Chapter 8

Electronic Mail Security

In virtually all distributed environments, electronic mail is the most heavily used

network-based application. Users expect to be able to, and do, send e-mail to others

who are connected directly or indirectly to the Internet, regardless of host operating

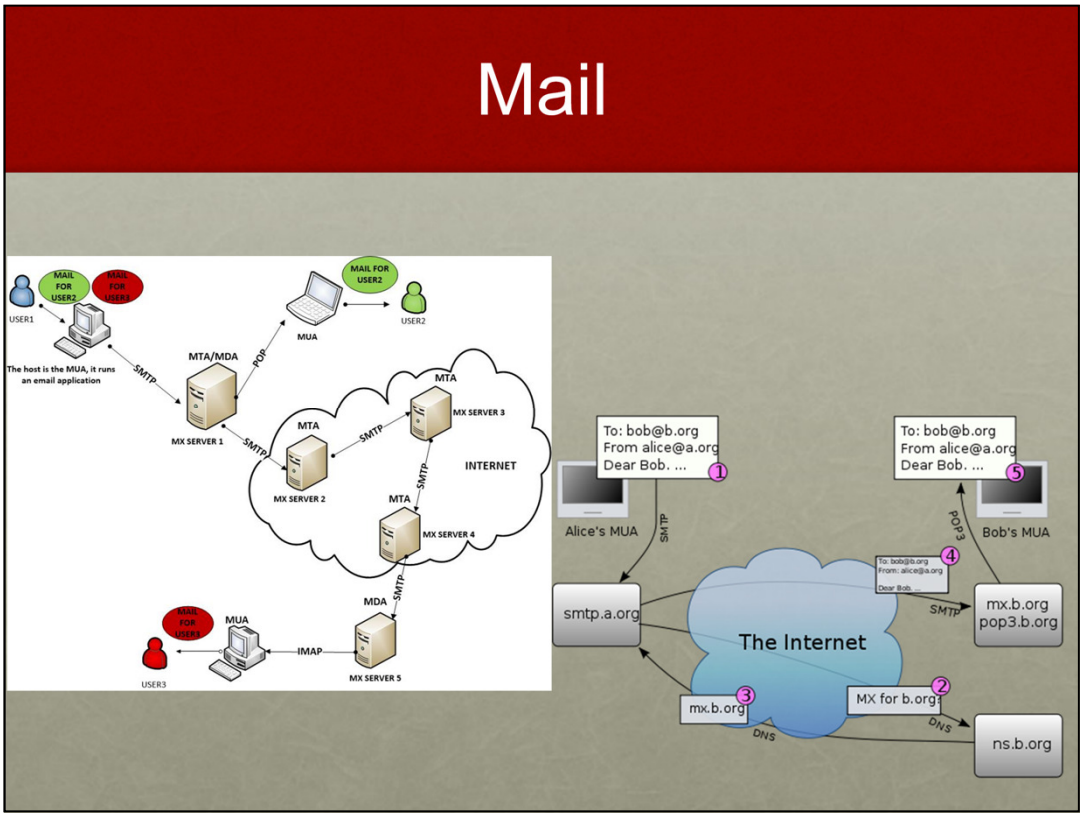
system or communications suite. With the explosively growing reliance on e-mail,

there grows a demand for authentication and confidentiality services. Two schemes

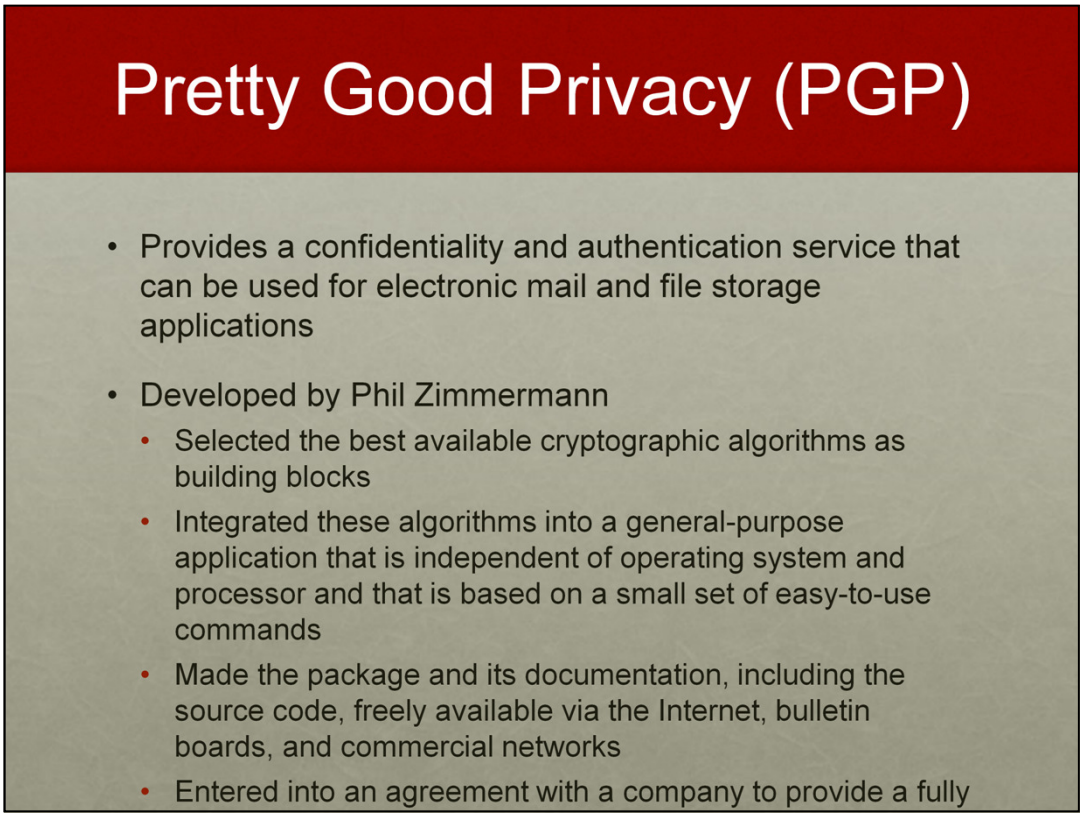
stand out as approaches that enjoy widespread use: Pretty Good Privacy (PGP) and

S/MIME. Both are examined in this chapter. The chapter closes with a discussion of

DomainKeys Identified Mail.



Opening quote.



