Network Working Group
Request For Comments: 1847
Category: Standards Track

J. Galvin
S. Murphy
Trusted Information Systems
S. Crocker
CyberCash, Inc.
N. Freed
Innosoft International, Inc.

October 1995

Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted

Status of this Memo

Abstract

This document defines a framework within which security services may be applied to MIME body parts. MIME, an acronym for "Multipurpose Internet Mail Extensions", defines the format of the contents of Internet mail messages and provides for multi-part textual and non-textual message bodies. The new content types are subtypes of multipart: signed and encrypted. Each will contain two body parts: one for the protected data and one for the control information necessary to remove the protection. The type and contents of the control information body parts are determined by the value of the protocol parameter of the enclosing multipart/signed or multipart/encrypted content type, which is required to be present.

Table of Contents

1. Introduction

An Internet electronic mail message consists of two parts: the headers and the body. The headers form a collection of field/value pairs structured according to STD 11, RFC 822 [1], whilst the body, if structured, is defined according to MIME [2]. The basic MIME specification does not provide specific security protection.

This document defines a framework whereby security protection provided by other protocols may be used with MIME in a complementary fashion. By itself, it does not specify security protection. A MIME agent must include support for both the framework defined here and a mechanism to interact with a security protocol defined in a separate document. The resulting combined service provides security for single-part and multi-part textual and non-textual messages.

The framework is provided by defining two new security subtypes of the MIME multipart content type: signed and encrypted. In each of the security subtypes, there are exactly two related body parts: one for the protected data and one for the control information. The type and contents of the control information body parts are determined by the value of the protocol parameter of the enclosing multipart/signed or multipart/encrypted content type, which is required to be present. By registering new values for the required protocol parameter, the framework is easily extended to accommodate a variety of protocols.

A MIME agent that includes support for this framework will be able to recognize a security multipart body part and to identify its protected data and control information body parts. If the value of the protocol parameter is unrecognized the MIME agent will not be able to process the security multipart. However, a MIME agent may continue to process any other body parts that may be present.

2. Definition of Security Subtypes of Multipart

The multipart/signed content type specifies how to support authentication and integrity services via digital signature. The control information is carried in the second of the two required body parts.

The multipart/encrypted content type specifies how to support confidentiality via encryption. The control information is carried in the first of the two required body parts.

A three-step process is described for the origination and reception of the multipart/signed and multipart/encrypted contents. The details of the processing performed during each step is left to be specified by the security protocol being used.

2.1. Definition of Multipart/Signed

(1) MIME type name: multipart

(2) MIME subtype name: signed

(3) Required parameters: boundary, protocol, and micalg

(4) Optional parameters: none

(5) Security considerations: Must be treated as opaque while in transit

The multipart/signed content type contains exactly two body parts. The first body part is the body part over which the digital signature was created, including its MIME headers. The second body part contains the control information necessary to verify the digital signature. The first body part may contain any valid MIME content type, labeled accordingly. The second body part is labeled according to the value of the protocol parameter.

The attribute token for the protocol parameter is "protocol", i.e.,

```
 parameter := "protocol" "=" value
```

The value token is comprised of the type and sub-type tokens of the Content-Type: header of the second body part, i.e.,

```
 value := <"> type "/" subtype <">
```

where the type and subtype tokens are defined by the MIME [2] specification. The semantics of the protocol parameter are defined according to its value.

The attribute token for the micalg parameter is "micalg", i.e.,

```
 parameter := "micalg" "=" value
```

The Message Integrity Check (MIC) is the name given to the quantity computed over the body part with a message digest or hash function, in support of the digital signature service. Valid value tokens are defined by the specification for the value of the protocol parameter. The value may be a comma (",") separated list of tokens, indicating the use of multiple MIC algorithms. As a result, the comma (",") character is explicitly excluded from the list of characters that may be included in a token used as a value of the micalg parameter. If multiple MIC algorithms are specified, the purpose and use of the multiple algorithms is defined by the protocol. If the MIC algorithm

is also specified in the control information and the value there does not agree with the value in this parameter, it must be treated as an error.

NOTE: The presence of the micalg parameter on the multipart/signed content type header is explicitly intended to support one-pass processing. MIME implementations may locate the second body part by inputting the first body part and computing the specified MIC values until the boundary identifying the second body part is found.

The entire contents of the multipart/signed container must be treated as opaque while it is in transit from an originator to a recipient. Intermediate message transfer agents must not alter the content of a multipart/signed in any way, including, but not limited to, changing the content transfer encoding of the body part or any of its encapsulated body parts.

The signature in a multipart/signed only applies to the material that is actually within the multipart/signed object. In particular, it does not apply to any enclosing message material, nor does it apply to entities that are referenced (e.g. via a MIME message/external-body) by rather than included in the signed content.

