Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 31, 2009

Lijun Liao, Joerg Schwenk
HGI, Ruhr-University Bochum
June 30, 2009

Header Protection for S/MIME
draft-liao-smimeheaderprotect-05

Status of This Memo

Copyright Notice

Abstract

In the current S/MIME Version 3.1 specification, the header protection is achieved by encoding the whole message as a message/rfc822 MIME media. Since this approach poses some practical problems, we propose to use signed attributes to implement a fully backward compatible S/MIME header protection scheme.

Table of Contents

1. Introduction

Mail message header fields as defined in [RFC5322] contain security critical information that is not protected cryptographically. The only exception is the address portion of the header fields From or Sender. Receiving agents MUST check that the message originator of a mail message matches an Internet mail address, if present, in the signer's certificate. Since there are no standards that cover this issue in S/MIME, each MUA behaves differently. For example, message originator is retrieved always from the "From" field in Outlook Express. Outlook does not check it at all. While in Thunderbird, message originator is retrieved from the "Sender" field if it exists; otherwise from the "From" field. A receiving agent SHOULD provide some explicit alternate processing of the message if the message originator does not match the signer's address, which may be to display a message that shows the recipient the addresses in the certificate or other certificate details [RFC3850]. Other header fields like "To", "Date", "Reply-To" and "Subject" remain totally unprotected.

In the solution described in this specification, a digest value is computed over the canonicalized version of some selected header fields. This technique resembles header protection in [RFC4871]. Then the digest value is included in a signed attribute field of a CMS signature.

This solution allows conforming clients to check if the selected header fields have been altered by simply re-computing the digest value. Non-conforming legacy clients will simply ignore that the signed attribute contains a digest value, and will only check the digest value computed over the message body according to S/MIME.

1.1. Terminology

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 [RFC2119].

1.2. Syntactic Notation

The following tokens are imported from other RFCs as noted. Those RFCs should be considered definitive.

The following tokens are imported from [RFC5322]:

o "field-name" (name of a header field)

Other tokens not defined herein are imported from [RFC4234]. These are intuitive primitives such as SP, HTAB, WSP, ALPHA, DIGIT, CRLF, etc.

1.3. Object Identifiers

The object identifiers defined in this specification is only for the experiment. When this memo moves to standards track within the IETF, it is intended that the IANA will maintain this registry.

1.4. Security Goals of Header Protection

The main security goal of mail message header protection is not to protect the whole RFC 822 header against manipulation, but to make it possible for the receiving client to detect whether the protected header fields have been changed.

1.5. Header Protection in S/MIME Version 3.1

S/MIME Version 3.1 [RFC3851] addresses the header protection by including all header fields as generated by the sending mail client, together with the body of the message, in a message/rfc822 MIME media, which can then be protected by S/MIME. It is up to the receiving client to decide how to present this message to the user.

This approach has, however, some limitations: If some of the message headers are changed during transport (e.g. when sent to a mailing list), this will either invalidate the whole message, or not be detected at all, depending on the receiving mail client's behavior.

This approach has the following disadvantages:

- o All inner header fields must also appear in the outer header (i.e., those headers must be presented doubly) so that the mail message is conform to [RFC5322] and the mail server and relay systems know how to send the mail message.

- o Only the inner header fields are protected, but not the outer header fields. As stated in [RFC3851], it is up to the receiving client to decide how to present the inner header along with the unprotected outer header. Usually the following header fields, if present, are shown in most clients: "From", "Sender", "To", "Cc", "Date", and "Subject". If the same header field is present in both inner and outer header, only the one in the inner header is presented. If a header field is only presented in the outer header, it will be also shown. Most mail messages do not contain the headers "Sender" and "Cc", hence one can add these header fields in the outer header to confuse the receivers.

- o It complicates the receiver to show the mail message. It is difficult to determine whether the message within the message/rfc822 wrapper is the top-level message or the complete message/rfc822 MIME entity is another encapsulated mail message.

1.6. Prototype Implementation

A prototype implementation of this memo is available in [FeLi08]. When this memo moves to standards track within the IETF, this section will be removed.

2. S/MIME Header Protection Entity

A S/MIME header protection entity contains names of header fields to be protected, the canonicalization algorithm, the digest algorithm and the corresponding digest value.

