Proc. of the Intl. Conf. on Future Trends in Computing and Communication -- FTCC 2013

ISBN: 978-981-07-7021-1

A Secure and Efficient Multi-Device Authentication Protocol Based on Secret Sharing in Heterogeneous Networks

[Haejun Jung, and Jooseok Song]

***Abstract*—Current EAP-AKA provides mutual authentication and key agreement in heterogeneous networks. However, with mobile devices increasing explosively, this protocol seems to be inefficient as each device needs to communicate with its remote home network to authenticate. In this paper, we propose a secure and efficient authentication protocol for owner of multiple devices by implementing shared group temporary key and secret sharing method. The devices which belong to a user generate group temporary key with shared secret. A master device communicates with home network for authentication. The other devices can be authenticated by serving network locally instead of communicating with remote home network. We will prove that the proposed protocol provides secure authentication and it is more efficient than the conventional EAP-AKA in multi-device scenario.**

***Keywords*—EAP-AKA, authentication, multi-device, secret sharing, group key**

I. Introduction

EAP-AKA provides mutual authentication between a mobile user/station(MS) and its home network(HN) in heterogeneous network. Each mobile station has an unique identification and pre-shared key(PSK). When the MS requests communication session to serving network(SN), SN sends the request message to HN. Then the authentication center(AuC) in HN generates a one-time authentication vector(AV) related to the MS and sends it to SN. After above procedure, SN can authenticate the MS.

However, with the trend of carrying multiple mobile devices by one user, EAP-AKA becomes inefficient because EAP-AKA is a device oriented protocol, which means that all of devices belonging to the same user should be authenticated by HN. It would cause severe communication overhead and considerable storage is needed in HN as the number of multi-device users increases.

Many papers were published to develop the EAP-AKA, yet most of them didn’t solve the problem of high communication overhead between SN and HN in multi-device environment. Huang et al. proposed the SEMMAP protocol using concept of peer ID and peer root key[3]. All of the devices belonging to one user(peer) have the same pre-shared key and peer identity. When the first device authenticates with HN, HN generates peer root key using pre-shared key and random number. The HN sends it to the SN with other authentication vector. The other devices can also generate the same root key when they authenticate with SN and there is no need to communicate with HN. However this protocol needs the pre-assumption that all of devices have the fixed same pre-shared key. If an attacker knows the pre-shared key, he can get all the communication contents of devices belonging to the peer. Yu-Wen et al. proposed the group-based authentication and key agreement protocol[4] based on UMTS X-AKA[5]. The HN has an index table which contains group information including group ID, member ID, initial value and so on. Each device and HN have the same group authentication key. When the HN is requested authentication from serving network related to a group, HN checks the index table and generates a group temporary key(GTK). Each device and SN can authenticate each other with the GTK and other components of AV from HN. This scheme can reduce the number of communications between SN and HN. However, HN needs to share the permanent GAK with mobile users and know group information, which will cause additional cost.

To solve the previous problem, we propose a secure and efficient authentication scheme with secret sharing method. The devices belonging to a user generate a group temporary key periodically by sharing the shareholder through non-secure channel. The procedure of generating temporary key is similar to group key transfer protocol[6] of Lein et al. Comparing with previous multi-device authentication protocols based on EAP-AKA, our protocol improves the security level because of periodic key TK generation. It also improves efficiency by reducing the communication between the SN and HN with group temporary key which is used in authentication between MSs and SN.

This paper is organized as follows. We will introduce the current EAP-AKA protocol and group key transfer protocol based on secret sharing in section II. In section III, we present our proposed scheme in three stages. Next we analyze our protocol in security and performance aspects in section Ⅳ. Finally, we conclude our paper in section V.

Haejun Jung
Department of Computer Science, Yonsei University
Republic of Korea(South Korea)

Jooseok Song
Department of Computer Science, Yonsei University
Republic of Korea(South Korea)

II. Related Work

A. *Current EAP-AKA Protocol*

EAP-AKA protocol is illustrated as Fig.1. When the MS receives the ID request message from SN, it sends its ID to the SN. Then the SN sends received ID to HN, and HN generates AV related to the MS as follows:

AV : RAND, XRES, CK, IK, AUTN

RAND=$f0$(internal state)

XRES=$f2$(K, RAND)

CK=$f3$(K, RAND)

IK=$f4$(K,RAND)

AUTN=SQN$\oplus$AK||AMF||MAC

where

MAC=$f1$(K, SQN||RAND||AMF)

AK=$f5$(K, RAND)

The HN sends back the AV so that SN authorizes the corresponding MS. The SN sends RAND, AUTN, MAC to the MS. Then the MS verifies the correctness of SQN by computing MAC and comparing it with the MAC from SN. The MS also generates RES and sends it to SN. The SN checks whether the RES from MS is same with the XRES from HN. The MS can also generate CK/IK using RAND. Finally SN and MS have a common session key.

