End-to-middle Security in the Session Initiation Protocol (SIP)
draft-ono-sipping-end2middle-security-03

Abstract

Some services provided by intermediaries depend on the ability to inspect the message bodies in the Session Initiation Protocol (SIP). When sensitive information is included in these bodies, a SIP User Agent (UA) needs to protect it from all intermediaries except for certain selected ones. This document proposes a mechanism for securing information passed between an end user and a selected intermediary using S/MIME. It also proposes mechanisms for notifying
the UA that an intermediary needs to inspect an S/MIME-secured message body, and that the message body needs to be transmitted securely.

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 ............................................. 3
2. Generating S/MIME-secured message body ................. 3
   2.1 Generating S/MIME CMS EnvelopedData ............. 3
   2.2 Generating S/MIME CMS SignedData ........... 4
3. Indicating the Target Content ......................... 5
4. Notification of the Proxy Server’s Policies .......... 6
5. Behavior of UAs and Proxy Servers ................. 7
   5.1 UAC Behavior ........................................ 7
   5.2 UAS Behavior ....................................... 8
   5.3 Proxy Behavior .................................... 8
6. Proxy-Required-Body Header Field Use ................ 9
7. Examples ................................................. 10
   7.1 Example of Request for End-to-Middle Confidentiality . . 10
   7.2 Example of Request for End-to-Middle Integrity .... 12
8. Security Considerations ................................ 12
9. IANA Considerations .................................. 13
10. Acknowledgments .................................... 13
11. References ........................................... 13
   11.1 Normative References ............................ 13
   11.2 Informative References ......................... 14
       Authors’ Addresses .................................. 14
       Intellectual Property and Copyright Statements .... 15
1. Introduction

When a SIP [2] UA requires services provided by intermediaries that depend on the message bodies in request/response messages, end-to-end confidentiality will currently have to be disabled. This problem is pointed out in Section 23 of [2]. Since such an intermediary is not always adjacent to the UA, this situation requires security between the UA and the intermediary for the message bodies. We call this "end-to-middle security", where by "end" we mean a UA and by "middle" we mean a specific intermediary, typically a proxy server.

This document describes proposed mechanisms for providing data confidentiality and integrity for end-to-middle security to meet the requirements discussed in [3]. Since the major requirement is to have little impact on the standardized end-to-end security mechanisms, the proposed mechanisms are based on S/MIME [4]. The mechanisms consist of generating S/MIME-secured message body and indicating the target message body for a selected proxy server. In addition, it also includes mechanisms for notifying the UA that an intermediary needs to inspect an S/MIME-secured message body, and that the message body needs to be transmitted securely.

2. Generating S/MIME-secured message body

2.1 Generating S/MIME CMS EnvelopedData

For end-to-middle confidentiality, a UA MUST generate S/MIME CMS [5] EnvelopedData, whose recipient list is specified in the "recipientInfos" field. The structure of the S/MIME CMS EnvelopedData contains data encrypted with a content-encryption-key (CEK) and the CEK encrypted with different key-encryption-keys (KEKs), one for each recipient as specified in [5]. The KEKs are the public keys of each recipient or the shared keys between the UA and each recipient.

If the data is encrypted only for a selected proxy server, the recipient list contains only the proxy server. If there is encrypted data destined for multiple proxy servers, the recipient list contains the targeted proxy servers. If there is encrypted data destined for each proxy server, the recipient list of each piece of encrypted data contains the targeted proxy server. In order to concatenate multiple pieces of encrypted data, the UAC MUST generate a multipart MIME body.

Since proxy servers are prohibited from deleting any body, the encrypted data for the proxy server is transmitted to the user agent server (UAS) but the UAS will be unable to decrypt it. In order to avoid causing unnecessary error conditions in the UAS, the user agent...
client (UAC) MUST set the value "optional" in the handling parameter of the "Content-Disposition" MIME header for the message body.

If the multipart MIME body consists of encrypted MIME bodies with the value "optional", the "Content-Disposition" MIME header of the multipart MIME body MUST also contain the value "optional" in the handling parameter. The UAS SHOULD NOT try to decrypt encrypted data that has the value "optional".

