INTERNATIONAL MESSAGING
ASSOCIATES

WHITEPAPER SERIES

MIME – Technical Overview

Version 1.0

July 1998

Hong Kong Computer Center, 3/F
54-62 Lockhart Road
Wan Chai
HONG KONG
Tel: +852 2520-0300
Fax: +852 2648-5913

USA Support/Sales: +1 (408) 481-9985
USA Fax: +1 (888) 562-3561

The Peak Tower, 15/F
107 Alfaro Street
Salcedo Village, Makati City
PHILIPPINES
+63 (2) 811-3999
+63 (2) 811-3939

Email: info@ima.com
Website: http://www.ima.com

Please send all comments and suggestions to the Whitepaper Series to *doc@ima.com*

Introduction

The Multipurpose Internet Mail Extensions (MIME) standard is a set of specifications that allow message contents in different data formats to be transported via various transports, including electronic mail or e-mail and HTTP. It was developed to make up for the limitations of *RFC 822: Standard Format the Format of ARPA Internet Text Messages*, which does not specify mechanisms for mail containing audio, video, Asian language text, or even text in most European languages.

One of the most notable limitations of RFC 822-based mail systems is that they limit the contents of electronic mail messages to relatively short lines of 7bit US-ASCII. This forces users to convert any non-textual data that they may wish to send into 7bit bytes representable as printable US-ASCII characters before invoking a local mail UA (User Agent, a program with which human users send and receive mail).

The limitations of RFC 822 mail become even more apparent as gateways are designed to allow for the exchange of mail messages between RFC 822 hosts and other environments like X.400, Lotus cc:Mail, etc. Unlike RFC 822, X.400 and cc:Mail, for instance, specify mechanisms for the inclusion of non-textual material within electronic messages. For example, the current standards for mapping of X.400 messages to RFC 822 messages specify either that X.400 non-textual material must be converted to IA5Text format, or that they must be discarded, notifying the RFC 822 user that discarding has occurred. This is clearly undesirable, as information that a user may wish to receive is lost. Even though a UA may not be capable of dealing with the non-textual material, the user might have some mechanism external to the UA that can extract useful information from the material. Moreover, the present standards for mapping between X.400 and RFC 822 do not allow messages to be gatewayed back into an X.400 message handling system, where the non-textual information would definitely become useful again.

MIME provides a solution to these problems by redefining the format of messages to allow for textual message bodies in character sets other than US-ASCII. MIME also allows for an extensible set of different formats for non-textual message bodies, multi-part message bodies, and textual header information in character sets other than US-ASCII. It was designed to cope with many of the most bizarre variations of SMTP, UUCP, and other Procrustean mail transport protocols that like to slice, dice, and stretch the headers and bodies of e-mail messages.

Header Fields

The various headers used to describe the structure of MIME messages are specified in *RFC 2045: Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies*. In particular, it describes:

1. A MIME-Version header field – this uses a version number to declare a message to be conformant with MIME and allows mail-processing agents to distinguish between such messages and those generated by older or non-conformant software, which are presumed to lack such a field.

2. A Content–Type header field, generalized from *RFC 1049: A Content-Type Header Field for Internet Messages*, which can be used to specify media type and subtype of data in the body of a message and to fully specify the native representation (canonical form) of such data.

3. A Content-Transfer-Encoding header field, which can be used to specify what sort of encoding transformation the body was subjected to and hence what decoding operation must be used to restore it to its original form. It also specifies the domain of the result. Encoding transformations other than the identity transformations are usually applied to data in order to allow it to pass through mail transport mechanisms that may have data or character set limitations.

4. Two additional header fields that can be used to further describe the data in the body, the Content-ID and Content-Description header fields.

All of these header fields are subject to the general syntactic rules for header fields specified in RFC 822.

