Global Information Assurance Certification Paper

Email – Exploring The Encrypted Solution

Neil Lindberg

October 4, 2001

Overview:

In the world of eavesdroppers, hackers, and now apparently high-tech terrorists[1], we need to combine as many defensive elements of communications as we can muster in order to thwart (or at least slowdown) these persistent adversaries. Of all communications I use daily, none is used as often as *email*. When communicating with telnet or ftp, there are many corresponding encrypted solutions available such as, the OpenSSH suite of applications containing ssh (telnet) and the accompanying scp (ftp), which is freely available at www.openssh.com. These solutions require very little additional manipulation by the user and may, therefore, be considered "user friendly".

That was the premise of my search for the encrypted email solution. What I was looking for was an easily integrated, widely accessible, "user friendly" feature for my corporate and home email. Also important is using an application that will encrypt and send mail and not be dependant on the mailer program receiving to message. A proprietary mail program that will only exchange encrypted mail by its own unique solution will not suffice. It should be standardized.

Lets start at the beginning. A brief history of email from the beginnings of the Internet to present follows.

Brief History:

2In 1982, an ASCII email capability was formulated by the IETF (Internet Engineering Task Force) through a Request For Comments, RFC 822. It specified the format in which a message should be constructed. It should contain a header and body sections. The header was defined in detail, but the body was assumed to contain only ASCII text. This worked well until the arrival of multi-media mail. Now it was necessary to send files that were not plain ASCII text. These were files that contained audio or image files. This caused the proliferation of proprietary formatted email from applications that would send enhanced messages that could be read only by users with the same application. Other email applications would turn it into meaningless garbled characters. By 1993, these limitations were unacceptable. Another document, RFC 1521, was prepared to propose another standard. This introduced the MIME standard which all email applications were to conform to. This new standard would allow numerous objects that were not text, to become part of the message body. These objects could be audio, images, or non-ASCII text. And now this standard has been taken even further to include security specifications. Enter the dreaded proliferation of multiple methods in an attempt to standardize the security aspects of encrypted messaging. Also listed are the encryption algorithms employed by the protocols.

The Players (Protocols):

This secure description then became S/MIME (Secure MIME). It is important to note here that there exist other and equally capable extensions. Another protocol is also being developed by an IMC (Internet Mail Consortium) workgroup called PGP/MIME (OpenPGP). [3]Then there are other less known protocols such as, PEM (Privacy Enhanced Mail), MOSS (Mime Object Security Services), command line PGP, PKCS#7, CMS, and XML. Following are the descriptions of protocols that were or are in use. The first two are the prevailing protocols in use today and are followed by the earliest to the latest.

1. S/MIME:

Starting in 1995, RSA Data Security, working with a private consortium, began developing an extension to MIME. This extension was named S/MIME. It was to provide authentication and privacy for email. Today the Internet Mail Consortium shows the current status of S/MIME up to its current version 3. Along with the related RFC documents and details. S/MIME version 2 was not IETF standard because of the RSA encryption algorithm was deemed too weak. S/MIME v3 was made an IETF standard in July, 1999. It consists of 4 parts:

- RFC 2630 – Cryptographic Message Syntax
- RFC 2633 – S/MIME Version 3 Message Specification
- RFC 2632 – S/MIME Version 3 Certificate Handling
- RFC 2631 – Diffie-Hellman Key Agreement Method

2. OpenPGP (PGP/MIME):

Based on PGP (Pretty-Good-Privacy), which was developed, and in use since the early '90's. It consists of 2 parts:

- RFC 1991 – PGP Message Exchange Formats
- RFC 2015 – MIME Security with Pretty Good Privacy

It was adopted by some email providers but has not been adopted as an IETF standard in its original form. The main reason for this is that it used the RSA key exchange and the IDEA encryption. Patents cover both of these algorithms and preclude further development of this as a standard.

Currently, IETF's OpenPGP Working Group is doing work on this protocol. This has produced OpenPGP under RFC 2440, which is on the standards track.

