2010

A secure end-to-end protocol for conference mobile call

Qifan Yang
University City College

Tiancheng Zhang
University of Wollongong, tz746@uowmail.edu.au

A secure end-to-end protocol for conference mobile call

Abstract

The advent of conference mobile call demands the security communication between end users. However, currently there is no efficient secure end-to-end protocol exists for conference mobile call. This slows down the steps of conference communication. In this paper a secure end-to-end protocol for remote conference is designed base on previous experts work, which is one-to-one end-to-end protocol. In addition, security analysis from perspectives of confidentiality, authenticity, anonymity, freshness as well as preventing from denial of service (DoS) attack on the protocol is made. At the end, the efficiency of this protocol is discussed.

Disciplines

Physical Sciences and Mathematics

Publication Details

Yang, Q. & Zhang, T. (2010). A secure end-to-end protocol for conference mobile call. 2010 2nd International Conference on Future Computer and Communication, ICFCC 2010 (pp. 430-433). Piscataway, NJ: IEEE.

A Secure End-to-End Protocol For Conference Mobile Call

Qifan Yang
School of Computer and Computing Science, Zhejiang University City College
Hangzhou, Peoples R China
Email:yangqf@zucc.edu.cn

Tiancheng Zhang
School of Computer Science & Software Engineering, University of Wollongong, Wollongong, NSW 2500, Australia.
Email:superztc@hotmail.com

***Abstract*—The advent of conference mobile call demands the security communication between end users. However, currently there is no efficient secure end-to-end protocol exists for conference mobile call. This slows down the steps of conference communication. In this paper a secure end-to-end protocol for remote conference is designed base on previous experts work, which is one-to-one end-to-end protocol. In addition, security analysis from perspectives of confidentiality, authenticity, anonymity, freshness as well as preventing from denial of service (DoS) attack on the protocol is made. At the end, the efficiency of this protocol is discussed.**

Keywords- end-to-end protocol, conference, mobile call, security, network

I. INTRODUCTION

Conference mobile call is getting popular in many companies. Instead of gathering together in a conference room, people can launch meetings through mobile devices. It makes conference possible wherever people are and creates more profits. Despite the convenience of it, not much concern is paid on the security of the conference mobile call. In the year 1996, Yi Mu and Vijay Varadharajan Vijay introduced a secure end-to-end protocol based on BCY and Carlsen's authentication protocols which are published in 1993 and 1994 respectively[1]. It can be used for one-to-one secure mobile device communication by both symmetric and asymmetric key based secure end-to-end protocols. However, an end-to-end protocol that supports conference mobile call has not been designed. In this paper, a secure end-to-end protocol for conference mobile call based on Yi's protocol is introduced together with the analysis of it. It enables members in remote conference communicate securely.

II. NOTATION USED

There are some notations used in the protocol and these notations are listed as follows.

1. A, B, C : end users A, B, C
2. As, Bs, Cs: subliminal ID of user A, B and C
3. As': new subliminal ID of A
4. AS1,AS2,AS3: authentication server 1, 2 and 3
5. A->B: A sends message to B
6. [Data]key: Encrypting the data with a key
7. h(…): a strong one way hash function
8. Ks: session key between A, B and C
9. n_A: a nonce generated by A
10. K_{AS1AS2}: shared key between authentication server AS1 and AS2
11. f(data): a function that generates key according to the data

III. PROTOCOL

According to Yi and Vijay's symmetric key base end-to-end protocol. In the first two steps, A requests to get authenticated herself from home authentication server (HAS) through home server (HS) and generates a nonce which is used for identify the session between. Then a session key with B, KAB, is generated in step three after HAS's successful verification on the A's identity. At the same time, HAS gives A a subliminal ID from further request. In the step four and five, the KAB will be delivered to A under the encryption of shared key between A and HAS. After A gets the message, she will response HAS that she has got the message. HAS sends the same KAB together with the nonce generated by A and a subliminal ID of B to B under the encryption of shared key between HAS and B in step six. B then responses HAS that he has got the message in step seven. After A and B decrypt the messages from HAS which contain the share key between them, they can communicate with each other securely based on this protocol.[1]

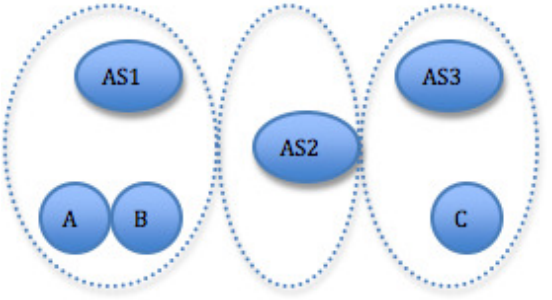

Figure 1. **Communication Scenario**

978-1-4244-5824-0/$26.00 ©2010 IEEE

However, this protocol cannot migrate to conference secure communication because of its limitation of authentication. Therefore, a protocol for conference mobile call which can be used in the scenario given in Figure 1 is introduced. As shown in Figure1, end user A is registered to AS1, B is registered to AS2 and C is registered to AS3. B moves to the AS1's domain and this domain regards B as a visitor. Assuming that A will launch a conference call and he wants to set up a secure end-to-end communication between A, B and C.