Encoding Standards

Many media types that can be transported via e-mail are represented in their "natural" format, as 8bit character or binary data. Such data cannot be transferred over some transfer protocols. *RFC 821: Simple Mail Transfer Protocol*, for instance, restricts mail messages to 7bit US-ASCII data with lines no longer than 1000 characters including any trailing CRLF line separator. Therefore, it is necessary to define a standard mechanism for encoding such data into a 7bit short line format.

RFC 2045 defines three transformations that may be included in the Content-Transfer-Encoding token.

1. ***Quoted-Printable Content-Transfer Encoding*** – this encoding technique is intended to represent data that largely consists of octets corresponding to printable characters in the US-ASCII character set. It encodes the data in such a way that the resulting octets are unlikely to be modified by mail transport. If the data being encoded are mostly US-ASCII text, the encoded form of data remains largely recognizable by humans. A body that is entirely US-ASCII may also be encoded in Quoted-Printable to ensure the integrity of the data should the message pass through a character-translating and/or line-wrapping gateway.

   Note: Bodies encoded with the quoted-printable encoding will work reliably over most mail gateways, but may not work perfectly over a few gateways, notably those involving translations into EBCDIC. A higher level of confidence is offered by the base64 encoding technique.

2. ***Base64 Content-Transfer Encoding*** – this encoding technique is designed to represent arbitrary sequences of octets in a form that needs to be readable by humans. Its is virtually identical to the one used in Privacy Enhanced Mail (PEM) applications, as defined in *RFC 1421: Privacy Enhancement for Internet Electronic Mail, Part 1: Message Encryption and Authentication Procedures*.

Base64 encoding uses a 65-character subset for US-ASCII, enabling 6 bits to be represented per printable character. The encoding process represents 24bit groups of input bits as output strings of 4 encoded characters. Proceeding from left to right, a 24bit group is formed by concatenating 3 8bit input groups. These 24 bits are then treated as 4 concatenated 6bit groups, each of which is translated into a single digit in the base64 alphabet. Any characters outside of the base64 alphabet are to be ignored in base64-encoded data.

Note: The 65-character subset used in base64 encoding is represented identically in all versions of ISO 646, including US-ASCII, and all characters in the subset are also represented identically in all versions of EBCDIC.

3. ***Identity*** - the Content-Transfer-Encoding values “7bit”, “8bit”, and “binary” all specify that the identity encoding transformation has been performed, meaning no encoding has been carried out. As such, these values serve simply as the indicators of the domain of the body data and provide information about the sort of encoding that might be needed for transmission in a given transport system.

Media Types Handled by MIME

The general structure of the MIME media typing system and the initial set of media types supported by MIME are defined in *RFC 2046: Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types*.

There are five discrete top-level media types and two composite top-level media types supported by MIME. The content of the discrete media types must be handled by non-MIME mechanisms since they are opaque to MIME processors.

Type	Subtype	Description
Text	Plain	Unformatted text
	Richtext	Text including simple formatting commands
Image	Gif	Still picture in Graphics Interchange Format (GIF)
	Jpeg	Still picture in JPEG format
Audio	Basic	Audible sound
Video	Mpeg	Movie in MPEG format
Application	(Various)	Binary data specific to an application
Message	Rfc822	Encapsulated RFC-822 message, with headers
	Partial	Message has been fragmented for transmission
	External-body	Message is stored elsewhere and must be fetched over the net
Multipart	Mixed	Independent parts designed to be viewed serially
	Alternative	Same message presented in different formats
	Parallel	Parts designed to be viewed simultaneously
	Digest	Each part is an RFC-822 message

1. ***Text Media Type*** – this media type is intended for sending material which is principally textual in form. A ”charset” parameter may be used to indicate the character set of the body text for “text” subtypes, notably including the subtype “text/plain”, which is a generic subtype for plain text.

   Beyond plain text, there are many formats for representing what might be known as “rich text.” These formats, to some extent, are readable even without the software that reads them. It is useful, then, to distinguish them at the highest level from such unreadable data such as images, audio, or text represented in an unreadable form.

   Unrecognized subtypes of “text” should be treated as subtype “plain” as long as the MIME implementation knows how to handle the charset. Unrecognized subtypes that also specify an unrecognized charset should also be treated as “application/octet-stream.”

