Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 467 – 475

ISSN 1896-7094

TLS handshake method based on SIP

Tadashi Kaji[1], Kazuyoshi Hoshino[2], Takahiro Fujishiro[1],
Osamu Takata[1], Akifumi Yato[1], Keisuke Takeuchi[3], Satoru Tezuka[1]

[1]Systems Development Laboratory, Hitachi, Ltd.,
890 Kashimada, Saiwai, Kawasaki, Kanagawa 212-8567 JAPAN
tadashi.kaji.ck@hitachi.com
[2]Network Systems Solutions Division, Hitachi, Ltd.,
890 Kashimada, Saiwai, Kawasaki, Kanagawa 212-8567 JAPAN
[3]Central Research Laboratory, Hitachi, Ltd.,
1-280 Higashi-Koigakubo, Kokubunji, Tokyo 185-8601, Japan

Abstract. As the result that there are many security problems around the Internet, various security measures are required to protect communications over the Internet. TLS is widely used to protect application layer protocol. However, TLS requires a handshake process to establish a TLS session. And the handshake process costs because of PKI based authentication and key calculation for each TLS session. This paper proposes the effective TLS handshake method based on SIP and TLS session resume. In this method, SIP server performs the PKI based authentication and key calculation for all TLS sessions on behalf of TLS server and TLS client.

1 Introduction

As the result that the Internet becomes a business infrastructure, many security problems are raised around the Internet. Against these problems, a lot of security measures are required to protect communications over the Internet. For this purpose, many security protocols and infrastructures are developed and standardized to protect communication data. TLS [1] and Ipsec [4] are well known secure communication protocols and both are standardized by IETF.

TLS is widely used to protect application layer protocol like HTTP, FTP and so on. Thus, many activities in regard to TLS have been done like improvement of TLS protocol to adopt certain environment, implementation technique to accelerate data transmission or session establishment, applications using TLS and so on. For example, there are some proposals to modify the specification of TLS to adopt mobile phone environment. Especially, WTLS [3] is standardized by OMA. And the mechanisms for caching handshake information on TLS clients were proposed to reduce the cost of handshake process [2].

On the other hand, new applications have been developed around the Internet continuously. (ex. IP telephony, P2P file sharing and so on.)

SIP [5] is known as a protocol to establish IP telephony (VoIP) sessions. However, SIP has not only a capability to establish voice or video sessions, but also a capability to establish and control various kinds of communications [6-9].

This paper proposes TLS handshake method based on SIP, which is the method for SIP to control TLS sessions. The proposed method works more effectively than current TLS handshake protocol, especially in the case that one TLS client establishes multiple TLS sessions with same TLS server or in the case that several entities communicate with each other by TLS.

2 Problems of TLS handshake

TLS performs "handshake process" before starting application data exchange between TLS server and TLS client.

During the handshake process, TLS server and client perform peer authentication, exchange a set of available cipher suites each other, select the cipher suite and calculate the session key to encrypt/decrypt application data. (Hereinafter, "security policy" or "SP" denotes a set of available cipher suites and "security association" or "SA" denotes a selected cipher suite and its session keys. "Negotiation of security association" denotes the selection of cipher suite and calculation of session keys)

Generally, the handshake process takes a lot of cost to verify the authenticity of the peer because the peer authentication of TLS is based on PKI. And also, key calculation in negotiation of security association is a costly process because public key encryption/decryption or DH key exchange methods are used.

The resuming session, known as "TLS session resume" shown in Figure 1 can improve the performance [12, 13].

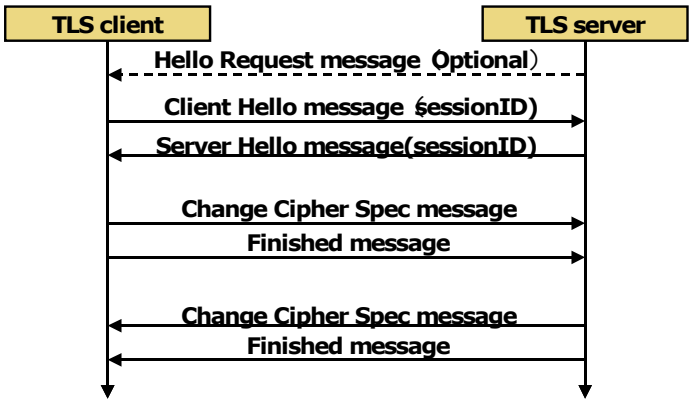

Fig. 1. Exchanged messages for resuming session

The overview of resuming session is as follows:

