A Hash Tree Based Authentication Scheme in SIP Applications

Ke Xu, Xiaowei Ma, Chunyu Liu

Department of Computer Science and Technology, Tsinghua University, Beijing, China
Email: {xuke, maxiaowei, liuchunyu}@csnet1.cs.tsinghua.edu.cn

***Abstract*—Being one of the leading signaling protocols of VoIP applications, SIP protocol becomes popular in IP-based multimedia services, and securing SIP has become a priority. In this paper, we develop a novel authentication scheme which relies only on one way hash functions. In contrast to the computationally expensive asymmetric RSA signature scheme, our scheme is efficient in signing and verifying procedures. And hash tree is exploited to store and verify key information. Our scheme can be used in SIP entities which have less computation power and limited memory.**

Keywords—SIP; One Time Signature;Hash Tree;PKI

I. INTRODUCTION

Due to its simplicity and flexibility, SIP(Session Initiation Protocol) [1] has become more popular in IP-based multimedia applications. In the audio/video communications or online games, SIP is used as a signaling protocol to control and manage the sessions, including initializing a session between two or more users, modifying the attributes of an existing session or closing an ongoing session.

SIP has been applied in many scenarios and security of SIP based applications has become a priority [2]. How to authenticate the UA (User Agent) in a registration procedure? How to prevent attackers modifying the transmitted SIP messages maliciously? How to mitigate eavesdropping attacks？And how to prevent attackers sending fake SIP messages to close ongoing sessions? These are all the security problems worth considering in SIP applications.

Most of the attacks result from the fact that there is no efficient authentication scheme in SIP applications. Most existing solutions are based on PKI and use computational expensive RSA [3] signature scheme, and are inadequate for SIP applications which require high efficiency. We develop a novel lightweight authentication scheme in this paper. Being very efficient, it only based on one way hash function. The signing and verifying processes require hash function operation for only several times. And hash tree is used in our scheme to reduce the memory required to store the key information.

Our paper is organized as following: Section Ⅱ introduces the SIP application system and SIP session; Section Ⅲ gives an overview of related work; We give the details of our solution in Section Ⅳ and the evaluation in Section Ⅴ; At last, in Section Ⅵ, we conclude our work.

II. SIP SESSION AND RELATED ATTACKS

Fig.1 depicts a typical SIP application system and how a SIP agent initializes a SIP session and sets up communication with another SIP agent. SIP entities involved in this system include SIP agents and SIP proxies.

The plain SIP session model is vulnerable to many attacks. E.g. In registration process, attacker Trudy can send a fake REGISTER message, saying her address is the current contact address related to Alice's URI. When Bob wants to communicate with Alice, he will actually set up a session with Trudy, and all traffic sent to Alice will be forwarded to Trudy. This is called registration hijack attack. In another scenario, when Alice and Bob set up a session, Trudy can either eavesdrop the ongoing traffic or modify the content of the SIP messages without being trapped. Another attack Trudy can make is sending BYE messages to the participators in an ongoing session, and the session will be closed immaturely [4][5].

Most of these attacks result from the absence of an efficient authentication scheme in the plain SIP session model. The SIP agents and proxies can't identify each other, for they have no idea whether the messages they received are true or false, and also the SIP entities in the SIP message transmitting path can't tell whether the messages they received from last hop has been modified or not. When the SIP agents get a BYE message, they just tear up the ongoing session without verifying the originator of this message. To mitigate all these attacks, an authentication scheme is critically needed.

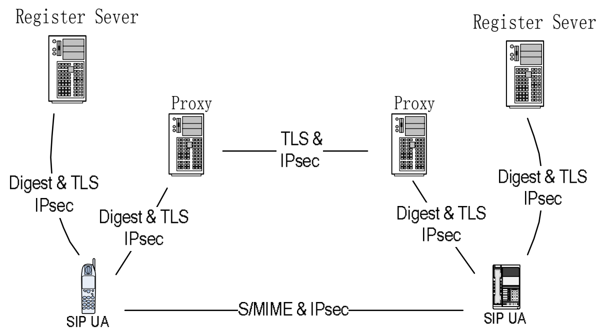

Fig.2. Existing security solutions in SIP applications

III. RELATED WORK

Fig.2 depicts the existing security solutions applied in SIP applications [5][7][8]. Most of these solutions are based on traditional security protocols, which can be used in a hop-by-hop style or an end-to-end style [5][6]. For example, the registration process can be protected by HTTP Digest authentication scheme or TLS scheme, IPsec in the network layer can also be used. A SIP client can set up a secure session with another client using S/MIME or IPsec protocols in an end-to-end way [9].