2. ***Image Media Type*** – a media type of “image” indicates that the body contains an image. The subtype names the specific image format. These names are not case sensitive. An initial subtype is “jpeg” for the JPEG format using JFIF encoding.

   Note: More “image” subtypes are described in *RFC 2048: Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures*.

3. ***Audio Media Type*** – a media type of “audio” indicates that the body contains audio data. Since there is no standard audio format yet for computers that will provide interoperable behavior, the initial subtype of “basic” is specified to meet this requirement. This subtype provides an absolutely minimal lowest common denominator audio format. It is expected that richer formats for high quality and/or lower bandwidth audio will be defined by a later document.

   The content of the “audio/basic” subtype is single channel audio encoded using 8bit ISDN mu-law [PCM] at a sample rate of 8000 Hz.

   Unrecognized subtypes of “audio” should at the minimum be treated as “application/octet-stream.” Implementations may optionally elect to pass subtypes of “audio” that they do not specifically recognize to a robust general-purpose audio playing application, if such an application is available.

4. ***Video Media Type*** – a media type of video indicates that the body contains a time-varying-picture image, possibly with color and coordinated sound. The term ‘video’ is used in its most generic sense, rather than with reference to any particular technology or format. The subtype “mpeg” refers to video coded according to the MPEG standard.

   Unrecognized subtypes of “video” should at the minimum be treated as “application/octet-stream.” Implementations may optionally elect to pass subtypes of “video” that they do not specifically recognize to a robust general-purpose video display application, if such an application is available.

5. ***Application Media Type*** – the “application” media type is used for discrete data that do not fit in any of the other categories, and particularly for data to be processed by some type of application program before it can be viewed by the user. Expected

uses for the “application” media type include file transfer, spreadsheets, data for mail-based scheduling systems, and languages for “active” (computational) materials.

“Active” languages may be defined as subtypes of the “application” media type. Examples of these subtypes are the octet-stream and PostScript.

6. ***Message Media Type*** – an encapsulated message. A body of media type “message” is itself all or a portion of some kind of message object. Such objects may or may not contain other entities. The “rfc822” subtype is used when the encapsulated content is itself an RFC 822 message. The “partial” subtype is defined for partial RFC 822 messages to permit the fragmented transmission of bodies that are thought to be too large to be passed through transport facilities in one piece. Another subtype, “external body,” indicates that the actual body data are not included, but merely referenced. In this case, the parameters describe a mechanism for accessing the external data.

7. ***Multipart Media Type*** – data consisting of multiple entities of independent data types. Four subtypes are initially defined, including the basic “mixed” subtype specifying a generic mixed set of parts, “alternative” for representing the same data in multiple formats, “parallel” for parts intended to be viewed simultaneously, and “digest” for multipart entities, in which each part has a default type of “message/rfc822.”

The following is an example of a multipart message that has two parts. One of the parts is typed explicitly while the other is typed implicitly.

```
From: rommelfajardo@hotmail.com (Rommel Fajardo)
To: kehres@ima.com  (Tim Kehres)
Date: Thu, 7 May 1998 21:27:43 -0800 (PST)
Subject: Simple example
MIME-Version: 1.0
Content-type: multipart/mixed; boundary="unique boundary"

This is the preamble to the message.  It is to be ignored, though it
is a handy place for composition agents to include an
explanatory note to non-MIME conformant rea ders.

--unique boundary

This is part 1 of the message. There was no Content-type or Content-transfer-encoding header, so
it defaults to type “text/plain” and encoding “7bit”. This is implicitly typed plain US-ASCII text.
It does not end with a linebreak.

--unique boundary
Content-type: text/plain; charset=US-ASCII

This is part 2 of the message. It is explicitly typed as “text/plain”. Its default encoding is also
“7bit”. It ends with a linebreak.

--unique boundary--
```

This is the epilogue. It is also to be ignored.

