Network Working Group
Request for Comments: 2015
Category: Standards Track

M. Elkins
The Aerospace Corporation

October 1996

MIME Security with Pretty Good Privacy (PGP)

Status of this Memo

Abstract

This document describes how Pretty Good Privacy (PGP) can be used to provide privacy and authentication using the Multipurpose Internet Mail Extensions (MIME) security content types described in RFC1847.

1. Introduction

Previous work on integrating PGP with MIME (including the since withdrawn application/pgp content type) has suffered from a number of problems, the most significant of which is the inability to recover signed message bodies without parsing data structures specific to PGP. This work makes use of the elegant solution proposed in RFC1847, which defines security multipart formats for MIME. The security multiparts clearly separate the signed message body from the signature, and have a number of other desirable properties. This document is styled after RFC 1848, which defines MIME Object Security Services (MOSS) for providing security and authentication.

This document defines three new content types for implementing security and privacy with PGP: application/pgp-encrypted, application/pgp-signature and application/pgp-keys.

1.1 Compliance

In order for an implementation to be compliant with this specification, is it absolutely necessary for it to obey all items labeled as MUST or REQUIRED.

2. PGP data formats

PGP can generate either ASCII armor (described in [3]) or 8-bit binary output when encrypting data, generating a digital signature, or extracting public key data. The ASCII armor output is the REQUIRED method for data transfer. This allows those users who do not have the means to interpret the formats described in this document to be able extract and use the PGP information in the message.

When the amount of data to be transmitted requires that it be sent in many parts, the MIME message/partial mechanism should be used rather than the multipart ASCII armor PGP format.

3. Content-Transfer-Encoding restrictions

Multipart/signed and multipart/encrypted are to be treated by agents as opaque, meaning that the data is not to be altered in any way [1]. However, many existing mail gateways will detect if the next hop does not support MIME or 8-bit data and perform conversion to either Quoted-Printable or Base64. This presents serious problems for multipart/signed, in particular, where the signature is invalidated when such an operation occurs. For this reason all data signed according to this protocol MUST be constrained to 7 bits (8- bit data should be encoded using either Quoted-Printable or Base64). Note that this also includes the case where a signed object is also encrypted (see section 6). This restriction will increase the likelihood that the signature will be valid upon receipt.

Data that is ONLY to be encrypted is allowed to contain 8-bit characters and therefore need not be converted to a 7-bit format.

> Implementor's note: It cannot be stressed enough that applications using this standard should follow MIME's suggestion that you "be conservative in what you generate, and liberal in what you accept." In this particular case it means it would be wise for an implementation to accept messages with any content-transfer-encoding, but restrict generation to the 7-bit format required by this memo. This will allow future compatibility in the event the Internet SMTP framework becomes 8-bit friendly.

4. PGP encrypted data

Before encryption with PGP, the data should be written in MIME canonical format (body and headers).

PGP encrypted data is denoted by the "multipart/encrypted" content type, described in [1], and MUST have a "protocol" parameter value of

"application/pgp-encrypted". Note that the value of the parameter MUST be enclosed in quotes.

The multipart/encrypted MUST consist of exactly two parts. The first MIME body part must have a content type of "application/pgp-encrypted". This body contains the control information. A message complying with this standard MUST contain a "Version: 1" field in this body. Since the PGP packet format contains all other information necessary for decrypting, no other information is required here.

The second MIME body part MUST contain the actual encrypted data. It must be labeled with a content type of "application/octet- stream".

Example message:

```
From: Michael Elkins <elkins@aero.org>
To: Michael Elkins <elkins@aero.org>
Mime-Version: 1.0
Content-Type: multipart/encrypted; boundary=foo;
   protocol="application/pgp-encrypted"

--foo
Content-Type: application/pgp-encrypted

Version: 1

--foo
Content-Type: application/octet-stream

-----BEGIN PGP MESSAGE-----
Version: 2.6.2

hIwDY32hYGCE8MkBA/wOu7d45aUxF4Q0RKJprD3v5Z9K1YcRJ2fve87lMlDlx4Oj
eW4GDdBfLbJE7VUpp13N19GL8e/AqbyyjHH4aS0YoTk10QQ9nnRvjY8nZL3MPXSZ
g9VGQxFeGqzykzmykU6A26MSMexR4ApeeON6xzZWfo+0yOqAq6lb46wsvldZ96YA
AABH78hyX7YX4uT1tNCWEIIBoqqvCeIMpp7UQ2IzBrXg6GtukS8NxbukLeamqVW3
1yt21DYOjuLzcMNe/JNsD9vDVCvOOG3OCi8=
=zzaA
-----END PGP MESSAGE-----

--foo--
```

5. PGP signed data

PGP signed messages are denoted by the "multipart/signed" content type, described in [1], with a "protocol" parameter which MUST have a value of "application/pgp-signature" (MUST be quoted). The "micalg"

parameter MUST have a value of "pgp-<hash-symbol>", where <hash-symbol> identifies the message integrity check (MIC) used to generate the signature. The currently defined values for <hash-symbol> are "md5" for the MD5 checksum, and "sha1" for the SHA.1 algorithm.

The multipart/signed body MUST consist of exactly two parts. The first part contains the signed data in MIME canonical format, including a set of appropriate content headers describing the data.

The second body MUST contain the PGP digital signature. It MUST be labeled with a content type of "application/pgp-signature".

When the PGP digital signature is generated:

(1) The data to be signed must first be converted to its type/subtype specific canonical form. For text/plain, this means conversion to an appropriate character set and conversion of line endings to the canonical <CR><LF> sequence.

(2) An appropriate Content-Transfer-Encoding is then applied. Each line of the encoded data MUST end with the canonical <CR><LF> sequence.