(1) TLS server sends Hello Request message to TLS client. (This is optional.)
(2) TLS client sends Client Hello message contains session ID, which is an ID of resuming session, to TLS server.
(3) TLS server sends Server Hello message contains same session ID to TLS client if TLS server finds any security association, which corresponds to the session ID indicated by Client Hello message.
(4) TLS client sends Change Cipher Spec message and Finished message.
(5) TLS server sends Change Cipher Spec message and Finished message if TLS server can decrypt Finished message, which is received from TLS client, successfully.
(6) TLS client and TLS server transmit application data if TLS client can decrypt Finished message, which is received from TLS server, successfully.

The problem of the TLS session resume is that the resuming session skips not only the peer authentication, but also both of the authorization of session and the negotiation of security association. However, in many cases, the authorization of the session and the negotiation of security association do not depend on peer entity, but

depends on certain session. Therefore the authorization and negotiation processes are required to perform by each session.

3 Proposed TLS handshake method based on SIP

This section describes the proposed TLS handshake method based on SIP and TLS session resume to resolve problems mentioned in section 2.

3.1 Overview of proposed method

The basic idea of the proposed TLS handshake method is to introduce a SIP server as the entity that performs costly process (peer authentication and security association generation) for TLS session on behalf of TLS client and TLS server to reduce these costs.

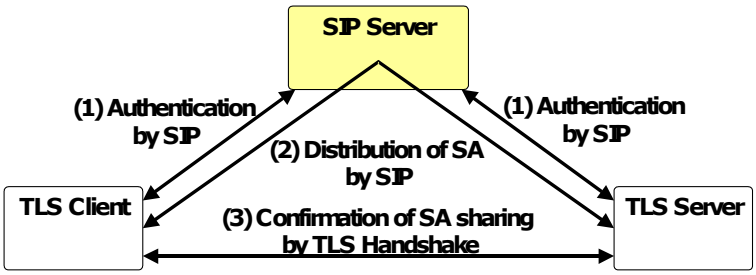

Fig. 2. Overview of proposed method

Figure 2 overviews the proposed TLS handshake method.

To establish a TLS session between TLS server and TLS client, the proposed method takes three phases: authentication, key distribution and SA confirmation.

At first, authentication phase is performed. SIP server authenticates TLS client and TLS server respectively. If authentication is success, SIP server establishes and keeps SIP sessions with TLS client and TLS server. And TLS client (TLS server) registers security policy to SIP server via the SIP session.

Next, key distribution phase is performed. SIP server generates a security association for TLS session between TLS client and TLS server and distributes it to TLS client and TLS server through the SIP sessions. This phase is triggered when TLS client tries to communicate with TLS server.

And then, SA confirmation phase is performed. As the resuming session procedure, TLS server and TLS client confirm if they share a security association. This phase follows the key distribution phase immediately. If this phase finishes successfully, TLS server and TLS client starts the application data transmission over the established TLS session.

It notes that the SIP messages exchanged in the proposed method will not contain SDP message, but contain XML to transmit security policy and security association.

3.2 Authentication phase

3.2.1 Message sequences and system behaviors

In authentication phase, TLS client (or TLS server) establish a TLS session with SIP server in the normal way at first. And then they exchange some SIP messages via the TLS session.

Figure 3 shows message sequence during authentication phase of TLS client.

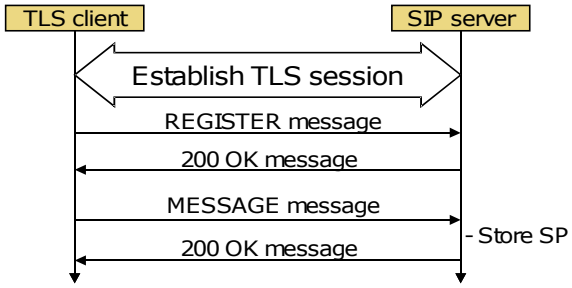

Fig. 3. Message sequence during authentication phase

(1) At first, TLS client establishes a TLS session with SIP server in the normal way. Mutual authentication is performed during the TLS session establishment.
(2) Then, TLS client sends a REGISTER message to SIP server.
(3) SIP server authenticates a sender of REGISTER message (i.e., TLS client), and returns a response message ("200 OK") if authentication is success.
(4) TLS client authenticates a sender of the response message (i.e., SIP server) when the message is received.
(5) And if authentication is success, MESSAGE message with security policy (i.e., a set of available cipher suites for TLS session) are sent to SIP server for key distribution phase.
(6) SIP server stores the security policy and returns a response message.