When creating a multipart/signed body part, the following sequence of steps describes the processing necessary. It must be emphasized that these steps are descriptive, not prescriptive, and in no way impose restrictions or requirements on implementations of this specification.

(1) The content of the body part to be protected is prepared according to a local convention. The content is then transformed into a MIME body part in canonical MIME format, including an appropriate set of MIME headers.

    In addition, if the multipart/signed object is EVER to be transferred over the standard Internet SMTP infrastructure, the resulting MIME body is constrained to 7 bits -- that is, the use of material requiring either 8bit or binary content-transfer-encoding is NOT allowed. Such 8bit or binary material MUST be encoded using either the quoted-printable or base64 encodings.

    This requirement exists because it is not generally possible, given the current characteristics of Internet SMTP, for a message originator to guarantee that a message will travel only along paths capable of carrying 8bit or binary material.

SMTP clients normally have the option of either converting the message to eliminate the use of 8bit or binary encoding or returning a nondelivery notification to the originator. However, conversion is not viable in the case of signed objects since conversion would necessarily invalidate the signature. This leaves a nondelivery notification as the only available option, which is not acceptable.

(2) The body part (headers and content) to be digitally signed is prepared for signature according to the value of the protocol parameter. The MIME headers of the signed body part are included in the signature to protect the integrity of the MIME labeling of the data that is signed.

(3) The prepared body part is made available to the signature creation process according to a local convention. The signature creation process must make available to a MIME implementation two data streams: the control information necessary to verify the signature, which the MIME implementation will place in the second body part and label according to the value of the protocol parameter, and the digitally signed body part, which the MIME implementation will use as the first body part.

When receiving a multipart/signed body part, the following sequence of steps describes the processing necessary to verify the signature or signatures. It must be emphasized that these steps are descriptive, not prescriptive, and in no way impose restrictions or requirements on implementations of this specification.

(1) The first body part and the control information in the second body part must be prepared for the signature verification process according to the value of the protocol parameter.

(2) The prepared body parts must be made available to the signature verification process according to a local convention. The signature verification process must make available to the MIME implementation the result of the signature verification and the body part that was digitally signed.

> NOTE: The result of the signature verification is likely to include a testament of the success or failure of the verification. Also, in the usual case, the body part returned after signature verification will be the same as the body part that was received. We do not insist that this be the case to allow for protocols that may modify the body part during the signature processing.

(3) The result of the signature verification process is made available to the user and the MIME implementation continues processing with the verified body part, i.e., the body part returned by the signature verification process.

The following example is an illustration of a multipart/signed body part. It is necessarily incomplete since the control information is defined by the security protocol, which must be specified in a separate document.

```
Content-Type: multipart/signed; protocol="TYPE/STYPE";
        micalg="MICALG"; boundary="Signed Boundary"

--Signed Boundary
Content-Type: text/plain; charset="us-ascii"

This is some text to be signed although it could be
any type of data, labeled accordingly, of course.

--Signed Boundary
Content-Type: TYPE/STYPE

CONTROL INFORMATION for protocol "TYPE/STYPE" would be here

--Signed Boundary--
```

2.2. Definition of Multipart/Encrypted

(1) MIME type name: multipart

(2) MIME subtype name: encrypted

(3) Required parameters: boundary, protocol

(4) Optional parameters: none

(5) Security considerations: none

The multipart/encrypted content type contains exactly two body parts. The first body part contains the control information necessary to decrypt the data in the second body part and is labeled according to the value of the protocol parameter. The second body part contains the data which was encrypted and is always labeled application/octet-stream.

The attribute token for the protocol parameter is "protocol", i.e.,

```
parameter := "protocol" "=" value
```

The value token is comprised of the type and sub-type tokens of the Content-Type: header of the first body part, i.e.,

value := <"> type "/" subtype <">

where the type and subtype tokens are defined by the MIME [2] specification. The semantics of the protocol parameter are defined according to its value.

When creating a multipart/encrypted body part, the following sequence of steps describes the processing necessary. It must be emphasized that these steps are descriptive, not prescriptive, and in no way impose restrictions or requirements on implementations of this specification.

(1) The contents of the body part to be protected is prepared according to a local convention. The contents are then transformed into a MIME body part in canonical MIME format, including an appropriate set of MIME headers.

(2) The body part (headers and content) to be encrypted is prepared for encryption according to the value of the protocol parameter. The MIME headers of the encrypted body part are included in the encryption to protect from disclosure the MIME labeling of the data that is encrypted.

(3) The prepared body part is made available to the encryption process according to a local convention. The encryption process must make available to a MIME implementation two data streams: the control information necessary to decrypt the body part, which the MIME implementation will place in the first body part and label according to the value of the protocol parameter, and the encrypted body part, which the MIME implementation will place in the second body part and label application/octet-stream. Thus, when used in a multipart/encrypted, the application/octet-stream data is comprised of a nested MIME body part.

