SIPPING
Internet-Draft
Expires: April 19, 2004

K. Ono
S. Tachimoto
NTT Corporation

October 20, 2003

Requirements for End-to-middle Security for the Session Initiation Protocol (SIP)
draft-ietf-sipping-end2middle-security-reqs-00

Status of this Memo

Copyright Notice

Abstract

A SIP UA does not always trust all proxy servers in a request path to decide whether to inspect the message bodies and/or headers contained in a message. The UA might want to protect the message bodies and/or headers from proxy servers excluding the particular proxy that provides some features based on reading them. This situation requires a mechanism for securing information passed between the UA and an intermediary proxy, also called "end-to-middle security", which can work with end-to-end security. This document defines a set of requirements for a mechanism to achieve end-to-middle security.

Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [1].

Table of Contents

1. Introduction

The Session Initiation Protocol (SIP) [2] supports hop-by-hop security using TLS [3] and end-to-end security using S/MIME [4]. This assumes that a SIP UA trusts all proxy servers in a request path to decide whether or not to inspect the message bodies contained in a message.

However, there is a model where trusted and partially-trusted proxy servers are mixed along a message path. The partially-trusted proxy servers are only trusted in terms of the SIP routing. Hop-by-hop confidentiality services using TLS are not suitable for this model. End-to-end confidentiality services using S/MIME are also not suitable when the intermediaries provide features based on reading the message bodies and/or headers. This problem is described in Section 23 of [2].

One example of such features is firewall traversal. A firewall entity that supports SIP protocol or a midcom [5] agent co-located with a proxy server controls a firewall based on certain SDP attributes in a SIP transaction.

Another example is transcoding [6]. A transcoder related to a SIP proxy transfers coding based on certain SDP attributes in a SIP transaction or transfers text-to-speech based on a message body in the MESSAGE [7] method.

A third example is the archiving of instant messaging traffic, where the archiving function co-located with a proxy server logs the message bodies in the MESSAGE method. This feature is deployed for financial or health care applications.

In these cases, a UA might want to protect the message bodies and/or headers from proxy servers excluding the particular proxy that provides these features. Conversely, a proxy might want to view the message bodies and/or headers to provide these features. Such a proxy is not always the first hop for the UA. These situations require security between the UA and the intermediary proxy for the message bodies and/or message headers. We call this "end-to-middle security".

End-to-middle security consists of authentication, message integrity, and message confidentiality. As for authentication, HTTP digest authentication described in [2] is used for user-to-proxy and proxy-to-user authentication. The authenticating proxy is not limited to the first hop for the UA. Thus, HTTP digest authentication can be used for end-to-middle security. Digital signatures in a Public Key Infrastructure, that is S/MIME CMS [8] SignedData body with certificate, can also be used for authentication. As for message

integrity, S/MIME CMS SignedData body can be used. S/MIME CMS SignedData body is created with the original data and the originator's private key, and anyone can verify the integrity using the originator's public key and the certificate. Thus, S/MIME CMS SignedData body can be used for end-to-middle security at the same time as end-to-end security. However, proxy servers usually transfer SIP messages without interpreting the S/MIME bodies.

This document mainly discusses requirements for the message confidentiality and integrity of end-to-middle security. Proposed mechanisms are discussed in [9].

2. Problems with the Existing Situations

We describe here examples of models in which trusted and partially-trusted proxy servers are mixed along a message path. These situations demonstrate the reasons for requiring end-to-middle security.

