{G_n-i}, ID_{HFS_k}, HFS_k, GDL, TS_{HFS_k})$.

(h) MME_j believes fresh (TS_{HFS_k}) **and** MME_j believes HFS_k said

$(R_{G_n-i}, ID_{HFS_k}, HFS_k, GDL, TS_{HFS_k})$,

then MME_j believes HFS_k believes

$(R_{G_n-i}, ID_{HFS_k}, HFS_k, GDL, TS_{HFS_k})$.

The conjunction can be broken and the result is MME_j believes HFS_k believes $(R_{G_n-i}, ID_{HFS_k}, HFS_k, GDL)$.

(i) MME_j believes HFS_k controls $(R_{G_n-i}, ID_{HFS_k}, HFS_k, GDL)$

and MME_j believes HFS_k believes $(R_{G_n-i}, ID_{HFS_k}, HFS_k, GDL)$,

then MME_j believes $(R_{G_n-i}, ID_{HFS_k}, HFS_k, GDL)$.

In steps *g) – i)*, MME_j gets message $(R_{G_n-i}, ID_{HFS_k}, HFS_k, GDL, \langle R_{G_n-i}, ID_{HFS_k}, HFS_k, GDL \rangle_{SK_{MME_j-HFS_k}})$ from HFS_k, and uses a long-term secret key $(SK_{MME_j-HFS_k})$ between MME_j and HFS_k to verify message from HFS_k. If the verification passes, MME_j believes that the message is from HFS_k.

After that, MME_j verifies the authentication message MAC_q from ME_i as follows.

(j) MME_j believes $ME_i \xleftrightarrow{SK_{ME_i-MME_j}} MME_j$ **and** MME_j see

$\langle ID_{G_n}, TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i}, bP \rangle_{SK_{ME_i-MME_j}}$,

then MME_j believes ME_i said

$(ID_{G_n}, TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i}, bP)$.

(k) MME_j believes fresh (TS_{G_n-i}) **and** MME_j believes ME_i said
(ID_{G_n}, TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i}, bP),
then MME_j believes ME_i believes
(ID_{G_n}, TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i}, bP).

The conjunction can be broken and the result is MME_j believes ME_i believes (ID_{G_n},TID_{ME_i}, R_{G_n-i}, HFS_k, LAI_{ME_i}, bP).

(l) MME_j believes ME_i controls (ID_{G_n},TID_{ME_i}, R_{G_n-i}, HFS_k, LAI_{ME_i}, bP)
and MME_j believe ME_i believes (ID_{G_n},TID_{ME_i}, R_{G_n-i}, HFS_k, LAI_{ME_i}, bP),
then MME_j believes (ID_{G_n},TID_{ME_i}, R_{G_n-i}, HFS_k, LAI_{ME_i}, bP).

In steps *j)* - *l)*, MME_j verifies message MAC_q (ID_{G_n}, TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i}, bP) from ME_i by using $SK_{ME_i-MME_j}$ to compute MAC'_q. If the verification passes, it is the correct ME_i. Then MME_j believes authentication message from ME_i.

After that, MME_j selects random number a and computes aP and uses bP in ME_i's message to compute abP. MME_j now can compute a shared session key $SSK_{MME_j-ME_i}$ between MME_j and ME_i. MME_j then sends the authentication message MAC_{MME_j} (ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, TS'_{MME_j}, aP) to ME_i.

(m) ME_i believes $MME_j \xleftrightarrow{SK_{ME_i-MME_j}} ME_i$ **and** ME_i see
$\langle ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, TS'_{MME_j}, aP\rangle_{SK_{ME_i-MME_j}}$,
then ME_i believes MME_j said
(ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, TS'_{MME_j}, aP).

(n) ME_i believes fresh (TS'_{MME_j}) **and** ME_i believes MME_j said
(ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, TS'_{MME_j}, aP),
then ME_i believes MME_j believes
(ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, TS'_{MME_j}, aP).

The conjunction can be broken and the result is ME_i believes MME_j believes (ID_{MME_j},ID_{G_n}, TID_{ME_i},R_{MME_j},R_{G_n-i}, aP).

(o) ME_i believes MME_j controls
(ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, aP)
and ME_i believes MME_j believes (ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, aP),
then ME_i believes (ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, aP).

In steps *m) - o)*, ME_i verifies message from MME_j by using $SK_{ME_i-MME_j}$ and believes that the message is from MME_j. ME_i uses aP in a message to compute abP. ME_i now can compute a shared session key $SSK_{MME_j-ME_i}$ between ME_i and MME_j.

