RECOMMENDATION TO GISB EXECUTIVE COMMITTEE

Requester: Systrends **Request No.: R01016**

1. Recommended Action:

_Accept as requested
_X_ Accept as modified below
__Decline

Effect of EC Vote to Accept Recommended Action:

_X Change to Existing Practice
_Status Quo

2. TYPE OF MAINTENANCE

Per Request:	**Per Recommendation:**
_Initiation	_Initiation
_Modification	_Modification
___Interpretation	___Interpretation
___Withdrawal	___Withdrawal
_Principle (x.1.z)	_Principle (x.1.z)
___Definition (x.2.z)	___Definition (x.2.z)
_X Business Practice Standard (x.3.z)	__X Business Practice Standard (x.3.z)
___Document (x.4.z)	___Document (x.4.z)
___Data Element (x.4.z)	___Data Element (x.4.z)
___Code Value (x.4.z)	___Code Value (x.4.z)
___X12 Implementation Guide	___X12 Implementation Guide
_X Business Process Documentation	__X Business Process Documentation

3. RECOMMENDATION

SUMMARY: Accept the modified request to supplement the use of PGP 2.6.2 compliant products, with the IETF developed open standard for PGP, called OpenPGP, as defined in RFC 2440,on a mutually agreed basis (http://www.ietf.org/rfc/rfc2440.txt).

RECOMMENDATION TO GISB EXECUTIVE COMMITTEE

Requester: Systrends **Request No.: R01016**

STANDARDS LANGUAGE:

See attached documentation.

4. SUPPORTING DOCUMENTATION

a. Description of Request:

Original request:

Please consider replacing the requirement to use proprietary PGP 2.6.2 compliant products, with the IETF developed open standard for PGP, called OpenPGP, as defined in RFC 2440 (http://www.ietf.org/rfc/rfc2440.txt).

b. Description of Recommendation:

The EDM Subcommittee evaluated a modified request to supplement the use of PGP 2.6.2 compliant products, with the IETF developed open standard for PGP, called OpenPGP, as defined in RFC 2440,on a mutually agreed basis. After considerable discussion, the Request was approved. The vote was not unanimous, and the attendance at the Subcommittee meeting limited.

c. Business Purpose:

OpenPGP is a non-proprietary protocol for encrypting documents using public key cryptography. It is based on PGP as originally developed by Phil Zimmermann. The OpenPGP specification defines standard formats for encrypted messages, signatures, and certificates for exchanging public keys, using open industry standard, freely available software.

Beginning in 1997, the OpenPGP Working Group was formed in the Internet Engineering Task Force (IETF) to define this standard that had formerly been a proprietary product since 1991. Over the past decade, PGP, and later OpenPGP, has become the de-facto standard for cryptography.

By becoming an IETF standard (RFC 2440), OpenPGP may be implemented by any company without paying any licensing fees to anyone.

The OpenPGP Alliance brings companies together to pursue a common goal of promoting the same standard for document encryption and to apply the PKI that has emerged from the OpenPGP community.

d. Commentary/Rationale of Subcommittee(s)/Task Force(s):

See minutes of the EDM Subcommittee for 2/28/02, 3/06/02, 3/11/02, and 3/13/02.

CHANGES TO STANDARD 4.3.15

TRADING PARTNERS SHOULD IMPLEMENT ALL SECURITY FEATURES (SECURE AUTHENTICATION, INTEGRITY, PRIVACY, AND NON-REPUDIATION) USING A FILE-BASED APPROACH VIA A COMMERCIALLY AVAILABLE IMPLEMENTATION OF PGP 2.6 OR GREATER (OR COMPATIBLE WITH PGP 2.6) OR, ON A MUTUALLY AGREED BASIS, AN OPENPGP COMPATIBLE PRODUCT. TRADING PARTNERS SHOULD ALSO IMPLEMENT BASIC AUTHENTICATION. THESE TECHNOLOGIES SUPPORT ALL OF THE ABOVE SECURITY FEATURES WHILE PROVIDING INDEPENDENCE OF CHOICE OF WEB SERVERS AND BROWSERS. ENCRYPTION KEYS SHOULD BE SELF-CERTIFIED AND THE MEANS OF EXCHANGE SHOULD BE SPECIFIED IN THE TRADING PARTNER AGREEMENT. ENCRYPTION KEYS SHOULD HAVE A LIMITED LIFETIME WHOSE DURATION IS DETERMINED BY THE KEY'S OWNER. A KEY'S END OF LIFE IS EXPRESSED IN THE EXPIRATION DATE FIELD CONTAINED IN EACH PUBLIC KEY. A LIFETIME OF ONE YEAR OR LESS IS RECOMMENDED.