However, this EAP-AKA protocol provides authentication based on the USIM, which means that each device needs communications between SN and HN to get the corresponding AV. With the multi-device users increasing, the communication overheads will increase and cause inefficient situations.

B. *Group Key Transfer protocol based on Secret Sharing*

Lein et al. proposed a group key transfer protocol based on secret sharing[6]. We take advantage of this secret sharing scheme in our multi-device environment. This protocol consists of three procedures: initialization of Key Generation Center(KGC), user registration, and group key generation and distribution. The brief description is as follows:

Initialization of KGC. The KGC randomly chooses two safe primes p and q and compute n=pq. n is made publicly known.

User Registration. During registration, the KGC shares a secret, (x_i, y_i), with each user U_i, where x_i, $y_i \in Z_n^*$

Group key generation and distribution. When KGC receive a group key generation request from user, it randomly selects a group key and accesses all shared secrets with group members. KGC distributes this group key to all group members. All communication between KGC and group members are in a broadcast channel. For example, we assume that a group consists of t members, {U_1, U_2, . . . , U_t}, and shared secrets are (x_i, y_i), for i= 1, . . . , t. The key generation and distribution process consists of the following steps.

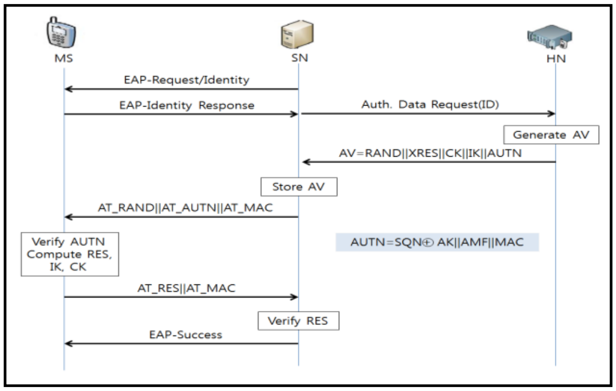

Figure 1. EAP-AKA full authentication

- Step 1. The initiator sends a key generation request to KGC with a list of group members
- Step 2. KGC broadcasts the list of all participating members.
- Step 3. Each participating group member sends a random challenge(R_i) to KGC.
- Step 4. KGC randomly selects a group key, K, and generates an interpolated polynomial $f(x)$ with degree t to pass through (t+1) points, (0, K) and (x_i, $y_i \oplus R_i$), for i=1, . . ., t. KGC also computes t additional points, P_i, for i=1, . . ., t, on $f(x)$ and $Auth$= h(K, U_1, . . . , U_t, R_2, . . ., R_t, P_1, . . . , P_t), where h is a one-way hash function. KGC broadcasts {$Auth$, P_i}, for i=1, . . ., t, to all group members. All computations are performed in Z_n^*
- Step 5. For each group member, U_i, knowing the shared secret, (x_i, $y_i \oplus R_i$), and t additional public points, P_i, for i=1, . . , t, on $f(x)$, is able to compute the polynomial $f(x)$ and recover the group key K=$f(0)$. Then, U_i computes h(K, U_1, . . , U_t, R_1, . . , R_t, P_1, . . , P_t) to check whether this hash value is identical to $Auth$. If these two values are identical, U_i authenticates the group key sent from KGC. Secret reconstruction algorithm is based on Lagrange Interpolation

III. Proposed Protocol

We propose an efficient multi-device authentication based on secret sharing. The key point of our scheme is that a master device acts like the KGC and each device can have fresh temporary key, which acts as a group key in authentication. It means that after a full authentication of master device, the other devices don't need any communication between SN and HN. Reducing communication messages in authentication makes this protocol more efficient than current EAP-AKA in the multi-device environment. It also improves the security level by using fresh temporary key rather than permanent pre-shared key.