The following example is that User#1 does not know the features or security policy on Proxy #1. User#1 sends an INVITE request including encrypted SDP for end-to-end security as shown in Figure 1. Proxy #1 may reject the request because of the impossibility of offering a firewall traversal feature. Or Proxy#1 may drop the encrypted data based on a security policy that prevents the sending of unknown data. Thus, there is a problem of discovering an intermediary's feature or security policy that may conflict with end-to-end confidentiality.

```
               Home network
               +---------------------+
               | +-----+     +-----+ |   +-----+     +-----+
   User#1------| | C   |-----| *   |-----| *   |-----| C   |-- User#2
               | +-----+     +-----+ |   +-----+     +-----+
               | UA#1        Proxy#1 |   Proxy#2     UA#2
               +---------------------+
```

C: Content that UA#1 allows the entity to inspect
*: Content that UA#1 prevent the entity from inspecting

Figure 1: Deployment example#1

In the second example, Proxy server#1 (Proxy#1) is the home proxy server of User#1 using UA#1. User#1 communicates with User#2 through Proxy#1 and Proxy#2 as shown in Figure 2.   UA#1 already knows the public key certificate of Proxy#1, and it allows Proxy#1 to inspect the message bodies in a request for some purpose. However, User#1 does not know whether Proxy#2 is trustworthy, and thus wants to protect the message bodies in the request. Thus, there is the problem of granting a trusted intermediary permission to inspect message bodies while preserving their confidentiality with respect to other intermediaries.

Even if UA#1's request message authorizes a selected proxy (Proxy#1) to see the message body, UA#1 is unable to authorize the same proxy to see the message body in the response from UA#2. Thus, there is the problem of designating and sharing a key that can be reused as a CEK for bidirectional exchanges of S/MIME-secured messages within SIP.

```
               Home network
               +---------------------+
               | +-----+     +-----+ |   +-----+     +-----+
   User#1------| | C   |-----| C   |-----| *   |-----| C   |-- User#2
               | +-----+     +-----+ |   +-----+     +-----+
               | UA#1        Proxy#1 |   Proxy#2     UA#2
               +---------------------+
```

C: Content that UA#1 needs to disclose
*: Content that UA#1 needs to protect

Figure 2: Deployment example#2

In the third example, User#1 connects UA#1 to a proxy server in a visited network, e.g. a hotspot service or a roaming service. Since User#1 wants to utilize certain home network services, UA#1 connects to a home proxy server, Proxy#1. However, UA#1 must connect to Proxy#1 via the proxy server of the visited network (Proxy A), because User#1 must follow the policy of that network. Proxy A may perform access control based on the destination addresses of calls. As shown in Figure 3, User#1 trusts Proxy A to route requests, but not to inspect the message bodies they contain. User#1 trusts Proxy#1 both to route requests and to inspect the message bodies for some purpose.

The same problems as in the second example exist.

```
              Visited network
             +---------------------+
             | +-----+     +-----+ |   +-----+     +-----+     +-----+
   User#1 -- | | C   |-----| *   |-----| C   |-----| *   |-----| C   |
             | +-----+     +-----+ |   +-----+     +-----+     +-----+
             | UA#1        Proxy A |   Proxy#1     Proxy#2      UA#2
             +---------------------+
```

C: Content that UA#1 needs to disclose
*: Content that UA#1 needs to protect

Figure 3: Deployment example#3

3. Requirements for a Solution

We describe here requirements for a solution. The requirements are mainly applied for the phase of a dialog creation or sending MESSAGE method.

3.1 Requirements from UA's Perspective

1. The solution MUST work even with SIP end-to-end encryption for confidentiality service enabled.

2. It SHOULD work even with SIP end-to-end integrity service enabled.

3. It SHOULD have little impact on the way of a UA handles messages with S/MIME bodies.

4. It SHOULD allow a UA to discover which proxy needs to view some data in a request/response for a certain feature.

   This requirement is for the case that the UA does not know the proxy or domain that provides the feature in advance.

5. It SHOULD allow a UA to discover what data in a request/response the proxy needs to view in order to provide the feature.

   This requirement is for the above case.

6. It MUST allow a UA to request selected proxy servers to view selected message bodies. The request itself SHOULD be secure.

7. It SHOULD allow a UA to request the UA on the opposite-side to impose the same type of data on the same proxy server. The request itself SHOULD be secure.

