EDIINT Working Group
draft-ietf-ediint-as2-06.txt

Dale Moberg
Dick Brooks
Rik Drummond

Expires March 2000

October 10 1999

HTTP Transport for Secure EDI

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

To view the current status of any Internet-Draft, please check the ``1id-abstracts.txt'' listing contained in an Internet-Drafts Shadow Directory, see http://www.ietf.org/shadow.html.

Abstract

This document describes how to exchange EDI documents securely using http transport for EDI data that is packaged in MIME messages that use public key security body parts.

Feedback Instructions:

If you want to provide feedback on this draft, follow these guidelines:

- Send feedback via e-mail to the ietf-ediint list for discussion, with "AS#2" in the Subject field. To enter/follow the discussion, you need to subscribe at ietf-ediint@imc.org.

- Be specific as to what section you are referring to, preferably quoting the portion that needs modification, after which you state your comments.

- If you are recommending some text to be replaced with your suggested text, again, quote the section to be replaced, and be clear on the section in question.

Table of Contents

1. Introduction

1.1 Purpose and relation to previous work

Early work on Internet EDI focused on specifying MIME content types for EDI data [MIMEEDI] The functional requirements document , "Requirements for Interoperable Internet EDI," [EDIINT] provides extensive information on EDI security and the business and user processes that can benefit from the use of EDI security. In addition, MIME structures appropriate for SMTP transport of the packaged EDI data are specified in ([AS1] "MIME-based Secure EDI") as well as details needed to support signed receipts as acknowledgments. The framework of [AS1] shows how to implement the security features--specifically data privacy, data integrity/authenticity, non-repudiation of origin and non-repudiation of receipt -- found to be requirements for secure EDI.

In this document, it is assumed that the reader is familiar with the SMTP/MIME transport document, the requirements document, and the RFCs applied or referenced in those documents.

This draft, like the SMTP/MIME transport document, builds on previous RFCs and is attempting to "re-invent" as little as possible. The goal here is to specify how previously specified MIME messaging structures and operations can be adapted for use with HTTP servers and clients to obtain secure, reliable, and acknowledged transport for EDI data.

The applicability statement, [AS1] "MIME-based Secure EDI," explained the basic EDI transaction using the concept of a "secure transmission loop" for EDI. This loop involves one organization sending a signed and encrypted EDI interchange to another organization, requesting a signed receipt, followed later by the receiving organization sending this signed receipt back to the sending organization. The transmission involves the following stages:

i. The organization sending EDI/EC data encrypts the data and provides a digital signature, using either PGP/MIME or S/MIME. In addition, they request a signed receipt.

ii. The receiving organization decrypts the message and verifies the signature, resulting in verified integrity of the data and authenticity of the sender.

iii. The receiving organization then sends a signed receipt using a signature over the hash of a message disposition notification, which contains a hash of the received message.

The above describes functionality which, when implemented, would satisfy all security requirements. Other restricted subsets of functionality (for example, only signature and no signed receipt) have also been characterized.
In this document, the goal is to make use of HTTP instead of SMTP

as a transport protocol, and make changes that are needed to adapt to protocol packaging differences.

In either transport case, the body of the message is a MIME structure, using MIME headers ("content-type" and other "content-X" tags) to convey information about the data being transported.

Also, one primary use of SMTP RFC 822 headers within SMTP based transport of secure EDI has been to enable requests for acknowledgements and to specify options for signatures over acknowledgements (asymmetric encryption and cryptographic hash algorithm preferences).

One way to convey this information within the HTTP transport context is to use either HTTP entity-headers or extension-headers [11, section 7.1] that have the syntax of SMTP headers. Only the "From" header is overloaded by possibly different usages in the SMTP and HTTP contexts. The From header normally contains machine-usable email addresses as defined in [SMTPMSG]. The usage of the FROM header in [HTTP] section 14.22 is to provide the email address of an administrative contact for the HTTP client. The function of the "From" header in the SMTP context of secure EDI transport has been to supply a value used in constructing the MDN style receipt. But the MDN receipt has been found to be too restrictive for some commercial EDI transport scenarios [GISB]. So alternative receipt mechanisms will be provided that, among other things, will remove any conflicts arising from trying to reuse the SMTP-MDN roles of "From" within the context of HTTP reserved usage of "FROM".

Also, it is currently difficult to make use of HTML [HTML] and simple scripting to send HTTP entity-headers as part of the HTML FORM tag construct. For HTML-based POST situations [GISB], it is useful to specify ways to convey 'metadata' needed for the secure transmission loop that do not make use of HTTP headers. One way to specify this data is by using the MIME multipart/form-data packaging specified in [FORMDATA].

For SMTP transport, the receipt and signed receipt functions are implemented using Message Dispostion Notifications [MDN] and Multipart/signed Message Disposition Notifications [AS1]. As mentioned previously, for HTTP transport, generalization of the Message Disposition Notification is useful. The multipart/report [REPORT] MIME package is retained to support these generalizations, and multipart/signed packages are used to support signatures over these generalized receipts.

Finally, within the HTTP transport context, it is useful to make use of Transport Layer Security [TLS] to provide privacy. Compression can be provided using HTTP content-codings [HTTP], sections 3.5, 14.3, 14.12]. (Content codings are not be be confused with the MIME concept of content transfer encodings.)

A variety of other minor differences (for example, absence of content-transfer-encoding) are noted below and summarized in the concluding section.

1.2 Overall operation

A HTTP POST operation [HTTP] is used to send appropriately packaged EDI or other business data. The Request-URI ([HTTP], section 9.5) identifies a process to unpack and handle the message data and to generate a reply for the client that contains a message disposition acknowledgement or a multipart/report, signed or unsigned, and possibly other turnaround transactions. This request/reply transactional interchange provides secure, reliable, and authenticated transport for EDI or other business data using HTTP; the security protocols and structures used also support auditable records of these transmissions, acknowledgements, and authentication.