[CHANGES TO TAB 6 OF THE EDM MANUAL]

TECHNICAL IMPLEMENTATION - INTERNET EDI/EDM & BATCH FF/EDM

...

SECURITY

Security Concepts

The security requirements include the current four primary security aspects: data privacy, data

integrity, authentication, and non-repudiation.

- Data privacy: unauthorized parties cannot decipher the content of the data.
- Data integrity: unauthorized parties cannot modify or corrupt the data.
- Authentication: the receiver is certain of the identity of the sender.
- Non-repudiation: the sender cannot deny ownership of the transaction if it was sent with his/her digital signature.

In general, these needs are met by using the Basic Authentication capability of the Web server and the encryption and digital signature capability of the PGP and OpenPGP security application for securing transactions.

Understanding PGP or OpenPGP

Pretty Good Privacy (PGP) is the name of the chosen security application. OpenPGP is the Internet Engineering Task Force standard version of PGP which excludes all patented algorithms, allowing free commercial use of the standard. See the GISB home page for information on software packages to implement the PGP or OpenPGP security application. Both OpenPGP and PGP utilize a public key/private key pair to accomplish secure file transfers. The private key must be known only to the company which generated it. The public key counterpart is shared with trading partners.

Deleted: s

Each company must generate its public key and private key pair. The RSA key generation algorithm should be chosen for versions of PGP which offer alternatives. Implementers of OpenPGP should choose DSA and El Gamal when creating their key pair. The public keys will be distributed using a secure method (eg., courier mail) to the company's trading partners. You must use the utmost care in protecting your private key. If it is compromised, the security is broken. It is recommended that a key size of 1024 be chosen when generating the key pair. This provides a significantly secure transaction.

When a company wishes to send transactions to its trading partner, it will use the partner's public key to encrypt the file. Encryption provides data privacy. Only the private key counterpart can decrypt this file. Hence, the need to guard your private key.

When the sending party encrypts the file, it also uses its own private key to "sign" the transaction. The receiving party can use the sender's public key to verify the signature. The digital signature provides non-repudiation.

Deleted: ¶

Encryption / Digital Signature

Encryption and signatures are applied to files already translated to a GISB standard data format, and before the data is sent to the batch browser." (Use of internal encryption such as X12.58 encryption is outside the scope of GISB encryption standards but does not conflict with PGP.)

Encryption and signatures can be accomplished manually for each file using the on-line PGP or, on a mutually agreed basis, OpenPGP software, or in an automated (or "batch") fashion using programs to encrypt and sign. Whether encrypting in a manual or automated fashion, it is essential that the correct public key of the trading partner be used to encrypt and just as essential that the correct sender's own private key be used to digitally sign the file.

Digital signatures may also be applied, on a mutually agreeable basis, to the HTTP response by the receiver of the transaction.

Decryption / Signature Verification

After a transaction is received and processed by the CGI program, it is ready to be decrypted and have its signature verified. PGP and OpenPGP software will utilize the appropriate key pair when encrypting, signing, and decrypting if given the correct userID in the key ring identifying the trading partner. Upon request for signature verification, the PGP and OpenPGP software will return a human-readable company name.

It is recommended that all implementors create a process where the name is used to look up the ID of the company in a database table. If the ID is passed along with the decrypted file, a process could be created to verify that the company which sent the transaction corresponds to the company identified within the file, once the data has been translated.

When digital signatures are applied, on a mutually agreeable basis, the HTTP response received by the sender of the transaction may be verified to ensure non-repudiation of receipt of the transaction.

Throughput Considerations

Encryption, digital signing, decryption and signature verification are all very CPU intensive. It is not recommended that decryption or signature verification be performed within the CGI that receives and processes the file. In fact, it would not be a good idea to have these steps performed on the same computer that is attempting to receive transactions at a time close to a deadline. Therefore, it is recommended that the secured or to-be-secured transaction be passed to a separate computer for security processing. This "passing" would likely be accomplished by using the File Transfer Protocol (FTP). The security processing computer should be optimized for CPU and memory.