The brief authentication steps of this figure is that A should first get authenticated from AS1, and then AS1 will send A's request to AS3 in order to inform AS3 that A want to talk to C. After A got the session key with C, he sends another request to tell AS1 that B is requested to be talked. Then AS1 will ask AS2 to authenticate end user B which is in to domain of AS1.

We will have the protocol of setting up the communication as follows:

Step1: $A\ \ AS1{:}As,AS1,n_A,[C]_{KAAS1},[h(As,AS1,n_A,C)]K_{AAS1}$

Step2: $AS1\ \ AS3{:}AS1,\ \ AS3,\ \ n_{AS1},\ \ [A,As,C,n_A,Ks]_{KAS1AS3},\ [h(AS1,AS3,n_{AS1},n_A,A,As,C,Ks)]K_{AS1AS3}$

Step3: $AS3\ \ C{:}AS3,Cs,n_{AS3},[A,As,Cs',C,n_A,Ks]_{KCAS3},\ [h(AS3,Cs,n_{AS3},n_A,A,As,C,Ks)]_{KCAS3}$

Step4: $C\ \ AS3{:}Cs,AS3,n_{AS3},[h(Cs,AS3,n_{AS3},n_A,A,C,Ks)]_{KCAS3}$

Step5: $AS3\ \ AS1{:}AS3,AS1,n_{AS1},[Cs]_{KAS1AS2},[h(AS3,AS1,n_{AS1},n_A,A,C,Cs,Ks)]_{KAS1AS3}$

Step6: $AS1\ \ A{:}AS1,As,[Ks,As,Cs]_{KAAS1},n_A,[h(AS1,A,C,Cs,As,As',n_A,Ks)]_{KAAS1}$

Step7: $A\ \ AS1{:}AS1,n_A',[As,A,B]_{KAAS1},[h(AS1,n_A',A,As,B)]_{KAAS1}$

Step8: $AS1\ \ AS2{:}AS1,AS2,n_{AS1}',[A,As,n_A',B,Ks]_{KAS1AS2},[h(AS1,AS2,n_A',n_{AS1}',A,As,B,Ks)]_{KAS1AS2}$

Step9: $AS2\ \ AS1{:}AS2,n_{AS1}',[Bs,B,K_{BAS1}]_{KAS1AS2},\ [h(AS2,n_{AS1}',Bs,B)]_{KAS1AS2},[B,Bs',A,As,n_A',Ks]_{KBAS2},[h(B,Bs',A,As,n_A',Ks)]_{KBAS2}$
$where K_{BAS1} = f(AS2,Bs,K_{BAS2})$

Step10: $AS1\ \ B{:}\ AS1,\ Bs,\ n_{AS1}'',\ [B,\ \ Bs',A,As,n_A',Ks]_{KBAS2},\ [h(B,Bs',A,As,\ \ n_A',Ks)]_{KBAS2},\ [h(AS1,Bs,n_{AS1}'')]_{KBAS1}$

Step11: $B\ \ AS1{:}Bs,AS1,n_{AS1}'',[h(Bs,AS1,n_{AS1}'')]_{KBAS1}$

Step12: $B\ \ \ A\ :\ Bs\ ,\ As\ ,[message,n_A']_{Ks}\ ,[h(Bs\ ,\ As\ ,\ message,\ n_A')]_{Ks}$

Step13: $A\ \ C{:}As,Cs,n_A'',[message']_{Ks},[h(As,Cs,n_A'',message')]_{Ks}$

Step14: $B\ \ AS1{:}AS1,Bs,[C]_{KBAS1},n_B,[h(AS1,Bs,C,n_B)]_{KBAS1},[B,Bs,n_B]_{Ks},[h(B,Bs,n_B)]_{Ks}$

Step15: $AS1\ \ AS3{:}AS1,AS3,[C,Bs]_{KAS1AS3},n_{AS1}''',[h(AS1,AS3,C,Bs,n_{AS1}''')]_{KAS1AS3},[B,Bs,n_B]_{Ks},[h(B,Bs,n_B)]_{Ks}$