PGP is a remarkable phenomenon. Largely the effort of a single person, Phil

Zimmermann, PGP provides a confidentiality and authentication service that can

be used for electronic mail and file storage applications. In essence, Zimmermann

has done the following:

1. Selected the best available cryptographic algorithms as building blocks.

2. Integrated these algorithms into a general-purpose application that is independent

of operating system and processor and that is based on a small set of

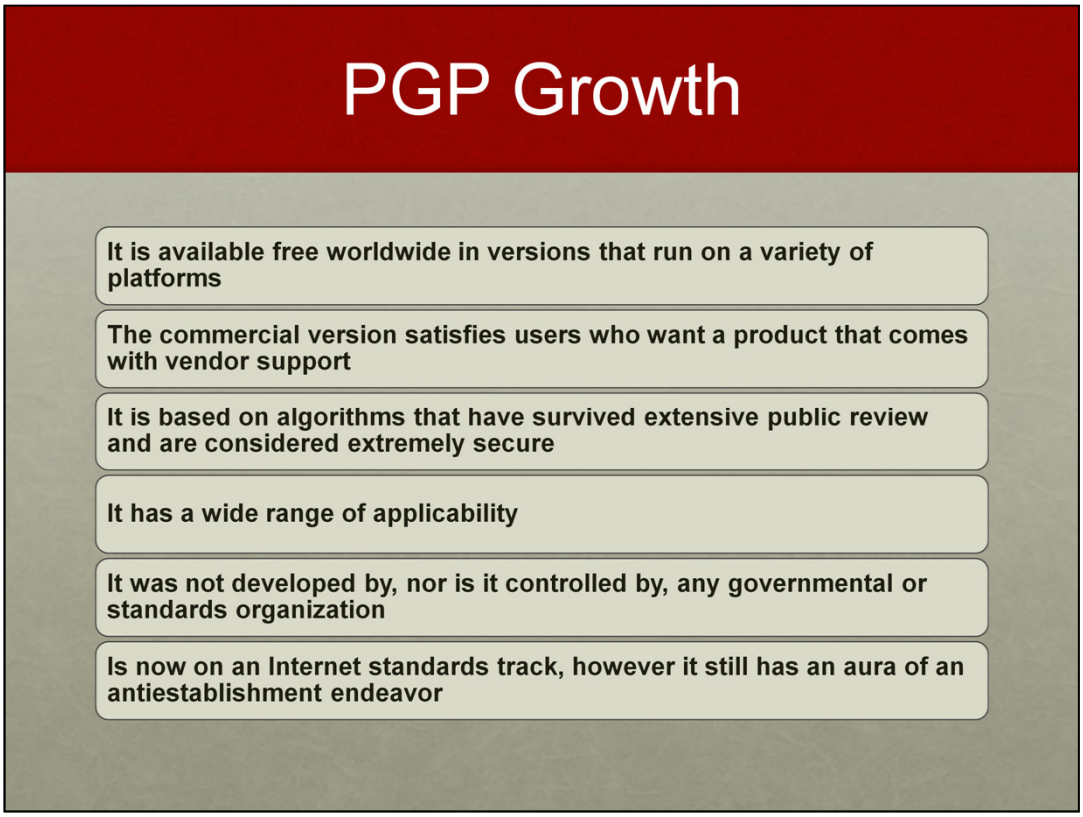
easy-to-use commands.

3. Made the package and its documentation, including the source code, freely

available via the Internet, bulletin boards, and commercial networks such as

AOL (America On Line).

4. Entered into an agreement with a company (Viacrypt, now Network Associates) to provide a fully compatible, low-cost commercial version of PGP.



PGP has grown explosively and is now widely used. A number of reasons can be cited for this growth.

1. It is available free worldwide in versions that run on a variety of platforms, including
Windows, UNIX, Macintosh, and many more. In addition, the commercial version satisfies users who want a product that comes with vendor support.

2. It is based on algorithms that have survived extensive public review and are considered extremely secure. Specifically, the package includes RSA, DSS, and Diffie-Hellman for public-key encryption; CAST-128, IDEA, and 3DES for symmetric encryption; and SHA-1 for hash coding.

3. It has a wide range of applicability, from corporations that wish to select and enforce a standardized scheme for encrypting files and messages to individuals who wish to communicate securely with others worldwide over the Internet and other networks.

4. It was not developed by, nor is it controlled by, any governmental or standards organization. For those with an instinctive distrust of “the establishment,” this makes PGP attractive.

5. PGP is now on an Internet standards track (RFC 3156; MIME Security with

OpenPGP). Nevertheless, PGP still has an aura of an antiestablishment

endeavor.

Table 8.1
Summary of PGP Services

Function	Algorithms Used	Description
Digital signature	DSS/SHA or RSA/SHA	A hash code of a message is created using SHA-1. This message digest is encrypted using DSS or RSA with the sender's private key and included with the message.
Message encryption	CAST or IDEA or Three-key Triple DES with Diffie-Hellman or RSA	A message is encrypted using CAST-128 or IDEA or 3DES with a one-time session key generated by the sender. The session key is encrypted using Diffie-Hellman or RSA with the recipient's public key and included with the message.
Compression	ZIP	A message may be compressed for storage or transmission using ZIP.
E-mail compatibility	Radix-64 conversion	To provide transparency for e-mail applications, an encrypted message may be converted to an ASCII string using radix-64 conversion.

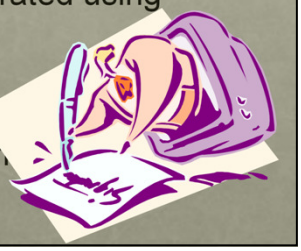
The actual operation of PGP, as opposed to the management of keys, consists of

four services: authentication, confidentiality, compression, and e-mail compatibility

(Table 8.1).

PGP Authentication

- Combination of SHA-1 and RSA provides an effective digital signature scheme
  - Because of the strength of RSA the recipient is assured that only the possessor of the matching private key can generate the signature
  - Because of the strength of SHA-1 the recipient is assured that no one else could generate a new message that matches the hash code
  - As an alternative, signatures can be generated using DSS/SHA-1
  - Detached signatures are supported
  - Each person's signature is independent and therefore applied only to the documen



The combination of SHA-1 and RSA provides an effective digital signature scheme. Because of the strength of RSA, the recipient is assured that only the possessor of the matching private key can generate the signature. Because of the strength of SHA-1, the recipient is assured that no one else could generate a new message that matches the hash code and, hence, the signature of the original message.

As an alternative, signatures can be generated using DSS/SHA-1.