Aside from the seven media types already mentioned, *RFC 2077: The Model Primary Content Type for Multipurpose Internet Mail Extensions* defines a new media type that can be handled by MIME. This is known as the model primary MIME type, an electronically exchangeable behavioral or physical representation within a given domain. Each subtype in the model structure has unique features, just as does each subtype in the other primary types. The important fact is that these various subtypes can be converted between each other with less loss of information than those subtypes belonging to the seven primary types previously mentioned. For the model primary MIME type, base64 encoding is recommended.

MIME was designed to be extensible. Thus, the set of content types, subtypes, and options is easily extensible, as well as the set of transfer encodings. The MIME standard RFC 1521 requires that new values for these basic types be registered with the Internet Assigned Numbers Authority or IANA to prevent confusion and name collision. Among such media types are:

- application/green-commerce for commercial transactions
- application/safe-tcl for enabled-mail
- application/x-macbinhex40 for Mac BinHex 4.0
- audio/x-macaudio for unsampled Macintosh audio
- audio/x-next for self-describing SunOS/NeXT audio data
- image-x-pgm for PGM data
- image/x-fits for FITS files
- video/x-qtv for QuickTime TV video/audio broadcasts
- video/x-qtc for QuickTime TV conference calls

To view IANA's master list of new MIME mappings, go to:

http://www.isi.edu/in-notes/iana/assignments/media-types/media-types

You can also find more information in Internet Exchange Gateway Administrator's Manual Version 3.0 Appendix D.

Message Header Extensions for Non-ASCII Text

When encountering non-ASCII text in RFC 822 message headers, several mail relaying programs are known to delete some message header fields while retaining others, rearrange the order of addresses in To or Cc fields, rearrange the vertical order of header fields, or "wrap" message headers at different places than those in the original message. Some programs are also known to experience difficulty in parsing message headers correctly.

Techniques for encoding of non-ASCII text in various portions of a RFC 822 message header without confusing existing Internet mail handling programs are described in detail in *RFC 2047: Multipurpose Internet Mail Extensions (MIME) Part Three: Message Header Extensions for Non-ASCII Text*.

The techniques described in RFC 2047 use certain sequences of “ordinary” printable ASCII characters, known as “encoded-words,” as encoded data. The syntax of encoded-words is such that that they are unlikely to accidentally appear as normal text in message headers. Furthermore, the characters used in encoded-words are restricted to those that do not have special meanings in the context in which the encoded-word appears.

Generally, an encoded-word is a sequence of printable ASCII characters that begins with “=?”, and ends with “?=”, and has two “?”s in between. It specifies a character set and an encoding method, and also includes the original text encoded as graphic ASCII characters, according to the rules for that particular encoding method.

A mail composer that implements this specification will provide a means of putting non-ASCII text in header fields, but will have to translate these fields (or appropriate portions of these fields) into encoded-words before inserting them into the message header. A mail reader that implements this specification will recognize encoded-words when they appear in certain portions of the message header. Instead of displaying the encoded-word “as is”, it will reverse the encoding and display the original text in the designated character set.

The following are examples of message headers that contain encoded words:

```
From: =?US-ASCII?Q?Keith_Moore?= <moore@cs.utk.edu>
To: =?ISO-8859-1?Q?Keld_J=F8rn_Simonsen?= <keld@dkuug.dk>
CC: =?ISO-8859-1?Q?Andr=E9?= Pirard <PIRARD@ vm1.ulg.ac.be>
Subject: =?ISO-8859-1?B?SWYgeW91IGNhbiByZWFkIHRoaXMgeW8=?=
=?ISO-8859-2?B?dSB1bmRlcnN0YW5kIHRoZSBleGFtcGxlLg==?=
```