2. Stages and Details of HTTP EDI Transmission and Acknowledgment

An EDI data file or stream is first structured into one of the message templates described in [AS1], sections 4.2.1 to 4.2.4 or 4.3.1 to 4.3.4 for PGP/MIME or S/MIME security. If TLS is to be used, the typical packaging will be that provided in 4.2.2 or 4.3.2; that is, a multipart/signed message will be created with no encryption in the message. Otherwise, if privacy is desired, message templates 4.2.4 or 4.3.4 are used. Content transfer encoding is not used and a content-length field is to be provided.

If HTML-based POST is used (using the METHOD=POST attribute within the "FORM" tag) [HTML, 17 Forms], then the message templates will be packaged as the final part of a multipart/form-data. The metadata needed for application layer routing, identification, requesting a reply and other transaction operations can be packaged in message body parts in the multipart/form-data. The labels for the metadata values are found in the "name" parameter of the Content-Disposition header in each form-data part as discussed in [FORMDATA, section 3].

In general, both HTTP servers and HTTP clients handling the message templates of [AS1] should be prepared to process these basic EDIINT data formats when they are embedded within MIME multiparts.

In addition to the enveloping and MIME media type options defined in sections 4.2.x and 4.3.x of "MIME-based Secure EDI" [AS1], this specification enables the transport of payload objects containing other MIME media types. A listing of registered MIME media types is available from the Internet Assigned Numbers Authority (IANA) at: http://www.isi.edu/in-notes/iana/assignments/media-types/media-types [MIME-TYPES].

Implementors are to follow the appropriate specifications identified under "References" in [MIME-TYPES], for the type of object being transmitted. For example, to send an XML object, the MIME media type of text/xml is used in the Content-type MIME header and the specifications for enveloping the object are contained in RFC2376, (e.g. Content-type: text/xml; charset="utf-8").

Many of the specifications referenced by [MIME-TYPES] were designed for SMTP transports. Implementors are advised to make appropriate adjustments for HTTP transport as indicated in section 4 of this document.

Finally, several industry groups currently make use of "encapsulated" (or opaque) signatures within encrypted or signed objects. Encapsulated signatures should be supported in order to accommodate these existing practices. Objects containing encapsulated signatures must be prepared according to the specifications contained in section 3.4.2 of RFC 2311[SMIMEV2] or, in the case of PGP, according to the specifications contained in section 6.2 of "MIME Security with Pretty Good Privacy (PGP)" [MIMEPGP] and "OpenPGP Message Format" [RFC2440].

2.1.1 Requesting MDN-based receipts

For requesting MDN based receipts, the originator supplies metadata using the syntax of extension headers (the [SMTPMSG] header syntax) that precede the message body. The header "tags" are as follows:

A Disposition-Notification-To header is added to indicate that a message disposition notification is requested in the reply to the POST request. This header is specified in [MDN].

A Message-ID header is added to support message reconciliation, so that an Original-Message-Id value can be returned in the MDN body part of the receipt. (Receipts is here used to refer to the signed or unsigned multipart/report body parts.)

Both "From" and "To" extension headers are to be supplied. The "From" value needs to have an email address as specified in [SMTPMSG] and [HTTP]. If other uses of "From" are needed, the generalized receipts to be next discussed should be used. There the role of From is replaced by tags not having a reserved HTTP and SMTP usage.

Other headers, especially "Subject" and "Date", should be supplied; the values of these headers are often mentioned in the human-readable section of a MDN to aid in identifying the original message.

A Disposition-Notification-Options header is used to request a signed message disposition notification. The parameters used to select protocols for signed message disposition

notification are found in [AS1].

A Receipt-delivery-option is a header whose value is a URL that indicates how the receipt is to be delivered. While the default mode of operation within HTTP transport is to return the receipt (be it MDN, signed MDN, generalized report receipt, or signed report receipt) in the reply body, asynchronous reply is allowed through use of this name. The "mailto", "http", and "https" will be the more common types of value for this information. If this header is found and contains the value "default" (case-insensitive), then any receipt is to be returned in the reply to the request.

2.1.2 Requesting Generalized Receipts

For requesting generalized receipts using the MIME template for multipart/reports [REPORTS], the following metadata can be supplied using the syntax of the name parameter within the Content-Disposition header of the multipart/form-data structure. Within HTML, these names correspond to the name attribute within the INPUT element, where the "type" attribute has a "text" value. [HTML], section 18.

The To name contains an identifier identifying the intended recipient of a data exchange and may be D&B D-U-N-S number [DUNS] or other agreed upon identifier system. Applications should allow users to configure these elements in the automated HTTP agents processing these values. For example, the body part MIME header line looks like the following line:

Content-Disposition: form-data; name="To"

The From name contains a textual value identifying the sender of a data exchange, such as the a D&B D-U-N-S number [DUNS] as in [GISB]. Because "From" has a specified use within [HTTP], the From name parameter is not to be considered equivalent to the extension header. If an extension header "From" is to be used it should within HTTP, it should conform to the usage, syntax, and semantics of [HTTTP] section 14.22. The extension header counterpart of the sender of a data exchange is the extension header version of "Receipt-disposition-to".

The Input-format name identifies the type of data contained in a data file.

The Agent name parameter indicates the network or agent where the data exchange originated.

The Application name identifies the application used to process

the data next(after the URI-request process has finished with the stream).

The DateTime name provides the date and time the data was created and uses the format specified in [SMTPMSG] as updated by RFC 1123.