4.2 Authentication Proof for the Remaining MEs

We need to prove that the MME_j which has believed ME_i's long term public key in GDL uses the key to compute a long-term secret key ($SK_{ME_i-MME_j}$) between ME_i and MME_j. MME_j uses $SK_{ME_i-MME_j}$ to verify ME_i's message. It then can believe ME_i's session public key, bP. Further, the proof is that ME_i can believe MME_j's session public key, aP. Both MME_j and ME_i can use aP and bP to compute a shared session key, abP.

To analyze this protocol, the following assumptions are made.

1) ME_i believes $MME_j \xleftrightarrow{SK_{ME_i-MME_j}} ME_i$.
2) MME_j believes fresh (TS_{G_n-i}).

3) ME_i believes fresh (TS'_{MME_j}).

4) MME_j believes ME_i controls (ID_{G_n},TID_{ME_i}, R_{G_n-i}, HFS_k, LAI_{ME_i}, bP).

5) ME_i believes MME_j controls (ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, aP).

The steps of the proof are as follows:

(a) MME_j believes $ME_i \xleftrightarrow{SK_{ME_i-MME_j}} MME_j$ **and** MME_j see $\langle ID_{G_n}, TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i}, bP\rangle_{SK_{ME_i-MME_j}}$,
then MME_j believes ME_i said (ID_{G_n},TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i}, bP).

(b) MME_j believes fresh (TS_{G_n-i}) **and** MME_j believes ME_i said (ID_{G_n},TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i}, bP),
then MME_j believes ME_i believes (ID_{G_n},TID_{ME_i}, R_{G_n-i}, TS_{G_n-i}, HFS_k, LAI_{ME_i}, bP).

The conjunction can be broken and the result is MME_j believes ME_i believes (ID_{G_n}, TID_{ME_i}, R_{G_n-i}, HFS_k, LAI_{ME_i}, bP).

(c) MME_j believes ME_i controls (ID_{G_n}, TID_{ME_i}, R_{G_n-i}, HFS_k, LAI_{ME_i}, bP)
and MME_j believe ME_i believes (ID_{G_n},TID_{ME_i}, R_{G_n-i}, HFS_k, LAI_{ME_i}, bP),
then MME_j believes (ID_{G_n},TID_{ME_i}, R_{G_n-i}, HFS_k, LAI_{ME_i}, bP).

In steps *a) – c)*, MME_j verifies message from ME_i by using $SK_{ME_i-MME_j}$.

After that, MME_j selects random number a and computes aP. It then uses bP in ME_i 's message to compute abP. MME_j now can compute a shared session key $SSK_{MME_j-ME_i}$ between MME_j and ME_i. MME_j then sends the authentication message MAC_{MME_j} (ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, TS'_{MME_j}, aP) to ME_i.

(d) ME_i believes $MME_j \xleftrightarrow{SK_{ME_i-MME_j}} ME_i$ **and** ME_i see

$\langle ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, TS'_{MME_j}, aP \rangle_{SK_{ME_i-MME_j}}$,

then ME_i believes MME_j said

$(ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, TS'_{MME_j}, aP)$.

(e) ME_i believes fresh (TS'_{MME_j}) **and** ME_i believes MME_j said

$(ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, TS'_{MME_j}, aP)$,

then ME_i believes MME_j believes

$(ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, TS'_{MME_j}, aP)$.

The conjunction can be broken and the result is ME_i believes MME_j believes $(ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, aP)$.

(f) ME_i believes MME_j controls $(ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, aP)$

and ME_i believes MME_j believes $(ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, aP)$,

then ME_i believes $(ID_{MME_j}, ID_{G_n}, TID_{ME_i}, R_{MME_j}, R_{G_n-i}, aP)$.

In steps *d) - f)*, ME_i verifies message from MME_j by using $SK_{ME_i-MME_j}$ and believes that the message is from MME_j. ME_i uses aP in the message to compute abP. ME_i now can compute a shared session key ($SSK_{MME_j-ME_i}$) between ME_i and MME_j.

CHAPTER 5

SECURITY ANALYSIS OF PROTOCOL

In this chapter, the SE-GA security is analyzed and compared with the SE-AKA and GLARM protocols.