Step16: $AS3\ \ C{:}AS3,Cs',[C,Cs'',n_{AS3}']_{KCAS3},[h(AS3,C,Cs',Cs'',n_{AS3}')]_{KCAS3},[B,Bs,n_B]_{Ks},[h(B,Bs,n_B)]_{Ks}$

Step17: $C\ \ AS3{:}Cs',AS3,n_{AS3}',[h(Cs',AS3,n_{AS3}')]_{KCAS3},[C,Cs',B,Bs,n_B]_{Ks},[h(C,Cs',B,Bs,n_B)]_{Ks}$

Step18: $AS3\ \ AS1{:}AS3,AS1,n_{AS1}''',[h(AS3,AS1,n_{AS1}''')]_{KAS1AS3},[C,Cs',B,Bs,n_B]_{Ks},[h(C,Cs',B,Bs,n_B)]_{Ks}$

Step19: $AS1\ \ B{:}AS1,Bs,n_B,[h(AS1,\ \ Bs,n_B)]_{KBAS1},[C,Cs',B,Bs,n_B]_{Ks},[h(C,Cs',B,Bs,n_B)]_{Ks}$

Step20: $B\ \ C{:}Bs,Cs',[message'',n_B']_{Ks},[h(Bs,Cs',message'',\ n_B')]_{Ks}$

IV. SECURITY ANALYSIS

A. *Confidentiality*

In order to ensure the confidentiality of the communication, checksum is used in the protocol. The message sent to the destination is first hashed using a strong one why hash function, for instance, MD5, CRC32, SHA-1 etc. However, a hash value can be forged by an attacker who launches the man-in-the-middle attack or brute force attack[2]. Therefore, the hash value is then encrypted using the share key between the sender and the receiver. When the destination end gets that message, it can decrypt the message and get the hash value. Then, it composites all the elements except the encrypted hash value from the sender into a string and hash that string using the corresponding hash algorithm. The hash value got from the destination machine is compared with the value from decryption. If these two values are equivalent, then the receiver can make sure that the message is not forged or modified.

B. *Authenticity*

When it comes to authentication, there are two scenarios. The first one is the end user stay at the domain of his or her home authentication server (HAS) domain. The other one is the end user who is registered with his or her HAS domain moves to VAS (visitor authenticate server).

In the first scenario, as can be seen in the step1 A sends the message [C]KAAS1 and [h(As,AS1,nA,C)]KAAS1 to AS1. At the same time, AS1 gets A's subliminal ID, As, as well. It will match As with A's real ID, and then find out the share key between A and itself. Then this share key is used to decrypted [C]KAAS1 and [h(As,AS1,nA,C)]KAAS1. A successful decryption can make AS1 sure that it is talking to A. However, if AS1 cannot decrypt the message, then it will reject the request of the sender. AS1 can also record the IP

address of the sender. If the same IP address causes more three times failed attempt, the AS1 will block the IP address for a period of time to protect the shared key from being compromised from brute force attack.

When it comes to the second scenario which is the end user migrating to a VAS, the authentication method turns to be a little complex. In the protocol, B, which is registered to the AS2, moves to AS1. And as can be seen in figure 1, AS1 is B's VAS. B has to get authenticated from AS2 and AS1 is used as a media between B and AS2. In the protocol, step 7, 8, 9, 10 and 11 do the authentication for B. A firstly sends the AS1 the request of communicating to B. Then AS1 sends this request to AS2. When AS2 gets this request from B, it will give AS1 a message containing a token, [B, Bs ', A, As, nA', Ks]KBAS2, which is encrypted under KBAS2 and a message containing KBAS1 which can be decrypted using the share key between AS1 and AS2. AS1 then sends the token to B, but AS1 itself has no able to get the information inside of the token. This protects the communication between AS2 and B. After B receiving the message from AS1, it can calculate the shared key between AS1 and itself. Because B knows AS2, Bs, KBAS2 and the mobile device can calculate the where KBAS1 □f(AS2,Bs,KBAS2). This key can be used to ensure the confidentiality of the message from AS1. If B can decrypt the token, then B is authenticated.

C. *Anonymity*

In order to protect end users' actual identity, subliminal ID is used in the protocol.The real identity is stored in the AS. When the end AS receives the subliminal ID from the end user, it will match the subliminal ID with the actual ID in its database.

A subliminal ID creates a subliminal channel between the sender and the receiver and prevents the sender's private information from being exposed to the public[3]. A man-in-the-middle attack can intercept the identity of the end user and know the identity of the end user.