The RefNum is an integer value used to uniquely identify the communication exchange and is in a textual format. The RefNum is similar to the Message-ID and Content-Id headers of SMTP that are used in constructing values in receipts based on MDNs.

The UserParam is a user defined parameter.

GISB-Version is a protocol version number [GISB].

Transaction-set is an optional data element identifying the EDI transaction.

Input-data is the sending side's local file system name for the file being sent

Receipt-disposition-to name contains an identifer identifying the party to receive positive notification of delivery. This identifier may be a D&B D-U-N-S number [DUNS] as in [GISB].

Receipt-delivery-option is a URL containing a value indicating how the receipt is to be delivered. While the default mode of operation within HTTP transport is to return the receipt (be it MDN, signed MDN, generalized report receipt, or signed report receipt) in the reply body, asynchronous reply is allowed through use of this name. The "MAILTO", "HTTP", and "HTTPS" will be the more common types of value for this information. For the HTTP and HTTPS schemes, the POST method is to be used.

Receipt-report-type indicates the desired value of the "report-type" parameter in the multipart/report content type of a specific version of the generalized receipt. If multiple types are to be indicated, use a semi-colon as a separator. An example for this value (discussed below) is "GISB-Acknowledgement-Receipt."

Receipt-security-selection is a name that indicates the protocol and algorithm choices for a digital signature over the receipt. Signatures are always in multipart/signed packages. The format for protocol and algorithm choices is that used in [AS1] and [MDN].

Disposition-Notification-To is a name that, if present, indicates that the MDN style of receipt is to be used. It may have values other than email addresses when it is found as a name parameter in a form-data body part, such as a D-U-N-S number. When this tag is used in HTTP extension headers or in SMTP headers it follows

the MDN usage.

Disposition-notification-options identifies characteristics of message disposition notification in accordance with [AS1] and [MDN].

Application are encouraged to support handling all metadata values whether they make use of the name parameter syntax within a multipart/form-data or whether they use the message header syntax used in SMTP or HTTP headers [SMTPMSG]. If metadata items are repeated in extension headers and in form-data parts, but the values are not the same, the extension header values will be selected for use.

The presence of the metadata value "Receipt-Disposition-To" using the extension header syntax indicates a request for a generalized receipt. This value also fulfills the role of "From" when a value other than an email address needs to be used. The value in Receipt-Disposition-To may have no significance for setting up the transport connection. Therefore, the extension header " Receipt-delivery-option" should be used to provide that information as a URL or as the special value "default." For signed generalized receipts, an extension header of "Receipt-security-selection" should be added to indicate the desired security protocol for the multipart/signed over the multipart/report.

2.1.3 Summary of Receipt request options

In summary, the receipt request and construction process now has the following options. Receipt requests are made by conveying metadata values using a syntax of either the name parameter in a multipart/form-data's Content-Disposition headers or using a syntax of extension headers.

Both MDN and generalized receipts can be requested in either way, though using an extension header syntax and requesting a MDN receipt means restricting the "From" values to email addresses. And either type of receipt comes in signed or unsigned versions. Finally, receipts may be delivered synchronously (HTTP only) or asynchronously by using the " Receipt-delivery-option" header.

2.2 Sending EDI in HTTP Client Requests using POST

For sending EDI, the following protocol elements are typically present: a request line ([HTTP], section 5.1), entity headers, a CRLF pair to mark the end of the entity headers, followed by the message-body.
The request line will have the form: "POST Request-URI HTTP/1.1", with spaces and followed by a CRLF. The Request-URI is typically

exchanged out of band, as part of setting up a bilateral trading partner agreement. Applications should be prepared to deal with an initial reply containing a status indicating a need for authentication of the usual types used for authorizing access to the Request-URI ([HTTP], section 10.4.2 and elsewhere).

Automation of this process is not discussed in this document but might involve obtaining a session URL from a page requesting authentication and possibly other information about proposed EDI standard versions and other trading conventions to be used.

The request line is followed by entity headers specifying content length ([HTTP] section 14.14) and content type [HTTP] section 14.18. The Host request header ([HTTP] sections 9 and 14.23) is also included.

The entity or extension headers used for requesting a MDN (unsigned or signed) have previously been mentioned, as have those ("To" "From" "Message-Id") that are needed as values for MDN fields or for other receipt requests.

For generalized receipts based on the multipart/report content type, the metadata can be the values found in extension headers, but can also be placed in body parts of a multipart/form-data using "name" parameters in the content-disposition header.

Finally, the payload is found in any of the message patterns of [AS1] sections 4.2.1 to 4.2.4 or 4.3.1 to 4.3.4 for PGP/MIME or S/MIME security. These payloads may arrive as the final part of a multipart/form-data or may even be enclosed in some other multipart.

2.3 Using Transport Layer Security

To use Transport Layer Security [TLS], the request-URI should indicate the appropriate scheme value, HTTPS. Usually only a multipart/signed message body would be sent using TLS, as encrypted message bodies would be redundant. Encrypted message bodies are not prohibited, however. For asynchronous receipt delivery requests, use the "Receipt-delivery- option" header with a URL value making use of the HTTPS scheme to obtain security privacy.

2.4 Response Status Codes in Replies

The status line for response to errors in the POST request line will be provided by a status line with the following protocol elements present ([HTTP], section 6.1) : HTTP version (normally, HTTP/1.1), a status code, reason phrase, and CRLF.

The status codes return status concerning HTTP operations. For example, the status code 401, together with the WWW-Authenticate header, is used to challenge the client to repeat the request with an Authorization header. Other explicit status codes are

documented in [HTTP], sections 6.1.1 and throughout section 10.
For errors in the request-URI, 400 ("Bad Request"),
404 ("Not Found") and similar codes are appropriate status codes.
These codes and their semantics are specified by [HTTP].
A careful examination of these codes and their semantics
should be made before implementing any retry functionality
that is described below; specifically, retries should not
be made if the error is not transient or if retries are
explicitly discouraged (for real authentication failures, for example.)