Note: In the first encoded work of the Subject field above, the last "=" at the end of the 'encoded-text' is necessary because each 'encoded-word' must be self-contained (the "=" character completes a group of 4 base64 characters representing 2 octets). An additional octet could have been encoded in the first 'encoded-word' (so that the encoded-word would contain an exact multiple of 3 encoded octets), except that the second 'encoded-word' uses a different 'charset' than the first one.

```
From: =?ISO-8859-1?Q?Olle_J=E4rnefors?= <ojarnef@admin.kth.se>
To: ietf-822@dimacs.rutgers.edu, ojarnef@admin.kth.se
Subject: Time for ISO 10646?
To: Dave Crocker <dcrocker@mordor.stanford.edu>
Cc: ietf-822@dimacs.rutgers.edu, paf@comsol.se
From: =?ISO-8859-1?Q?Patrik_F=E4ltstr=F6m?= <paf@nada.kth.se>
Subject: Re: RFC-HDR care and feeding

From: Nathaniel Borenstein <nsb@thumper.bellcore.com>
(=?iso-8859-8?b?7eXs+SDv4SDp7Oj08A==?=)
To: Greg Vaudreuil <gvaudre@NRI.Reston.VA.US>, Ned Freed
<ned@innosoft.com>, Keith Moore <moore@cs.utk.edu>
Subject: Test of new header generator
MIME-Version: 1.0
Content-type: text/plain; charset=ISO-8859-1
```

Mapping Between X.400/ISO 10021 and RFC 822

There is a large community using RFC 822 based protocols for mail services who will wish to communicate with users of the Interpersonal Messaging System (IPMS) provided by the OSI Message Handling System (MHS) or X.400. This will also be a requirement in cases where communities intend to make a transition to use of an X.400 IPMS, as conversion will be needed to ensure a smooth service transition. It is expected that there will be more than one gateway, and this specification will enable them to behave in a consistent manner. Note that the term gateway is used to describe a component performing the protocol mappings between RFC 822 and X.400. This is standard usage among mail implementors, but should be noted carefully by transport and network service implementors.

Messages for X.400 MHS comprise of an IPMS.heading and an IPMS.body. The IPMS.body is a sequence of IPMS.Body Parts. An IPMS.Body Part may be a nested message (IPMS.MessageBodyPart).

A MIME message consists of headers and a content. The content may be structured (multipart or message) or atomic (otherwise). An element of a structured content may be a message or a content. Both message and structured content have subtypes that do not have direct analogies in MHS.

The mapping between X.400 and RFC 822 message bodies, as defined by *RFC 1495: Mapping Between X.400 and RFC-822 Message Bodies,* is symmetrical for the following cases:

- any atomic body part
- multipart: digest and mixed subtypes
- message/rfc822

The mappings have been specifically designed to provide optimal behavior for three different scenarios:

- Allow a MIME user and an MHS user to exchange an arbitrary binary content;
- Allow MIME content-types to "tunnel" through an MHS relay, that is, two MIME users can exchange content-types without loss
- Allow MHS body parts to "tunnel" through a MIME relay, that is, two MHS users can exchange body parts without loss of data through a MIME relay.

Other, related, scenarios can also be easily accommodated. The mappings for the headers are specified in *RFC 1327: Mapping Between X.400 (1988)/ISO 10021 and RFC 822*.

Security Issues

An Internet e-mail message consists of two parts: the headers and body. The headers form a collection of field/value pairs structured according to STD 11, RFC 822, while the body, if structured is defined according to MIME. The basic MIME specification, however, does not provide security protection.

A framework whereby security protection provided by other protocols may be used with MIME in a complementary manner is described in *RFC 1847: Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted*. This framework is provided by defining two new security subtypes of the MIME multipart content type: signed and encrypted. The multipart/signed content type specifies how to support authentication and integrity services via digital signature. The multipart/encrypted content type specifies how to support confidentiality via encryption. In each of these security subtypes, there are two related body parts: one for the protected data and one for the control information.