Comparison of S/MIME and OpenPGP:

Mandatory Features	S/MIME	OpenPGP
Message Format	Binary, based on CMS	Binary, based on previous PGP
Certificate format	Binary, based on X.509v3	Binary, based on previous PGP
Symmetric encryption algorithm	TripleDES (DES EDE3 CBC)	TripleDES (DES EDE3 Eccentric CFB)
Signature algorithm	Diffie-Hellman (X9.42) with DSS	ElGamal with DSS
Hash Algorithm	SHA-1	SHA-1
MIME encapsulation of signed data	Choice of multipart/signed or CMS format	Multipart/signed with ASCII armor
MIME encapsulation of encrypted data	Application/pkcs7-mime	Multipart/encrypted

Table 1[4]

(Reproduced from http://www.imc.org/smime-pgpmime.html)

Although both protocols are designed to add authentication and privacy to email messages, they differ in many ways. Until there is a single standard for secure Internet mail, both will be developed.

1. **PGP (Command line PGP):** PGP has for years been the de facto standard for email encryption. The PGP application is not dependant on a large infrastructure for use. It is made up of a single binary for encryption, decryption, signing, verification, and key management. Basic PGP however is not suited for a transparent email solution because it lacks MIME integration. It will only work for ASCII mail reliably. The message body contains lines identifying it as PGP signed. This is to be picked up by the email application and processed accordingly.
2. **PEM (Privacy Enhanced Mail):** PEM was the first notable secure email implementation. Designed and specified in the late 1980's, at first PEM relied on symmetric encryption. PEM was specified by RFC 1421-1424. Today, PEM is not widely used due in part to the fact that it was based on a flawed design[7], and PEM's vulnerability to surreptitious forwarding is mostly just a matter of historical interest. By this, it is meant that if a user A sent a message to user C, there is no way to tell if the message has been intercepted by B and then forwarded to C. The vulnerability was introduced in PEM and was propagated to the descendants of this standard. This includes S/MIME, OpenPGP, PGP, PKCS#7, PEM, and MOSS.
3. **MOSS (MIME Object Security Services):** MOSS in its reference implementation is known as TIS/MOSS 7.1. This reference build uses 56-bit DES, which is in direct violation of the standards and is not in current development[3]. It however did offer an elegant and simple means of encryption. Since MOSS was developed as a sort of framework, many different forms have proliferated. Today it is possible for two different implementations of MOSS to not speak to each other. MOSS used PEM as a baseline and added cryptographic support for MIME and a couple of other less notable changes. Like

PEM, MOSS was eclipsed by S/MIME and by PGP, and is little heard-of today.

4. **PKCS#7 (Public Key Cryptography Standard #7 - Cryptographic Message Syntax Version 1.5):** Adapted from PEM. Specified by RFC 2315
5. **CMS (Cryptographic Message Syntax):** Almost identical to PKCS#7 from which it came. This specification defines data formats and procedures for encryption and signing. It is used together within S/MIME (and others) to form the message format. It is specified by RFC 2630

1. **XML (Extensible Markup Language):**

XML is the latest attempt to standardize. The XML workgroup was nearing completion on its standards specification in the spring of 2001. This protocol specification is to eventually provide an "idiot proof" means of security. However, at this time it is not for anyone that is not a security specialist. "An emerging XML-based standard and messaging protocol called Security Assertion Markup Language (SAML) is designed to facilitate the secure exchange of authentication and authorization information between partners regardless of their security or e-commerce platforms."[9]

Algorithms:

Along with the different methods of encrypting email, there are a number of encryption algorithms in use by email protocols. The following are a few of the symmetric algorithms in use today[6].

1. **Blowfish:** one of the fastest. Variable length key, from 32 to 448 bits
2. **DES (Data Encryption Standard):** Fairly fast. Uses a 56-bit key to each 64-bit block of data. Adopted by the Department of Defense. It is capable of being broken. "While this code has been cracked, it is still considered a safe method of encryption, as breaking the DES code takes a considerable amount of time".
3. **IDEA (International Data Encryption Algorithm):** Uses a 128-bit key. It is patented but available for free usage.
4. **3DES (Triple DES):** Applies the DES algorithm 3 times with separate keys. This makes it extremely slow.

Digital Certificates:

While investigating this subject, there were many references to Digital Certificates. Digital Certificates are public-private key pair generation. Like the OpenPGP model, these require a key server to be available to the message sender and recipient for the exchange of public keys. A definition of Digital Certificates follows:

"An attachment to an electronic message used for security purposes. The most common use of a digital certificate is to verify that a user sending a message is who he or she claims to be, and to provide the receiver with the means to encode a reply.

