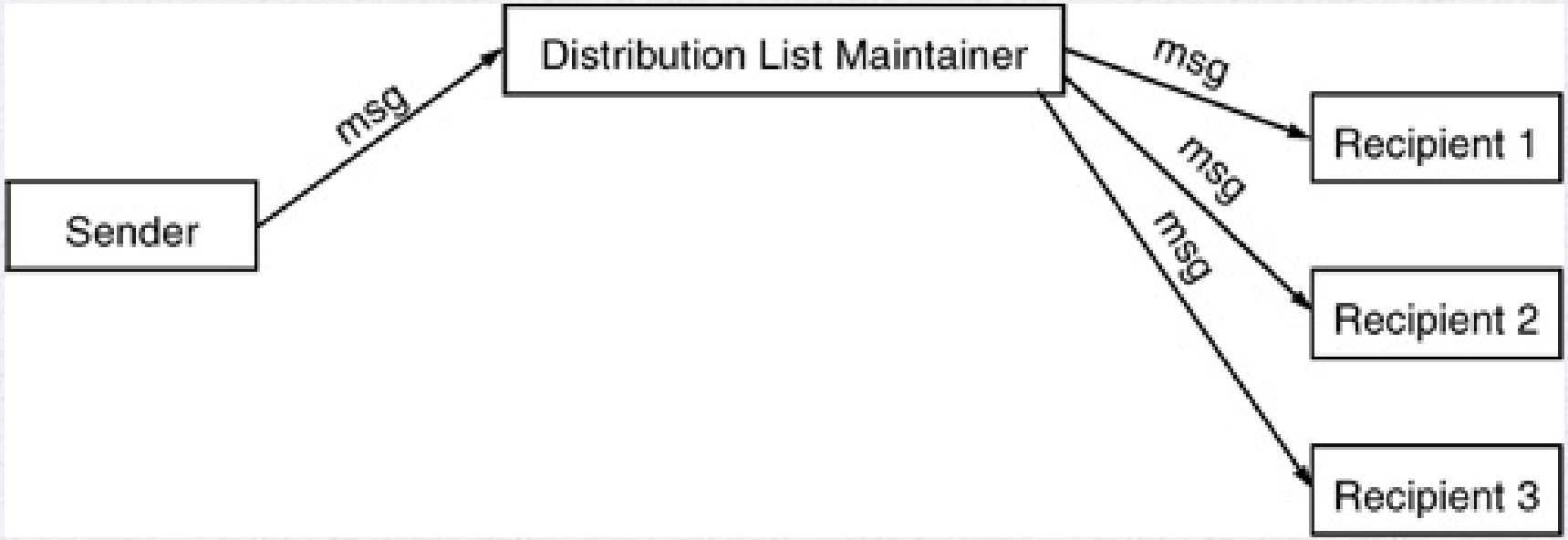
Pretty Good Privacy (PGP)

- Provides a confidentiality and authentication service that can be used for electronic mail and file storage applications
- Developed by Phil Zimmermann
  - Selected the best available cryptographic algorithms as building blocks
  - 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
  - Made the package and its documentation, including the source code, freely available via the Internet, bulletin boards, and commercial networks
  - Entered into an agreement with a company to provide a fully compatible, low-cost commercial version of PGP

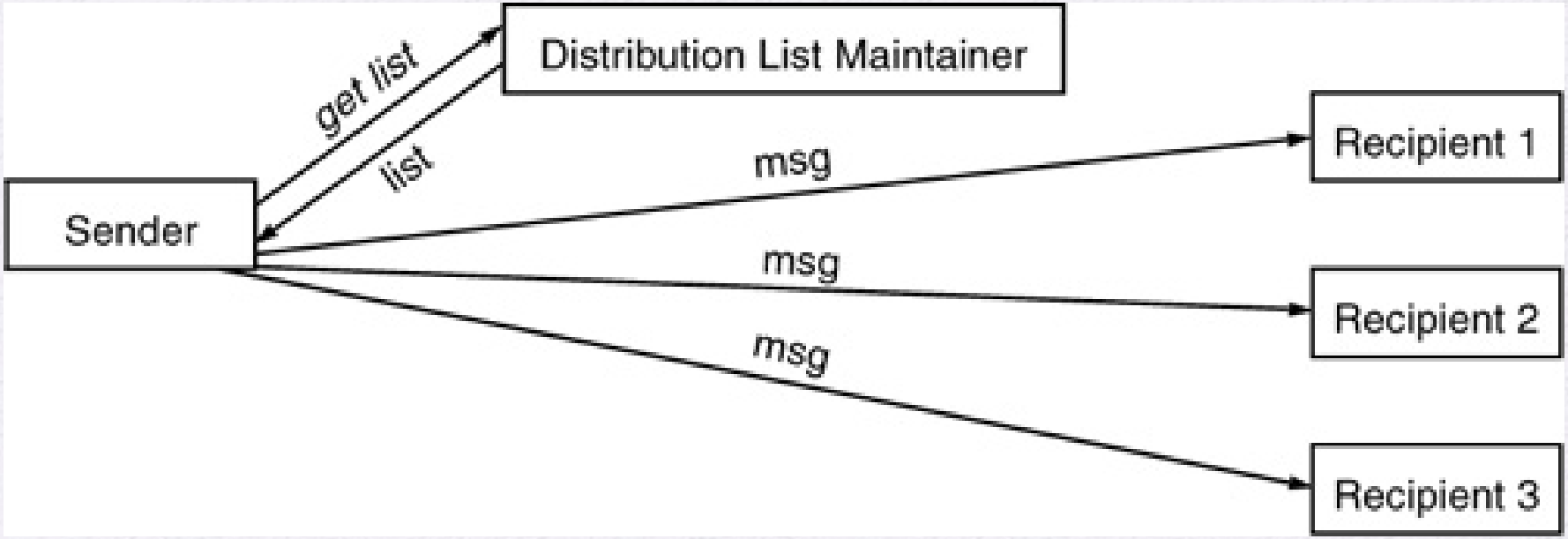
Distribution List

- user to send a message to one or more recipients
- mail is often sent to a distribution list.