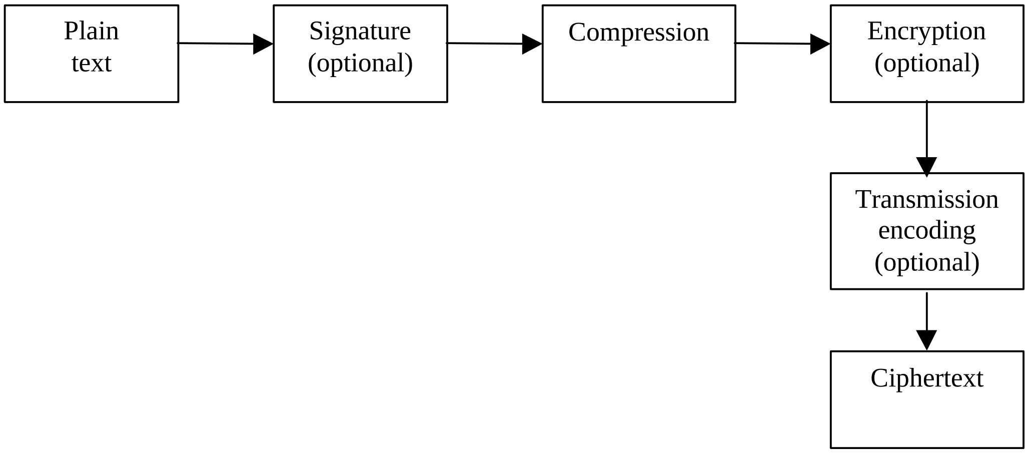
One such protocol that uses the multipart/signed and multipart/encrypted framework is the MIME Object Security Services (MOSS) protocol, which is described in detail in *RFC 1848: MIME Object Security Services*. MOSS is based largely on the Privacy Enhanced Mail (PEM) protocol, which defines message encryption and message authentication procedures for text-based e-mail messages using a certificate-based key management mechanism. PEM is designed to be compatible with most Internet e-mail systems, thus, PEM messages can be created with text editors. In addition, most e-mail systems do not destroy PEM messages. PEM is described in detail in *RFC 1421: Privacy Enhancement for Internet Electronic Mail, Part I: Message Encryption and Authentication Procedures, RFC 1422: Privacy Enhancement for Internet Electronic Mail, Part II: Certificate-Based Key Management,* and *RFC 1423: Privacy Enhancement for Internet Electronic Mail, Part III: Algorithms, Modes, and Identifiers*.

Under the MOSS protocol, an originator's private key (symmetric cryptography) is used to digitally sign MIME objects; a recipient would use the originator's public key (asymmetric cryptography) to verify the digital signature. A recipient's public key is used to encrypt the data-encrypting key that is used to encrypt the MIME objects; a recipient would use the corresponding private key to decrypt the data encrypting key so that the MIME objects can be decrypted. The following is an example of a PEM message using symmetric (secret-key) cryptography:

```
- - - - BEGIN PRIVACY-ENHANCED MESSAGE - - - -

Proc-Type: 4 , ENCRYPTED
Content-Domain: RFC 822
DEK-Info: DES-CBC , F8143EDE5960C597
Originator-ID-Symmetric: schneier@chinet.com , ,
Recipient-ID-Symmetric: schneier@chinet.com,ptf-kmc , 3
Key-Info: DES-ESB, RSA-
MD2 , 9FD3AAD2F2691 , B70665BB9BF7CBCDA60195DB94F727D3
Recipient-ID-Symmetric: pem-dev@tis.com , ptf-kmc , 4
Key-Info: DES-ECB , RSA-
MD2, 161A3F75DC82EF26 , E2EF532C65CBCFF79F83A2658132DB47

LLrHBOeJzyhP+/fSStdW8okeEnv47jxe7SJ/iN72ohNcUk2jHEUSoH1nvNSIWL9M

8tEjmF/zxB+bATMtPjCUWbz8Lr9wloXIkjHUlBLpvXROUrUzYbkNpk0agV21zUpk

J6UiRRGcDSvzrsoK+oNvqu6z7Xs5Xfz5rDqUcMlK1Z6720dcBWGGs DLpTSCnpot

dXd/H5LMDWnonNvPCwQUHt==
- - - - END PRIVACY-ENHANCED MESSAGE - - - -
```

Another protocol that can be used to provide privacy and authentication using the MIME security content types described in RFC 1847 is Pretty Good Privacy (PGP). *RFC 2015: MIME Security With Pretty Good Privacy (PGP)* defines three new content types for implementing security and privacy ith PGP: application/pgp-encrypted; application/pgp-signature, and application/pgp-keys. PGP generates either ASCII armor or 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. 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.