After the end user being authenticated, a new subliminal ID for the corresponding end user is sent back. Then the end user's mobile device can record this subliminal ID for future usage. This can ensure no same subliminal ID is used for the same end user. In the protocol given above, the new subliminal ID As' is encrypted under the shared key between A and AS1. This can guarantee that only A can decrypt this message and record this new subliminal ID.

D. *Freshness*

nonce is used in the protocol for freshness. The nonce is a non-repeat number. It can be used for preventing the server from replay attack. How to make the nonce unpredictable is significant for freshness[4].

A new replay attack against Tor, which is a real-world, circuit-based low latency anonymous communication network, is introduced in 2008. And the countermeasure given is to monitor duplicate cells. [5] The nonce in the protocol, for instance nA, can uniquely identify a session between the sender and the receiver, so this end-to-end protocol is secure from replay attack.

E. *Preventing from DoS attack*

The Denial of Service(DoS) attack from in the home domain is easy to prevent. If the authentication failed, the HAS can just reject the connection and free the memory. If a same IP continuously send request to HAS, the HAS can simply block that IP address. Liu X, Yang X and Lu Y introduced a filter-based DoS defence system called StopIt in 2008, which is a very efficient measure of preventing from DoS attack[6].

However, if a user goes to a VAS and he or she wants launch a conference call first. The VAS is vulnerable from DoS attack. Because, VAS cannot confirm the sender's identity before sending that request to the sender's HAS. The solution to this defect is to authenticate the sender automatically when the sender goes to a VAS domain. In this way, the sender's authentication information can be migrated from HAS's database to VAS' database. Because VAS and HAS trust each other and they have a share key for communication between them. So the customers' authentication information is secure. After the data migration, VAS can authenticate sender's identity without the sender's home domain when the sender wants to launch a conference call first. What worth mentioning is that VAS should delete visitors' authentication information after visitor leaves VAS domain to decrease rate of loss if VAS's database is compromised.

V. EFFICIENCY

The end-to-end protocol for conference call introduce in this paper has high efficiency. Hash function is used for generating checksum. If cells in message are simply concatenated and then encrypted under encryption algorithms, it brings great burden to the encryption algorithms like AES, DES etc. There are many different problems of hashing such as Dynamic hashing, Cryptographic hashing, Geometric hashing, Robust hashing, Bloom hash, String hashing[7]. Which algorithm is to be used in the protocol depends on the time and security requests from companies. Because of the variation of request and condition, simulation used for comparing efficiency on this protocol hasn't been done. That means there is no best algorithm, but the most suitable for a specific company depends on its business request.

VI. CONCLUSION

The end-to-end protocol for conference mobile call introduced in this paper can ensure the secure communication between end users in difference AS domains. It protects end user's private information and communication message. At the same time, authentication server can be protected from different kind of attacks. The analysis on the protocol makes a deeper exploration in the protocol from the aspects of confidentiality, authenticity, anonymity, freshness and preventing from DoS.

REFERENCES

[1] Y. Mu, and V. Varadharajan, "Design of Secure End-to-End Protocols for Mobile Systems, in Mobile Communications," Ed. L. Jose Encarnacao, and M. KanRabaey. Chapman & Hall, 1996, pp. 258-266.

[2] J. Stone, and M. Greenwald , "Performance of checksum and CRCs over real data," Networking, IEEE/ACM Transactions, vol 6, no. 12, pp 1628-1641, 1998.

[3] L. Harn, and G. Gong , "Digital signature with a subliminal channel", Computers and Digital Techniques, IEE Proceedings , vol 144, no. 6, pp. 387-289, 1997.

[4] M. Khan, A. Cheema, and A. Hasan, "Improved Nonce Construction Scheme for AES CCMP to Evade Initial Counter Prediction," Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2008. SNPD '08. Ninth ACIS International Conference, IEEE conference proceeding, pp 207-212, 2008 .

[5] R. Pries, W. Yu, X. Fu and W. A. Zhao, "New replay attack against anonymous communication networks," Communications, 2008. ICC '08. IEEE International Conference, IEEE conference proceeding, Beijing, China pp. 1578-1582, 2008.

[6] X. Liu, X. Yang, and Y. Lu, "To filter or to authorize: network-layer DoS defence against multimillion-node botnets," Proceedings of the ACM SIGCOMM 2008 conference on Data communication, ACM, pp 195-206, 2008.

[7] M. Singh and D. Garg "Choosing best hashing function Strategies and hash functions," Advance Computing Conference, 2009. IACC 2009. IEEE International, IEEE conference proceeding, Patiala, India, pp 20-55.