If the multipart MIME body contains a body with the value "required" and another body with the value "optional", the multipart MIME body SHOULD have the value "required" in the handling parameter of the "Content-Disposition" MIME header.

If the encrypted data is meant to be shared with the UAS and selected proxy servers, the recipient list SHOULD be addressed to the UAS and the selected proxy servers. The UAC SHOULD set the value "required" in the handling parameter of the "Content-Disposition" MIME header for the message body. The UAS MUST try to decrypt the encrypted data that has the value "required" in the handling parameter. If the handling parameter is not set, the default behavior is the same as for setting the value "required", as specified in [2].

If a piece of encrypted data is destined for a selected proxy server and another piece of encrypted data for the UAS, the recipient of each piece of encrypted data is each respective entity. In this case, the UAC MUST generate a multipart MIME body to concatenate the two.

For example, a UA uses this mechanism when keying materials, such as keys for use by Secure RTP (SRTP), are included in the SDP[9]. One CMS EnvelopedData contains SDP that includes keying materials of an SRTP stream only for the UA. The other CMS EnvelopedData contains an SDP that does not include the keying materials of the SRTP stream only for a selected proxy server that needs to view SDP (i.e., for a firewall traversal service).

2.2 Generating S/MIME CMS SignedData

For end-to-end data integrity, a UA generates S/MIME CMS SignedData that can be verified by any entity that knows the public key of the UA. For end-to-middle data integrity, a UA MUST generate S/MIME CMS SignedData in the same way as for end-to-end data integrity.

Note: Even if the handling parameter of the signature of the whole message body is set to the value "optional", the UAS SHOULD validate the signature of the whole message body since the "Content-Disposition" might be modified by a malicious entity.
3. Indicating the Target Content

A UA needs a way to indicate content that they expect to be viewed by a proxy server, in order for the proxy server to easily determine whether to process MIME bodies and if so, which one. To meet this requirement, the UAs SHOULD set a label to indicate a selected proxy server and the target content. This document defines a new SIP header, "Proxy-Required-Body", for the label.

Open Issue: There are four options for the label: a new SIP header, a new parameter of an existing SIP header, a new MIME header, or a new parameter for an existing MIME header.

1) Using a new SIP header, "Proxy-Required-Body", to indicate the selected proxy server and the target body. The new SIP header contains one or more "content-id" parameter(s) for setting the "Content-ID" MIME header into the target body. When a message has no target body destined for a proxy server, it is easier for this option to determine that fact, as this only needs to look at the SIP header.

To protect the integrity of the label in the SIP headers, UAs need to generate an application/sipfrag body and attach a digital signature for the whole body to protect the data integrity of the label. Generating the application/sipfrag body would require an additional multipart MIME structure. The validation cost for integrity protection of these headers reduces the merit of using a new SIP header.

2) Using a new parameter of the Route header, "content-id". Since a proxy server is allowed to modify the Route header, there is no way to protect the data integrity of the label against other proxy servers along the signaling path.

3) Using a new MIME header, "Content-Target", as described in the last version of this draft. When a message has a target body destined for a proxy server, this shows better performance than using a SIP header in searching the target body. This requires only the inspection of the MIME header, while using a SIP header requires the inspection of the SIP header "Content-ID" MIME header. All that the UA needs to do to protect the integrity of the label is to attach a digital signature for the whole body. This is simpler than using a SIP header.

4) Using a new parameter of "Content-Disposition" MIME header, "required-entity". This MIME header is ambiguous to indicate the target entity.

When a UA labels the encrypted data, it SHOULD set the "Proxy-Required-Body" SIP header that contains the address of the server and "content-id" parameter indicating the S/MIME CMS EnvelopedData. When a UA labels the data with signature, it SHOULD
set the "Proxy-Required-Body" SIP header that contains the address of
the server and "content-id" parameter indicating the S/MIME CMS
SignedData. When a proxy server receives a message, it SHOULD
inspect the "Proxy-Required-Body" SIP headers.