Figure 1. Preparation of a PGP message



An emerging version of MIME that promises to provide optimum data security is the Secure Multipurpose Internet Mail Extensions (SMIME), which uses a uniform method to encrypt browser-based e-mail. SMIME is described as a protocol capable of providing a consistent way to send and receive secure MIME data. It provides the following cryptographic security services for electronic messaging applications: authentication, message integrity and non-repudiation of origin (using digital signatures) and privacy and data security (using encryption).

SMIME can be used by traditional mail user agents (MUAs) to add cryptographic security services to mail that is sent, and to interpret cryptographic security services in mail that is received. However, SMIME is not restricted to mail; it can be used with any transport mechanism that transports MIME data, such as HTTP. As such, SMIME takes advantage of the object-based features of MIME and allows secure messages to be exchanged in mixed-transport systems. In addition, SMIME can be used in automated message transfer agents that use cryptographic security services that do not require any human intervention, such as the signing of software-generated documents and the encryption of FAX messages sent over the Internet.

The following is an example of a multipart/signed SMIME message:

```
Content-Type: multipart/signed;
protocol="application/pkcs7-signature";
micalg=sha1; boundary=boundary42
```

```
--boundary42
Content-Type: text/plain

This is a clear-signed message.

--boundary42
Content-Type: application/pkcs7-signature; name= smime.p7s
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7s

ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB9HG4VQpfyF467GhIGfHfYT6
4VQpfyF467GhIGfHfYT6jH77n8HHGghyHhHUujhJh756tbB9HGTrfvbnj
n8HHGTrfvhJhjH776tbB9HG4VQbnj7567GhIGfHfYT6ghyHhHUujpfyF4
7GhIGfHfYT64VQbnj756

--boundary42--
```

MIME Conformance

RFC 2049: Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples describes what portions of MIME must be supported by a conformant MIME implementation to allow the useful interworking of messages with content that differs from US-ASCII.

As defined in RFC 2049, a mail user agent that is MIME-conformant MUST:

1. Always generate a “MIME-Version: 1.0” header field in any message it creates

2. Recognize the Content-Transfer-Encoding header field and decode all received data encoded by either quoted-printable or base64 methods. The identity transformations 7bit, 8bit, and binary must also be recognized.

3. Treat any unrecognized Content-Transfer-Encoding as if it had Content-Type of “application/octet-stream”, regardless of whether or not the actual Content-Type is recognized.

4. Recognize and interpret the Content-Type header field and avoid showing users raw data with a Content-Type field other than the text.

5. Ignore any content type parameters whose names they do not recognize.

6. Explicitly handle the media types mentioned above to certain extents.

7. Treat unrecognized Content-Type field as if it had a media type of “application/octet-stream” with no parameter sub-arguments.

8. Conformant user agents are required, if they provide non-standard support for non-MIME messages employing character sets other than US-ASCII, to do so on received messages only. Conforming user agents must not send non-MIME

messages containing anything other than US-ASCII text. In particular, the use of non-US-ASCII text in mail messages without a MIME-Version field is strongly discouraged as it impedes interoperability when sending messages between regions with different localization conventions. Conforming user agents MUST include proper MIME labeling when sending anything other than plain text in the US-ASCII character set. In addition, non-MIME user agents should be upgraded if at all possible to include appropriate MIME header information in the messages they send even if nothing else in MIME is supported. This upgrade will have little, if any effect on non-MIME recipients and will aid MIME in correctly displaying such messages. It also provides a smooth transition path to eventual adoption of other MIME capabilities.