5.1 Entity Mutual Authentication

The main goal is to have an authentication between MME and ME in order to create a secure channel for sending data. For the first ME, it will authenticate itself with the home facilitator server (HFS) because the information of ME and the group is at the Home of ME. After ME has confirmed its success, the Home will send ME's Group Detail List (GDL) to MME. An MME trusts ME and the authentication message from ME because MME gets a correct response from ME's Home.

The rest of the group members can authenticate directly with MME because the information of MEs and the group has been sent to MME after the first ME has finished its authentication process.

For example, ME and HFS have a shared key (SK_{ME-HFS}) generated from Diffie-Hellman key exchange in the initialization stage. For authentication of the first ME, ME generates $AUTH_i$ and sends it to the MME. The MME verifies Home of ME from $AUTH_i$ and then forwards $AUTH_i$ to the Home. Home verifies the first ME by function MAC_i which MAC_i is computed by using a shared secret key (SK_{ME-HFS}) between ME and Home. For authentication between ME and MME, MME uses the information obtained from ME's Home to generate a key (SK_{MME-ME}) between ME and MME to validate MAC_q. If it is valid, MME trusts ME and sends $AUTH_{MME_j}$ to ME. An ME checks the MME by verifying MAC_{MME_j} in $AUTH_{MME_j}$ using the key (SK_{MME-ME}) between ME and MME. If the verification passes, ME believes MME.

For the rest of the group, the mutual authentication between ME and MME is made by using function MAC_q and MAC_{MME_j} which are computed by using a long-term secret key (SK_{MME-ME}).

5.2 Confidentiality

After the authentication process, the key data used for generating the session key (SSK/KGK) between MME and ME is *abP*. This *abP* is computed by using the Diffie-Hellman key exchange. The session key (SSK/KGK) is utilized to encrypt data between ME and MME. Thus, SSK/KGK can provide the data confidentiality.

5.3 Data Integrity

The integrity of messages between ME and MME, and between ME and Home are controlled by MAC function calculated from key SK_{MME-ME}, SK_{HFS-ME}, respectively. These keys are computed by using the Diffie-Hellman key exchange and known only between the two parties. Then every message sent in the protocol has a MAC function to achieve integrity control.

5.4 Enhanced Privacy-Preservation

For the first time when ME registers with the HFS, the ME gets a pair of permanent/temporary identity (PID_{ME} / TID_{ME}) to register in 3GPP networks. In the real case, ME does not send PID_{ME} into the communication network without protection because PID_{ME} is ME's privacy which may cause harm if it is sniffed. In SE-GA protocol, ME can send TID_{ME} into the communication network to the other party with MAC and the party can verify TID_{ME} by MAC function. In addition, in the case that the network needs ME to send PID_{ME} to the home network, the PID_{ME} may be encrypted with a long-term secret key between ME and HFS.

5.5 Secure Key Derivation

In the SE-GA, the SSK_{MME-ME} is created from a function which uses a shared secret between MME and ME. As described in section 5, MME and ME send a session public key of their own (*aP/bP*) to compute a shared secret *abP* between them. This *abP* is computed by making use of Diffie-Hellman key exchange which is secure. After that, both MME and ME use *abP* to generate SSK_{MME-ME}.

5.6 Key Forward / Backward Secrecy (KFS/KBS)

In the SE-GA, the session public keys *(aP/bP)*, which are used to compute session key, are sent between MME and ME, while a long-term secret SK_{ME-MME} is calculated from a long-term public/private keys of MME and ME respectively. Then, the session public keys are not related to the calculation of the SK_{ME-MME}. In addition, the SSK key value between ME and MME is very difficult to attack because this value is based on *abP* and known only between ME and MME. Then the KFS/KBS can be achieved.

5.7 Group Key Forward / Backward Secrecy (GKFS/GKBS)

When group members join or leave the group, the group key needs to update in order to preserve backward and forward secrecy. Up to now, several protocols have been proposed for dynamic group key agreement, such as Pipat Hiranvanichakorn (2017) and Zhu (2016). After updating the group key, the group will send a group's information such as the public keys of new members/leaving members, group members' numbers to each member's Home. Then the member who has newly joined or left will not know any information before joining or after leaving.

5.8 Resistance to Replay Attack

While MME and ME are communicating, authentication messages are sent with timestamps and random numbers, thus preventing replay attacks. For example of case 1, between MME and ME, there is a chance of replay attack, so while ME is sending a message to MME in step 1 to request services, a timestamp (TS_{G_n-i}) is included into the message. Similarly, when MME responds to ME in step 4, a timestamp (TS'_{MME}) is attached to the message to prevent replay attack.