One of the disadvantages of these solutions is that they are based on PKI. It's difficult to find a globally trusted organization to be the root CA (Certification Authority), and it will also take labor to deal with the certificate-related issues. Another important reason for these schemes to be impractical in SIP applications is that they are computationally expensive. RSA signature signing operation is 3 to 4 orders of magnitude slower than hash function operation, while verifying operation is 2 to 3 orders of magnitude slower than hash function operation [15]. In SIP based VoIP or video conference systems, real time communication is important and this requires a more efficient authentication scheme rather than a RSA-based one.

Lamport raised the idea of replacing the computationally expensive asymmetric signature scheme with one time signature scheme [10]. In his original model, when the message sender, say Alice, wants to sign a one-bit message, she will choose two random values x_0 and x_1 to be her private keys, and their hash values $y_0=H(x_o)$ and $y_1=H(x_1)$ will be published as her public keys. Here $H(x)$ denotes a one way hash function. If the one-bit message is '0', then Alice will send x_0 to message receiver Bob as her signature, or she will send x_1. Bob will check the hash value of the received signature to authenticate the received message. In the original scheme, when sending a multi-bit message, Alice has to choose two random values for each bit and sets the hash value of these random values as her public keys.

In Lamport's scheme, storing public key will take abundant of memory, and the signature size is big. To address these disadvantages, Merkle[11][12] made some improvements.The differences in his proposal are that sender Alice will attach redundant bits at the end of the message to record the number of bits which are '1' in the message, and then choose only one random value for each bit in the message (including the redundant bits). The corresponding random values of those '1' bits then will be sent as the signature. Receiver Bob will check whether the hash values of the elements in the signature equal the corresponding public key values. To reduce the memory request for storing the public keys, Merkle proposed using a hash tree to authenticate the received public keys [16].

Reyzin [13] developed their HORS scheme to improve the signing and verifying efficiency in one time signature scheme. In their scheme, Alice will choose *n* random values as her private keys, and their hash values as her public keys. When signing a message *m*, Alice will compute *H(m)* first, and split the result into several parts, then interpret each part as an integer number. And the private key elements indexed by these integers will be chosen to constitute the signature. Bob will also perform the message's hash value computation and encoding processes, and then check the hash values of the elements in the signature to authenticate the message. This scheme was applied in secure routing by Hu et al [14]. Seys [15] evaluated the efficiency of different one time signature schemes and their applications in ad hoc network settings.

IV. LIGHTWEIGHT AUTHENTICATION SCHEME

A. *Authentication Framework*

Fig.3 shows the lightweight authentication framework developed in this paper, which is different from the PKI based solutions from the following aspects:

1) Instead of getting the public key and private key from the CA, the SIP clients in this system will generate their own keys in the one-time-signature way as we described in the former part. It means each SIP client will generate random values as its private key and their corresponding hash value as the public key.

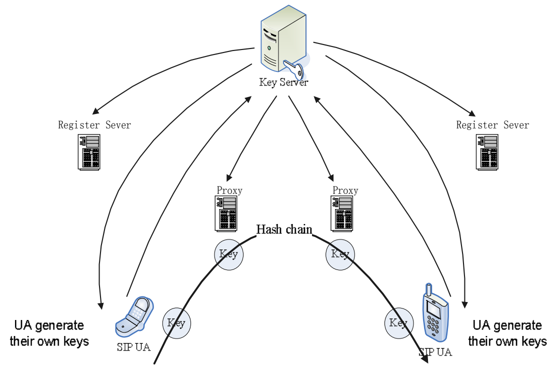

Fig.3. Lightweight authentication framework

2) Each SIP client will construct a hash tree based on the key values it generated. Fig.4 gives an example of an 8-leaf hash tree. In this tree, each leaf in the bottom layer represents a public key of the SIP client. Each node in the upper layer will be the hash value of each leaf node, and each node in the third layer will be the hash value of the concatenation result by its right child node and left child node. Thus we will get the upper layers and finally the root value of this tree. A key server is added into this framework, and the root value of the hash tree will be transmitted to the key server, which will distribute this root value to other SIP clients. The root value will be used in sequential steps to verify the SIP client's public keys. Fig.5 depicts the other nodes (the shadowed ones) needed to verify the public key represented by the leaf node v_2 [16]. Hash tree in this system will benefit the SIP clients to reduce the memory required to store other clients' key information.