Remote Exploder



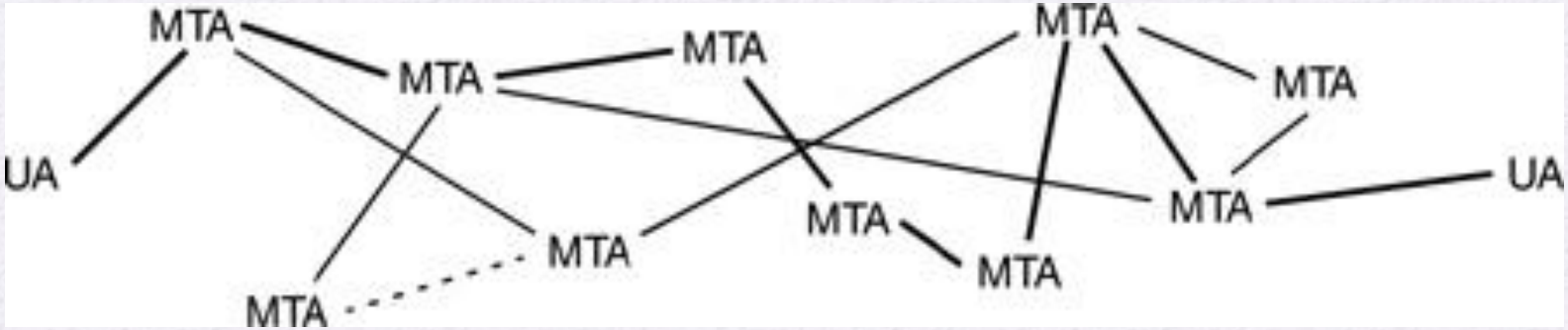
Local Explder



Store & Forward

- it is necessary for both the source and destination machines to be running, and reachable from each other on the network. This might be especially inconvenient if the user machines are only occasionally connected.

MTA-msg t/f agent UA-user agent



- **Privacy** ability to keep anyone but the intended recipient from reading the message. **Authentication** assurance to the recipient of the identity of the ender
- **Integrity** assurance to the recipient that the message has not been altered since it was transmitted by the sender

- **non-repudiation** the ability of the recipient to prove to a third party that the sender really did send the message.
- proof of submission verification given to the sender that the message was handed to the mail delivery system
- proof of delivery verification that the recipient received the message.

- message flow confidentiality: an extension of privacy such that Carol not only cannot know the content of the message Alice sent Bob, but cannot even determine whether Alice sent Bob a message.

- Anonymity the ability to send a message so that the recipient can't find out the identity of the sender.
- Containment the ability of the network to keep certain security levels of information from leaking out of a particular region.
- **Audit** the ability of the network to record events that might have some security relevance
- Accounting the ability of the mail system to maintain system usage statistics

- self destruct: Snapmail does not allow to forward.
- message sequence integrity

Establishing Keys

- **Establishing Public Keys:**
- If Alice wants to send a signed message to Bob, she can just sign the message and send it, either hoping Bob will already have her certificate, can obtain it if necessary, or she can include her certificate in the email message

Session keys

- Alice to obtain a ticket for Bob from a KDC

Privacy of Email

- Most people think that electronic mail is private.
- Ways to read your messages.
  - An eavesdropper can listen to the message while it is being transmitted
  - Relay nodes (routers or mail forwarders) might have software to store messages and divulge them to people

End-to-End Privacy

- if Alice wants to send Bob an encrypted message, she encrypts it using Bob's public key.
- If Alice has a long message to send to multiple recipients.

- Bob's name; $K_{Bob}\{S\}$
- Carol's name; $K_{Carol}\{S\}$
- Ted's name; $K_{Ted}\{S\}$
- $S\{m\}$

Privacy with Distribution List Exploders

- Alice is sending a message to a distribution list which will be remotely exploded.
- Alice will choose a random per-message secret key S, and encrypt the message with S. The distribution list exploder will decrypt S, and re-encrypt S with the key for each recipient

Authentication of the Source

- **Source Authentication Based on Public Key Technology**
- Assuming Bob knows Alice's public key, Alice can digitally sign the message, using her private key, which will assure Bob that Alice wrote the message. The method usually chosen to sign a message is for Alice to first compute a hash of the message (for instance using MD5), and then to sign the message digest

- To: Bob
- Subject: I got PEM working
- Date: Fri, 01 Apr 94 10:12:37 -0400
- From: Alice
- -----BEGIN PRIVACY-ENHANCED MESSAGE-----
- Proc-Type: 4,MIC-CLEAR
- Content-Domain: RFC822
- DEK-Info: DES-CBC,31747476B4831B1D
- Originator-ID-Asymmetric: MEMxCzAJBgNVBAYTAlVTMSYwJAYDVQQKEx1EaWd
- pdGFsIEVxdWlwbWVudCBDb3Jwb3JhdGlvbjEMMAoGA1UECxMDTEtH,02
- MIC-Info: RSA-MD5,RSA,u1OHP1RwLqePAoaN5nPk9W7Q2EfjaP+yDBSaLyMcSgc
- MK2YicGSAqLz47Ol+TUR4YmMD/JnHMtsCJerV2QFtFQ==

- ... and I'm sending this message with PEM.
- -----END PRIVACY-ENHANCED MESSAGE-----

Source Authentication Based on Secret Keys

- If Alice has a shared key with Bob, then she can reassure Bob that she is Alice by proving she knows the shared secret key. She does this by performing some cryptographic computation on the message using the shared secret key.