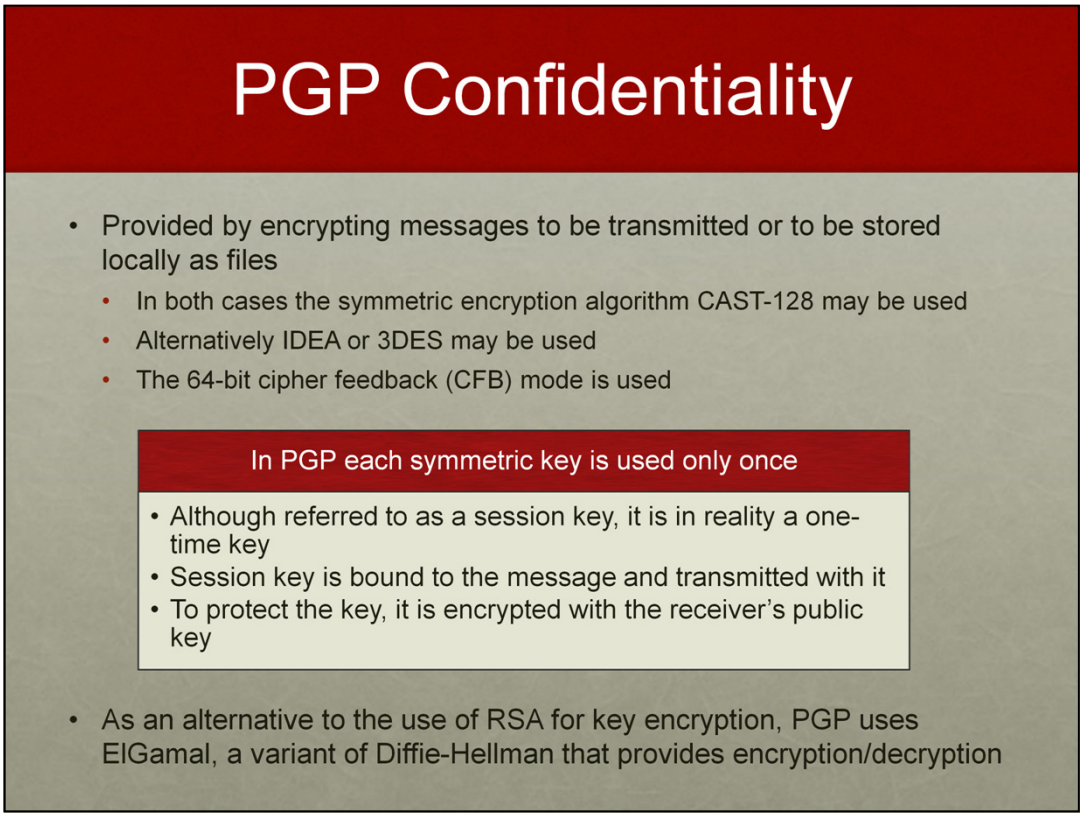
Although signatures normally are found attached to the message or file that they sign, this is not always the case: Detached signatures are supported. A detached signature may be stored and transmitted separately from the message it signs. This is useful in several contexts. A user may wish to maintain a separate signature log of all messages sent or received. A detached signature of an executable program can detect subsequent virus infection. Finally, detached signatures can be used when

more than one party must sign a document, such as a legal contract. Each person’s

signature is independent and therefore is applied only to the document. Otherwise,

signatures would have to be nested, with the second signer signing both the document

and the first signature, and so on.



Another basic service provided by PGP is confidentiality, which is provided by encrypting messages to be transmitted or to be stored locally as files. In both cases, the symmetric encryption algorithm CAST-128 may be used. Alternatively, IDEA or 3DES may be used. The 64-bit cipher feedback (CFB) mode is used.

As always, one must address the problem of key distribution. In PGP, each symmetric key is used only once. That is, a new key is generated as a random 128-bit number for each message. Thus, although this is referred to in the documentation as a session key, it is in reality a one-time key. Because it is to be used only once, the session key is bound to the message and transmitted with it. To protect the key, it is encrypted with the receiver's public key. Figure 8.1b illustrates the sequence.

As an alternative to the use of RSA for key encryption, PGP provides an option referred to as Diffie-Hellman . As was explained in Chapter 3, Diffie-Hellman is a key exchange algorithm. In fact, PGP uses a variant of Diffie-Hellman that does provide encryption/decryption, known as ElGamal.

Several observations may be made. First, to reduce encryption time, the combination

of symmetric and public-key encryption is used in preference to simply

using RSA or ElGamal to encrypt the message directly: CAST-128 and the other

symmetric algorithms are substantially faster than RSA or ElGamal. Second, the

use of the public-key algorithm solves the session-key distribution problem, because

only the recipient is able to recover the session key that is bound to the message.

Note that we do not need a session-key exchange protocol of the type discussed in

Chapter 14, because we are not beginning an ongoing session. Rather, each message

is a one-time independent event with its own key. Furthermore, given the store-andforward

nature of electronic mail, the use of handshaking to assure that both sides

have the same session key is not practical. Finally, the use of one-time symmetric

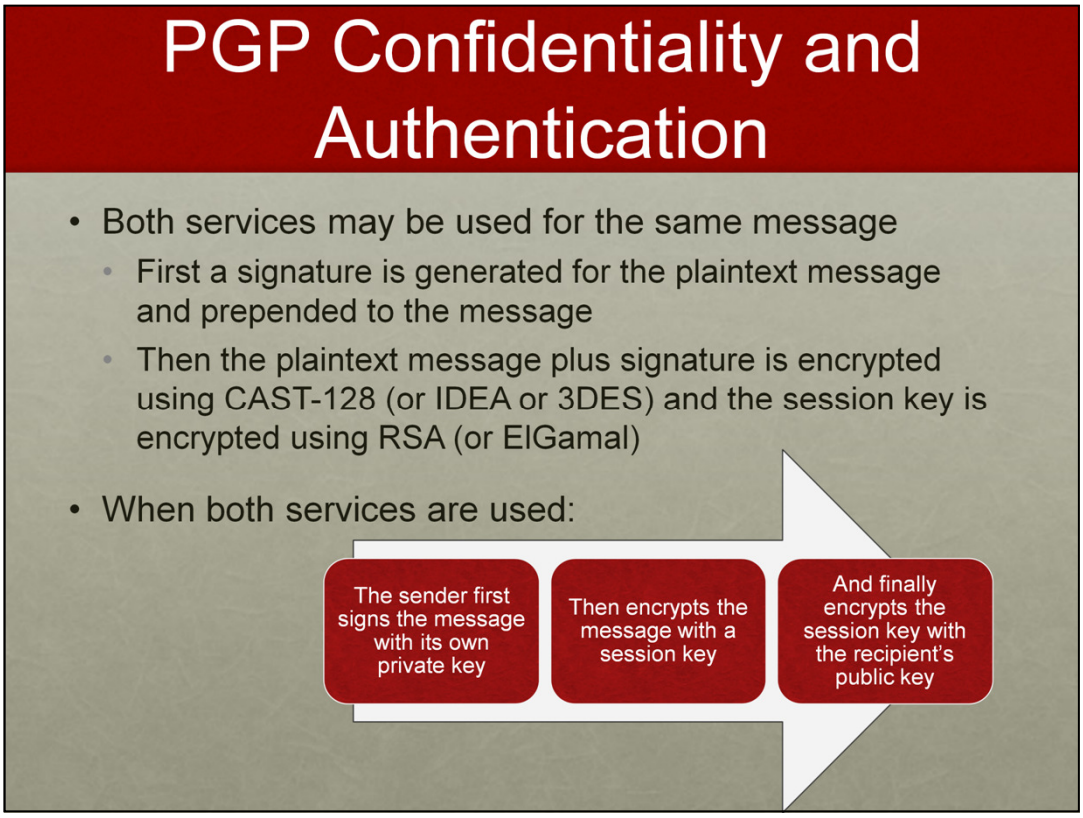
keys strengthens what is already a strong symmetric encryption approach. Only a

small amount of plaintext is encrypted with each key, and there is no relationship

among the keys. Thus, to the extent that the public-key algorithm is secure, the

entire scheme is secure. To this end, PGP provides the user with a range of key size

options from 768 to 3072 bits (the DSS key for signatures is limited to 1024 bits).