3) The SIP entities in the SIP session path are connected by a hash chain, by using which, the SIP entities except the one which initiates this SIP session in this session path don't need to generate their own keys. They just use the keys transmitted by the last hop using this hash chain. In fact, this chain can spare many work for the proxies in the SIP

application system and mitigate many potential attacks made by these proxies.

4) The most important change in this framework is the one time signature scheme, which will be used to sign the sent SIP messages and verify the messages received by the SIP entities in this system. As we described in the former part, in this signature scheme, the signing and verifying processes just need do the hash function operation for several times, being much faster than the RSA scheme used in the existing solutions.

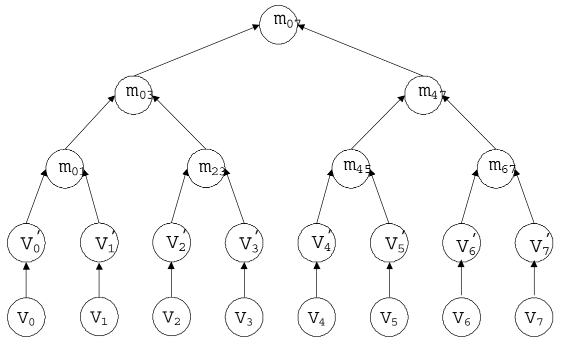

Fig.4. 8-leaf hash tree

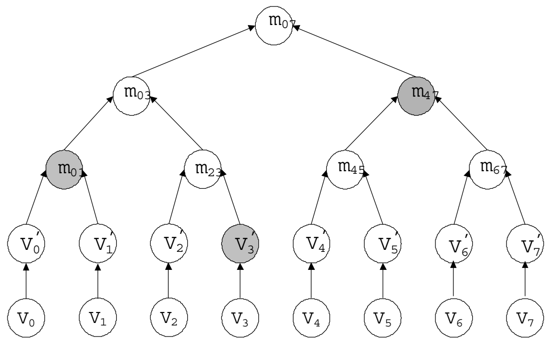

Fig.5. Authentication path in a hash tree

B. Authentication Scheme

The proposed authentication scheme in a SIP application involved the following steps:

1) Generation of private and public keys

SIP entity generates $m \cdot n$ random values using pseudo-random function. And the key used in the pseudo-random function will be the corresponding node value in the hash chain. Here *m* is the number of hash tree's leaf nodes, and *n* is the number of elements in a private key. The m private keys are:

$$SK_i = < s_{i0}, s_{i1}, ..., s_{i(n-1)} > \qquad 0 \le i \le m-1 \qquad (1)$$

SIP entity computes each element's hash value,

$$p_{i0} = H(s_{i0}) \qquad (2)$$

and get the *m* public keys as following:

$$PK_i = < p_{i0}, p_{i1}, ..., p_{i(n-1)} > \qquad 0 \le i \le m-1 \qquad (3)$$

2) Hash tree generation

SIP entity concatenates each element in a public key and computes the result's hash value. *m* public keys will result in *m* nodes $< v_0, v_1, ..., v_{m-1} >$;

SIP entity computes every node in the hash tree using the following equation:

$$\begin{cases} Leaf_i = v[i,i] = H(v_i) \\ v[i,j] = H(v[i,(i+j-1)/2]) \parallel v[i,(i+j+1)/2]; \end{cases} \qquad (4)$$

3) Signature signing

SIP entity computes the hash value of the SIP message $m, h = H(m)$;

SIP entity will select *k*elements to constitute the signature, so it encodes the first $k \cdot \log_2 n$ bits of *h*, and interprets every $\log_2 n$ bits as an integer $I_t\ (0 \le t \le k-1)$.

SIP entity chooses the elements in SK_i indexed by I_t $(0 \le t \le k-1)$; these elements constitute the signature.