5.9 Resistance to Redirection Attack

Because the authentication message ($AUTH_i$) from ME included with LAI_{ME}, MAC_q and MAC_i. The LAI_{ME} indicates the BS which ME contacts at that time. If the MME forwards $AUTH_i$ to HFS, then the HFS uses LAI_{MME_j} to compare with LAI_{ME}. In the case $LAI_{ME} = LAI_{MME_j}$, the HFS computes MAC'_i and compares with MAC_i in step (3) of authentication for the first ME. If $MAC'_i = MAC_i$ then HFS accepts the authentication. It rejects the authentication if the verification of MAC_i fails. For the remaining ME, the MME uses LAI_{MME_j} getting from the BS to compare with LAI_{ME} embedded in $AUTH_i$. If LAI_{ME} has the same value as LAI_{MME_j} then MME verify MAC_q with MAC'_q. Thus, SE-GA protocol can prevent the redirection attack.

5.10 Resistance to Man-in-the-Middle Attack

During the first confirmation of ME, an attacker may disguise as MME to sniff the information. Then the attacker disguises as the ME and sends the information to the real MME. As the attacker does not know the value *b*, he/she may try to perform man-in-the-middle attack by replacing *bP* with b_1P. However, it cannot fool the Home because the attacker does not know the secret key SK_{ME-HFS} which is utilized to compute MAC between ME and its Home.

In the case of the remaining ME, the secret key (SK_{ME-MME}) is utilized to protect messages between ME and MME. If an attacker changes messages, the MME can know messages which are not sent from the real ME. Thus, the protocol can prevent a man-in-the-middle attack.

5.11 Resistance to Denial-of-Service (DoS) Attack

While performing the authentication process, a malicious ME can run DoS attack on either HFS or MME. If a malicious ME forges the message, HFS or MME can detect the forged message by checking TS and comparing LAI in the message from the ME with LAI from MME.

5.12 Resistance to Impersonate Attack

The SE-GA protocol makes use of each ME's long-term private and public keys to achieve secure authentication between ME and MME. It is very difficult for an ME to disguise itself as another ME.

5.13 Comparison between SE-GA and Some Other Protocols

The comparison of security and flexibility based on an actual usage in some group authentication protocols as shown in Table 5.1. It was found that the SE-GA has better features.

Table 5.1 Comparison of the Proposed Protocol (SE-GA) with Some Schemes

PROTOCOL / FEATURES	SE-AKA	GLARM	SE-GA
AK	Symmetric Keys & Group Key*	Symmetric Keys**	Diffie- Hellman***
RMA	Yes	Yes	Yes
RRA	Yes	Yes	Yes
GMD	No	No	Yes
GMS	No	No	Yes
GDO	Yes	No	No

Note: AK: Authentication Key.

RMA: Resistance to Man-in-the-Middle-Attack.

RRA: Resistance to Redirection Attack.

GMD: Group Members can Come from the Different Home Networks.

GMS: Group Members can Use Different Networks Simultaneously.

GDO: Group Members Disguised as Others.

* The first ME uses a pre-shared key which is received from the Home in the initial stage to authenticate with the Home in order to use the network service, while the remaining MEs use the group key to authenticate with the MME.

** Each ME uses the symmetric key defined by its Home when it first registered with the Home in order to authenticate itself with the service network.

*** The key used in the authentication process can be created on the fly between the two parties by making use of the Diffie-Hellman key exchange.

CHAPTER 6

CONCLUSION

This research has developed the SE-GA protocol that assists group authentication on LTE networks. Group members can access the service networks from different LTE networks and can locate in different locations for authentication in order to access the service networks at the same time which is better than GLARM protocol. The authentication protocol uses the long-term private keys and public keys between parties to create shared secret keys used in the authentication process. This authentication process helps the group members to protect the information in the real scenarios. In particular, group members cannot disguise themselves as other members for authentication to use the service network. This feature is better than the SE-AKA which group members can disguise as another one. Before authentication between ME and MME, a shared secret key is used between ME and Home for the first member of the group, or ME and MME for the rest of the group. The MAC is used to control the data integrity while the data is being sent. After confirming the authentication, the session key between ME and MME is used to send the information which provides secrecy, so the information sent after authentication is confidential. By using this technique, SE-GA can be flexible and scalable. In addition, this reduces the provider's network traffic as well as network delays. Hence, the authentication of the group members excluding the first one, SE-GA needs only three steps for the authentication of each member while the former SE-AKA needs at least four steps.