An individual wishing to send an encrypted message applies for a digital certificate from a Certificate Authority (CA). The CA issues an encrypted digital certificate containing the applicant's public key and a variety of other identification information. The CA makes its own public key readily available through print publicity or perhaps on the Internet.

The recipient of an encrypted message uses the CA's public key to decode the digital certificate attached to the message, verifies it as issued by the CA and then obtains the sender's public key and identification information held within the certificate. With this information, the recipient can send an encrypted reply. The most widely used standard for digital certificates is X.509." [10]

Hence, the Digital Certificate would be used for encryption using asymmetric encryption. Digital Certificates may be produced by the sender using freely available tools such as "keytool" or obtained from a Certificate Authority (CA) of your choice. Everyone may not accept self-produced Digital Certificates, as a prior trust relationship would have to be established. By obtaining a certificate from a neutral source, the CA, it is more readily acceptable. The whole idea of certificates was to by-pass the public key distribution problem.

The Bottom Line:

At this time, there is not a standard to cover all encrypted email packages to be used "out of the box" to send secure email to everyone you want without having to configure plug-ins or send or obtain "public-keys" with your intended recipient (asymmetric encryption). This would most likely deter the average user from attempting to encrypted messages. The OpenPGP plug-in seems to offer the most diverse coverage for secure messaging and seamlessly integrates into the major packages; such as, Outlook, Outlook Express, Netscape, and Eudora. These mailers also work with S/MIME that uses a digital certificate that provides your signature to your messages.

One of the biggest problems is that there are too many protocols for application programmers to choose from. That problem manifests, as application programmers are not usually cryptographers, who possess the knowledge required to make the decision of which protocol is best[7].

While most email packages offer SSL (now known as TLS) connections between the desktop and mail server machines, this will only provide security for communications between the two and most likely not beyond the current LAN.

Having a VPN installed between local and remote sites is another way of providing secure messaging. This will cause all communications to be encrypted and, thereby, encrypting the email messages along with all the other networks traffic. This works fine in a corporate setting where you have support personnel dedicated to making it happen, but for the home user sending messages to family and friends, it is of little use.

5Currently, the PKI (Public Key Infrastructure) X.509 based digital certificates may offer the required flexibility in encrypting your messages for you "on-the-fly" and remove the problem of key distribution by making available local key servers.

While most of the above-defined protocols employ asymmetric encryption, I have read papers promoting symmetric encryption as the better choice. Enter XML. The XML protocol has been around for a few years and is getting closer to the integrated vision of the future. With XML, applications won't need the understand PKI[8]. The two specifications handle different aspects of the message. There is a workgroup that is integrating XML and PKI into specifications such as XKMS, XML Signature, and SAML that combine PKI and XML to create easy-to-implement, interoperable solutions for trust.

We seem to be on the right track to connect these pieces together. We have the knowledge to obtain our goal. And until encrypted email is as easy to use as a web browser, it will not come into the main stream of communications. Encrypted email should be as available as e-commerce solutions and offer my kids the chance to send secure messages as easily as it does IT professionals. Without this ability there remains a wide hole in the Internet's suite of applications for communications.

Reference:

1. Secret Codes: Authorities Say Bin Laden Using Encryption Software – TechTV
   http://abcnews.go.com/sections/scitech/TechTV/techtv_encryption010925.html
2. E-mail For Everybody: S/MIME Brings Compatibility & Security To Electronic Messages: How The Internet Works – Part I Vol.5, Pages 95-99
   URL: http://www.smartcomputing.com
3. A Brief Comparison of Email Encryption Protocols: Ralph Leven
   URL: http://www.arraydev.com/commerce/JIBC/9603-2.htm
4. S/MIME and OpenPGP: The Internet Mail Consortium
   http://www.imc.org/smime-pgpmime.html
5. Encryption Pipe Dream: Computerworld – Russell Kay, Aug. 13, 2001
   URL: http://www.computerworld.com
6. Protect Your Privacy- Use Encryption To Ensure E-mail Integrity: Maximize PC Performance, July 2001, Vol.9 Issue 7, Pages 133-135
   URL: http://www.smartcomputing.com
7. Defective Sign & Encrypt in S/MIME, PKCS#7, MOSS, PEM, PGP, and XML: Don Davis, June 2001
   URL: http://world.std.com/~dtd/sign_encrypt/sign_encrypt7.html
8. Microsoft, others offer XML-based encryption scheme: InfoWorld - Tom Sullivan and James Evans, November 29, 2000
   URL: http://www.infoworld.com/articles/hn/xml/00/11/29/001129hnxkms.xml
9. Net Infrastructure: XML Standard To Keep Web Services Secure: InternetWeek, Rutrell Yasin, July 2001
10. Webopedia: URL: http://webopedia.internet.com/TERM/d/digital_certificate.html