SIP entity sends *m*, the signature, PK_i, the authentication path of PK_i and the next hop value in the hash chain to the next hop SIP entity.

4) Signature verifying

SIP entity checks the validity of PK_i, using PK_i's authentication path;

SIP entity computes the hash value of the received SIP message *m*, and gets $h' = H(m)$, then encodes the first $k \cdot \log_2 n$ bits of h', interpreting each $\log_2 n$ bits as an integer $I'_t (0 \le t \le k-1)$;

SIP entity computes the hash value of each element in the signature, and checks whether the results equal the PK_i's values at the corresponding positions indexed by $I'_t (0 \le t \le k-1)$. If the results conform, then the received message is valid.

V. Evaluation

A. Security Evaluation

1) Authentication and Integrity:

Our authentication scheme guarantees the verification of every participator's identity. In SIP registration, the register server can check the signature's validity of the REGISTER message, and this will mitigate the register hijack attack. In the ongoing session, the signatures attached to the SIP messages will prevent the personating attack, and the attackers can't send fake messages to close the ongoing sessions.

Also, the integrity of the SIP messages will be protected by our signature scheme. By checking the signatures of the SIP entities before itself, every SIP entity can decide whether the SIP messages have been modified maliciously in the transmitting path.

2) "One Time" problem:

Attackers can collect the signed signatures, so as the number of signatures signed increases, the security of the system will decrease, which is the shortcoming of one time signature scheme. In our scheme, the number of signatures every private key can sign is limited. We assume attacker Trudy eavesdrops the SIP session traffic and collects the signatures Alice signed. When the number of collected signatures rises to a threshold, Trudy will then be able to forge a signature and personate Alice. Assuming Alice signs r signatures using every private key, and each signature consists of k elements, the number of elements in a private key is n. Then when Trudy collects r signatures, the probability for Trudy to forge a signature is:

$$p = (r \cdot k / n)^k \qquad (5)$$

From (5), we define the system's security level L as following:

$$L = \lg(1/p) = k \cdot (\lg(n) - \lg(r) - \lg(k)) \qquad (6)$$

Fig.6 depicts the system's security levels when n=128 and n=256. For example, if the security of a system is tuned to 20, then the probability for an attacker to forge a signature is $1/10^{20}$. From Fig.6 we can see, when the number of elements in a private key increases, the system has a high security level, however, the cost in storing and communication increase. When the number of signatures signed by a private key increases, the system's security level will decrease, but the number of signatures every hash tree can sign increase, which is the product of the number of leaf nodes in the hash tree and the number of signatures signed using one private key.

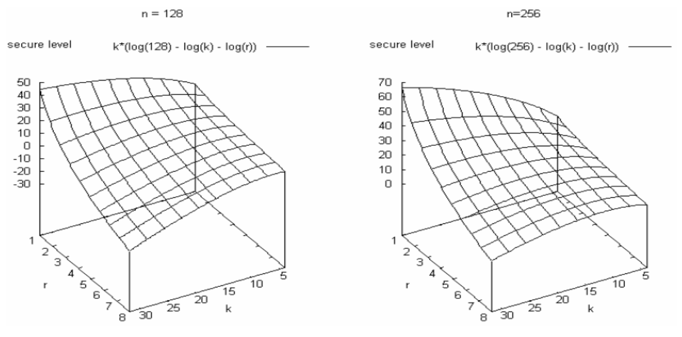

Fig.6. Security level when n=128 and n=256

B. Performance Evaluation

In our scheme, to authenticate the public keys of other SIP entities, every SIP entity is required to store the root values of the hash trees. Assuming we select SHA1 as the hash function in our scheme, and the hash chain's length is 10. The length of SHA1 function's output is 160 bits, so the size of the data required to authenticate the public keys of the entities in a hash chain is:

$$D = 10 \cdot 160 / 8 = 200Byte \approx 0.2KB \qquad (7)$$

In asymmetric RSA signature scheme, to authenticate other entities' public keys, a SIP entity should store their PKI certificates, e.g.X.509 certificates, each certificate is about 1KB, and the sum is much bigger than 0.2KB.