2.5 Receipt Reply

The details of the response to the POST command vary depending
upon whether a receipt has been requested and upon what kind of receipt
has been requested.

With no extended header requesting a receipt, and no errors
accessing the request-URI specified processing, the status
line in the Response to the POST request should be in the
200 range. Status codes in the 200 range should also be used when
an entity is returned (a signed receipt in a multipart/signed
content type or an unsigned receipt in a multipart/report).
That is, even when the disposition of the data was an error condition
at the authentication, decryption or other higher level, the
HTTP status code should indicate success at the HTTP level.

The HTTP server-side application may respond with an unsolicited
multipart/report as a message body that the HTTP client
might not have solicited, but this may be discarded by the
client. Applications should avoid emitting unsolicited receipt replies
because bandwidth or processing limitations might have led
administators to suspend asking for acknowledgements.

When a Disposition-Notification-To extension header is present
in the POST request entity headers, then entity headers for
the MDN should be included. The content type for the MDN receipt
(multipart/report [REPORT] or multipart/signed [SECURITY])
should be included in the Response entity headers. The

The basic responsibilities of responding to requests are discussed
at length in [AS1] section 5, and in detail within section 5.2.1.

2.5.1 MDN based Receipts and Signed MDN Receipts

Message Disposition Notifications, when used in the HTTP reply
context, will closely parallel a SMTP MDN. For example,
the disposition field is a required element in the machine
readable second part of a multipart/report for a MDN.
The final-recipient-field([MDN] section 3.1) value should
be derived from the entity headers of the request
If the "To" field is missing, for signed messages,

the value for Original-recipient may be the email address field from the signer's X.509 attribute for email addresses, if that value is available. For a MDN, an application must report the Message-ID of the request. The human readable part (the first part of the multipart/report) should include items such as the subject, date and other information when those fields are present in entity header fields following the POST request

The HTTP reply should normally omit the third optional part of the multipart/report (used to return the original message or its headers in the SMTP context).

2.5.2 Generalized Receipts and Signed Receipts

In addition, a generalized receipt using the multipart/report [REPORT] and multipart/signed containing a multipart/report as the signed data is allowed. The basic structure of the multipart/report is used so that the first part is a "human-readable" message concerning the received message. The second part should be for automated process utilization so should at least possess some common Internet syntax for expressing names and values, such as the [SMTPMSG] header syntax, XML, or some MIME content type correlated with automated processing. The MDN requirements are removed for this second part but fields used in MDNs may be used here. The third part of the multipart report is usually omitted in the HTTP context, but would include the extension headers when used to provide diagnostic hints. A multipart/signed over a multipart/report is constructed precisely in the same way as a multipart/signed over a MDN [AS1].

To indicate a desire for a generalized receipt (which may be fulfilled by a MDN), the following metadata elements are to be included in the original message in either the extended header format or the form-data format:

The Receipt-Disposition-To metadata element contains an identifier identifying the party to receive positive notification of delivery. This is usually the same value contained in the "From" metadata element.

The Receipt-report-type metadata element is used by the sender to request a specific type of general acknowledgement receipt. At present only one report type is defined, GISB-acknowledgement-receipt. The content and format of this report type is defined in section 2.5.3 of this document.

Receipt-delivery-option is either the key word (case-insensitive) value "default" or a URL that indicates how and where the receipt is to be delivered. While the default mode of operation within HTTP transport is to return the receipt (be it MDN, signed MDN, generalized report receipt, or signed report receipt) in the reply body, asynchronous reply is

allowed through use of this metadata value.
The "MAILTO", "HTTP", and "HTTPS"
will be the more common schemes found in the URL value.

Receipt-security-selection is a name that indicates the
protocol and algorithm choices for a digital signature
over the receipt. Signatures are always in multipart/signed
packages. The format for protocol and algorithm choices is
that used in [AS1] and [MDN].

One metadata element should, if at all possible,
be within the automated part.
This is the Received-Content-MIC (also allowing X-Received-Content-MIC).
This value is constructed and formatted as described in [AS1] and
the syntax should be either RFC822:

```
Received-Content-MIC: w7AguNJEmhF/qIjJw6LnnA==,rsa-md5
```

or simple XML

```
<ReceivedContentMic algid=rsa-md5 encode=base64 >
 w7AguNJEmhF/qIjJw6LnnA==
</ReceivedContentMic>
```

Any original metadata thought useful to include in the automated part
may be reflected back using "Original-X", as in

```
Original-Message-ID: <43141asfioufasd@somewhere.com>
```

Otherwise the automated acknowledgement semantics are left open
to further semantic specification by specific electronic commerce
communities, such as in [GISB]. Each specialization of the
generalized receipt should make use of a specific identifying
value to be placed in the parameter "report-type,"

```
Content-Type: multipart/report;
              report-type="organizationalid";
              boundary="=-Trfds88fd99"
```

2.5.3 Example of GISB Acknowledgement Receipt

An "Acknowledgement Receipt" contains information indicating the
success/failure of a file transfer. Acknowledgement Receipts are
communicated on the same session connection as the HTTP POST, by the
receiving party's system, immediately after receiving all of the MIME parts
contained in an HTTP POST request. These receipts contain the following
data
elements:
time-c the time of transfer completion at the server. The format
is yyyymmddhhmmss.

request-status a text status indicator by the server. The defined
value

for
a successful transfer is "ok". If an error is identified, the server should supply a descriptive indication of the error detected following the standards for error codes and messages presented in APPENDIX A.

server-id hostname.domainname uniquely identifying the server associated with the program that received and processed the file.

trans-id a number (integer) up to 15 characters in length uniquely identifying the received transaction file at the server. The trans-id will uniquely identify the file only at the receiving server. A client may receive non-unique trans-ids across multiple servers.