Implementers of Internet EDM sites should review and evaluate Domain Name Server (DNS) cache refresh intervals so as to ensure trading partner address changes are recognized on a timely basis. A refresh interval of 24 hours or less is common.

Because decryption and signature verification are not handled at the time the file is received, the sender will get an HTTP response of successful transfer but doesn't know if the file can be decrypted by the receiver. Guidelines for communicating the status of the decryption step have been developed. See Section "Sending Error Notification Transactions" and Table A, "Internet EDM Standard Error Codes and Messages".

Security Requirements

Basic Authentication

Basic authentication, also known as realm one security, has been defined as one of the security standards for transmission on the Internet. The userid and password will be assigned by the server party according to site standards. The trading party agreement must identify the userid and password for this security as well as procedures for changing the password, if applicable.

PGP or OpenPGP File Encryption

File encryption of the EDI file is also selected as a security standard for transmission on the Internet. The encryption software employed is required to be compatible with PGP 2.6 or greater (using keys generated with the RSA algorithm) or the OpenPGP standard, specified in IETF RFC 2440, on a mutually agreed basis. There are freely available software implementations the OpenPGP standard available at http://www.gnupg.org/.

General Security Recommendations

Firewall

A firewall is one or more computers running special software which is designed to provide control of communications between two networks. Its purpose is to limit the types of services between these two networks. Often, a company's connection to the Internet is intended to provide several other services to its employees who are connected by an internal network such as a Local Area Network or Wide Area Network (LAN or WAN). Examples of these services include access to the World Wide Web, use of e-mail, use of file transfer capabilities and publishing content intended for viewing by the external world on a Web server. In addition, the internal network will likely have connections to host computers which provide internal services such as file and print sharing, fax and database capabilities. So that availability of these services and confidential internal data are not compromised by unwelcome intruders from the Internet, there should exist a protective mechanism between the internal network and the public Internet, the firewall.

There are two general mechanisms employed by firewalls to provide this control: packet filtering and proxy services. Packet filtering examines important components of the messages such as the address of the sending and target computers and the designator (port number) for a specific application running on the target computer. By doing this, it can prevent access to specific computers or programs on those computers. It can also reject messages from certain computers. Proxy servers have various capabilities. They can act as relay agents that can examine attempted use of certain features within an application thus limiting access to these features. They can also hide (by substituting its own address) the internal addresses of clients communicating with external hosts. This hiding makes it difficult for potential attackers to focus on specific internal hosts.

Because firewalls are designed to deal with a broad set of security issues, which may vary at each organization, and are not specific to the use of HTTP, this guide does not attempt to provide specific implementation information. Deciding on a specific firewall architecture, organizational security policies, and choosing between numerous products may require outside resources to address these issues.

SENDING ERROR NOTIFICATION TRANSACTIONS

Error Notification

When a client sends a file to a server, the server responds to the receipt of the file. Though the file may be received correctly, some further processing must be done, such as decryption and X12 translation. The decryption step which will have a pass/fail status and then the X12 general translation step which will have a pass/fail status. The X12 general translation is merely the check that the file meets the X12 standards and has not been corrupted. Further translation and processing of specific transactions and elements is outside the Internet EDM scope.

When a file passes the decryption step and passes the general translation step, no notifying communication is sent back to the client. However, if either the decryption step or the general translation step fails, an error notification must be sent to the client.

In general, this standard format for error notification applies to the posting of an error message after 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 (WEDM999 format) for the request-status value, not an error (EEDM999).

Additionally, trading partners are permitted to utilize digitally signed error notifications, if both parties

mutually agree to do so.

Error Notification Data Elements

The data elements for the error notification are the same as those described in Section "Sending Transactions", with the exception of the "input-format" and "input-data" elements. The file containing the data elements for error notification should not be encrypted.