TABLE Ⅰ EFFIENCY OF SHA1 AND RSA OPERATIONS

SHA1	64 size	256 size	1024 size	8192 size
	387.12K	272.1K	124.72K	20.563K
RSA	512 S/s	512 V/s	1024 S/s	1024 V/s
	761.9	8344.0	146.6	2691.4

Table.□depicts the efficiency of SHA1 function and RSA signature scheme operations. These results are obtained from a Pentium 4 running at 1.7 GHz and having 256 MB memory. Table.□ gives the number of SHA1 hash operations per second for different size blocks (64 size means 64 size block data), and the number of RSA signature signing and verifying operations per second (S/s and V/s means signing per second and verifying per second respectively). We can see in this table that SHA1 hash operation is 2 to 3 orders of magnitude faster than RSA verifying operation and 3 to 4 orders of magnitude faster than RSA signing. In our authentication scheme, the signing process just need 1 hash operation, and the verifying operation need $1+k$ hash operations only, in which 1 denotes computing hash value of the SIP message and k represents computing the hash values of the k elements in the signature. For k will be a number between 10 and 100, our scheme is about 1 to 2 orders of magnitude faster than RSA scheme theoretically.

The communication overhead will grow because besides the SIP message, each SIP entity sends the signature, public key, the authentication path corresponding to the public key and the chain node value of next hop to the next hop entity. From Fig.5 we can see the size of authentication path equals the height of the hash tree. So when the signature consists of k elements and the public key consists of n ones, and assuming SHA1 is the hash function in our scheme, the communication overhead will be:

$$C = (k + n + \log_2 n + 1) \cdot 160 / (8 \cdot 10^3) KB \qquad (8)$$

When we set k=16 and n=128, we will get C=2.97KB, which is bigger than the 1KB cost by transmitting the PKI certificate and the signature in RSA signature scheme.

In Fig.7 and Fig.8, we show the experiment results of our scheme. The open source SIP stack Osip[17] was used to implement our scheme. We built the SIP client and proxy based on this stack, and added the routines to support our authentication scheme. The end clients and proxies all ran on 256M, 1.7GHz Pentium 4 boxes. For cryptographic operations, we used libcrypto, which comes from the OpenSSL [18] project.

To compare our scheme with the existing authentication schemes, we also implemented the PSK (Pre-shared Key) scheme and RSA scheme in our experiments. We measured the SIP session setup delay and bandwidth usage in each scheme. In order to measure the effect made by both two SIP clients, we defined the session setup delay as the interval between dialing the Call button and receiving the first 2xx message.

From Fig.7 we can see PSK scheme has the least effect on the session setup delay, for this scheme doesn't involve much computation, clients just need to deal with the challenge string and challenge response. RSA scheme brings much higher session setup delay, for the signing and verifying processes take long time. As our analysis above, our scheme should be 1 to 2 orders faster than RSA scheme, the experiment results show it's about 10 times faster, for the encoding, decoding processes and the key elements selecting process consume some time also.