2.1. Fieldname List

The fieldname-list is a colon-separated list of header field names that identify the header fields presented to the digest algorithm; it is defined as follows:

```
   fieldname-list  = lowercase-field-name *(":" lowercase-field-name)
   lowercase-field-name = field-name in lowercase
```

The fieldname-list contains the complete list of header fields input to the hash algorithm. The order of the names in the list does not matter. The header fields specified by the list are presented to the hash algorithm in order of their appearance in the header block, from the top to the bottom. The field name is lowercase. The field may contain names of header fields that do not exist when digested. This is useful to prevent adding of undesired header fields. The fieldname-list is compared against the actual header field names in a case insensitive manner.

INFORMATIVE EXAMPLE:

Given a mail message as follows:

```
  Received: A <CRLF>
  Message-ID: B <CRLF>
  Date: C <CRLF>
  From: D <CRLF>
  To: E <CRLF>
  Cc: F <CRLF>
  Subject: G <CRLF>
  Comments: H <CRLF>

  Body
```

If the signer wishes to sign the header fields "From", "To", "Cc" and "Subject", then the fieldname-list may be:

```
  from:to:cc:subject
```

and the following header fields will be digested in the order:

```
  From: D <CRLF>
  To: E <CRLF>
  Cc: F <CRLF>
  Subject: G <CRLF>
```

If the signer wishes to protect additional header fields "Date", "Comments" and "Message-ID" then the fieldname-list may be:

```
  from:to:cc:subject:date:comments:message-id
```

and the following header fields will be digested in the order:

```
Message-ID: B <CRLF>
Date: C <CRLF>
From: D <CRLF>
To: E <CRLF>
Cc: F <CRLF>
Subject: G <CRLF>
Comments: H <CRLF>
```

Signers MUST NOT digest header fields that might have additional instances added later in the delivery process, since such header fields will change the input of the digest algorithm.

To prevent modifying header fields as far as possible, headers fields which are added before the signature creation and will not be modified after that SHOULD be included in the fieldname-list. Thus, a reasonable fieldname-list SHOULD contain at least the following content:

```
date:from:sender:reply-to:to:cc:message-id:in-reply-to:references:
subject:comments:keywords.
```

2.2. Canonicalization of Headers

Mail message, specially the mail message header, may be modified by some mail servers and relay systems. Some signers may demand that any modification of the mail message header result in a signature failure, while some other signers may accept modification of the header within the bounds of mail message standards such as [RFC5322].

To satisfy all requirements, two canonicalization algorithms are defined for each of the header: a "simple" algorithm stated in Section 3.4.1 of [RFC4871] that tolerates almost no modification and a "relaxed" algorithm stated in Section 3.4.2 of [RFC4871] that tolerates common modifications such as white-space replacement and header field line re-wrapping.

3. CMS Fields

3.1. CanonAlgorithmIdentifier

The CanonAlgorithmIdentifier type identifies a canonicalization algorithm. Examples include "simple" header canonicalization, and "relaxed" header canonicalization.

```
CanonAlgorithmIdentifier ::= AlgorithmIdentifier
```

AlgorithmIdentifier is defined in [RFC5280] as follows:

```
    AlgorithmIdentifier  ::=  SEQUENCE  {
      algorithm               OBJECT IDENTIFIER,
      parameters              ANY DEFINED BY algorithm OPTIONAL  }
```

The algorithm identifier is used to identify a canonicalization algorithm.

The "simple" canonicalization algorithm is identified by the following object:

```
  id-alg-simpleHeaderCanon OBJECT IDENTIFIER ::= {iso(1)
     member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
     smime(16) alg(3) 101}
```

The "relaxed" canonicalization algorithm is identified by the following object:

```
  id-alg-relaxedHeaderCanon OBJECT IDENTIFIER ::= {iso(1)
     member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
     smime(16) alg(3) 102}
```

For the canonicalization algorithms "simple" and "relaxed" the parameters field is NULL.

3.2. SMIME Header Protection

The smime-header-protection attribute type specifies the S/MIME header protection entity. It MUST be a signed attribute or an authenticated attribute; it MUST NOT be an unsigned attribute, unauthenticated attribute, or unprotected attribute in CMS signature.

The following object identifier identifies the smime-header-protection attribute:

```
  id-smimeHeaderProtection OBJECT IDENTIFIER :: = {iso(1)
     member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
     smime(16) aa(2) 101}
```

The attrValues of the smime-header-protection attribute contains only one value that has ASN.1 type SMIMEHeaderProtectionEntity:

```
SMIMEHeaderProtectionEntity ::= SEQUENCE {
    canonAlgorithm       CanonAlgorithmIdentifier,
    digestAlgorithm      DigestAlgorithmIdentifier,
    headerfieldNames     PrintableString,
    digest               Digest
```

```
   }
```

The canonAlgorithm field specifies the canonicalization algorithm. The digestAlgorithm field specifies the digest algorithm. The format of an headerfieldNames is a "headername-list" field specified in Section 2.1, which specifies the list of header field names. The digest field carries the the digest value.