Required Data Elements for Error Notification (listed in the required order)

Data Element Name	Description	
from	Common Code Identifier of sending/client company, the server company which detected the error	
to	Common Code Identifier of receiving/server company, the client company which sent the data set in error	
input-format	"error"	
input-data	A text block containing the following items:	
	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 response.
	resp-server-id	The "server-id" value from the original response.
	resp-trans-id	The "trans-id" value from the original response.
	request-status	The new status of the transaction based on some process beyond CGI such as decryption; see Table A, "Internet EDM Standard Error Codes and Messages".
	comments	Any comments the original receiving server wishes to include.

Mutually Agreed Upon Data Elements for Error Notification

none defined at this time

Error Notification "input-data" Element Specifications:

The file containing the data elements for error notification should not be encrypted.

All data element names will be in lower case in the Error Notification.

Carriage returns and line feeds will be ignored in all files.

A field delimiter of "*" will be used in the Error Notification. Please refrain from displaying a "*" anywhere else in the error notification so as not to confuse programs that need to parse on this basis.

No spaces should surround the equal sign or the field delimiter.

The required data elements must appear first in the response.

Additional information can be included after the required elements at the server's discretion.

The first occurrence of the field name within the response will contain the value.

If an error notification is given, a GISB Error Notification contains two body parts nested within a multipart/report outer envelope. The first body part contains human readable content in HTML. The second body part contains machine readable content in HTML. Additionally, consenting trading partners can mutually agree to digitally sign error notifications. If digital signatures are used, the multipart/report containing the GISB Error Notification is used to create a digital signature body part, identified by a content-type of application/pgp-signature. Both the multipart/report GISB Error Notification and application/pgp-encrypted digital signature body parts are combined in a multipart/signed envelope.

Error Notification Example:

```
POST /cgi-bin/AS2dispatcher HTTP/1.1
Referer: http://www.get.a.life/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="version"

1.4
------------------------------87453838942833
Content-Disposition: form-data; name="receipt-disposition-to"

123456789
------------------------------87453838942833
Content-Disposition: form-data; name="receipt-report-type"

gisb-acknowledgement-receipt
------------------------------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—

Signed Error Notification

Content-Type:multipart/signed; micalg=pgp-md5; protocol="application/pgp-signature";
boundary=8760

--8760

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--
--8760

```
Content-Type: application/pgp-signature
-----BEGIN PGP MESSAGE-----

Version: 2.6.2

iQCVAwUBMJrRF2N9oWBghPDJAQE9UQQAtl7LuRVndBjrk4EqYBIb3h5QXIX/L
C//JV5bNvkZIGPIcEmI5iFd9boEgvpirHtlREEqLQRkYNoBActFBZmh9GC3C041
WGquMbrbxc+nIs1TIKlA08rVi9ig/2Yh7LFrK5Ein57U/W72vgSxLhe/zhdfolT9BrnH
OxEa44b+EI=
=ndaj

