MIME Security with Pretty Good Privacy (PGP)

Status of this Memo

This document is an Internet-Draft. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

To learn the current status of any Internet-Draft, please check the "1id-abstracts.txt" listing contained in the Internet-Drafts Shadow Directories on ftp.is.co.za (Africa), nic.nordu.net (Europe), munnari.oz.au (Pacific Rim), ds.internic.net (US East Coast), or ftp.isi.edu (US West Coast).

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 RFCXXXX (draft-ietf-pem-sigenc-03.txt).

1. Introduction

This document is based on RFCXXXX [1] which defines security mechanisms for MIME messages, and specifically addresses how PGP can be used within this framework in order to effect it. This document is styled after RFCXXXX [2], which defines MIME Object Security Services (MOSS) for providing security and authentication. This work was prompted by the fact that implementations of MOSS may not be available outside of the U.S. because of export restrictions. PGP, on the other hand, is widely available throughout the world (with a few notable exceptions), and is gaining support as the de-facto standard for electronic mail privacy. This document defines three new content types form implementing security and privacy with
PGP: application/pgp-encrypted, application/pgp-signature and application/pgp-keys.

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. Except where noted, the ascii armor output is the preferred 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.

Ascii armor output should NEVER be BASE64 or QUOTED-PRINTABLE encoded since it is already in 7 bit format.

When 8 bit to 7 bit conversion of PGP data is necessary, BASE64 encoding is the preferred method.

3. PGP encrypted data

Before encryption with PGP, the data should be written in MIME canonical format (body and headers). This MIME body is constrained to 7 bit format.

PGP encrypted data is denoted by the "multipart/encrypted" content type, described in [1], and REQUIRES a "protocol" parameter value of "application/pgp-encrypted". Note that the value of the parameter MUST be enclosed in quotes. The "multipart/encrypted" data is constrained to 7 bit format.

The multipart/encrypted MUST consist of exactly two parts. The first MIME body part must have a content type of "application/pgp-encrypted". The data portion of the body may be zero length or contain human readable information about the encrypted data.

The second MIME body part MUST contain the actual encrypted data. It must be labeled with a content type of "application/octet-stream", and is REQUIRED to be in 7 bit format.

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"
Content-Transfer-Encoding: 7bit
4. PGP signed data

Prior to generating the PGP signature, the following steps MUST be taken to ensure a valid signature:

(1) The data to be signed MUST be written in MIME canonical format, including the appropriate set of headers.

(2) The data MUST be constrained to 7 bit format.

(3) The data MUST use the canonical CRLF sequence to denote the end of a line, regardless of the local newline convention.

PGP signed messages are denoted by the "multipart/signed" content type, described in [1], and REQUIRE the "protocol" parameter to have a value of "application/pgp-signature" (MUST be quoted), and the "micalg" parameter to have a value of "pgp-md5".

The multipart/signed body MUST consist of exactly two parts, the first of which MUST be the signed data in the form described above.

The second body MUST contain the PGP digital signature. It MUST be labeled with a content type of "application/pgp-signature", and is REQUIRED to be in 7 bit format.

When the PGP digital signature is generated:

(1) As described in [1], the digital signature MUST be calculated over both the headers AND the data to be signed.

(2) 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>
Did you know that talking to yourself is a sign of senility?

Implementor’s note: when verifying a digital signature, it is VERY important that the message uses the canonical CRLF as the end line sequence. Depending on your local convention, the CRLF may have been converted to a different sequence during transmission, which could cause the signature verification to fail. Therefore, if the message has been altered, it MUST be written in the canonical form before the signature is checked in order to ensure verification.

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!

5. Encrypted and Signed Data

Sometimes it is desirable to both digitally sign and then encrypt a message to be sent. In [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, ie.,

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

```
--foo
Content-Type: application/pgp-encrypted
```
6. 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. Ascii armor output and 7BIT encoding are HIGHLY recommended.

7. Notes

   PGP and Pretty Good Privacy are trademarks of Philip Zimmermann.

References


[2] James Galvin, Gale Murphy, Steve Crocker, Ned Freed. MIME Object
Security Services. RFCXXXX (draft-ietf-pem-mime-08.txt), 1995