Source Authentication with Distribution Lists

- With public keys, source authentication is easy, even with distribution lists
- With secret keys it is more complicated. We can't assume Alice will share a secret key with every recipient

Message Integrity

- When Bob receives a message from Alice, how does he know that Carol did not intercept the message and modify it? For instance, she might have changed the message
- Fire Carol immediately to Promote Carol immediately.

Message Integrity without Source Authentication

- **Message Integrity without Source Authentication:**
  - If the message is encrypted (where the sender knows the public key of the recipient), it is possible to do something that could be considered integrity protection without source authentication

Non Repudiation

- **Repudiation** is the act of denying that you sent a message
- it means that if Alice sends a message to Bob, Alice cannot later deny that she sent the message. Bob can prove to a third party that Alice did indeed send the message

Non-Repudiation Based on Public Key Technology

- **Non-Repudiation Based on Public Key Technology:**
  - Use public key signature on the message digest of the message, using her private key

  Non-Repudiation with Secret Keys

  Alice sends the message to N, and does source authentication with N so that N knows the message came from Alice. N does some computation on the message and Alice's name using a secret quantity S_N that N shares with nobody.

Proof of submission

- Some electronic mail systems offer a similar service, but by using cryptography on the contents of the message, it turns out to be more powerful than the postal service certified mail concept.

Proof of delivery

- If the mail service gives the proof of delivery, it would be done after transmitting the message to the destination. If the destination gives the proof of delivery, it would be done after the destination receives the message.

PEM

- PEM (Privacy Enhanced Mail) was developed by the Internet community in the late '80s and early '90s as a means of adding encryption, source authentication, and integrity protection to ordinary text messages.
- RFC 1421-1424

SMTP

- SMTP cannot transmit executable files.
- SMTP cannot transmit text data that includes national language characters,
- SMTP servers may reject mail message over a certain size
- SMTP gateways to X.400 electronic mail networks.

MIME (Multipurpose Internet Mail Extensions)

- Secure S/MIME (RFC 2633) took many of the design principles of PEM, and incorporated them into the MIME structure, adding signed data and encrypted data as new types of data.

S/MIME

- signed data and encrypted data as new types of data

Structure of a PEM Message

- -----BEGIN PRIVACY-ENHANCED MESSAGE-----
- -----END PRIVACY-ENHANCED MESSAGE-----
- ordinary, unsecured data
- integrity-protected unmodified data
- integrity-protected encoded data
- encoded encrypted integrity-protected data

- From: Alice
- To: Bob
- Subject: Keep this proposition private!
- Date: Fri, 01 Apr 94 10:12:37 -0400
- Care to meet me at my apartment tonight?

- From: Alice
- To: Bob
- Subject: Keep this proposition private!
- Date: Fri, 01 Apr 94 10:12:37 -0400
- -----BEGIN PRIVACY-ENHANCED MESSAGE-----
- Proc-Type: 4, MIC-CLEAR
- Content-Domain: RFC822
- Originator-ID-Asymmetric: MEMxCzAJBgNVBAYTAlVTMSYwJAYDVQQKEx1EaWd
- pdGFsIEVxdWlwbWVudCBDb3Jwb3JhdGlvbjEMMAoGA1UECxMDTEtH,02
- MIC-Info: RSA-MD5,RSA,u1OHP1RwLqePAoaN5nPk9W7Q2EfjaP+yDBSaLyMcSgc
- MK2YicGSAqLz47Ol+TUR4YmMD/JnHMtsCJerV2QFtFQ==

- Care to meet me at my apartment tonight?
- -----END PRIVACY-ENHANCED MESSAGE-----

PGP Growth

- It is available free worldwide in versions that run on a variety of platforms
- The commercial version satisfies users who want a product that comes with vendor support
- It is based on algorithms that have survived extensive public review and are considered extremely secure
- It has a wide range of applicability
- It was not developed by, nor is it controlled by, any governmental or standards organization
- Is now on an Internet standards track, however it still has an aura of an antiestablishment endeavor

Notation

K_s	=	session key used in symmetric encryption scheme
PR_a	=	private key of user A, used in public-key encryption scheme
PU_a	=	public key of user A, used in public-key encryption scheme
EP	=	public-key encryption
DP	=	public-key decryption
EC	=	symmetric encryption
DC	=	symmetric decryption
H	=	hash function
$\|$	=	concatenation
Z	=	compression using ZIP algorithm
R64	=	conversion to radix 64 ASCII format[1]

Table 19.1
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.

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 document

Authentication

1. The sender creates a message.
2. SHA-1 is used to generate a 160-bit hash code of the message.
3. The hash code is encrypted with RSA using the sender's private key, and the result is prepended to the message.
4. The receiver uses RSA with the sender's public key to decrypt and recover the hash code.
5. The receiver generates a new hash code for the message and compares it with the decrypted hash code. If the two match, the message is accepted as authentic.