-----END PGP MESSAGE-----

--8760--.
```

Pre-validation before Decryption

Proper trapping of the range of decryption process errors listed in Table A (Internet EDM Standard Error Messages and Codes) may require program code which is external to the decryption algorithm. Some versions of the PGP software do not explicitly discriminate between EEDM601, EEDM602, EEDM603, and EEDM699 type errors. Under such a circumstance, files inbound to the decryption process should be preprocessed to trap the errors not identified by the PGP version being used. For example, searching the file for the text strings "BEGIN PGP MESSAGE" and "END PGP MESSAGE" can quickly identify "EEDM602 File not encrypted" and "EEDM603 Encrypted file truncated" type errors when the implemented PGP version only identifies decryption success, invalid public key (EEDM601), and decryption failure (EEDM699).

Deleted: i

CHECKLIST OF TESTING STEPS

Purpose

Preliminary steps in testing are helpful before the full batch browser and server applications are completed. This checklist is intended to provide a series of small achievements leading up to the complete solution.

Client/Browser

NOTE: Throughout all transfer tests, compare files stored on the server against the source file to ensure that the file transferred intact. While transferring to another company's server, you may have to contact that company to send the file back to you so that you can perform the compare.

1. Install an interactive browser. Identify an existing Web server from among GISB compliant servers offering interactive upload for test. See the GISB home page for a list of organizations willing to act as testing partners. These organizations should have a URL complete with the CGI program name to which a tester may send test files. File content does not need to be X12 or other GISB standard format to accomplish this step in testing.
2. Develop or acquire a batch browser that uses multipart for the encoding methodology. Transfer the same test file as in step 1 to the URL not requiring Realm One security.
3. Add Realm One security to your file transfer, and change the URL to the secure URL. Continue transfer tests with your batch browser.
4. Acquire and install PGP or OpenPGP compliant software. Generate your public and private key pair. Make sure to choose the RSA key generation algorithm for PGP or DSA and El Gamal

for OpenPGP. Download the test server's test public key. Encrypt your data file using this key. Modify your file transfer to send the encrypted file. Continue transfer tests. Request that the test server contact decrypt your file.

HTTP Server and CGI

1. Install Web server. Establish an Internet connection to your server. Ensure that you have ample storage space for transferred files. Ensure that permissions are granted to the directories.
2. As an optional preliminary step, acquire or develop an HTML page for interactive file upload (sample code is earlier in this document). Test interactive file upload to your own server using an interactive browser.
3. Acquire or develop a CGI program to receive file transfers and process according to GISB standards. Test transfers to your CGI using your batch browser.
4. Transfer a X12 or other GISB standard format dataset to your server and process it through your translator or other appropriate processes.
5. Copy the CGI to a "secure" directory where Realm One security, or basic authentication, is enabled. Using your batch browser, transfer to both URLs, with and without authentication. Thoroughly test using the incorrect userid and password against the secure directory.
6. Generate a second public/private key pair. Use the second key to encrypt a file and transfer the file to your server. Decrypt the file.
7. Once your site security is established, contact a trading partner to test transfers against your server.
8. Test with various file sizes to ensure that your CGI can process small and large files.
9. Request that several other trading partners and/or several clients within your own company transfer concurrently to ensure that your server can withstand the load.
10. Test application with various simulated errors in both file transfers and in PGP or OpenPGP decryption.

FREQUENTLY ASKED QUESTIONS

As an end user, do I need a continuously connected internet Web server to participate in the Internet EDM in the gas industry, or can I just use a dial-up connection to my ISP and my favorite shrink-wrapped browser software?

An interactive browser connection is not enough to actively participate in the system. It is not necessary to have a private Web server, you can use a service, however the system requires that you have access to a permanent internet connection which is capable of both sending and receiving files (with CGI or BGI) without operator intervention.

If we use ANSI X12.58 encryption do we still need to use PGP or OpenPGP encryption?

The use of internal encryption such as X12.58 is outside the scope of the GISB encryption standards.

TABLE A - Internet EDM Standard Error Codes And Messages

These errors and warnings are strictly related to problems found in the recipient CGI or decryption levels of processing before translation. Errors and warnings generated by the client batch browser are assumed to be documented at the client site to distinguish them from problems occurring in the recipient CGI or decryption. Numbering schemes and descriptions should aid in this distinction.

Note: For HTTP error codes see the GISB home page for information sources.

EEDM### standard error format with ### representing a numeric value further processing will not take place

WEDM### standard warning format with ### representing a numeric value further processing will take place

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

Internet EDM Standard Error Codes and Messages

Validation Code	Description	Data Element	Required vs. Mutually Agreed
EEDM100	Missing "from" Common Code Identifier code	From	required
EEDM101	Missing "to" Common Code Identifier	To	required
EEDM102	Missing input format	input-format	required
EEDM103	Miss ing data file	input-data	required