As Figure 8.1c illustrates, both services may be used for the same message. First, a signature is generated for the plaintext message and prepended to the message. Then the plaintext message plus signature is encrypted using CAST-128 (or IDEA or 3DES), and the session key is encrypted using RSA (or ElGamal). This sequence is preferable to the opposite: encrypting the message and then generating a signature for the encrypted message. It is generally more convenient to store a signature with a plaintext version of a message. Furthermore, for purposes of third-party verification, if the signature is performed first, a third party need not be concerned with the symmetric key when verifying the signature.

In summary, when both services are used, the sender first signs the message with its own private key, then encrypts the message with a session key, and finally encrypts the session key with the recipient's public key.

PGP Compression

- As a default, PGP compresses the message after applying the signature but before encryption
  - This has the benefit of saving space both for e-mail transmission and for file storage
  - The placement of the compression algorithm is critical
    - Applying the hash function and signature after compression would constrain all PGP implementations to the same version of the compression algorithm
    - Message encryption is applied after compression to strengthen cryptographic security
  - The compression algorithm used is ZIP

As a default, PGP compresses the message after applying the signature but before encryption. This has the benefit of saving space both for e-mail transmission and for file storage.

The placement of the compression algorithm is critical.

1. The signature is generated before compression for two reasons:

a. It is preferable to sign an uncompressed message so that one can store only

the uncompressed message together with the signature for future verification.

If one signed a compressed document, then it would be necessary either to store a compressed version of the message for later verification or to recompress the message when verification is required.

b. Even if one were willing to generate dynamically a recompressed message

for verification, PGP's compression algorithm presents a difficulty. The algorithm is not deterministic; various implementations of the algorithm achieve different tradeoffs in running speed versus compression ratio and, as a result, produce different compressed forms. However, these different compression algorithms are interoperable because any version of the algorithm can correctly decompress the output of any other version. Applying the hash function and signature after compression would constrain all PGP implementations to the same version of the compression algorithm.

2. Message encryption is applied after compression to strengthen cryptographic security. Because the compressed message has less redundancy than the original plaintext, cryptanalysis is more difficult.

The compression algorithm used is ZIP, which is described in Appendix G.

PGP E-mail Compatibility

- Many electronic mail systems only permit the use of blocks consisting of ASCII text
  - To accommodate this restriction, PGP provides the service of converting the raw 8-bit binary stream to a stream of printable ASCII characters
  - The scheme used for this purpose is radix-64 conversion
    - Each group of three octets of binary data is mapped into four ASCII characters
    - This format also appends a CRC to detect transmission errors

When PGP is used, at least part of the block to be transmitted

is encrypted. If only the signature service is used, then the message digest is

encrypted (with the sender's private key). If the confidentiality service is used, the

message plus signature (if present) are encrypted (with a one-time symmetric key).

Thus, part or all of the resulting block consists of a stream of arbitrary 8-bit octets.

However, many electronic mail systems only permit the use of blocks consisting of

ASCII text. To accommodate this restriction, PGP provides the service of converting

the raw 8-bit binary stream to a stream of printable ASCII characters.

The scheme used for this purpose is radix-64 conversion. Each group of three

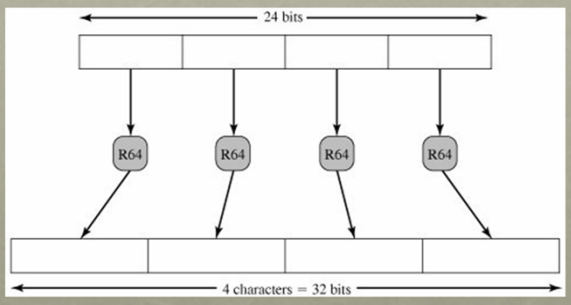
octets of binary data is mapped into four ASCII characters. This format also appends

a CRC to detect transmission errors. See Appendix 19A for a description.

The use of radix 64 expands a message by 33%. Fortunately, the session key
and signature portions of the message are relatively compact, and the plaintext message
has been compressed. In fact, the compression should be more than enough to
compensate for the radix-64 expansion. For example, [HELD96] reports an average
compression ratio of about 2.0 using ZIP. If we ignore the relatively small signature
and key components, the typical overall effect of compression and expansion of a
file of length X would be 1.33 * 0.5 * X = 0.665 * X . Thus, there is still an overall
compression of about one-third.

One noteworthy aspect of the radix-64 algorithm is that it blindly converts the
input stream to radix-64 format regardless of content, even if the input happens to
be ASCII text. Thus, if a message is signed but not encrypted and the conversion
is applied to the entire block, the output will be unreadable to the casual observer,
which provides a certain level of confidentiality. As an option, PGP can be configured
to convert to radix-64 format only the signature portion of signed plaintext
messages. This enables the human recipient to read the message without using PGP.
PGP would still have to be used to verify the signature.