A. *Assumptions*

- Devices belonging to a user configure a network and they communicate in an insecure channel.
- Each group has a group ID and all members know it.
- There is a master device among the devices. The master device acts as the key generation center(KGC) and has a group member list.
- Both of master device and HN know shared secrets (x_i, y_i) for each device.
- SN and HN trust each other and they have established a secure communication channel.

B. *Overall procedure*

Our protocol is divided into three stages. The first stage is the temporal key(TK) generation among devices before authentication with SN and HN. The second stage is TK generation in HN and mutual authentication. The last stage is the authentication of the other devices.

- **The First stage** TK generation procedure is illustrated in Fig. 2. We assume that devices belonging to the same user establish a network. In a case that a user owns three devices(MS$_1$, MS$_2$, MS$_3$) and MS$_1$ is the master device, MS$_1$ initiates TK generation process. MS$_1$ already knows the list of group members and the secret (x_i, y_i) of group members. At first, MS$_1$ broadcasts group key refresh message. Each member selects a random challenge $R_i \in Z_n^*$ and sends it to master device MS$_1$. MS$_1$ also selects R_1. Then, MS$_1$ randomly selects a TK, and generates an interpolated polynomial $f(x)$ with degree 3 to pass the four points, (0,TK), (x_1,$y_1 \oplus R_1$), (x_2,$y_2 \oplus R_2$), (x_3,$y_3 \oplus R_3$). MS$_1$ also computes three additional points, P_i(P_1, P_2, P_3) and $Auth$=h(TK, ID$_G$, P_1, P_2, P_3), where h is a one-way hash function. MS$_1$ broadcasts {$Auth$, P_1, P_2, P_3} to all group members. Now, each member knows the shared secret, (x_i, $y_i \oplus R_i$), and three additional public points(P_1, P_2, P_3). They can compute the polynomial $f(x)$ and recover the TK=$f(0)$. They also check the $Auth$ to ensure MS$_1$'s legitimacy.

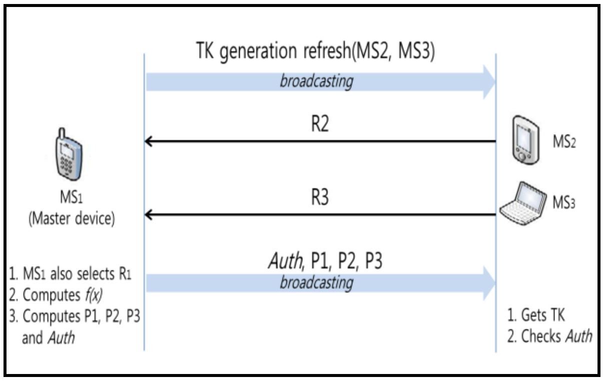

Figure 2. TK generation procedure

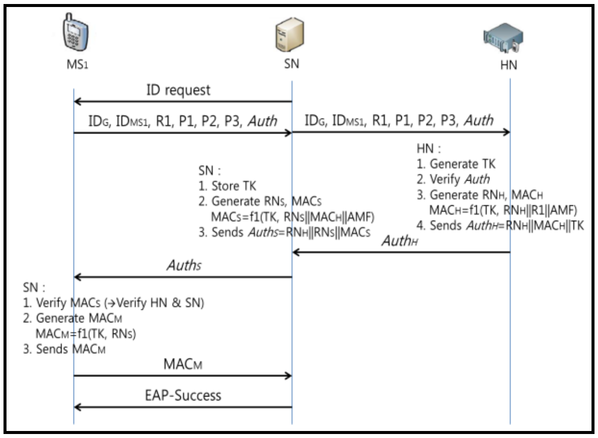
MS1
SN
HN
ID request
IDG, IDMS1, R1, P1, P2, P3, Auth
IDG, IDMS1, R1, P1, P2, P3, Auth
HN :
1. Generate TK
2. Verify Auth
3. Generate RNH, MACH
MACH=f1(TK, RNH||R1||AMF)
4. Sends AuthH=RNH||MACH||TK
SN :
1. Store TK
2. Generate RNs, MACs
MACs=f1(TK, RNs||MACH||AMF)
3. Sends Auths=RNH||RNs||MACs
AuthH
Auths
SN :
1. Verify MACs (→Verify HN & SN)
2. Generate MACM
MACM=f1(TK, RNs)
3. Sends MACM
MACM
EAP-Success