4. Creating Signed S/MIME Messages with Header Protection

The signed S/MIME messages with header protection are created in the same way as in [RFC3851] except the followings:

- Before computing the digest value over the signedAttrs field, the smime-header-protection attribute MUST be prepared (see Section 4.1) and added to the signedAttrs field.

- All header fields that are protected MUST be prepared before the preparing the smime-header-protection.

4.1. Preparing an SMIME-Header-Protection Attribute

An smime-header-protection attribute is prepared as follows:

Step 1. Choose the canonicalization algorithm, the digest algorithm, and the list of names of message header fields to be digested. The digest algorithm SHOULD be the same as the digest algorithm in the SignerInfo to which the smime-header-protection attribute should be added.

Step 2. Retrieve the message header fields from the message according to the protected header fields from Step 1.

Step 3. Canonicalize the retrieved header fields from Step 2 according to the canonicalization algorithm.

Step 4. Compute the digest value over the canonicalization result in Step 3 according to the digest algorithm.

Step 5. Create an smime-header-protection attribute. Store the chosen canonicalization algorithm, the digest algorithm, and the list of names from Step 1 in the fields canonAlgorithm, digestAlgorithm, and headerfieldNames, respectively. Store the digest value from Step 4 in the the field digest.

5. Verifying Signed S/MIME Message with Header Protection

The signed S/MIME message with header protection are first verified in the same way as in [RFC3851], then the smime-header-protection attribute is verified as stated in Section 5.1.

5.1. Verifying an SMIME-Header-Protection Attribute

An smime-header-protection attribute is verified as follows:

Step 1. Retrieve the canonicalization algorithm, the digest algorithm, and the list of names of message header fields, and the digest value from the smime-header-protection attribute.

Step 2. Retrieve the message header fields from the message according to the list of protected header fields from Step 1.

Step 3. Canonicalize the retrieved header fields from Step 2 according to the canonicalization algorithm.

Step 4. Compute the digest value over the canonicalization result in Step 3 according to the digest algorithm.

Step 5. Compares the computed digest value from Step 4 and the stored one from Step 1. If both digest values are different, then the verification fails; otherwise the verification successes.

6. Security Considerations

All security considerations from [RFC3851] and [RFC3852] apply to applications that use procedures described in this document.

7. References

7.1 Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3850] Ramsdell, B. (Editor), "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Certificate Handling", RFC 3850, July 2004.

[RFC3851] Ramsdell, B. (Editor), " Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification", RFC 3851, July 2004.

[RFC3852] Housley, R., "Cryptographic Message Syntax (CMS), RFC 3852, July 2004.

[RFC4234] Crocker, D. (Editor), Overell, P., "Augmented BNF for Syntax Specifications: ABNF", RFC 4234, October 2005

[RFC4871] Allman, E. et. al., "DomainKeys Identified Mail (DKIM) Signatures", RFC 4871, May 2007

[RFC5280] Cooper, D., Santesson S., Farrell S., Boeyen S., Housley R., Polk W., "Internet X.509 Public Key Infrastructure, Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, April 2002.

[RFC5322] Resnick, P. (Editor), "Internet Message Format", RFC 5322, October 2008.7.2 Informative References

7.2 Informative References

[FeLi08] Feldmann, F., Liao, L., Prototype Implementation of Header Protection for S/MIME (this draft). URL: http://nds.hgi.rub.de/liao/works/headerprotect/index.html