The results of this research show that this developed protocol is consistent with the actual situation which was the starting point of our interest as stated in the source of the problem in Chapter 1.

BIBLIOGRAPHY

Bauer, R.K.; Berson, T.A. and Feiertag, R.J. 1983. A Key Distribution protocol using event markers. **ACM Transactions on Computer Systems**. 1(3): 249-255.

Burrows, M.; Abadi, M. and Needham, R. 1990. A Logic of Authentication. **ACM Transactions on Computer Systems**. 8 (1): 18-36.

Cao, J.; Ma, M. and Li, H. 2012. A Group-Based Authentication and Key Agreement for MTC in LTE Networks. **IEEE Global Communications Conference (GLOBECOM December 3-7, 2012)**. 1017-1022. doi: 10.1109/GLOCOM.2012.6503246

Chen, Y.-W.; Wang, J.-T.; Chi, K. H. and Tseng, C.-C. 2012. Group-Based Authentication and Key Agreement. **Wireless Personal Communications**. 62 (4): 965-979.

Denning, D.E. and Sacco, G.M. 1981. Timestamps in Key Distribution Protocol. **Communication of the ACM**. 24(8): 533-536.

Katz, J. and Lindell, A.Y. 2008. Aggregate message authentication codes. **Topics in Cryptology–CT-RSA 2008**. 155–169. doi: https://doi.org/10.1007/978-3-540-79263-5_10

Lai, C.; Li, H.; Lu, R. and Shen, X. 2013. SE-AKA: A Secure and Efficient Group Authentication and Key Agreement Protocol for LTE Networks. **Computer Networks**. 57 (17): 3492-3510.

Lai, C.; Lu, R.; Zheng, D.; Li, H. and Shen X. 2016. GLARM: Group-Based Light Weight Authentication Scheme for Resource-Constrained Machine to Machine Communications. **Computer Networks**. 99: 66-81.

Miller, S.P.; Neuman, C.; Schiller, J.I. and Saltzer, J.H. 1988. Kerberos Authentication and Authorization System. **Project Athena Technical Plan**. Section E.2.1: 1-36. Retrieved December 20, 2018 from http://web.mit.edu/Saltzer/www/publications/athenaplan/e.2.1.pdf

Needham, R.M. and Schroeder, M.D. 1978. Using Encryption for Authentication in Large Networks of Computers. **Communication of the ACM**. 21 (12): 993-999.

Nohrborg, M. 2018. **3GPP**. Retrieved November 18, 2018 from http://www.3gpp.org/LTE

Ou, H.-H.; Hwang, M.-S. and Jan, J.-K. 2010. A Cocktail Protocol with the Authentication and Key Agreement on the UMTS. **Journal of Systems and Software**. 83 (2): 316–325.

Pipat Hiranvanichakorn. 2017. Provably Authenticated Group Key Agreement Based on Braid Groups - The Dynamic Case**. International Journal of Network Security**. 19 (4): 517-527.

Wang, F.; Chang, C. and Chou, Y. 2015. Group Authentication and Group Key Distribution for Ad Hoc Networks. **International Journal of Network Security**. 17 (2): 199-207.

Wu, S.; Zhu, Y. and Pu, Q. 2010. Security Analysis of a Cocktail Protocol with the Authentication and Key Agreement on the UMTS. **IEEE Communications Letters**. 14 (4): 366-368.

Zhu, H. F. 2016. Secure Chaotic Maps-B ased Group Key Agreement Scheme with Privacy Preserving. **International Journal of Network Security**. 18 (6): 1001-1009.

BIOGRAPHY

NAME	Mr. Boriphat Kijjabuncha
ACADEMIC BACKGROUND	B.S. degree in Computer Science from Silpakorn University, Nakorn Pathom, Thailand, in 2000**.** M.S. degree in Applied Information System at School of Applied Statistics, National Institute of Development Administration (NIDA), Thailand in 2006.
PRESENT POSITION	Lecturer in Business Information Technology, Department of Information Business, Faculty of Information and Communication Technology, Silpakorn University, Petchburi, Thailand.

PUBLICATIONS

Boriphat Kijjabuncha and Pipat Hiranvanichakorn. 2019. A Provably Secure Group Authentication Protocol for Various LTE Networks**. International Journal of Network Security**. 21 (3): 838-851.

Note:

This article is accepted to be published in the International Journal of Network Security (IJNS).