The above data elements must be formatted into Name/Value pairs with "=" separating Names from Values (e.g. server-id=www.mydomain.com) and "*" separating each Name/Value pair (e.g. time-c=19990804090000*request-status=ok*server-id=www.me.com*trans-id=12345*). The receipt can be identified as either text/plain or text/html. If text/html is used the set of all name/value pairs must be encapsulated in <html> </html> tags.

The report-type parameter value for the Acknowledgement Receipt is "GISB-Acknowledgement-Receipt," and a case insensitive match is used for testing the value.

See Appendix A.4 and A.5 for formatted examples.

2.5.4 Example of GISB Error Notification

Though a file may be received correctly initially, and a positive "Acknowledgement Recipt" has been delivered, errors can occur during a later
stage of processing, such as decryption. When a file passes the decryption step no notifying communication is sent back to the original sender. However,
if the decryption step fails, an Error Notification must be sent to the original sender of the file. In general, the standard format for Error Notification applies to the posting of an error message after the sender's session has been disconnected. This Error Notification has the potential of occurring only after the original HTTP Response is returned with an "ok" or a warning.

Error Notifications are sent using HTTP POST, the same method used to send EDI
files. The sender of a Error Notification must adhere to the specifications in section 2 of this document, using a value of "error" for the input-format
data element. Error Notifications are sent in cleartext (non-encrypted) within the input-data element and contain the following data elements:

orig-from The "from" value from the original transmission

orig-to	The "to" value from the original transmission.
orig-input-format	The "input-format" value from the original transmission.
resp-time-c	The "time-c" value from the original Acknowledgement Receipt.
resp-server-id	The "server-id" value from the original Acknowledgement Receipt
resp-trans-id	The "trans-id" value from the original Acknowledgement Receipt.
request-status	The new status of the transaction based on the occurance of an error during a later stage of processing; see APPENDIX A, "Standard Error Codes and Messages" for a listing of possible values.
comments	Any comments the original receiving server wishes to include.

The above data elements must be formatted into Name/Value pairs with "=" separating Names from Values (e.g. server-id=www.mydomain.com) and "*" separating each Name/Value pair (e.g. time-c=19990804090000*request-status=ok*server-id=www.me.com*trans-id=12345*). The data elements must be formatted as text/plain.

The report-type parameter value for the Error Notification is "GISB-Error-Notification," and a case insensitive match is used for testing the value. " The extension header "Receipt-delivery-option" (or its form data correlate) may be used to indicate the desired destination for any GISB-Error-Notifications.

See Appendix A.6 for a formatted example.

2.6 Additional Reply Content

In general, both HTTP servers and HTTP clients should be prepared to process the basic EDIINT data formats when they are embedded

within MIME multiparts. This is true for HTTP request payloads as well as HTTP reply payloads.

So, as previosuly mentioned, for HTML-based POSTS,

any of the EDIINT templates described in [AS1], sections 4.2.1 to 4.2.4 or 4.3.1 to 4.3.4 for PGP/MIME or S/MIME security, may be found as parts of a multipart/form-data.

In addition, the response to the POST operation may include other MIME wrapped content besides an MDN Receipt, Signed MDN, Generalized Report Receipt or Signed Report Receipt. If a receipt was requested within the POST data, and additional content is to be returned, the receipt multipart/report must be combined with the other data using some MIME multipart pattern.
Real-time EDI processing systems may use MIME multipart content-types to include a response EDI message, for example, a Quote in response to a Request-For-Quote transaction.

Also, if requested, the sender may request an asynchronous mode for return of receipt. This mode is indicated by including the metadata for Receipt-delivery-option as explained above and choosing a URL rather than the value "default".

2.7 Non-Repudiation of the POST Reply

If the reply to a POST operation needs a MDN receipt for non-repudiation (for example, the reply includes content other than a receipt), the top-level headers in the response include the same headers required for POST data described above: Disposition-Notification-To, Message-ID, From, and To. Other headers described above used in a MDN should be included, for example, Date and Subject.

The MDN receipt of the response data is returned using a subsequent POST operation. A POST operation used only to transmit an MDN should not include the Disposition-Notification-To receipt request, and only a 200 ("OK") response would be expected.

An MDN in response to a reply may be combined with a subsequent EDI message sent with a POST operation, for example a Purchase-Order transaction in response to a Quote. The MIME multipart/mixed form is used to combine the MDN with the other data, the same as for a POST reply.

2.8 Error Recovery

If the HTTP client fails to read the HTTP server response data, the POST operation with identical content (including

Message-ID) should be repeated, if the error condition is transient. The Message-ID on a POST operation can be reused if and only if all of the content (including the original Date) is identical.

Details of the retry process -- including time intervals to pause, number of retries to attempt, timeouts for retrying -- are implementation dependent.

Servers should be prepared to receive a POST with a repeated Message-ID. The MIME reply body previously sent should be resent, including the MDN and other MIME parts.

3. Referenced RFCs and their contribution

3.1 RFC 2068 [HTTP] : The HyperText Transfer Protocol, HTTP, provides an application level protocol for distributed hypermedia information systems. This standard specifies the protocol HTTP/1.1.

3.2 RFC 2246 [TLS] : Transport Layer Security
Security specifies a protocol similar to SSL version 3 that provides communications privacy over the Internet. Applications can communicate without eavesdropping, tampering, or message forgery.