When receiving a multipart/encrypted body part, the following sequence of steps describes the processing necessary to decrypt the enclosed data. It must be emphasized that these steps are descriptive, not prescriptive, and in no way impose restrictions or requirements on implementations of this specification.

(1) The second body part and the control information in the first body part must be prepared for the decryption process according to the value of the protocol parameter.

(2) The prepared body parts must be made available to the decryption process according to a local convention. The decryption process must make available to the MIME implementation the result of the decryption and the decrypted form of the encrypted body part.

NOTE: The result of the decryption process is likely to include a testament of the success or failure of the decryption. Failure may be due to an inability to locate the proper decryption key or the proper recipient field, etc. Implementors should note that the data, if any, of a failed decryption process is pretty much guaranteed to be garbage.

(3) The result of the decryption process is made available to the user and the MIME implementation continues processing with the decrypted body part, i.e., the body part returned by the decryption process.

NOTE: A MIME implementation will not be able to display the received form of the second body part because the application of encryption will transform the body part. This transformation will not be described in the MIME headers (Content-Type: and Content-Transfer-Encoding:) but, rather, will be described in the content of the first body part. Therefore, an implementation should wait until the encryption has been removed before attempting to display the content.

The following example is an illustration of a multipart/encrypted body part. It is necessarily incomplete since the control information is defined by the security protocol, which must be specified in a separate document.

```
 Content-Type: multipart/encrypted; protocol="TYPE/STYPE";
        boundary="Encrypted Boundary"

 --Encrypted Boundary
 Content-Type: TYPE/STYPE

 CONTROL INFORMATION for protocol "TYPE/STYPE" would be here

 --Encrypted Boundary
 Content-Type: application/octet-stream

     Content-Type: text/plain; charset="us-ascii"
```

> All of this indented text, including the indented headers, would be unreadable since it would have been encrypted by the protocol "TYPE/STYPE". Also, this encrypted data could be any type of data, labeled accordingly, of course.

--Encrypted Boundary--

3. Definition of Control Information Content Types

This document defines a framework within which security services may be applied to MIME body parts. A minimal MIME implementation will be able to recognize multipart/signed and multipart/encrypted body parts and be able to identify the protected data and control information body parts within them.

Complete support for security services requires the MIME agent to recognize the value of the protocol parameter and to continue processing based on its value. The value of the protocol parameter is the same value used to label the content type of the control information.

The value of the protocol parameter and the resulting processing required must be specified in the document defining the security protocol used. That document must also precisely specify the contents of the control information body part.

4. Definition of Key Management Content Types

This specification recognizes that the complete specification of a MIME-based security protocol must include a mechanism for distributing the cryptographic material used in support of the security services. For example, a digital signature service implemented with asymmetric cryptography requires that a signer’s public key be available to the signee.

One possible mechanism for distributing cryptographic material is to define two additional body parts: one for the purpose of requesting cryptographic information and one for the purpose of returning the cryptographic information requested. The specification of a security protocol may include a definition of two such body parts or it may specify an alternate mechanism for the distribution of cryptographic material.

5. Security Considerations

This specification describes an enhancement to MIME to support signed and encrypted body parts. In that context this entire document is about security.

6. Acknowledgements

David H. Crocker suggested the use of a multipart structure for the MIME and PEM interaction.

7. References

[1] Crocker, D., "Standard for the Format of ARPA Internet Text Messages", STD 11, RFC 822, University of Delaware, August 1982.

[2] Borenstein, N., and N. Freed, "MIME (Multipurpose Internet Mail Extension) Part One: Mechanisms for Specifying and Describing the Format of Internet Message Bodies", RFC 1521, Bellcore and Innosoft, September 1993.

8. Authors' Addresses

Jim Galvin
Trusted Information Systems
3060 Washington Road
Glenwood, MD 21738

Phone: +1 301 854 6889
Fax: +1 301 854 5363
EMail: galvin@tis.com

Sandy Murphy
Trusted Information Systems
3060 Washington Road
Glenwood, MD 21738

Phone: +1 301 854 6889
Fax: +1 301 854 5363
EMail: sandy@tis.com

Steve Crocker
CyberCash, Inc.
2086 Hunters Crest Way
Vienna, VA 22181

Phone:: +1 703 620 1222
Fax: +1 703 391 2651
EMail: crocker@cybercash.com

Ned Freed
Innosoft International, Inc.
1050 East Garvey Avenue South
West Covina, CA 91790

Phone: +1 818 919 3600
Fax: +1 818 919 3614
EMail: ned@innosoft.com