PGP Confidentiality

- Provided by encrypting messages to be transmitted or to be stored locally as files
  - In both cases the symmetric encryption algorithm CAST-128 [Carlisle Adams and Stafford Tavares] may be used
  - Alternatively IDEA or 3DES may be used
  - The 64-bit cipher feedback (CFB) mode is used

In PGP each symmetric key is used only once

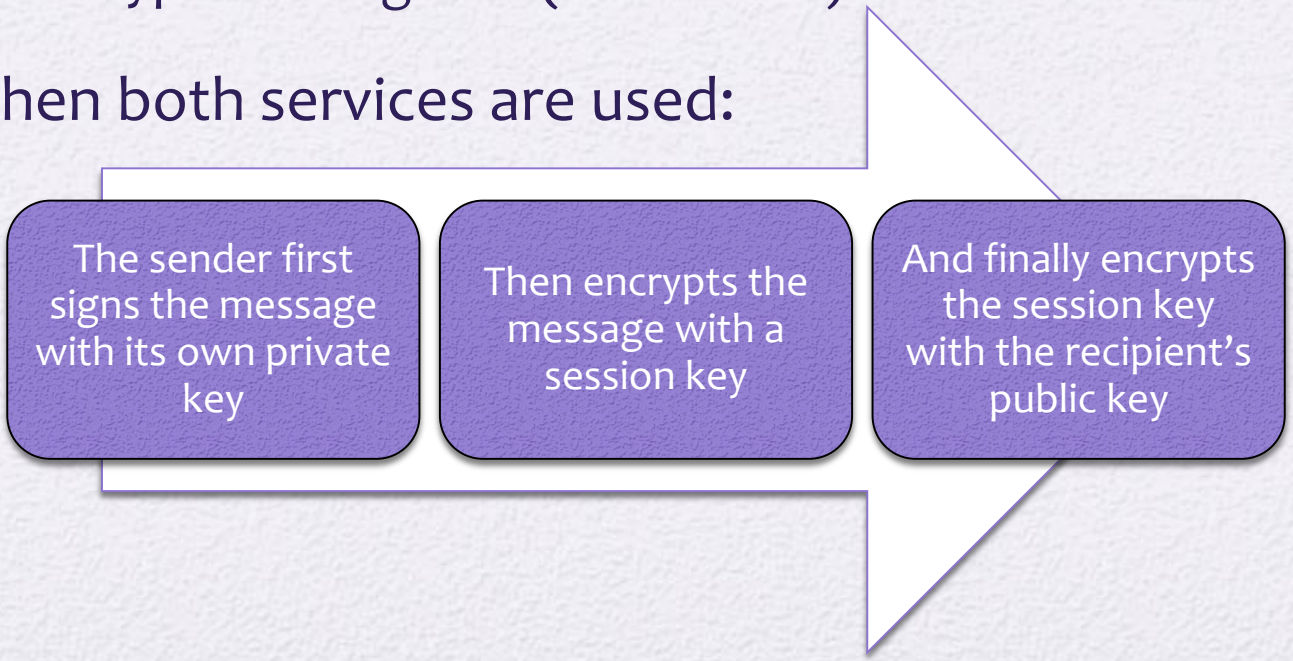
- Although referred to as a session key, it is in reality a one-time key
- Session key is bound to the message and transmitted with it
- To protect the key, it is encrypted with the receiver's public key

- As an alternative to the use of RSA for key encryption, PGP uses ElGamal, a variant of Diffie-Hellman that provides encryption/decryption

1. The sender generates a message and a random 128-bit number to be used as a session key for this message only.
2. The message is encrypted using CAST-128 (or IDEA or 3DES) with the session key.
3. The session key is encrypted with RSA using the recipient's public key and is prepended to the message.
4. The receiver uses RSA with its private key to decrypt and recover the session key.
5. The session key is used to decrypt the message.

PGP Confidentiality and Authentication

- 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)
- When both services are used:



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

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

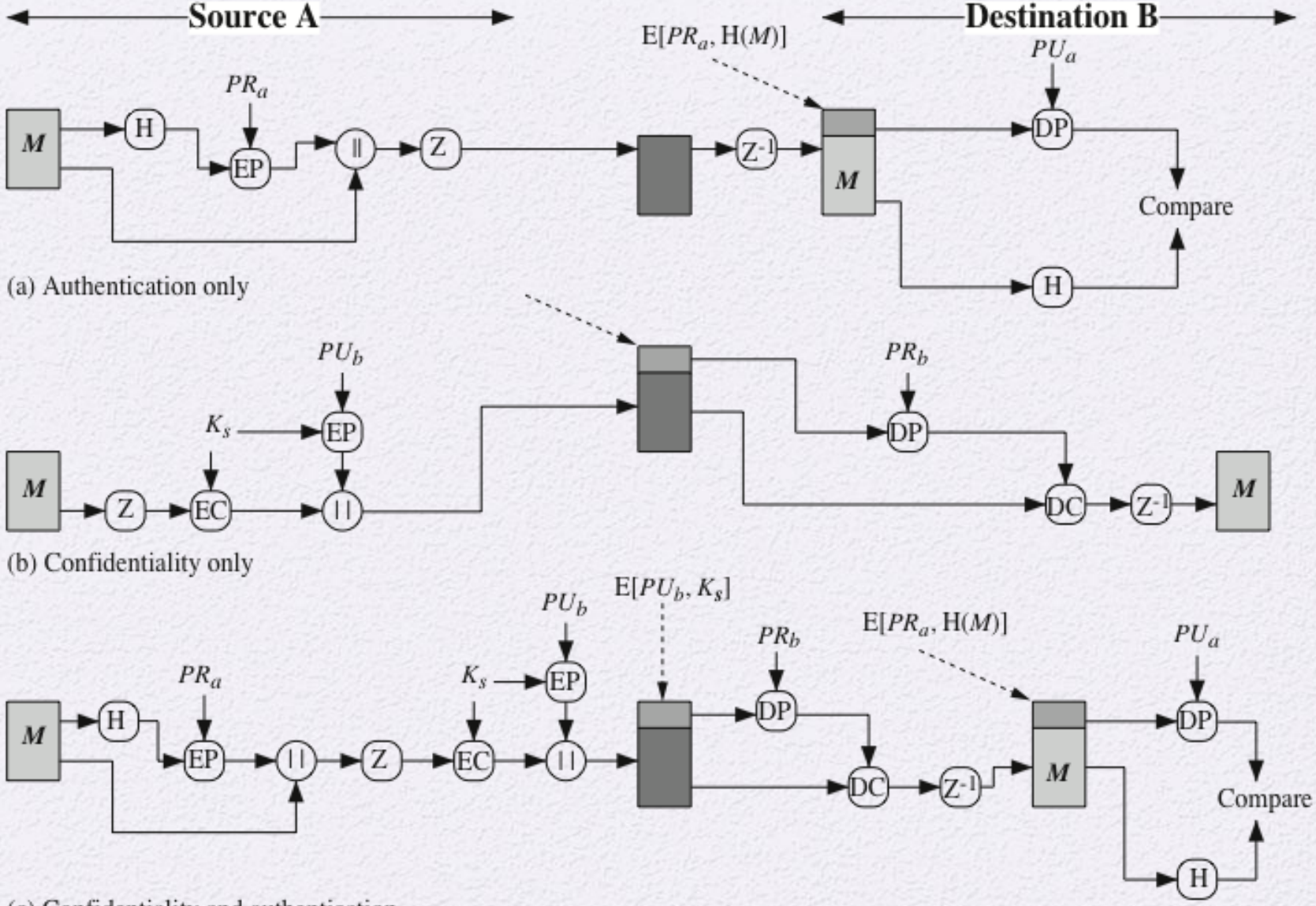
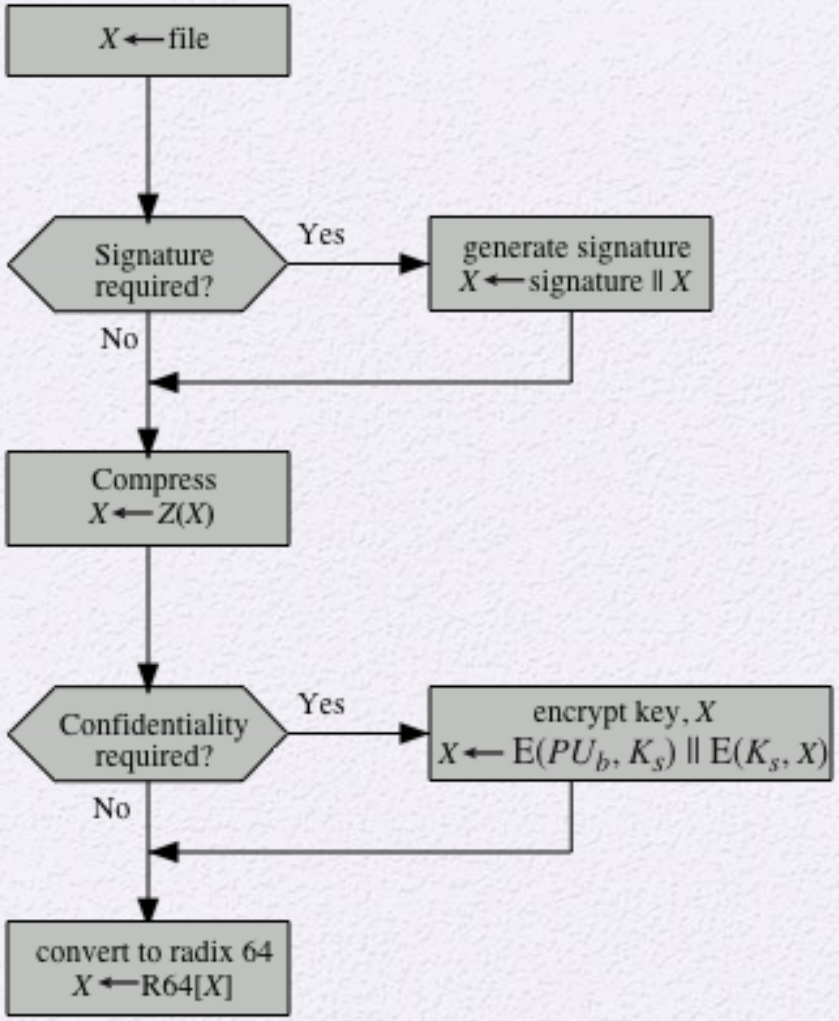
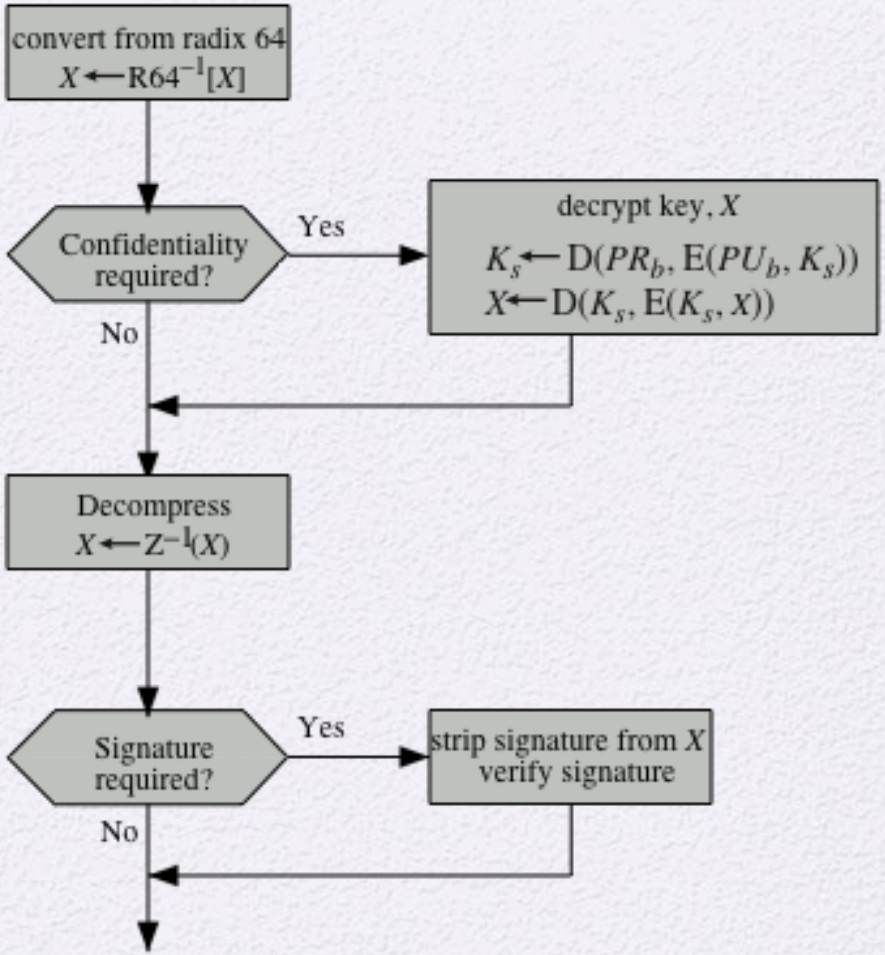