3.2.2 Messages

In authentication phase, four types of messages are exchanged: RESISGER message, RESISGER response message, MESSAGE message and MESSAGE response message.

(a) REGISTER message

REGISTER message defined in [5] is sent from TLS client (TLS server) to SIP server.

(b) REGISTER response message

REGISTER response message defined in [5] is sent from SIP server to the sender of REGISTER. If SIP server accepts the REGISGER message, "200 OK" message is sent as REGISTER response message.

(c) MESSAGE message

MESSAGE message defined in [10] is sent from TLS client (TLS server) to SIP server. The body of this message carries security policy like Figure 4.

```
<TLS>
  <Cipher Suite>TLS_RSA_WITH_AES_128_CBC_SHA</Cipher Suite>
  <Cipher Suite>TLS_RSA_WITH_3DES_EDE_CBC_SHA</Cipher Suite>
</TLS>
```

Fig. 4. Example of security policy in MESSAGE message during authentication phase

The security policy in Figure 4 states that TLS client (or TLS server) has two available cipher suites for TLS session: one is the cipher suite "TLS_RSA_WITH_AES_128_CBC_SHA" defined in [11] and another is the cipher suite "TLS_RSA_WITH_3DES_EDE_CBC_SHA" defined in [1].

(d) MESSAGE response message

MESSAGE response message defined in [10] is sent from SIP server to the sender of MESSAGE message. If SIP server accepts the MESSAGE message, "200 OK" message is sent as MESSAGE response message.

3.3 Key distribution phase

3.3.1 Message sequences and system behaviors

In key distribution phase, TLS client, TLS server and SIP server exchange SIP messages via SIP sessions which are established during authentication phase.

Figure 5 shows message sequence during key distribution phase as follows:

(1) In Figure 5, TLS client sends an INVITE message to SIP server to request TLS session establishment with TLS server at first.

(2) SIP server, which receives INVITE message from TLS client, generates a security association from security policies of TLS client and TLS server. If no security association is generated (ex. there is no available algorithms), SIP server returns 4xx response to TLS client. Otherwise, SIP server sends INVITE message, which contains the security association, to TLS server.

(3) TLS server that receives the INVITE message from SIP server stores the security association and returns a response message ("200 OK").

(4) SIP server that receives the response message from TLS server adds the security association to the response message and sends the response message with the security association to TLS client.

(5) If TLS client that receives "200 OK" response message, TLS client stores the security association. Otherwise, TLS client disposes of the message.

(6) Finally, (if TLS client stores the security association successfully,) TLS client sends ACK message to TLS server via SIP server.

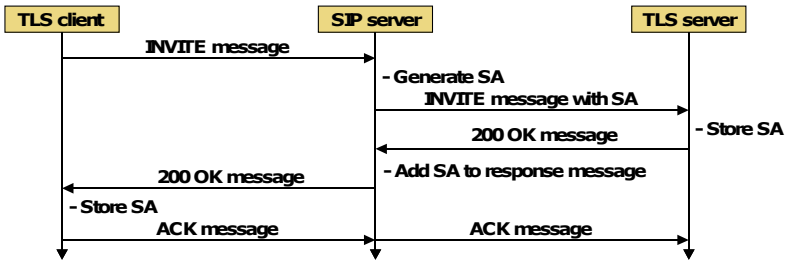

Fig. 5. Message sequence during key distribution phase

3.3.2 Messages

In key distribution phase, three types of messages are exchanged: INVITE message, INVITE response message and ACK message.

(a) INVITE message

INVITE message defined in [5] and is sent from TLS client to TLS server via SIP server. This message is used to notify that TLS client ask TLS session establishment to TLS server via SIP server. The body of this message is used to carry security association from SIP server to TLS server like Figure 6.