9. Conforming user agents must ensure that any string of non-white-space printable US-ASCII characters within a "*text" or "*ctext" that begins with "=?" and ends with "?=" be a valid encoded-word. ("begins" means: At the start of the field-body or immediately following linear-white-space; "ends" means: At the end of the field-body or immediately preceding linear-white-space.) In addition, any "word" within a "phrase" that begins with "=?" and ends with “?=" must be a valid encoded-word.

10. Conforming user agents must be able to distinguish encoded-words from "text", "ctext", or "word"s anytime they appear in appropriate places in message headers. It must support both the "B" and "Q" encodings for any character set which it supports. The program must be able to display the unencoded text if the character set is "US-ASCII". For the ISO-8859-* character sets, the mail reading program must at least be able to display the characters which are also in the US-ASCII set.

A user agent that meets the above conditions is said to be MIME-conformant. The meaning of this phrase is that it is assumed to be "safe" to send virtually any kind of properly-marked data to users of such mail systems, because such systems will at least be able to treat the data as undifferentiated binary, and will not simply splash it onto the screen of unsuspecting users.

There is another sense in which it is always "safe" to send data in a format that is MIME-conformant, which is that such data will not break or be broken by any known systems that are conformant with RFC 821 and RFC 822. User agents that are MIME-conformant have the additional guarantee that the user will not be shown data that were never intended to be viewed as text.

References

RFC 821
Jonathan B. Postel, Network Working Group, “Simple Mail Transfer Protocol,” August 1992.

RFC 822
David H. Crocker, “Standard for the Format of ARPA Internet Text Messages,” August 13, 1982.

RFC 2045
N. Freed and N. Borenstein, Network Working Group, “Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies,” November 1996.

RFC 1049
M. Sirbu, "A Content-Type Header Field for Internet Messages," March 1988.

RFC 2046
N. Freed and N. Borenstein, Network Working Group, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types," November 1996.

RFC 2047
K. Moore, Network Working Group, "Multipurpose Internet Mail Extensions (MIME) Part Three: Message Header Extensions for Non-ASCII Text," November 1996.

RFC 1495
H. Alvestrand, S. Kille, R. Miles, M. Rose, and S. Thompson, Network Working Group, "Mapping Between X.400 and RFC-822 Message Bodies," August 1993.

RFC 2048
N. Freed, J. Klensin, and J. Postel, Network Working Group, "Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures," November 1996.

RFC 1847
J. Galvin, S. Murphy, S. Crocker, and N. Freed, Network Working Group, "Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted," October 1995.

RFC 2015
M. Elkins, Network Working Group, "MIME Security With Pretty Good Privacy (PGP)," October 1996.

RFC 1421
J. Linn, Network Working Group, "Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures," February 1993.

RFC 1422
S. Kent, Network Working Group, "Privacy Enhancement for Internet Electronic Mail: Part II: Certificate-Based Key Management," February 1993.

RFC 1423
D. Balenson, Network Working Group, "Privacy Enhancement for Internet Electronic Mail: Part III: Algorithms, Modes, and Identifiers," February 1993.

RFC 2049
N. Freed and N. Borenstein, Network Working Group, "Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples," November 1996.

RFC 2077
S. Nelson and C. Parks, Network Working Group, "The Model Primary Content Type for Multipurpose Internet Mail Extensions," January 1997.

RFC 1521
N. Borenstein and N. Freed, Network Working Group, “MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for Specifying and Describing the Format of Internet Message Bodies,” September 1993.

RFC 1327
S. Hardcastle-Kille, Network Working Group, “Mapping Between X.400 (1988)/ISO 10021 and RFC 822,” May 1992.

RFC 1848
S. Crocker, N. Freed, J. Galvin, and S. Murphy, Network Working Group, “MIME Object Security Services,” October 1995.

Bruce Schneier, “E-Mail Security: How to Keep Your Electronic Messages Private,” John Wiley & Sons Inc., 1995.