A. ASN.1 Module

```
SMIMEHeaderProtectionService
     { iso(1) member-body(2) us(840) rsadsi(113549)
       pkcs(1) pkcs-9(9) smime(16) modules(0) shps(101) }

DEFINITIONS IMPLICIT TAGS ::=
BEGIN

IMPORTS
    -- Imports from RFC 5280
           AlgorithmIdentifier
              FROM PKIX1Explicit88
                   { iso(1) identified-organization(3) dod(6)
                     internet(1) security(5) mechanisms(5) pkix(7)
                     mod(0) pkix1-explicit(18) }

    -- Imports from RFC 3852
           DigestAlgorithmIdentifier, Digest
              FROM CryptographicMessageSyntax2004
                  { iso(1) member-body(2) us(840) rsadsi(113549)
                    pkcs(1) pkcs-9(9) smime(16) modules(0) cms-2004(24)}

CanonAlgorithmIdentifier ::= AlgorithmIdentifier

id-alg-simpleHeaderCanon OBJECT IDENTIFIER ::= {iso(1)
      member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
      smime(16) alg(3) 101}

id-alg-relaxedHeaderCanon OBJECT IDENTIFIER ::= {iso(1)
      member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
      smime(16) alg(3) 102}
```

```
id-smimeHeaderProtection OBJECT IDENTIFIER :: = {iso(1)
      member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
      smime(16) aa(2) 101}

SMIMEHeaderProtectionEntity ::= SEQUENCE {
      canonAlgorithm    CanonAlgorithmIdentifier,
      digestAlgorithm   DigestAlgorithmIdentifier,
      headerfieldNames  PrintableString,
                                    -- The format of a headerfieldNames
                                    -- is a "fieldname-list" field
                                    -- specified in Section 2.1.
      digest            Digest
}

END
```

B. Examples

B.1. SMIME-Header-Protection Attribute with "simple" and "SHA256"

This section contains an annotated hex dump of a 178 byte smime-header-protection attribute which is contained in the signedAttrs of a signature. The attribute contains the following information:

```
(a) the canocalization algorithm is "simple" header canonicalization;
(b) the digest algorithm is "SHA256";
(c) the list of header field names is
    "date:from:sender:reply-to:to:cc:message-id:in-reply-to:
     references:subject:comments:keywords";
(d) the digest value (32 hex).
```

```
  0 30  178: SEQUENCE {
  2 06   11:   OBJECT IDENTIFIER
           :     smime-header-protection {1 2 840 113549 1 9 16 2
           :       101}
 15 31  163:   SET {
 17 30  161:     SEQUENCE {
 19 30   15:       SEQUENCE {
 21 06   11:         OBJECT IDENTIFIER
           :           simple { 1 2 840 113549 1 9 16 3 101 }
 34 05    0:         NULL
           :       }
 36 30   13:       SEQUENCE {
 38 06    9:         OBJECT IDENTIFIER
           :           SHA256 { 2 16 840 1 101 3 4 2 1 }
 49 05    0:         NULL
           :       }
```

```
51 16   93:       PrintableString "date:from:sender:reply-to:to:cc:
                                   message-id:in-reply-to:references:
                                   subject:comments:keywords"

146 04   32:       OCTET STRING
           :         BA F1 D4 FD 95 EB 8B FA 55 F6 31 52 E7 86 50 53 AB
           :         6B 79 C7 93 F1 87 89 A1 11 66 A8 10 83 42 24
           :     }
           :   }
           : }
```

B.2. SMIME-Header-Protection Attribute with "relaxed" and "SHA1"

This section contains an annotated hex dump of a 163 byte smime-header-protection attribute which is contained in the signedAttrs of a signature. The attribute contains the following information:

(a) the canocalization algorithm is "relaxed" header
    canonicalization;
(b) the digest algorithm is "SHA1";
(c) the list of header field names is
    "date:from:sender:reply-to:to:cc:message-id:in-reply-to:
     references:subject:comments:keywords";
(d) the digest value (20 hex)

```
  0 30  163: SEQUENCE {
  2 06   11:   OBJECT IDENTIFIER
           :     smime-header-protection {1 2 840 113549 1 9 16 2
           :       101}
 15 31  147:   SET {
 17 30  145:     SEQUENCE {
 19 30   15:       SEQUENCE {
 21 06   11:         OBJECT IDENTIFIER
           :           relaxed { 1 2 840 113549 1 9 16 3 102 }
 34 05    0:         NULL
           :       }
 36 30    9:       SEQUENCE {
 38 06    5:         OBJECT IDENTIFIER
           :           SHA1 { 1 3 14 3 2 26 }
 45 05    0:         NULL
           :       }
 47 16   93:       PrintableString "date:from:sender:reply-to:to:cc:
                                   message-id:in-reply-to:references:
                                   subject:comments:keywords"
```

```
142 04   20:       OCTET STRING
           :         61 8D A3 CA 54 E2 F7 71 38 CD 76 A2 AA 2A 3D ED
                     79 EC 3A 86
           :
           :     }
           :   }
           : }
```

C. Authors' Addresses

Lijun Liao
Chair for Network and Data Security
Ruhr-University Bochum
44780 Bochum
Germany
Mail message: lijun.liao@nds.rub.de

Joerg Schwenk
Chair for Network and Data Security
Ruhr-University Bochum
44780 Bochum
Germany
Mail message: joerg.schwenk@nds.rub.de