Further Reading:

1. Secure Messaging: Ron Hilton, November 7, 2000 – SANS Information Security Reading Room
2. Defective Sign & Encrypt in S/MIME, PKCS#7, MOSS, PEM, PGP, and XML: Don Davis, June 2001
   URL: http://world.std.com/~dtd/sign_encrypt/sign_encrypt7.html
3. Distributed XML : The role played by XML in the next-generation Web: Ed Dumbill, September 06, 2000
   URL: http://www.xml.com/pub/a/2000/09/06/distributed.html
4. XML Trust Center – URL: http://www.xmltrustcenter.org/index.htm
5. Internet Mail Consortium – URL: http://www.imc.org

Last Updated: July 22nd, 2017

Upcoming Training

SANS Prague 2017	Prague, Czech Republic	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS New York City 2017	New York City, NY	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS Salt Lake City 2017	Salt Lake City, UT	Aug 14, 2017 - Aug 19, 2017	Live Event
Community SANS Omaha SEC401*	Omaha, NE	Aug 14, 2017 - Aug 19, 2017	Community SANS
Community SANS Trenton SEC401	Trenton, NJ	Aug 21, 2017 - Aug 26, 2017	Community SANS
Virginia Beach 2017 - SEC401: Security Essentials Bootcamp Style	Virginia Beach, VA	Aug 21, 2017 - Aug 26, 2017	vLive
SANS Chicago 2017	Chicago, IL	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
Community SANS Pasadena SEC401 @ NASA	Pasadena, CA	Aug 23, 2017 - Aug 30, 2017	Community SANS
Mentor Session - SEC401	Minneapolis, MN	Aug 29, 2017 - Oct 10, 2017	Mentor
SANS San Francisco Fall 2017	San Francisco, CA	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS Tampa - Clearwater 2017	Clearwater, FL	Sep 05, 2017 - Sep 10, 2017	Live Event
Mentor Session - SEC401	Edmonton, AB	Sep 06, 2017 - Oct 18, 2017	Mentor
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
Mentor Session - SEC401	Ventura, CA	Sep 11, 2017 - Oct 12, 2017	Mentor
Community SANS Albany SEC401	Albany, NY	Sep 11, 2017 - Sep 16, 2017	Community SANS
Community SANS Columbia SEC401	Columbia, MD	Sep 18, 2017 - Sep 23, 2017	Community SANS
Community SANS Dallas SEC401	Dallas, TX	Sep 18, 2017 - Sep 23, 2017	Community SANS
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Copenhagen 2017	Copenhagen, Denmark	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Boise SEC401	Boise, ID	Sep 25, 2017 - Sep 30, 2017	Community SANS
Baltimore Fall 2017 - SEC401: Security Essentials Bootcamp Style	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	vLive
Community SANS New York SEC401	New York, NY	Sep 25, 2017 - Sep 30, 2017	Community SANS
Rocky Mountain Fall 2017	Denver, CO	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Sacramento SEC401	Sacramento, CA	Oct 02, 2017 - Oct 07, 2017	Community SANS
SANS DFIR Prague 2017	Prague, Czech Republic	Oct 02, 2017 - Oct 08, 2017	Live Event
Community SANS Charleston SEC401	Charleston, SC	Oct 02, 2017 - Oct 07, 2017	Community SANS
Mentor Session - SEC401	Arlington, VA	Oct 04, 2017 - Nov 15, 2017	Mentor