Figure 3. TK generation of HN and mutual authentication

- **The Second stage** After the first stage, each device knows the TK. When the user visit SN, the SN sends ID request message. The master device sends group ID, device ID, R_1, P_1, P_2, P_3 and $Auth$ to the SN. Then the SN sends this message to HN. Upon receiving the message, HN generates TK using R_1, P_1, P_2 and P_3. After that, HN can verify MS$_1$ by checking $Auth$. The HN selects a random number RN$_H$ and calculates MAC$_H$ to prove itself to MS$_1$ later. MAC$_H$ is $f1$(TK, RN$_H$||R_1||AMF) and AMF is Authentication Management Field in EAP-AKA protocol. Lastly HN sends Auth$_H$(RN$_H$|| MAC$_H$|| TK) to SN. When SN receives Auth$_H$ from HN, it stores TK and generates MAC$_S$ using TK with a random number which SN selects, MAC$_H$ and AMF. Then SN sends Auth$_S$(RN$_H$ ||RN$_S$||MAC$_S$) to MS$_1$.

  Now MS$_1$ can verify HN and SN simultaneously by checking MAC$_S$ with TK, RN$_H$, RN$_S$. MS$_1$ generates MAC$_M$ using $f1$ function with TK and RN$_S$. MS$_1$ sends it to SN to prove itself. After verifying MAC$_M$, the SN sends EAP-Success message to MS$_1$.

- **The last stage** Upon receiving ID request message from SN, MS$_2$ response group ID and device ID. Then SN selects a new random number RN$_S$ and generate new MAC$_S$ with TK(generated in previous stage). The SN sends new Auth$_S$(RN$_S$||MAC$_S$) to MS$_2$. MS$_2$ can verify the SN by checking the MAC$_S$. If SN is legitimate, MS$_2$ generates MAC$_{M2}$ and sends it to SN. Finally, SN and MS$_2$ can authenticate each other.

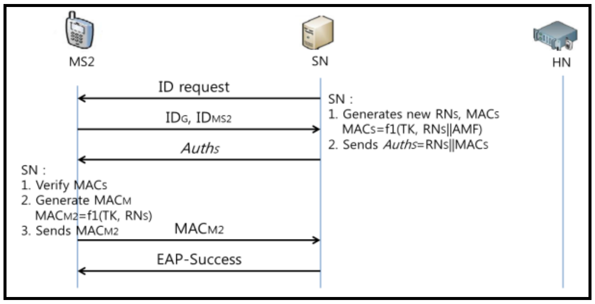

Figure 4. Authentication procedure of the other devices

IV. Security and Performance Analysis

Our protocol provides mutual authentication between MS and HN or MS and SN like EAP-AKA. In case of master device, HN can verify that MS is legitimate by checking *Auth*. MS also verifies SN and HN simultaneously by checking MACs containing MACH. Moreover we implement secret sharing key distribution in the first stage. In this way, the devices possess a fresh TK periodically. Previous protocols related to multi-device authentication assume that all of devices belonging to a user have a fixed pre-shared key. Therefore if an attacker get this pre shared key of one device, he can watch all messages continuously. In the same situation of our scheme, the attacker cannot watch messages continuously because TK is updated efficiently.

In the view of performance, our protocol is more efficient than the current EAP-AKA in multi-device environment. Considering that *n* devices from *k* users and that each device initiates *m* authentications, Table 1 shows the number of messages communicated in our protocol and current EAP-AKA. We apply the method of Yu-Wen et al.[5] to calculate the number of signaling messages in authentication.

TABLE I. NUNMER OF MESSAGES IN AUTHENTICATION

Protocol	**1 MS(*n*=1)**		***n* MSs**	
	m=1	***m>1***	***m=1***	***m>1***
EAP-AKA	7	7*m*	7*n*	7*mn*
Our protocol	7	7+5(*m*-1)	7*k*+5(*n*-*k*)	7*k*+5(*n*-*k*) +5*n*(*m-1*)

In the case that a user has only one device, the total number of messages in authentication grows linearly with *m* in EAP-AKA. On the other hand, our protocol needs 7 messages for master device, 5 messages for the other devices. Considering the multi-device environment, total messages in EAP-AKA grows linearly with *m* and *n*. In comparison, the only master device needs full authentication, the other devices authenticate with SN instead of HN in our protocol.