The security association in Figure 6 is a candidate of security association. The security association contains a session ID that is wedged between "<SessionID>" and "</SessionID>", a selected cipher suite that is

wedged between "<CipherSuite>" and "</CipherSuite>", a master secret value that is wedged between "<MasterSecret>" and "</MasterSecret>", the life time of this security association that is wedged between "<Lifetime>" and "</Lifetime>", TLS client information for this session, that is wedged between "<ClientInfo>" and "</ClientInfo>" include a random value for this TLS session, and TLS server information for this session, that is wedged between "<ServerInfo>" and "</ServerInfo>" include a random value for this TLS session. TLS client information and TLS server information may contain IP address and port number used for this TLS session.

```
<TLS>
  <SessionID>114454-132344-1475452@bar.hitachi.com</SessionID>
  <Cipher Suite>TLS_RSA_WITH_AES_128_CBC_SHA</Cipher Suite>
  <MasterSecret>MK8wuzC5jiMQEwTVH•••  MQAwADgA8w</MasterSecret>
  <Lifetime>36000</Lifetime>
  <Client Info>
    <random>ajCKMAEwTVJ0XgxUH2dL •••  5wZWcwMQA</random>
  </Client Info>
  <Server Info>
    <random>EGIxADgAdF4yAAhnazBnT•••  BiMQA3ALMA</random>
  </Server Info>
</TLS>
```

Fig. 6. Example of security association in INVITE message during key distribution phase

(b) INVITE response message format

INVITE response message defined in [5] is sent from TLS server to TLS client via SIP server. If TLS server accepts the INVITE request, the status of this response message is "200OK." The body of this message is also used to carry security association from SIP server to TLS client like Figure 6.

(c) ACK message format

ACK message defined in [5] is sent from TLS client to TLS server via SIP server to notify that TLS client receive the INVITE response message to TLS server.

3.4 SA confirmation phase

3.4.1 Message sequences and system behaviors

In SA confirmation phase, TLS client and TLS server exchange TLS Handshake messages shown as Figure 1. Namely, the following messages are exchanged between TLS client and TLS server.

(1) TLS client refers the security association shared at key distribution phase and sends Client Hello message with the session ID of security association to TLS server.

(2) TLS server, which receives Client Hello message, returns Server Hello message with same session ID to TLS client if TLS server can find the security association, which corresponds to the session ID.

(3) Then, TLS client sends Change Cipher Spec message and Finished message.

(4) TLS server sends Change Cipher Spec message and Finished message if TLS server can decrypt Finished message that received from TLS client

(5) TLS client and TLS server transmit application data if TLS client can decrypt Finished message that received from TLS server.

3.4.2 Messages

In SA confirmation phase, four types of messages are exchanged: Client Hello message, Server Hello message, Change Cipher Spec massage and Finished message. These messages are compliant with

RFC2246. And Client Hello message and Server Hello message contains the session ID of the security association shared at key distribution phase.

4. Evaluations

This section describes evaluations of the proposed TLS handshake method from the number of authentication and the cost of SA sharing.

4.1 The number of authentication

The proposed method can reduce a number of peer authentications than the traditional TLS handshake method. Because TLS server and TLS client are authenticated by SIP server in advance of the initiation of TLS session establishment and there is no authentication in key distribution phase. This has the advantage in the case that TLS client establishes multiple TLS sessions with same TLS server (case 1) or in the case that several entities communicate with each other by TLS (case 2).

In regard to case 1, it supposes that there are m types of applications are running on TLS client and TLS server. In this case, the traditional TLS handshake method needs 2m times of authentication processes because each TLS session needs 2 authentication processes and there are m TLS sessions totally. On the other hand, the proposed TLS handshake method needs only 4 times of authentication processes because this method does not depend on the number of TLS sessions between TLS client and TLS server, but depends on the number of SIP sessions. There are 2 SIP sessions and one SIP session needs 2 authentication processes (one is for SIP server to authenticate TLS client (TLS server) and another is for TLS client (TLS server) to authenticate SIP server).

In regard to case 2, it supposes that there are n terminals running on the application that communicates with n-1 isomorphic applications on other n-1 terminals. In this case, the traditional TLS needs n(n-1) times of authentication processes because there are n(n-1)/2 TLS sessions totally. On the other hand, the proposed method needs only 2n times of authentication processes because there are n SIP sessions totally.