Radix-64



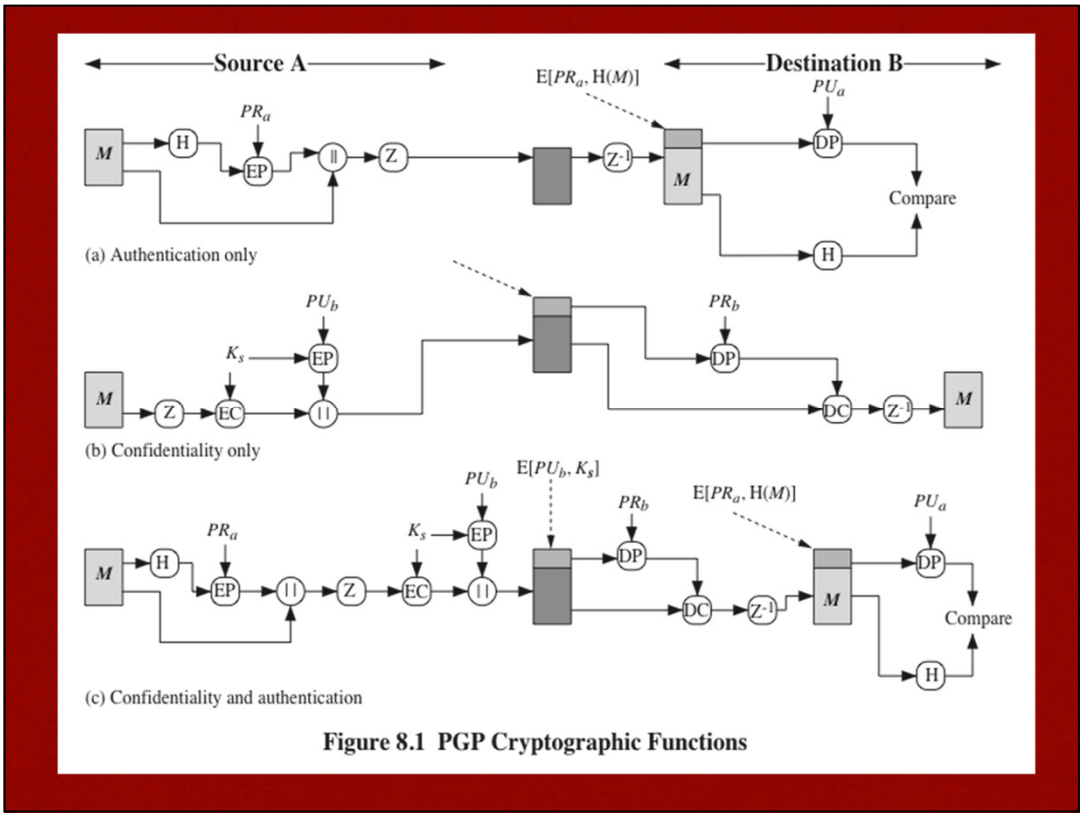


Figure 8.1 PGP Cryptographic Functions

Figure 8.1a illustrates the digital signature service provided by

PGP. This is the digital signature scheme discussed in Chapter 4 and illustrated in

Figure 4.5.

To protect the key, it is encrypted with the receiver's public key. Figure 8.1b illustrates the sequence.

As Figure 8.1c illustrates, both services

may be used for the same message. First, a signature is generated for the plaintext

message and prepended to the message. Then the plaintext message plus signature

is encrypted using CAST-128 (or IDEA or 3DES), and the session key is encrypted

using RSA (or ElGamal). This sequence is preferable to the opposite: encrypting

the message and then generating a signature for the encrypted message. It is generally

more convenient to store a signature with a plaintext version of a message.

Furthermore, for purposes of third-party verification, if the signature is performed

first, a third party need not be concerned with the symmetric key when verifying the

signature.

In summary, when both services are used, the sender first signs the message

with its own private key, then encrypts the message with a session key, and finally

encrypts the session key with the recipient’s public key.

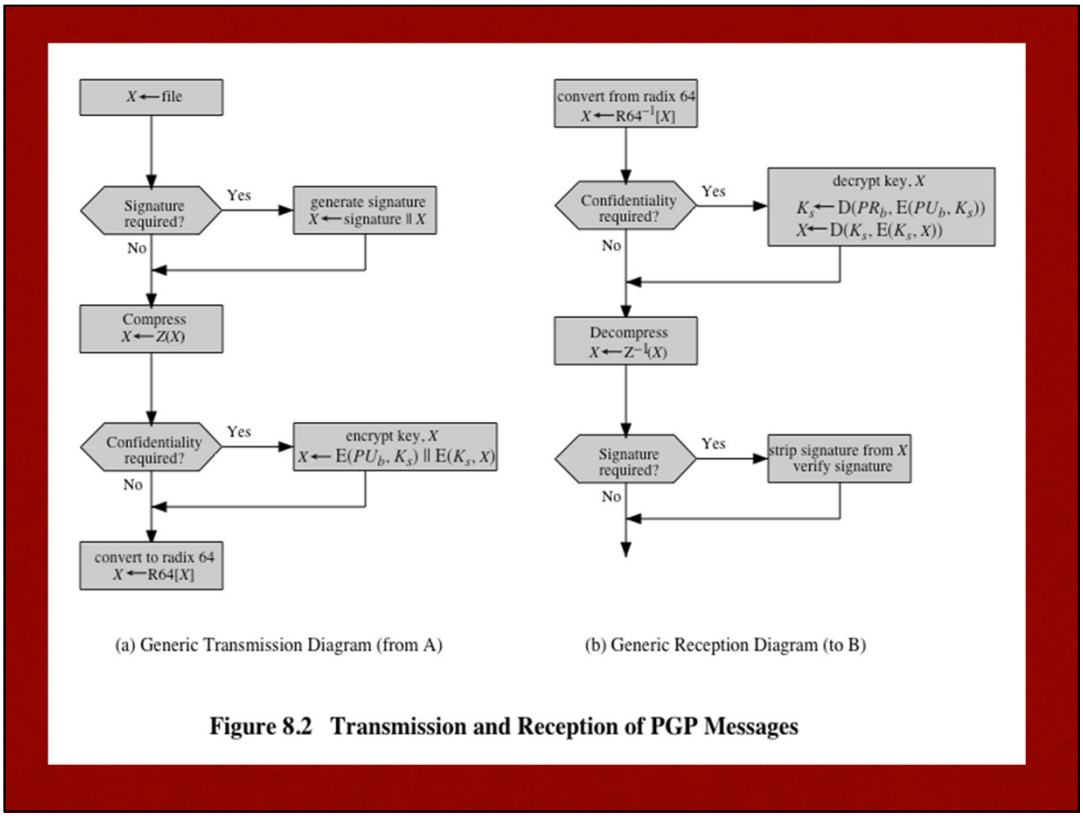


Figure 8.2 Transmission and Reception of PGP Messages

Figure 8.2 shows the relationship among the four services so far discussed.

On transmission (if it is required), a signature is generated using a hash code of

the uncompressed plaintext. Then the plaintext (plus signature if present) is compressed.

Next, if confidentiality is required, the block (compressed plaintext or

compressed signature plus plaintext) is encrypted and prepended with the publickey-

encrypted symmetric encryption key. Finally, the entire block is converted to

radix-64 format.

On reception, the incoming block is first converted back from radix-64 format

to binary. Then, if the message is encrypted, the recipient recovers the session key

and decrypts the message. The resulting block is then decompressed. If the message

is signed, the recipient recovers the transmitted hash code and compares it to its

own calculation of the hash code.