Figure 19.1 PGP Cryptographic Functions



(a) Generic Transmission Diagram (from A)



(b) Generic Reception Diagram (to B)

Figure 19.2 Transmission and Reception of PGP Messages

Secure/Multipurpose Internet Mail Extension (S/MIME)

- A security enhancement to the MIME Internet e-mail format standard based on technology from RSA Data Security
- Defined in:
  - RFCs 3370, 3850, 3851, 3852

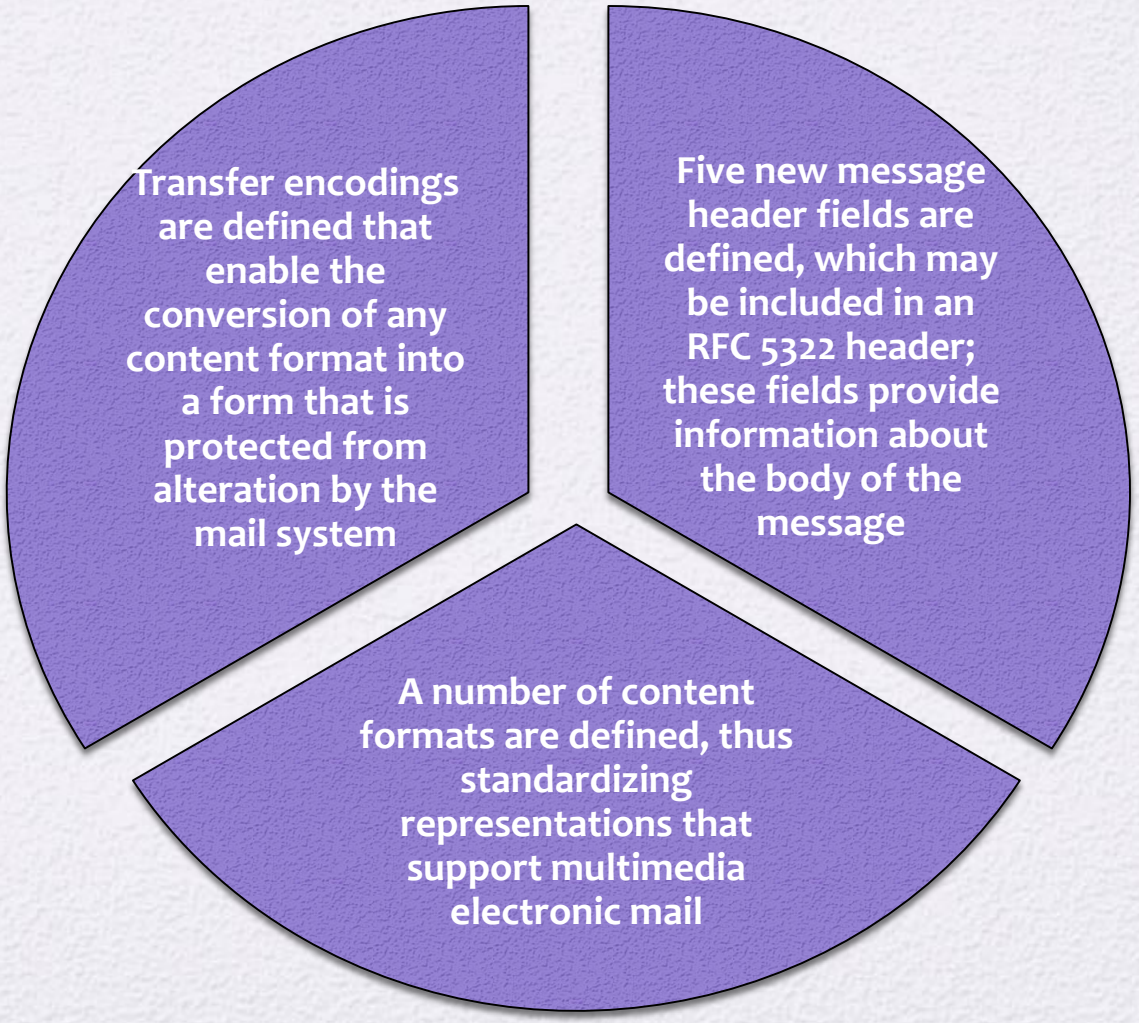
RFC 5322

- Defines a format for text messages that are sent using electronic mail
- Messages are viewed as having an envelope and contents
  - The envelope contains whatever information is needed to accomplish transmission and delivery
  - The contents compose the object to be delivered to the recipient
  - RFC 5322 standard applies only to the contents
- The content standard includes a set of header fields that may be used by the mail system to create the envelope

Multipurpose Internet Mail Extensions (MIME)

- An extension to the RFC 5322 framework that is intended to address some of the problems and limitations of the use of Simple Mail Transfer Protocol (SMTP)
  - Is intended to resolve these problems in a manner that is compatible with existing RFC 5322 implementations
  - The specification is provided in RFCs 2045 through 2049

MIME specification includes the following elements:



The Five Header Fields Defined in MIME

MIME-Version

- Must have the parameter value 1.0
- This field indicates that the message conforms to RFCs 2045 and 2046

Content-Type

- Describes the data contained in the body with sufficient detail that the receiving user agent can pick an appropriate agent or mechanism to represent the data to the user or otherwise deal with the data in an appropriate manner

Content-Transfer-Encoding

- Indicates the type of transformation that has been used to represent the body of the message in a way that is acceptable for mail transport

Content-ID

- Used to identify MIME entities uniquely in multiple contexts

Content-Description

- A text description of the object with the body; this is useful when the object is not readable

Type	Subtype	Description
Text	Plain	Unformatted text; may be ASCII or ISO 8859.
	Enriched	Provides greater format flexibility.
Multipart	Mixed	The different parts are independent but are to be transmitted together. They should be presented to the receiver in the order that they appear in the mail message.
	Parallel	Differs from Mixed only in that no order is defined for delivering the parts to the receiver.
	Alternative	The different parts are alternative versions of the same information. They are ordered in increasing faithfulness to the original, and the recipient's mail system should display the "best" version to the user.
	Digest	Similar to Mixed, but the default type/subtype of each part is message/rfc822.
Message	rfc822	The body is itself an encapsulated message that conforms to RFC 822.
	Partial	Used to allow fragmentation of large mail items, in a way that is transparent to the recipient.
	External-body	Contains a pointer to an object that exists elsewhere.
Image	jpeg	The image is in JPEG format, JFIF encoding.
	gif	The image is in GIF format.
Video	mpeg	MPEG format.
Audio	Basic	Single-channel 8-bit ISDN mu-law encoding at a sample rate of 8 kHz.
Application	PostScript	Adobe Postscript format.
	octet-stream	General binary data consisting of 8-bit bytes.

Table 19.2

MIME Content Types

```
MIME-Version: 1.0
Content-type: multipart/mixed; boundary="simple boundary"

This is the preamble. It is to be ignored, though it is a
handy place for mail composers to include an explanatory
note to non-MIME conformant readers.
–simple boundary

This is implicitly typed plain ASCII text. It does NOT
end with a linebreak.
–simple boundary
Content-type: text/plain; charset=us-ascii

This is explicitly typed plain ASCII text. It DOES end
with a linebreak.
–simple boundary–
This is the epilogue. It is also to be ignored.
```

Table 19.3
MIME Transfer Encodings

7bit	The data are all represented by short lines of ASCII characters.
8bit	The lines are short, but there may be non-ASCII characters (octets with the high-order bit set).
binary	Not only may non-ASCII characters be present but the lines are not necessarily short enough for SMTP transport.
quoted-printable	Encodes the data in such a way that if the data being encoded are mostly ASCII text, the encoded form of the data remains largely recognizable by humans.
base64	Encodes data by mapping 6-bit blocks of input to 8-bit blocks of output, all of which are printable ASCII characters.
x-token	A named nonstandard encoding.