Table 1 shows that the proposed method is fewer authentications than the traditional TLS handshake if the number of terminals or applications is 3 or more. Because the authentication process takes a lot of time and cost for cryptographic calculations or user's interactions (ex. entering PIN), the proposed method can establish TLS session more effectively than the traditional TLS handshake.

Table 1. Comparison of authentication between the proposed method and the traditional TLS

		Proposed method	**Traditional TLS**
Case1: TLS client establishes multiple TLS sessions with same TLS server	# of terminals	*2*	*2*
	# of applications	*m*	*m*
	# of TLS sessions	*m*	*m*
	# of SIP sessions	*2*	-
	# of authentications	***4***	***2m***
Case 2: Several entities communicate with each other by TLS	# of terminals	*n*	*n*
	# of applications	*1*	*1*
	# of TLS sessions	*n(n-1)*	*n(n-1)*
	# of SIP sessions	*n*	-
	# of authentications	***2n***	***2 n(n-1)***

4.2 The cost of SA sharing

The proposed method can also reduce the cost of SA sharing than the traditional TLS handshake method.

In the traditional TLS handshake method, TLS client and TLS server exchange redundant data because the negotiation of security association and the key exchange are integrated. For example, although Client Hello message contains 32 bytes random data, this is not mandatory in the case that RSA is selected as key exchange method. In addition, because the traditional TLS handshake method supposes that there is no secure session and no shared secret between TLS client and TLS server, TLS client and TLS server require to process public key cryptography to share the secret.

On the other hand, the proposed method requires for both TLS client and TLS server to register their security policies to SIP server in advance of the initiation of TLS session establishment. Therefore, SIP server can select available cipher suite and then generate the security association. In addition, because there are secure sessions between TLS client (TLS server) and SIP server in the proposed method, SIP server can transmit the security association to TLS client (TLS server) directly.

5. Conclusion

This paper proposed another TLS handshake method, which is based on SIP and TLS session resume. This proposed method divides the processing of TLS handshake into three phases: authentication, key distribution and SA confirmation, and the first two phases are performed over SIP and the last one phase is performed over TLS session resume. This method reduces the cost of peer authentication because SIP server authenticates TLS client (TLS server) on behalf of TLS server (TLS client). This method also reduces the cost of key distribution between TLS client and TLS server though the authorization of the session or the negotiation of security association is performed for each session.

Acknowledgments

This paper contains the result of “R&D of technologies for advanced network authentication - technologies for secure communication platform based on user authentication” (Ministry of Internal Affairs and Communications, Japan).

References

1. T. Dierks etc., RFC2246, *The TLS Protocol Version 1.0,* IETF (1999).
2. H. Shacham etc., *Client-Side Caching for TLS,* ACM Transactions on Information and System Security, Vol. **7**, No. 4, November 2004, pp. 553-575.
3. OMA, *Wireless Transport Layer Security,* OMA (2001).
4. S. Kent etc., *RFC2401, Security Architecture for the Internet Protocol,* IETF (1998).
5. J. Rosenberg, etc., *RFC3261, SIP: Session Initiation Protocol,* IETF (2002).
6. H. Schulzrinne, *Signaling for internet telephony services,* Proc. of Opensig'96 (1996).
7. N. Kausar etc., *An architecture of Conference Control Functions,* Proc. of Photonics East, Boston, Massachusetts, September 20-22, 1999.
8. E. M. Schooler, *Case study: multimedia conference control in a packet-switched teleconferencing system,* Journal of Internetworking: Research and Experience, vol. **4**, pp. 99-120, June 1993. ISI reprint series ISI/RS-93-359.

9. H. Schulzrinne etc., *Application-Layer Mobility using SIP,* Mobile Computing and Communications Review (MC2R), Volume **4**, Number 3, July 2000.
10. E. B. Campbell etc., *RFC3428 Session Initiation Protocol (SIP) Extension for Instant Messaging,* IETF (2002).
11. P. Chown, *RFC3268 Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS),* IETF (2002).
12. A. Goldberg, etc., *Secure web server performance dramatically improved by caching SSL session keys,* Proceedings of the Workshop on Internet Server Performance (1998).
13. C. Coarfa, etc., *Performance Analysis of TLS Web Servers,* Network and Distributed Systems Security Symposium '02 (2002).