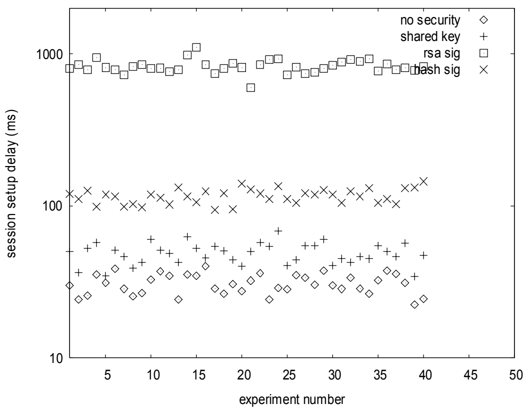

Fig.7. Session setup delay for different security schemes

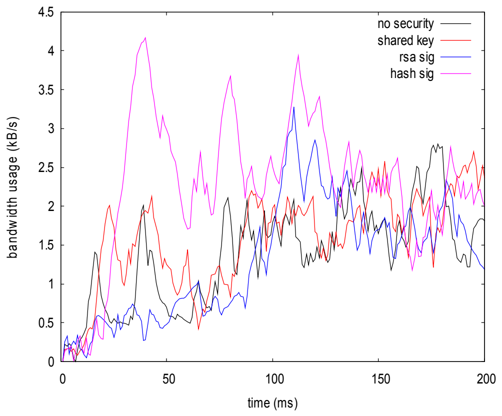

Fig.8. Bandwidth usage for different security schemes

Fig.8 shows the different bandwidth usage of the three schemes. In the experiment, we only consider the signaling session phase, the media session phase is not included. The results show that in fact all the schemes don't bring much effect on bandwidth usage to the SIP application system. PSK scheme's bandwidth usage effect is almost the same, as we don't use any security scheme. Our scheme has the largest bandwidth usage, because signature, public key, its verifying path values and the corresponding value in the hash chain should all be transmitted in our scheme. But after all, the total bandwidth usage is under 5 KB/s in our scheme, which is acceptable in the scenarios where SIP protocol in used.

VI. CONCLUSION

In this paper, we first list the potential attacks in SIP applications, and analyze the disadvantages of the existing PKI based authentication schemes, whose inefficiency in storing the public keys and the asymmetric primitives' expensive computation cost make it difficult to deploy them in a real world scenario. We develop our authentication scheme based on computationally efficient hash function, and each SIP entity in the SIP message transmitting path can sign a SIP message using 1 hash operation and verifying a SIP message in 1+*k* ones. The key information needed by a SIP entity to verify other SIP entities is greatly reduced by our scheme also. Experiment results show that our scheme's session setup delay is 10 times lower than RSA based scheme, and the communication overhead of our scheme is acceptable. Our scheme will be suitable to be applied in the SIP devices which have limited computation power and the SIP applications, in which security and efficiency are important.

REFERENCES

[1] J. Rosenberg, H. Schulzrinne and G. Camanilo, "SIP: Session Initiation Protocol," Internet RFC3261, 2002.

[2] D. Richard Kuhn, J. Walsh Thomas, Fries Steffen. Security Considerations for Voice Over IP Systems. NIST SP 800-58. 2005

[3] R. L. Rivest, A. Shamir and L .M .Adleman, "A method for obtaining digital signatures and public-key cryptosystems," Communications of the ACM, vol.21, pp. 120-126, 1978.

[4] M. Collier. Basic Vulnerability for SIP Security. 2005

[5] A. Steffen, D. Kaufmann and A. Stricker, "SIP Security," Security Group, Zurcher Hochschule Winterthur, 2004..

[6] S. Stefano, V. Luca, P. Donald. SIP security issues: The SIP authentication procedure and its processing load. IEEE Network, 2002, 38-44

[7] DF. Si, XH. Han, Q. Long, AM. Pan. A survey on the core technique and research development in SIP standard. Journal of Software, 2005, 16(2): 239-259.

[8] J. Arkko, V. Torvinen, G. Camarillo, A. Niemi, T. Haukka. Security mechanism agreement for the session initiation protocol (SIP). Internet RFC 3329, 2003.

[9] B. Ramsdell. Secure/Multipurpose Internet mail extension version 3 message specification. Internet RFC 2633, 1999

[10] L. Lamport, "Constructing Digital Signatures from a One-Way Function," Technical Report SRI-CSL-98, SRI International Computer Science Laboratory, October 1979.

[11] R. Merkle, "Protocols for Public Key Cryptosystems," In 1980 IEEE Symposium on Security and Privacy, 1980.

[12] R. Merkle, "A Digital Signature Based on a Conventional Encryption Function," In Advances in Cryptology-CRYPTO'87, pages 369-378, 1988.

[13] L. Reyzin and N. Reyzin, "Better than Biba: Short One-Time Signatures with Fast Signing and Verifying," In Information Security and Privacy-7th Australasian Conference (ACSIP 2002), July 2002..

[14] Yih-Chun. Hu, A. Perrig and M. Sirbu, "SPV: Secure Path Vector Routing for Securing BGP," In Proceedings of the ACM SIGCOMM Conference, pages 179-192, 2004..

[15] S. Seys, "Cryptographic Algorithms and Protocols for Security and Privacy in Ad Hoc Networks," PhD thesis, Katholieke Universiteit Leuven, 2006

[16] R.Merkle, "A certified digital signature," In Advances in Cryptology-CRYPTO 1989, vol.435 of lecture Notes in Computer Science, pp.218-238, Springer-Verlag, 1990.

[17] http://www.gnu.org/software/osip/

[18] http://www.openssl.org/