DomainKeys Identified Mail (DKIM)

- A specification for cryptographically signing e-mail messages, permitting a signing domain to claim responsibility for a message in the mail stream
- Message recipients can verify the signature by querying the signer's domain directly to retrieve the appropriate public key and can thereby confirm that the message was attested to by a party in possession of the private key for the signing domain
- Proposed Internet Standard RFC 4871
- Has been widely adopted by a range of e-mail providers and Internet Service Providers (ISPs)

DomainKeys Identified Mail (DKIM) is a specification for cryptographically signing

e-mail messages, permitting a signing domain to claim responsibility for a message

in the mail stream. Message recipients (or agents acting in their behalf) can verify

the signature by querying the signer's domain directly to retrieve the appropriate

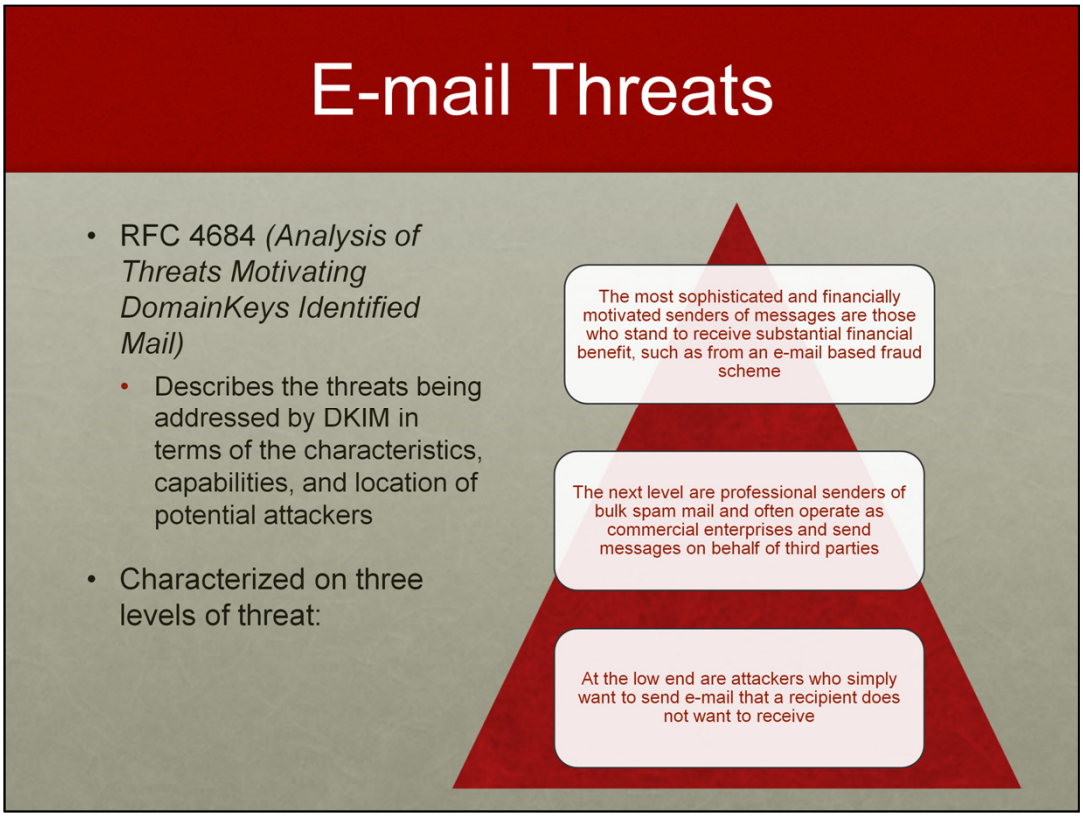
public key and thereby can confirm that the message was attested to by a party in

possession of the private key for the signing domain. DKIM is a proposed Internet

Standard (RFC 4871: DomainKeys Identified Mail (DKIM) Signatures). DKIM has

been widely adopted by a range of e-mail providers, including corporations, government

agencies, gmail, yahoo, and many Internet Service Providers (ISPs).



RFC 4686 characterizes the range of attackers on a spectrum of three levels of threat.

1. At the low end are attackers who simply want to send e-mail that a recipient does not want to receive. The attacker can use one of a number of commercially available tools that allow the sender to falsify the origin address of messages. This makes it difficult for the receiver to filter spam on the basis of originating address or domain.

2. At the next level are professional senders of bulk spam mail. These attackers often operate as commercial enterprises and send messages on behalf of third parties. They employ more comprehensive tools for attack, including Mail Transfer Agents (MTAs) and registered domains and networks of compromised

computers (zombies) to send messages and (in some cases) to harvest addresses to which to send.

3. The most sophisticated and financially motivated senders of messages are those who stand to receive substantial financial benefit, such as from an E-mail-based fraud scheme. These attackers can be expected to employ all of the above mechanisms and additionally may attack the Internet infrastructure itself, including DNS cache-poisoning attacks and IP routing attacks.

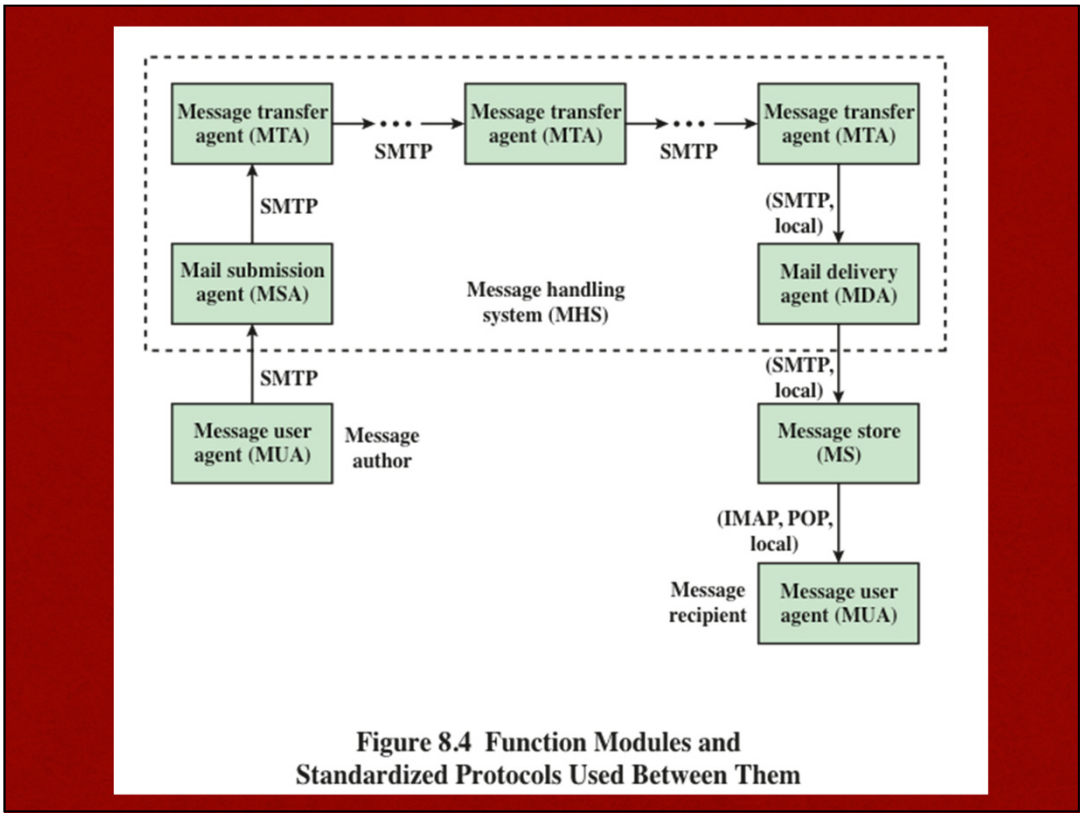


Figure 8.4 Function Modules and Standardized Protocols Used Between Them

To understand the operation of DKIM, it is useful to have a basic grasp of the

Internet mail architecture, which is currently defined in RFC 5598. This subsection

provides an overview of the basic concepts.