EEDM104	Missing transaction set	transaction-set	mutually agreed
EEDM105	Invalid "from" Common Code Identifier	From	required
EEDM106	Invalid "to" Common Code Identifier	To	required
EEDM107	Invalid input format	input-format	required
EEDM108	Invalid transaction set	transaction-set	mutually agreed
EEDM109	No parameters supplied	parameter string	required
EEDM110	Invalid "version"	Version	required
EEDM111	Missing "version"	Version	required
EEDM112	"receipt-security-selection" not mutually agreed	receipt-security-selection	mutually agreed
EEDM113	Invalid "receipt-security-selection"	receipt-security-selection	mutually agreed

Validation Code	Description	Data Element	Required vs. Mutually Agreed
EEDM114	Missing "receipt-disposition-to"	receipt-disposition-to	required
EEDM115	Invalid "receipt-disposition-to"	receipt-disposition-to	required
EEDM116	Missing "receipt-report-type"	receipt-report-type	required
EEDM117	Invalid "receipt-report-type"	receipt-report-type	required
EEDM118	Missing "receipt-security-selection"	receipt-security-selection	mutually agreed
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 by PGP or OpenPGP messages or exit codes
EEDM701	EDM party not associated with EDI party		required
EEDM702	Data Structure Error		required if the translator does not handle this exception
EEDM703	Data set exchange not established for Trading Partner		required if the translator does not handle this exception
EEDM999	System error		required for general system errors to indicate severe errors in processing at the receiving site
WEDM100	Transaction set sent not mutually agreed	transaction-set	mutually agreed
WEDM102	"receipt-security-selection" not mutually agreed	receipt-security-selection	mutually agreed
WEDM103	Missing "receipt-security-selection"	receipt-security-selection	mutually agreed

[CHANGES TO TAB 4 OF THE EDM MANUAL]

Page 7

Security

Though many decisions as to overall security measures are left to each trading partner and their environment, several security measures were established as standards to ensure a minimum level of confidence in conducting business over the Internet and to provide some uniformity in the implementation of security. Four primary security aspects were considered as vital in providing the level of protection of transactions needed for gas industry commerce: data privacy, data integrity, authentication, and non-repudiation. The FTTF found that these concerns are addressed by the use of encryption and digital signature capability of the Pretty Good Privacy (PGP) security application. Any process used for encryption and decryption compatible with PGP 2.6 (using keys generated with the RSA algorithm) meets the minimum standard to be applied to files transmitted over the Internet. Additionally, the OpenPGP standard, defined by IETF RFC 2440, is a mutually agreed upon, supported alternative to PGP 2.6. Implementers of the PGP product should consider upgrading to PGP version 6.5 for compatibility with the OpenPGP standard and all previous versions of PGP. To prevent unwanted intruders from connecting to the Web sites, basic authentication is the required standard. Additional issues such as firewall security are discussed in the standards, but are considered implementation issues to be addressed by each organization.

Page 16 - Conforming change to Standard 4.3.15

Trading partners should implement all security features (secure authentication, integrity, privacy, and non-repudiation) using a file-based approach via a commercially available implementation of PGP 2.6 or greater (or compatible with PGP 2.6) or, on a mutually agreed basis, an OpenPGP compatible product. Trading partners should also implement basic authentication. These technologies support all of the above security features while providing independence of choice of Web servers and browsers. Encryption keys should be self-certified and the means of exchange should be specified in the trading partner agreement. Encryption keys should have a limited lifetime whose duration is determined by the key's owner. A key's end of life is expressed in the expiration date field contained in each public key. A lifetime of one year or less is recommended.

[CHANGES TO TAB 10 OF THE EDM MANUAL]

Page 2 (Appendix A)

• • •

PGP Software

PGP is available for a variety of operating systems and platforms. For more information contact Network Associates (http://www.nai.com)

OpenPGP Software

The IETF OpenPGP standard is available at http://www.ietf.org/rfc/rfc2440.txt

Software implementations of the OpenPGP standard are freely available for commercial use from the Free Software Foundation at http://www.gnupg.org.

Time Synchronization

Testing has shown that the clocks on all computer systems drift. It has also been surprising to see

• • •

Page 3 (Appendix B)

• • •

Does the HTTP sender match the party who encrypted and signed the file?

The next validation, determining that the HTTP sender is the same as the signer, requires that the following information be available:

The 'from' common code identifier (9 digit D-U-N-S® Number). This is the second field in the HTTP post message sent to the CGI. This information must be preserved from that earlier process and passed to the 'post-CGI' process.

The Pretty Good Privacy (PGP) or OpenPGP User ID associated with that same party

To compare these items a 'table' would most likely be established that would allow the post-CGI process to identify that there is a correlation between these identifiers. The origin of the 'from' identifier is the HTTP POST 'from' field. The origin of the PGP or OpenPGP user ID is the decryption process. The PGP or OpenPGP User ID of the signer is a byproduct of file decryption on a signed file. If PGP or OpenPGP software is executed from the command line the output would be presented in a format like:

Good signature from user "ENRON CORP".
Signature made 1997/05/13 19:30 GMT
Plaintext filename: test3

If PGP or OpenPGP software is executed using a program interface the User ID that signed the file will be provided in a buffer. Comparing this buffer to the expected User ID would serve to verify this value.

...