(3) MIME content headers are then added to the body, each ending with the canonical <CR><LF> sequence.

(4) As described in [1], the digital signature MUST be calculated over both the data to be signed and its set of content headers.

(5) The signature MUST be generated detached from the signed data so that the process does not alter the signed data in any way.

Example message:

```
From: Michael Elkins <elkins@aero.org>
To: Michael Elkins <elkins@aero.org>
Mime-Version: 1.0
Content-Type: multipart/signed; boundary=bar; micalg=pgp-md5;
protocol="application/pgp-signature"

--bar
& Content-Type: text/plain; charset=iso-8859-1
& Content-Transfer-Encoding: quoted-printable
&
& =A1Hola!
&
& Did you know that talking to yourself is a sign of senility?
&
```

```
  & It's generally a good idea to encode lines that begin with
  & From=20because some mail transport agents will insert a greater-
  & than (>) sign, thus invalidating the signature.
  &
  & Also, in some cases it might be desirable to encode any   =20
  &railing whitespace that occurs on lines in order to ensure  =20
  & that the message signature is not invalidated when passing =20
  & a gateway that modifies such whitespace (like BITNET). =20
  &
  & me

  --bar
  Content-Type: application/pgp-signature

 -----BEGIN PGP MESSAGE-----
Version: 2.6.2

iQCVAwUBMJrRF2N9oWBghPDJAQE9UQQAtl7LuRVndBjrk4EqYBIb3h5QXIX/LC//
jJV5bNvkZIGPIcEmI5iFd9boEgvpirHtIREEqLQRkYNoBActFBZmh9GC3C041WGq
uMbrbxc+nIs1TIKlA08rVi9ig/2Yh7LFrK5Ein57U/W72vgSxLhe/zhdfolT9Brn
HOxEa44b+EI=
=ndaj
-----END PGP MESSAGE-----

--bar--
```

The "&"s in the previous example indicate the portion of the data over which the signature was calculated.

Though not required, it is generally a good idea to use Quoted-Printable encoding in the first step (writing out the data to be signed in MIME canonical format) if any of the lines in the data begin with "From ", and encode the "F". This will avoid an MTA inserting a ">" in front of the line, thus invalidating the signature!

Upon receipt of a signed message, an application MUST:

(1) Convert line endings to the canonical <CR><LF> sequence before the signature can be verified. This is necessary since the local MTA may have converted to a local end of line convention.

(2) Pass both the signed data and its associated content headers along with the PGP signature to the signature verification service.

6. Encrypted and Signed Data

Sometimes it is desirable to both digitally sign and then encrypt a message to be sent. This protocol allows for two methods of accomplishing this task.

6.1 RFC1847 Encapsulation

[1], it is stated that the data should first be signed as a multipart/signature body, and then encrypted to form the final multipart/encrypted body, i.e.,

```
Content-Type: multipart/encrypted;
   protocol="application/pgp-encrypted"; boundary=foo

--foo
Content-Type: application/pgp-encrypted

Version: 1

--foo
Content-Type: application/octet-stream

-----BEGIN PGP MESSAGE-----
& Content-Type: multipart/signed; micalg=pgp-md5
&     protocol="application/pgp-signature"; boundary=bar
&
& --bar
& Content-Type: text/plain; charset=us-ascii
&
& This message was first signed, and then encrypted.
&
& --bar
& Content-Type: application/pgp-signature
&
& -----BEGIN PGP MESSAGE-----
& Version: 2.6.2
&
& iQCVAwUBMJrRF2N9oWBghPDJAQE9UQQAtl7LuRVndBjrk4EqYBIb3h5QXIX/LC//
& jJV5bNvkZIGPIcEmI5iFd9boEgvpirHtIREEqLQRkYNoBActFBZmh9GC3C041WGq
& uMbrbxc+nIs1TIKlA08rVi9ig/2Yh7LFrK5Ein57U/W72vgSxLhe/zhdfolT9Brn
& HOxEa44b+EI=
& =ndaj
& -----END PGP MESSAGE-----
&
& --bar--
-----END PGP MESSAGE-----
```

```
--foo--
```

(The text preceded by '&' indicates that it is really encrypted, but presented as text for clarity.)

6.2 Combined method

Versions 2.x of PGP also allow data to be signed and encrypted in one operation. This method is an acceptable shortcut, and has the benefit of less overhead. The resulting data should be formed as a "multipart/encrypted" object as described above.

Messages which are encrypted and signed in this combined fashion are REQUIRED to follow the same canonicalization rules as for multipart/signed objects.

It is explicitly allowed for an agent to decrypt a combined message and rewrite it as a multipart/signed object using the signature data embedded in the encrypted version.

7. Distribution of PGP public keys

Content-Type: application/pgp-keys
Required parameters: none
Optional parameters: none

This is the content type which should be used for relaying public key blocks.

8. Notes

PGP and Pretty Good Privacy are trademarks of Philip Zimmermann.

9. Security Considerations

Use of this protocol has the same security considerations as PGP, and is not known to either increase or decrease the security of messages using it; see [3] for more information.

10. Author's Address

Michael Elkins
P.O. Box 92957 - M1/102
Los Angeles, CA 90009-2957

Phone: +1 310 336 8040
Fax: +1 310 336 4402

References

[1] Galvin, J., Murphy, G., Crocker, S., and N. Freed, "Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted", RFC 1847, October 1995.

[2] Galvin, J., Murphy, G., Crocker, S., and N. Freed, "MIME Object Security Services", RFC 1848, October 1995.

[3] Atkins, D., Stallings, W., and P. Zimmermann, "PGP Message Exchange Formats", RFC 1991, August 1996.