   It is not appropriate for the UA on the opposite-side to have knowledge of the public key certificate of the proxy server on the originating network. This last requirement can be modified into the following:

   + The solution SHOULD allow a UA to request the opposite-side UA to reuse a content-encryption-key in subsequent messages during a dialog.

   + It SHOULD allow a UA to request a selected proxy server to keep a content-encryption-key in a message during a dialog. The requests themselves SHOULD be secure.

8. It MAY allow a UA to notify the opposite-side UA which proxy needs to view some data in a request/response for the services.

9. It MAY allow a UA to notify the opposite-side UA what data the proxy is permitted to view in a request/response for the services.

   These last two requirements might be applied for a registration phase.

3.2 Requirements from Proxy's Perspective

1. It SHOULD have no impact on proxy servers that do not provide features based on S/MIME bodies in terms of handling the existing SIP headers.

2. It SHOULD have little impact on standardized mechanism of proxy servers that provide features based on S/MIME bodies.

   When a proxy server receives an S/MIME message, it should be able to quickly and easily determine the need to investigate the S/MIME body. This last requirement can be modified into the following:

   + It SHOULD allow proxy servers to quickly and easily determine whether to handle S/MIME bodies and, if so, how and which ones.

3. It SHOULD allow a proxy to notify a UA its own security policy for a request/response.

4. It SHOULD allow a proxy to notify a UA what data in a request/response is needed in order to provide a feature.

4. Security Considerations

This documents presents requirements including security viewpoints in Section 3.

5. IANA Considerations

This document requires no additional considerations.

6. Acknowledgments

Thanks to Rohan Mahy and Cullen Jennings for their initial support of this concept, and to Jon Peterson, Gonzalo Camarillo, and Sean Olson for their helpful comments.

References

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, BCP 14, March 1997.

[2] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M. and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[3] Allen, C. and T. Dierks, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[4] Ramsdell, B., "S/MIME Version 3 Message Specification", RFC 2633, June 1992.

[5] Srisuresh, P., Kuthan, J., Rosenberg, J., Brim, S., Molitor, A. and A. Rayhan, "Middlebox communication architecture and framework", RFC 3303, August 2002.

[6] Camarillo, G., "Framework for trasnscoding with the Session Initiation Protocol", draft-camarillo-sipping-transc-framework-00.txt (work in progress), August 2003.

[7] Campbell, Ed., B., Rosenberg, J., Schulzrinne, H., Huitema, C. and D. Gurle, "Session Initiation Protocol (SIP) Extension for Instant Messaging", RFC 3428, December 2002.

[8] Housley, R., "Cryptographic Message Syntax", RFC 2630, June 1999.

[9] Ono, K. and S. Tachimoto, "End-to-middle security in the Session Initiation Protocol(SIP)", draft-ono-sipping-end2middle-security-00 (work in progress), June 2003.

[10] Rosenberg, J., "Requirements for Session Policy for the Session Initiation Protocol (SIP)", draft-ietf-sipping-session-policy-req-00 (work in progress), June 2003.

[11] Farrell, S. and S. Turner, "Reuse of CMS Content Encryption Keys", RFC 3185, October 2001.

[12] Sparks, R., "Internet Media Type message/sipfrag", RFC 3420, November 2002.

[13] Crocker, D. and P. Overell, "Augmented BNF for Syntax

Specifications: ABNF", RFC 2234, November 1997.

Authors' Addresses

Kumiko Ono
Network Service Systems Laboratories
NTT Corporation
9-11, Midori-Cho 3-Chome
Musashino-shi, Tokyo 180-8585
Japan

EMail: ono.kumiko@lab.ntt.co.jp

Shinya Tachimoto
Network Service Systems Laboratories
NTT Corporation
9-11, Midori-Cho 3-Chome
Musashino-shi, Tokyo 180-8585
Japan

EMail: tachimoto.shinya@lab.ntt.co.jp

Intellectual Property Statement

Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.