3.3 RFC 1847 [SECURITY] : MIME Security Multiparts

This document defines security multiparts for MIME: multipart/encrypted and multipart/signed.

3.4 RFC 1892 [REPORT] : Multipart/report

This RFC defines the use of the multipart/report content type that the MDN RFC 2298 [MDN] presupposes.

3.5 RFC 1767 [MIMEEDI] : EDI Content

This RFC defines the use of content type "application" for ANSI X12 (application/EDI-X12), EDIFACT (application/EDIFACT) and mutually defined EDI (application/EDI-Consent).

3.6 RFC 2015 [MIMEPGP] : PGP/MIME

This RFC defines the use of content types "multipart/encrypted", "multipart/signed", "application/pgp encrypted" and "application/pgp-signature" for defining MIME PGP content.

3.7 RFC 2045, 2046, and 2049 [MIME] : MIME

These are the basic MIME standards, upon which all MIME related RFCs build, including this one. Key contributions include definition of "content type", "sub-type" and "multipart", as well as encoding

guidelines, which establishes 7-bit US-ASCII as the canonical character set to be used in Internet messaging.

3.8 RFC 2298 [MDN] : Message Disposition Notification

This RFC defines how a message disposition notification (MDN) is requested, and the format and syntax of the MDN. The MDN is the basis upon which receipts and signed receipts are defined in this and in [AS1].

3.9 RFC 2311 [SMIMEV2] : S/MIME Version 2 Message Specification
This specification describes how MIME shall carry PKCS7 1.5 envelopes.

3.10 RFC 2633 [SMIMEV3] : This specification updates formats and procedures for combining cryptographic encryption and signature services with MIME processing. See also [CMS].

3.11 RFC XXXX X [AS1] : MIME-based Secure EDI
This applicability statement describes security patterns, MIME content types, and acknowledgement policies and mechanisms for EDI or business data transport.

3.12 RFC 2388 [FORMDATA] : This specifies a standard Internet Media Type useful for returning a set of values as the result of a user filling out a form.

4. Other differences to notice in HTTP and SMTP based transport

For HTTP version 1.1, TCP persistent connections are the default, ([HTTP] sections 8.1.2, 8.2, and 19.7.1).

A number of other differences exist because HTTP does not conform to MIME [MIME] as used in SMTP transport. Relevant differences are summarized below.

4.1 Unused MIME headers and operations

4.1.1 Content-Transfer-Encoding not used in HTTP transport

HTTP can handle binary data and so there is no need to use the Content transfer encodings of MIME [MIME]. This difference is discussed in [HTTP] section 19.4.4.

4.1.2 Epilogue must be empty

The EBNF for a multipart [MIME] RFC 2046, section 5.1.1 allows a multipart to have trailing octets after the close delimiter. In [HTTP] section 3.7.2, it is explicitly noted that multiparts must have null epilogues.

4.1.3 Lengthy message bodies

In [AS1], section 5.4.1, options for large file processing are discussed for SMTP transport. For HTTP, large files should be handled correctly by the TCP layer. However, [HTTP] sections 3.5 and 3.6 discuss some options for compressing or chunking entities to be transferred. Section 8.1.2.2 discusses a pipelining option that is useful for segmenting large amounts of data.

4.2 Differences in MIME or other headers or parameters used

4.2.1 Content-Length

Because connections are persistent, closing a connection cannot be used to indicate the end of an entity. Therefore, [HTTP] sections 4.4 and 14.14 indicate the need for a Content-Length entity header in a request.

4.2.2 Final and Original Recipient

The final and original recipient distinction should not arise for HTTP transport because SMTP aliases and mailing lists should not be used.

4.2.3 Message-Id and Original-Message-Id

The Message-Id and Original-Message-Id distinction should not arise for HTTP transport because SMTP MTA alterations should not occur.

4.2.4 Host header

The host request header field must be included in the POST request made when sending business data. This field is to allow one server IP address to service multiple hostnames, and potentially conserve IP addresses. See [HTTP], sections 14.23 and 19.5.1.

5. Acknowledgments

Carl Hage, Karen Rosenfeld, Chuck Fenton, Dick Brooks, and many others have provided valuable suggestions improving this applicability statement.

6. References

[MIME] N. Borenstein, N.Freed, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, December 02, 1996.

N. Borenstein, N.Freed, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, December 02, 1996.

N. Borenstein, N.Freed, "Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples", RFC 2049 , December 02, 1996.

[MIMEEDI] D. Crocker, "MIME Encapsulation of EDI Objects", RFC 1767, March 2, 1995.

[SMTPMSG] D. Crocker, "Standard for the Format of ARPA Internet Text Messages", STD 11, RFC 822, August 13, 1982.
(Also RFC 1123 provides important updates on date and time formats as well as email addresses.)

[MIMEPGP] M. Elkins, "MIME Security With Pretty Good Privacy (PGP)", RFC 2015, Sept. 1996.

[MDN] R. Fajman, "An Extensible Message Format for Message Disposition Notifications", RFC 2298, March 1998.

[SECURITY] J. Galvin, S. Murphy, S. Crocker, N. Freed, "Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted", RFC 1847, Oct. 3, 1995

[SMTP] J. Postel, "Simple Mail Transfer Protocol", STD 10, RFC 821, August 1, 1982.

[SMIMEV2] S. Dusse, P. Hoffman, B. Ramsdell, L. Lundblade, L. Repka, "S/MIME Version 2 Message Specification", RFC 2311.

[SMIMEV3] B. Ramsdell, "S/MIME Version 3 Message Specification", RFC 2633, June 1999.

[CMS] R. Housley, "Cryptographic Message Syntax", RFC 2630, June 1999.