A UA SHOULD generate a digital signature of the whole message body
including an application/sipfrag body that contains the new SIP
header in order to protect the integrity of the label. The proxy
server SHOULD validate the signature of the whole message body to
check the integrity of the indication, even when the "content-id"
parameter contained in the "Proxy-Required-Body" SIP header is not
set for the whole message body.

Open Issue: Should the target proxy server remove the label before
forwarding the message?

If a UA already knows that a selected proxy server needs to inspect a
message body, the UA MAY NOT set such label. The proxy server MAY
view a message body independently of the label, in order to inspect
the target body using "recipientinfo" in CMS EnvelopedData or
"Content-Type" MIME header.

4. Notification of the Proxy Server’s Policies

A notification mechanism for the proxy server’s policies is needed
when a UA does not know the policies of the proxy server in a
signaling path and the proxy server has its own policies for
providing some services. There are two ways in which a UA can learn
the policies of the proxy server. The UA MAY learn them by getting a
policy package from a policy server[10]. When a proxy server needs
to inspect the message body contained in the response, it needs to
learn the policies from a policy server before sending the response.
Alternatively, the UA MAY learn them by receiving a response with an
error code.

When the proxy server receives a request in which it cannot view the
message body that has to be read in order to proceed, the proxy
server MUST send a response with an error code. If the request
contains encrypted data, the error code SHOULD be 493
(Undecipherable), accompanied by a proxy’s public key certificate and
required Content-Type.

When the proxy server receives a request whose sending condition
cannot be accepted, the proxy server MUST also send a response with
an error code. If a digital signature is not attached to the request
and it required for an integrity check, the error code SHOULD be 403
(Forbidden) accompanied by a required Content-Type that is
"multipart/signed".
Open Issues:
Is it better to define a new error code for requiring a signature attachment? How should the error message indicate the Content-Type to which a signature needs to be attached? Can these Content-Types be nested such as "Content-Type: multipart/signed" for "Content-Type: application/sdp"? When proxy servers require both disclosure and an integrity check, how should it be described?

When the UAC receives one of the above error codes, it needs to authenticate the proxy server. Therefore, the error code SHOULD contain the digital signature of the proxy server.

In the worst case, this notification mechanism requires two messages for each proxy server in the signaling path to establish a session between the UAs. In addition, it requires validation procedures using the digital signatures for all proxy servers. Since this causes an increase in the delay before session establishment, it is recommended that a UA learns in advance the policies of as many proxy servers as they can.

5. Behavior of UAs and Proxy Servers

We describe here an example of the behavior of UAs and proxy servers in a model in which a proxy server that provides a logging service for instant messages exists in a message path as shown in Figure 1.

```
+-----+     +-----+     +-----+     +-----+
| C   |-----| C   |-----| *   |-----| C   |
|-----+     +-----+     +-----+     +-----+
UA #1      Proxy #1    Proxy #2     UA #2
w/Logging function
```

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

Figure 1: Deployment example

5.1 UAC Behavior

When a UAC sends a MESSAGE [11] request including encrypted message content for end-to-end and end-to-middle confidentiality, it MUST use S/MIME CMS EnvelopedData to encrypt them. In this example, UA #1 is assumed to know the services and the public key of Proxy #1. UA #1 MUST use S/MIME CMS EnvelopedData body for UA #2 and Proxy #1. UA #1 SHOULD specify the hostname of Proxy #1 and Content-ID of the S/MIME CMS EnvelopedData to be decrypted in the "Proxy-Required-Body" SIP header.
When the UAC sends a request and needs end-to-end and end-to-middle integrity for the message body, it MUST use S/MIME CMS SignedData to attach a digital signature. In this example, UA #1 MUST use the CMS SignedData body of the contents. UA #1 SHOULD specify the hostname of Proxy#1 and Content-ID of the CMS SignedData to be validated in the "Proxy-Required-Body" SIP header.