At its most fundamental level, the Internet mail architecture consists of a

user world in the form of Message User Agents (MUA), and the transfer world, in

the form of the Message Handling Service (MHS), which is composed of Message

Transfer Agents (MTA). The MHS accepts a message from one user and delivers

it to one or more other users, creating a virtual MUA-to-MUA exchange environment.

This architecture involves three types of interoperability. One is directly between

users: messages must be formatted by the MUA on behalf of the message

author so that the message can be displayed to the message recipient by the destination
MUA. There are also interoperability requirements between the MUA and the
MHS—first when a message is posted from an MUA to the MHS and later when
it is delivered from the MHS to the destination MUA. Interoperability is required
among the MTA components along the transfer path through the MHS.

Figure 8.4 illustrates the key components of the Internet mail architecture, which include the following.

• Message User Agent (MUA): Operates on behalf of user actors and user applications.
It is their representative within the e-mail service. Typically, this
function is housed in the user's computer and is referred to as a client e-mail
program or a local network e-mail server. The author MUA formats a message
and performs initial submission into the MHS via a MSA. The recipient
MUA processes received mail for storage and/or display to the recipient user.

• Mail Submission Agent (MSA): Accepts the message submitted by an MUA
and enforces the policies of the hosting domain and the requirements of
Internet standards. This function may be located together with the MUA or
as a separate functional model. In the latter case, the Simple Mail Transfer
Protocol (SMTP) is used between the MUA and the MSA.

• Message Transfer Agent (MTA): Relays mail for one application-level hop.
It is like a packet switch or IP router in that its job is to make routing assessments
and to move the message closer to the recipients. Relaying is performed

by a sequence of MTAs until the message reaches a destination MDA. An MTA also adds trace information to the message header. SMTP is used between MTAs and between an MTA and an MSA or MDA.

• Mail Delivery Agent (MDA): Responsible for transferring the message from the MHS to the MS.

• Message Store (MS): An MUA can employ a long-term MS. An MS can be located on a remote server or on the same machine as the MUA. Typically, an MUA retrieves messages from a remote server using POP (Post Office Protocol) or IMAP (Internet Message Access Protocol).

Two other concepts need to be defined. An administrative management domain (ADMD) is an Internet e-mail provider. Examples include a department that operates a local mail relay (MTA), an IT department that operates an enterprise mail relay, and an ISP that operates a public shared e-mail service. Each ADMD can have different operating policies and trust-based decision making. One obvious example is the distinction between mail that is exchanged within an organization and mail that is exchanged between independent organizations. The rules for handling the two types of traffic tend to be quite different.

The Domain Name System (DNS) is a directory lookup service that provides a mapping between the name of a host on the Internet and its numerical address.

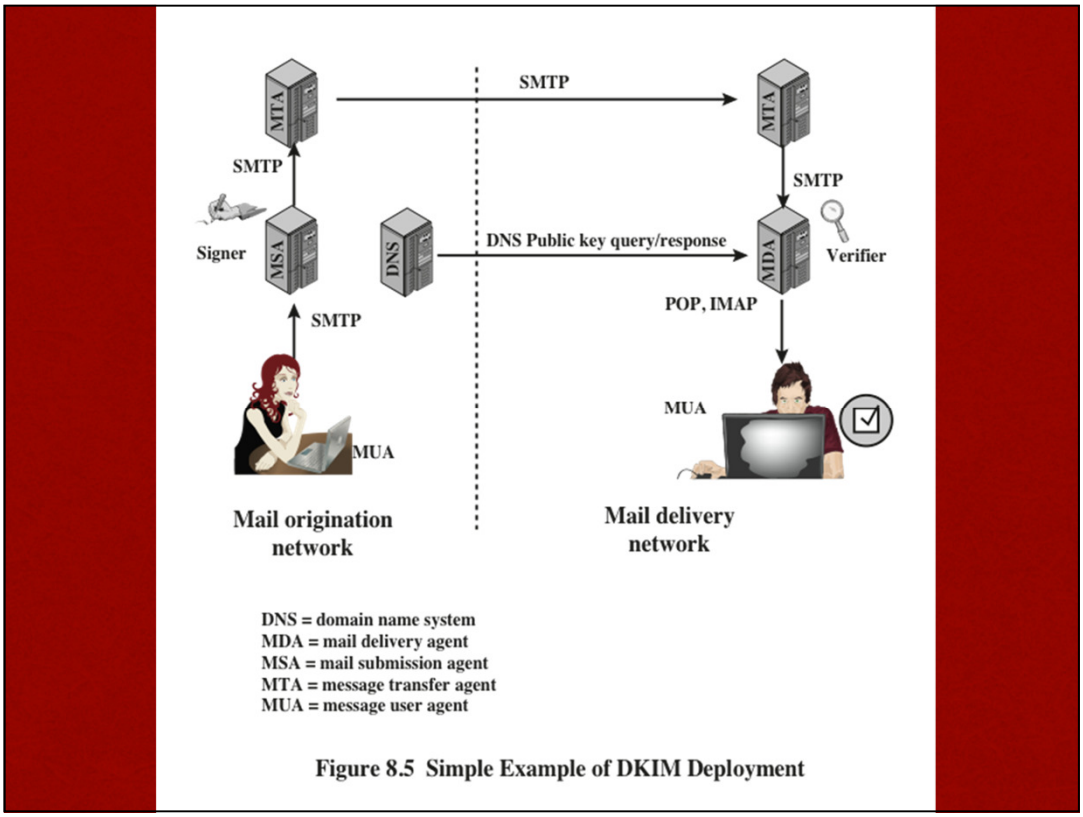


Figure 8.5 Simple Example of DKIM Deployment

DKIM is designed to provide an e-mail authentication technique that is transparent to the end user. In essence, a user's e-mail message is signed by a private key of the administrative domain from which the e-mail originates. The signature covers all of the content of the message and some of the RFC 5322 message headers. At the receiving end, the MDA can access the corresponding public key via a DNS and verify the signature, thus authenticating that the message comes from the claimed administrative domain. Thus, mail that originates from somewhere else but claims to come from a given domain will not pass the authentication test and can be rejected. This approach differs from that of S/MIME and PGP, which use the originator's private key to sign the content of the message. The motivation for DKIM is based on the following reasoning.