[REPORT] G. Vaudreuil, "The Multipart/Report Content Type for the Reporting of Mail System Administrative Messages", RFC 1892, January 15, 1996.

[HTTP] R. Fielding, J.Gettys, J. Mogul, H. Frystyk, T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2068, January 1997.

[AS1] T. Harding, R. Drummond, "MIME-based Secure EDI", Internet draft: draft-ietf-ediint-as1-10.txt.

[TLS] T. Dierks,C. Allen, "The TLS Protocol Version 1.0" RFC 2246, January 1999.

[FORMDATA] L. Masinter, "Returning Values from Forms: multipart/form-data", RFC 2388, August, 1998.

[HTML] D. Raggett, A. Le Hors, I. Jacobs. "HTML 4.0 Specification", World Wide Web Consortium Technical Report

"REC-html40", December, 1997. <http://www.w3.org/TR/REC-html40/>

[GISB] Gas Industry Standards Board, "Electronic Delivery Mechanism Related Standards", Version 1.3 July 31, 1998

[MIME-TYPES] "Media Types," http://www.isi.edu/in-notes/iana/assignments/media-types/media-types.

[EDIINT] T. Harding, R. Drummond , "Requirements for Interoperable Internet EDI",
Internet draft: draft-ietf-ediint-req06.txt.

7. Authors' Addresses

Dale Moberg
dale_moberg@stercomm.com
Sterling Commerce
4600 Lakehurst Ct.
Dublin, OH 43016 USA

Dick Brooks
Group 8760
110 12th Street North
Suite F103
Birmingham, Alabama 35203
Tel: 205-250-8053
E-mail: dick@8760.com

Rik Drummond
drummond@onramp.net
The Drummond Group
5008 Bentwood Ct.
Ft. Worth, TX 76132 USA

Chuck Shih

Appendix A. Example Exchanges.

NOTE: Examples are provided as illustration only.
If the example conflicts with the previous text,
the example is wrong.

Example A.1

Sending a multipart signed for trading partner 1 back to
trading partner 2. "#" indicates a comment line.

```
POST https://tp2server.company2.com/cgi-bin/tp1drawer.pl HTTP/1.1
Host: tp2server.company2.com
```

```
From: tp1@company1.com
To: tp2@company2.com
Date: Tue, 06 Nov 2001 12:53:01 UT
Subject: Purchase orders for 6 November 2001
Message-Id: <20011106@company1.com>
Disposition-Notification-To: tp1@company1.com
# continuation lines not used in actual HTTP protocol data unit
Content-Type: multipart/signed; boundary="20011106RsXgYlvCNW" ;
 protocol=application/pkcs7-signature;  micalg=rsa-md5
Content-Length: 3056

--20011106RsXgYlvCNW
Content-Type: application/edi-x12
Content-Disposition: Attachment; filename=rfc1767.dat
Content-Length: 2605

ISA ...
# EDI transaction data
IEA  ...
--20011106RsXgYlvCNW
Content-Type: application/pkcs7-signature
Content-Length: 804

# omitted binary data
--20011106RsXgYlvCNW--
```

Example A.2
Returning a signed MDN (using the previously established TLS security) from trading partner 2 back to trading partner 1.
"#" indicates a comment line.

```
HTTP/1.0 200 OK
Server: HTTPEDI/1.1
Content-type: multipart/signed;
Content-Length: 1200

--boundary1
Content-type: multipart/report
Content-length: 1133

--boundary2
Content-type: text/plain
Content-length: 85

Message <20011106@company1.com> was authenticated;
EDI processing was initiated.
--boundary2
Content-type: message/disposition-notification
Content-length: 213

Reporting-UA: Company2UA
```

```
Original-Message-Id: <20011106@company1.com>
Original-Recipient: tp2@company2.com
X-Received-Content-MIC: w7AguNJEmhF/qIjJw6LnnA==,rsa-md5
Disposition: MDN-sent-automatically/processed

--boundary2--

--boundary1
Content-Type: application/pkcs7-signature
Content-Length: 560

# Signature data omitted
--boundary1--
```

Example A.3 PGP ENCRYPTED/SIGNED EDI Object sent using HTTP POST (ref [AS1], section 4.2.4 enveloping using multipart/form-data).
The entire payload contained within the multipart/encrypted part is treated as one opaque object.

```
POST /cgi-bin/receiver.cgi HTTP/1.1
Connection: Keep-Alive
User-Agent: Group 8760 WinBB  (Win98; I)
Host: localhost:2600
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, */*
Accept-Encoding: gzip
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8
Content-type: multipart/form-data;
boundary=---------------------------222875935764
Content-Length: 1373

-----------------------------222875935764
Content-Disposition: form-data; name="from"

FROM1234
-----------------------------222875935764
Content-Disposition: form-data; name="to"

TO9876
-----------------------------222875935764
Content-Disposition: form-data; name="input-format"

x12
-----------------------------222875935764
Content-Disposition: form-data; name="Receipt-report-type"

GISB-acknowledgement-receipt
-----------------------------222875935764
```

```
Content-Disposition: form-data; name="input-data";
filename="D:\GISBLite\as2testtextfile.aes"
Content-Type: multipart/encrypted; boundary=8760; protocol="application/pgp-
encrypted"

--8760
Content-Type: application/pgp-encrypted

Version: 1

--8760
Content-Type: application/octet-stream

-----BEGIN PGP MESSAGE-----
Version: PGP 6.5

hQCMAzRG1pEOIOvdAQP+JMr0m/9+8yOL60Z9Vr6fFV81FCExB/o0xmwiMkiwYsHs
z0e8sb7ErC340MrNA/dw3taGMjmI+CXYRF/PLEdg1NZE1ZCtNeL4YdIHAMLWwODG
lQxhSucz8rMSgQ5mZzcOJwBdWLW70efgsu/9UljuJjYc1uZ6C03eFQv/43fkB+al
ATtgydxX4g8QK664ad+Jo/XUICSmWBL66fqJR1KLeLf4wTaqGy174Aq48Wpwvg1E
h785zC03UAw0qg0ugMt86dPeyd91e2JigqwDYEf/DYEKD0J9BGiGpS/uAupNKj8O
cp2IWClxKOGUbxpVNOnNTqWHS/GntegvDE/7/ewCxDxsnmQS95pOl141QZ1RQbeN
aqx2Dq/ra9g65HNchOCzjul5Vi8HHf6Yhg2WnROe+npByyCue6rihqgNVOJwj0cV
zpb4JE+gMDf3q4ISUb1Fv7/+SSFHDdnhdC5YTpqf1Bc3B07hiLmtTXqNit31EbX9
UVElObzSa9ZhxbC6/eSl7Nuf5ZTDsh9nrk+QQJ6FeC9W4cqXLj7IZySaRO8Vtff+
4ktqeuhYusT4kSpnk027aw4O/5jomUkfb22CAe4=
=Oiuo
-----END PGP MESSAGE-----
--8760
----------------------------222875935764--
```