When the UAC sends multiple requests to the same UAS, the CEK reuse mechanism is beneficial in letting UAs efficiently encrypt/decrypt data. The CEK reuse mechanism is described in [6][7]. The UAC SHOULD use the "unprotectedAttrs" field to stipulate reuse of the CEK and indicate its identifier. When the UAC reuses the CEK in the previous request as the KEK, it generates CMS EnvelopedData with the type "KEKRecipientInfo" of "RecipientInfo" attribute.

5.2 UAS Behavior

When a UAS sends a response to the request with this mechanism, the use of the same type of S/MIME CMS data is recommended. For example, if the UAS receives an INVITE request in which the SDP is encrypted by using CMS EnvelopedData, the response is RECOMMENDED to be a "200 OK" containing the encrypted SDP based on the CMS EnvelopedData. In the above example, however, the response of the MESSAGE request does not need to use the same type of S/MIME CMS data since the response does not contain the message content.

In particular, when the CMS EnvelopedData body of the request contains the "unprotectedAttrs" attribute specifying reuse of the CEK, the UAS SHOULD keep the CEK with the identifier specified in the "unprotectedAttrs" attribute.

When the UAS receives a request that uses S/MIME, it decrypts and/or validates the S/MIME bodies as usual.

Even when the UAS receives a request without this mechanism, the UAS may need end-to-end and end-to-middle confidentiality of the message bodies and/or headers in the response. In this case, the UAS MUST use CMS EnvelopedData to encrypt them. When the UAS sends a response and needs end-to-end and end-to-middle integrity of the message bodies and/or headers, it MUST use CMS SignedData to attach a digital signature. This is the same way in which the UAC normally operates with this mechanism.

5.3 Proxy Behavior

When a proxy server supporting this mechanism receives a message, it MUST inspect the "Proxy-Required-Body" SIP header. If the SIP header includes the processing server’s own hostname, the proxy server MUST
inspect the specified body in the "content-id" parameter.

When the specified body is CMS EnvelopedData, the proxy server MUST inspect it and try to decrypt the "recipientInfos" field. If the proxy server fails to decrypt that, it SHOULD cancel the subsequent procedure and respond with a 493 (Undecipherable) response if it is a request, otherwise any existing dialog MAY be terminated. If the proxy server succeeds in this decryption, it MUST inspect the "unprotectedAttrs" field of the CMS EnvelopedData body. If the attribute gives the key's identifier, the proxy server MUST keep the CEK with its identifier until the lifetime of the CEK has expired. If it receives subsequent messages within the lifetime, it MUST try to decrypt the type "KEKRecipientInfo" of the "RecipientInfo" attribute by using this CEK.

When the specified content is CMS SignedData, the proxy server MUST inspect it and validate the digital signature. If the verification fails, the proxy server SHOULD reject the subsequent procedure and respond with a 403 (Forbidden) response if the message is a request, otherwise any existing dialog MAY be terminated.

When the proxy server forwards the request, it modifies the routing headers normally. It does not need to modify the message body.

If a proxy does not support this mechanism and receives a message with the "Proxy-Required-Body" SIP header, the proxy MUST ignore the header and operate as usual.

6. Proxy-Required-Body Header Field Use

The following syntax specification uses the augmented Backus-Naur Form (BNF) as described in RFC-2234 [8]. The new header "Proxy-Required-Body" is defined as a SIP header.

Proxy-Required-Body = "Proxy-Required-Body" HCOLON required-proxy SEMI target-body
required-proxy = host
target-body = cid-param *(COMMA cid-param)
cid-param = "cid" EQUAL content-id
content-id = LDQUOT dot-atom @ (dot-atom / host) RDQUOT
dot-atom = 1*( alphanum / \_ / \* / \@ / \# / \$ / \^ / & / \* / \( / \) / \[ / \] / \{ / \} )

Information about the use of headers in relation to SIP methods and proxy processing is summarized in Table 1.
Table 1: Summary of header field use

The "where" column gives the request and response types in which the header field can be used. The values in the "where" column are as follows:

* R: The header field may appear in requests
* 200-699: A numeral range indicates response codes with which the header field can be used.
* a: A proxy can add or concatenate the header field if it is not present.
* r: A proxy must be able to read the header field, so it cannot be encrypted.
* d: A proxy can delete a header field value.
* o: The header field is optional.