1. S/MIME depends on both the sending and receiving users employing S/MIME.

For almost all users, the bulk of incoming mail does not use S/MIME, and the

bulk of the mail the user wants to send is to recipients not using S/MIME.

2. S/MIME signs only the message content. Thus, RFC 5322 header information

concerning origin can be compromised.

3. DKIM is not implemented in client programs (MUAs) and is therefore transparent

to the user; the user need take no action.

4. DKIM applies to all mail from cooperating domains.

5. DKIM allows good senders to prove that they did send a particular message

and to prevent forgers from masquerading as good senders.

Figure 8.5 is a simple example of the operation of DKIM. We begin with a

message generated by a user and transmitted into the MHS to an MSA that is within

the user's administrative domain. An e-mail message is generated by an e-mail client

program. The content of the message, plus selected RFC 5322 headers, is signed

by the e-mail provider using the provider's private key. The signer is associated

with a domain, which could be a corporate local network, an ISP, or a public e-mail

facility such as gmail. The signed message then passes through the Internet via a

sequence of MTAs. At the destination, the MDA retrieves the public key for the

incoming signature and verifies the signature before passing the message

on to the

destination e-mail client. The default signing algorithm is RSA with SHA-256. RSA

with SHA-1 also may be used.

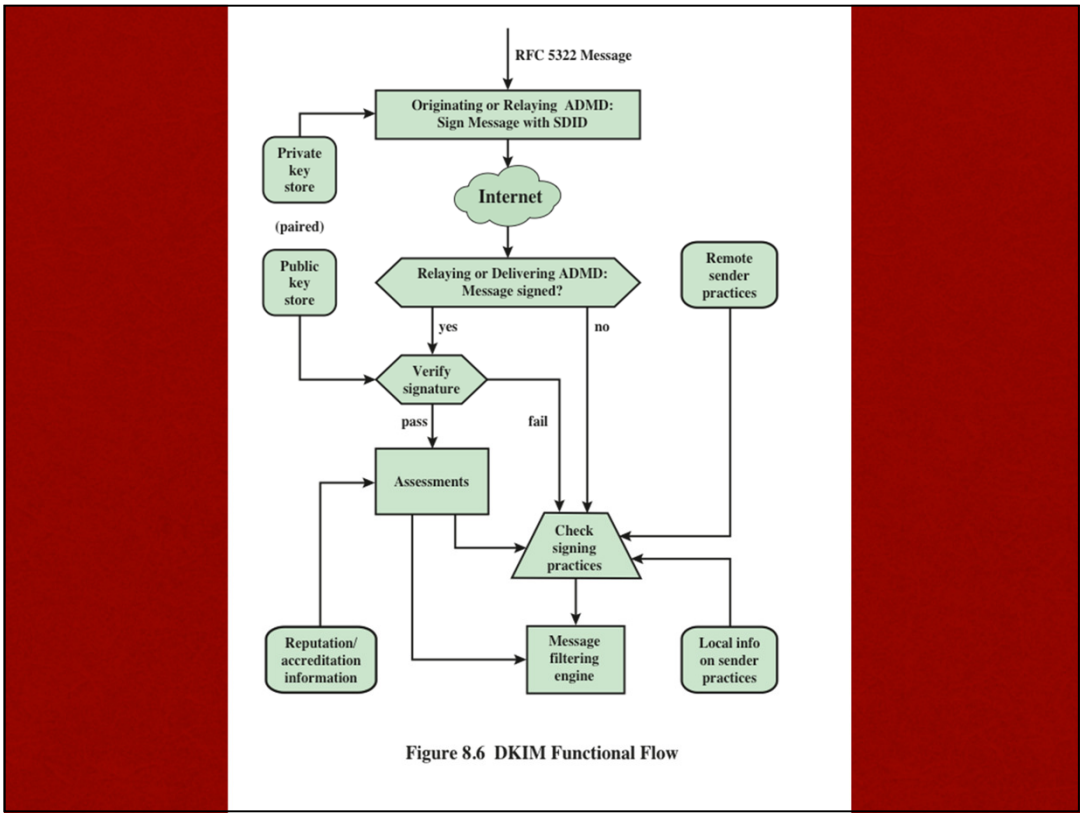


Figure 8.6 DKIM Functional Flow

Figure 8.6 provides a more detailed look at the elements of DKIM operation. Basic

message processing is divided between a signing Administrative Management Domain

(ADMD) and a verifying ADMD. At its simplest, this is between the originating ADMD

and the delivering ADMD, but it can involve other ADMDs in the handling path.

Signing is performed by an authorized module within the signing ADMD and

uses private information from a Key Store. Within the originating ADMD, this

might be performed by the MUA, MSA, or an MTA. Verifying is performed by

an authorized module within the verifying ADMD. Within a delivering ADMD,

verifying might be performed by an MTA, MDA, or MUA. The module verifies

the signature or determines whether a particular signature was required. Verifying
the signature uses public information from the Key Store. If the signature passes,
reputation information is used to assess the signer and that information is passed
to the message filtering system. If the signature fails or there is no signature using
the author's domain, information about signing practices related to the author can
be retrieved remotely and/or locally, and that information is passed to the message
filtering system. For example, if the sender (e.g., gmail) uses DKIM but no DKIM
signature is present, then the message may be considered fraudulent.

The signature is inserted into the RFC 5322 message as an additional header
entry, starting with the keyword Dkim-Signature . You can view examples from
your own incoming mail by using the View Long Headers (or similar wording)
option for an incoming message.

Before a message is signed, a process known as canonicalization is performed
on both the header and body of the RFC 5322 message. Canonicalization is necessary
to deal with the possibility of minor changes in the message made en route,
including character encoding, treatment of trailing white space in message lines, and
the "folding" and "unfolding" of header lines. The intent of canonicalization is to
make a minimal transformation of the message (for the purpose of signing; the message
itself is not changed, so the canonicalization must be performed again by the

verifier) that will give it its best chance of producing the same canonical value at the

receiving end. DKIM defines two header canonicalization algorithms (“simple” and

“relaxed”) and two for the body (with the same names). The simple algorithm tolerates

almost no modification, while the relaxed tolerates common modifications.

The signature includes a number of fields. Each field begins with a tag consisting

of a tag code followed by an equals sign and ends with a semicolon.

Summary

- Pretty good privacy
  - Notation
  - Operational description
- DomainKeys Identified Mail
  - Internet mail architecture
  - E-mail threats
  - DKIM strategy
  - DKIM functional flow
- S/MIME
  - RFC 5322
  - Multipurpose Internet mail extensions
  - S/MIME functionality
  - S/MIME messages
  - S/MIME certification processing
  - Enhanced security services

Chapter 8 summary.