If the *n* devices initiate multiple *m* authentications, master devices of each group need 7 messages, the others only need 5(*n*-*k*) messages. From the first authentication of each device, all of devices just need 5 messages in next rounds. Fig 5. Shows the number of messages in EAP-AKA and our protocol. When 20 users use 60 devices and each device initiates authentication 3 times, EAP-AKA needs 1,260 messages in authentication whereas our protocol needs 940 messages.

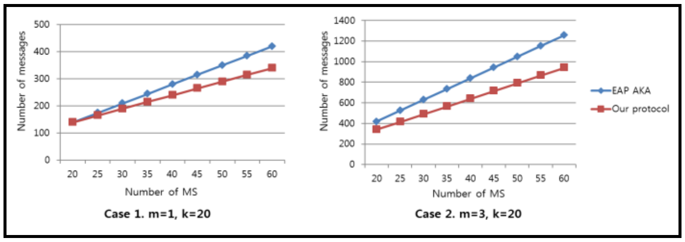

Figure 5. Number of messages in EAP-AKA and our protocol

V. Conclusion

As the number of mobile devices increases explosively, and many people tend to use more than one device, how to authenticate multiple devices becomes an important issue. In this paper, we propose secure and efficient authentication protocol in multi-device environment. We apply secret sharing method to generate fresh temporary key among the mobile devices and use this key as group session key to reduce the cost of message communication. The mutual authentication between MS and HN/SN also guarantees identity of each entity. However, we don't cover the communication overhead analysis among the devices in this paper because it is a separate part to the communication with SN or HN. We will focus on the overhead and constraints analysis to improve our protocol in future.

Acknowledgment

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2012 R1A1B3004161).

References

[1] J.Arkko, H.Haverinen, "Extensible Authentication Protocol Method for 3rd Generation Authentication and Key Agreement(EAP-AKA)" RFC 4187, Jan. 2006.

[2] Mun, Hyeran, Kyusuk Han, and Kwangjo Kim. "3G-WLAN interworking: security analysis and new authentication and key agreement based on EAP-AKA" Wireless Telecommunications Symposium, 2009. WTS 2009. IEEE, 2009..

[3] Jie Huang and Chin-Tser Huang, "A Secure and Efficient Multi-Device and Multi-Service Authentication Protocol(SEMMAP) for 3GPP-LTE Networks" Computer Communication and Network(ICCCN), 2012 21st International Conferece on. IEEE, 2012.

[4] Chen, Yu-Wen, et al. "Group-Based Authentication and Key Agreement" Wireless Personal Communications 62.4(2012):965-979.

[5] C.Huang and J Li, "Authentication and key agreement protocol for UMTS with low bandwidth consumption." Proceedings of the 19th International Conference on Advanced Information Networking and Application 2005, pp.392-397, Mar. 2005.

[6] Harn, Lien, and Changlu Lin, "Authenticated group key transfer protocol based on secret sharing" Computers, IEEE Transactions on 59.6(2010):842-846.

[7] Lin, Shen-Ho, Jung-Hui Chiu, and Sung-Shiou Shen. "A fast iterative localized re-authentication protocol for UMTS-WLAN heterogeneous mobile communication networks" EURASIP Journal on Wireless Communication and Networking, 2011(1), 1-16.

[8] Li, Xinghua, et al. "Authentications and key management in 3G-WLAN interworking" Mobile Networks and applications 16.3 (2011): 394-407

About Author (s):

Haejun Jung received the B.S. degree in Mechanical Engineering from Korea Military Academy, Seoul, Korea, in 2002. He is currently working toward the M.S. degree in Computer Science at Yonsei University, Seoul, Korea. His research interests include wireless ad hoc networks and network security.

Jooseok Song received the B.S. degree in Electrical Engineering from Seoul National University, Korea, in 1976, and the M.S. degree in Electrical Engineering from Korea Advanced Institute of Science and Technology, Korea, in 1979. In 1988, he received the Ph.D. degree in Computer Science from University of California at Berkeley. From 1988 to 1989, he was an Assistant Professor at the Naval Postgraduate School, Monterey, CA. He was the president of Korea Institute of Information Security and Cryptology in 2006. He is currently a Professor of computer science at Yonsei University, Seoul. His research interests include cryptography and network security.