7. Examples

The following examples illustrate the use of the mechanism defined in the previous section.

7.1 Example of Request for End-to-Middle Confidentiality

In the following example, a UA needs the message content in a MESSAGE request to be confidential and it allows a selected proxy server to view the message content. It also needs to protect the label of the target content. In addition, it needs to reuse the CEK in the subsequent request messages. In the example encrypted message below, the text with the box of asterisks ("*") is encrypted:

MESSAGE alice@atlanta.example.com --> ss1.atlanta.example.com

MESSAGE sip:bob@biloxi.example.com SIP/2.0
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
Route: <sip:ss1.atlanta.example.com;lr>
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
To: Bob <sip:bob@biloxi.example.com>
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 MESSAGE
Date: Fri, 20 June 2003 13:02:03 GMT
Proxy-Required-Body: ssl.atlanta.example.com;cid=1234@atlanta.example.com
Content-Type: multipart/signed;protocol="application/pkcs7-signature";
micalg=sha1;boundary=boundary1

--boundary1

Content-Type: application/pkcs7-mime;smime-type=enveloped-data;
            name=smime.p7m
Content-Transfer-Encoding: binary
Content-ID: 1234@atlanta.example.com
Content-Disposition: attachment;filename=smime.p7m;handling=required
Content-Length: ...

******************************************************************
* (encryptedContentInfo)                                         *
* Content-Type: text/plain                                       *
* Content-Length: ...                                            *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* Hello.                                                         *
* This is confidential.                                          *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* (recipientInfos)                                               *
* RecipientInfo[0] for ssl.atlanta.example.com public key         *
* RecipientInfo[1] for Bob’s public key                          *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* (unprotectedAttrs)                                             *
* CEKReference                                                  *
******************************************************************

--boundary1--

Content-Type: application/pkcs7-signature; name=smime.p7s
Content-Transfer-Encoding: binary
Content-Disposition: attachment;
filename=smime.p7s;handling=required

[ binary data ]

--boundary1--
7.2 Example of Request for End-to-Middle Integrity

In the following example, a UA needs the integrity of the message content in a MESSAGE request to be validated by a selected proxy server before it views the message content. It also needs to protect the label of the target content.

MESSAGE alice@atlanta.example.com --> ss1.atlanta.example.com

MESSAGE sip:bob@biloxi.example.com SIP/2.0
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
Route: <sip:ss1.atlanta.example.com;lr>
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
To: Bob <sip:bob@biloxi.example.com>
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 MESSAGE
Date: Fri, 20 June 2003 13:02:03 GMT
Proxy-Required-Body: ss1.atlanta.example.com;cid=1234@atlanta.example.com
Content-Type: multipart/signed; protocol="application/pkcs7-signature";
micalg=sha1;boundary=boundary1
Content-Length: ...

--boundary1

Content-Type: text/plain
Content-Length: ...

Hello.
This is protected with the signature.

--boundary1--

Content-Type: application/pkcs7-signature; name=smime.p7s
Content-Transfer-Encoding: binary
Content-ID:1234@atlanta.example.com
Content-Disposition: attachment; filename=smime.p7s; handling=required

[ binary data ]

--boundary1--

8. Security Considerations

This proposal allows a UA to encrypt data for multiple recipients, such as multiple proxy servers, or the UAS and proxy servers. When a
proxy server or the UAS receives an encrypted data, the encrypted data may be decrypted and modified at another entity on the recipient list. If the encrypted data is meant to be shared with multiple recipients, the UAC SHOULD generate S/MIME CMS SignedData for each piece of data before encryption or for the whole message body.

9. IANA Considerations

This document requires a new "Proxy-Required-Body" SIP header.

10. Acknowledgments

Thanks to Rohan Mahy and Cullen Jennings for their initial support of this concept and to many people for useful comments, especially Jon Peterson, Jonathan Rosenberg, Eric Burger.

11. References

11.1 Normative References


11.2 Informative References


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

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Copyright Statement

Copyright (C) The Internet Society (2004). This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

Acknowledgment

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