Example A.4 An Acknowledgement Receipt Indicating Errors.

```
Content-Type: multipart/report;  report-type="GISB-Acknowledgement-Receipt";
 boundary="GISB7866"

--GISB7866
Content-type: text/html

<HTML><HEAD><TITLE>Acknowledgement Receipt Error</TITLE></HEAD> <BODY><P>
time-c=19960619082855*
request-status=EEDM106: Invalid To Common Code Identifier*
server-id=coolhost*
trans-id=234423897*
</P> </BODY></HTML>
--GISB7866
Content-type: text/plain

time-c=19960619082855*
request-status=EEDM106: Invalid To Common Code Identifier*
```

```
server-id=coolhost*
trans-id=234423897*
--GISB7866--
```

Example A.5 An Acknowledgement Receipt Indicating Success

```
Content-Type: multipart/report;  report-type="GISB-Acknowledgement-Receipt";
 boundary="GISB7867"

--GISB7867
Content-type: text/html

<HTML><HEAD><TITLE>Acknowledgement Receipt Success</TITLE></HEAD> <BODY><P>
time-c=19960619082855*
request-status=ok*
server-id=coolhost*
trans-id=234423897*
</P> </BODY></HTML> --GISB7867
Content-type: text/plain

time-c=19960619082855*
request-status=ok*
server-id=coolhost*
trans-id=234423897*
--GISB7867--
```

Example A.6 A GISB Error Notification

```
POST URL HTTP/1.1
Referer: http://www.upload.com/upl.htm
Connection: Keep-Alive
User-Agent: brow v0.1 XYZ Corp.
Host: localhost
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*
Content-type: multipart/form-data; boundary=---------------------------
87453838942833
Content-Length: 1958

-----------------------------87453838942833
Content-Disposition: form-data; name="from"

234567890
-----------------------------87453838942833
Content-Disposition: form-data; name="to"

123456789
-----------------------------87453838942833
Content-Disposition: form-data; name="input-format"
```

```
error
-----------------------------87453838942833
Content-Disposition: form-data; name="input-data"; filename=c:\temp\error.not
Content-Type: multipart/report;  report-type="GISB-Error-Notification";
 boundary="GISB7868"

--GISB7868
Content-type: text/html

<HTML><HEAD><TITLE>Error Notification</TITLE></HEAD> <BODY><P>
orig-from=123456789*
orig-to=234567890*
orig-input-format=X12*
resp-time-c=19960619102855*
resp-server-id=coolhost*
resp-trans-id=234423897*
request-status=EEDM601: Public Key Invalid*
comments=Please contact 1-800-555-1212 for correct public key*
</P> </BODY></HTML>

--GISB7868
Content-Type: text/plain

orig-from=123456789*
orig-to=234567890*
orig-input-format=X12*
resp-time-c=19960619102855*
resp-server-id=coolhost*
resp-trans-id=234423897*
request-status=EEDM601: Public Key Invalid*
comments=Please contact 1-800-555-1212 for correct public key*
--GISB7868--
-----------------------------87453838942833--
```

Error Notification Standard Error Codes and Messages

Codes beginning with EEDM### indicate standard error format with ### representing a numeric value.

Codes beginning with WEDM### standard warning format with ### representing a numeric value.

The string for the error or warning should appear in the following format:

Validation Code: Description; supplemental message to be defined by the issuing site up to 80 characters.

The supplemental message is senders option, only the Validation Code and Description are required.

Example:

EEDM100:Missing from Common from required Code Identifier

Listing of Standard Error Codes and Messages

CODE	MESSAGE
EEDM100	Missing from Common from required Code Identifier
EEDM101	Missing to Common Code to required Identifier
EEDM102	Missing input format input-format required
EEDM103	Missing data file input-data required
EEDM104	Missing transaction set transaction-set mutually agreed
EEDM105	Invalid from Common from required Code Identifier
EEDM106	Invalid to Common Code to required Identifier
EEDM107	Invalid input format input-format required
EEDM108	Invalid transaction set transaction-set mutually agreed
EEDM109	No parameters supplied parameter required string
EEDM601	Public key invalid file itself required - security
EEDM602	File not encrypted file itself required - security
EEDM603	Encrypted file truncated file itself required - security
EEDM604	Encrypted file not signed or signature not matched
EEDM699	Decryption Error required for general decryption errors not specifically identified above
EEDM999	System error (required for general system errors to indicate severe errors in processing at the receiving site)
WEDM100	Transaction set sent not transaction-set mutually agreed mutually agreed