MIME T/f Encoding

- **quoted-printable transfer encoding is useful when the data consists largely** of octets that correspond to printable ASCII characters.
- **base64 transfer encoding, also known as radix-64 encoding, is a common** one for encoding arbitrary binary data

```
MIME-Version: 1.0
From: Nathaniel Borenstein <nsb@bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: A multipart example
Content-Type: multipart/mixed;
    boundary=unique-boundary-1
This is the preamble area of a multipart message. Mail readers that understand multipart format should ignore this preamble.
 If you are reading this text, you might want to consider changing to a mail reader that understands how to properly display
 multipart messages.

 --unique-boundary-1
  ...Some text appears here...
[Note that the preceding blank line means no header fields were given and this is text, with charset US ASCII.  It could have
 been done with explicit typing as in the next part.]

--unique-boundary-1
Content-type: text/plain; charset=US-ASCII
This could have been part of the previous part, but illustrates explicit versus implicit typing of body parts.

--unique-boundary-1
Content-Type: multipart/parallel;   boundary=unique-boundary-2

--unique-boundary-2
Content-Type: audio/basic
Content-Transfer-Encoding: base64
  ... base64-encoded 8000 Hz single-channel mu-law-format audio data goes here....

--unique-boundary-2
Content-Type: image/jpeg
Content-Transfer-Encoding: base64
  ... base64-encoded image data goes here....

--unique-boundary-2--
--unique-boundary-1
Content-type: text/enriched

This is <bold><italic>richtext.</italic></bold> <smaller>as defined in RFC 1896</smaller>

 Isn't it <bigger><bigger>cool?</bigger></bigger>

--unique-boundary-1
Content-Type: message/rfc822

From: (mailbox in US-ASCII)
To: (address in US-ASCII)
Subject: (subject in US-ASCII)
Content-Type: Text/plain; charset=ISO-8859-1
Content-Transfer-Encoding: Quoted-printable

  ... Additional text in ISO-8859-1 goes here ...

--unique-boundary-1--
```

Figure 19.3 Example MIME Message Structure

Table 19.4
Native and Canonical Form

Native Form	The body to be transmitted is created in the system's native format. The native character set is used and, where appropriate, local end-of-line conventions are used as well. The body may be a UNIX-style text file, or a Sun raster image, or a VMS indexed file, or audio data in a system-dependent format stored only in memory, or anything else that corresponds to the local model for the representation of some form of information. Fundamentally, the data is created in the "native" form that corresponds to the type specified by the media type.
Canonical Form	The entire body, including "out-of-band" information such as record lengths and possibly file attribute information, is converted to a universal canonical form. The specific media type of the body as well as its associated attributes dictate the nature of the canonical form that is used. Conversion to the proper canonical form may involve character set conversion, transformation of audio data, compression, or various other operations specific to the various media types. If character set conversion is involved, however, care must be taken to understand the semantics of the media type, which may have strong implications for any character set conversion (e.g. with regard to syntactically meaningful characters in a text subtype other than "plain").

Content Type

- Canonical form is a format, appropriate to the content type, that is standardized for use between systems.
- Native form, which is a format that may be peculiar to a particular system.

S/MIME Functionality

S/MIME

Enveloped data

- Consists of encrypted content of any type and encrypted content encryption keys for one or more recipients

Signed data

- A digital signature is formed by taking the message digest of the content to be signed and then encrypting that with the private key of the signer
- The content plus signature are then encoded using base64 encoding
- A signed data message can only be viewed by a recipient with S/MIME capability

Clear-signed data

- Only the digital signature is encoded using base64
- As a result recipients without S/MIME capability can view the message content, although they cannot verify the signature

Signed and enveloped data

- Signed-only and encrypted-only entities may be nested, so that encrypted data may be signed and signed data or clear-signed data may be encrypted

Function	Requirement
Create a message digest to be used in forming a digital signature.	MUST support SHA-1. Receiver SHOULD support MD5 for backward compatibility.
Encrypt message digest to form a digital signature.	Sending and receiving agents MUST support DSS. Sending agents SHOULD support RSA encryption. Receiving agents SHOULD support verification of RSA signatures with key sizes 512 bits to 1024 bits.
Encrypt session key for transmission with a message.	Sending and receiving agents SHOULD support Diffie-Hellman. Sending and receiving agents MUST support RSA encryption with key sizes 512 bits to 1024 bits.
Encrypt message for transmission with a one-time session key.	Sending and receiving agents MUST support encryption with tripleDES Sending agents SHOULD support encryption with AES. Sending agents SHOULD support encryption with RC2/40.
Create a message authentication code	Receiving agents MUST support HMAC with SHA-1. Sending agents SHOULD support HMAC with SHA-1.

Table 19.5

Cryptographic Algorithms Used in S/MIME

Table 19.6
S/MIME Content Types

Type	Subtype	smime Parameter	Description
Multipart	Signed		A clear-signed message in two parts: one is the message and the other is the signature.
Application	pkcs7-mime	signedData	A signed S/MIME entity.
	pkcs7-mime	envelopedData	An encrypted S/MIME entity.
	pkcs7-mime	degenerate signedData	An entity containing only public-key certificates.
	pkcs7-mime	CompressedData	A compressed S/MIME entity.
	pkcs7-signature	signedData	The content type of the signature subpart of a multipart/signed message.

Securing a MIME Entity

- S/MIME secures a MIME entity with a signature, encryption, or both
- The MIME entity is prepared according to the normal rules for MIME message preparation
  - The MIME entity plus some security-related data, such as algorithm identifiers and certificates, are processed by S/MIME to produce what is known as a PKCS object
  - A PKCS object is then treated as message content and wrapped in MIME
- In all cases the message to be sent is converted to canonical form

EnvelopedData

- The steps for preparing an envelopedData MIME are:

Generate a pseudorandom session key for a particular symmetric encryption algorithm

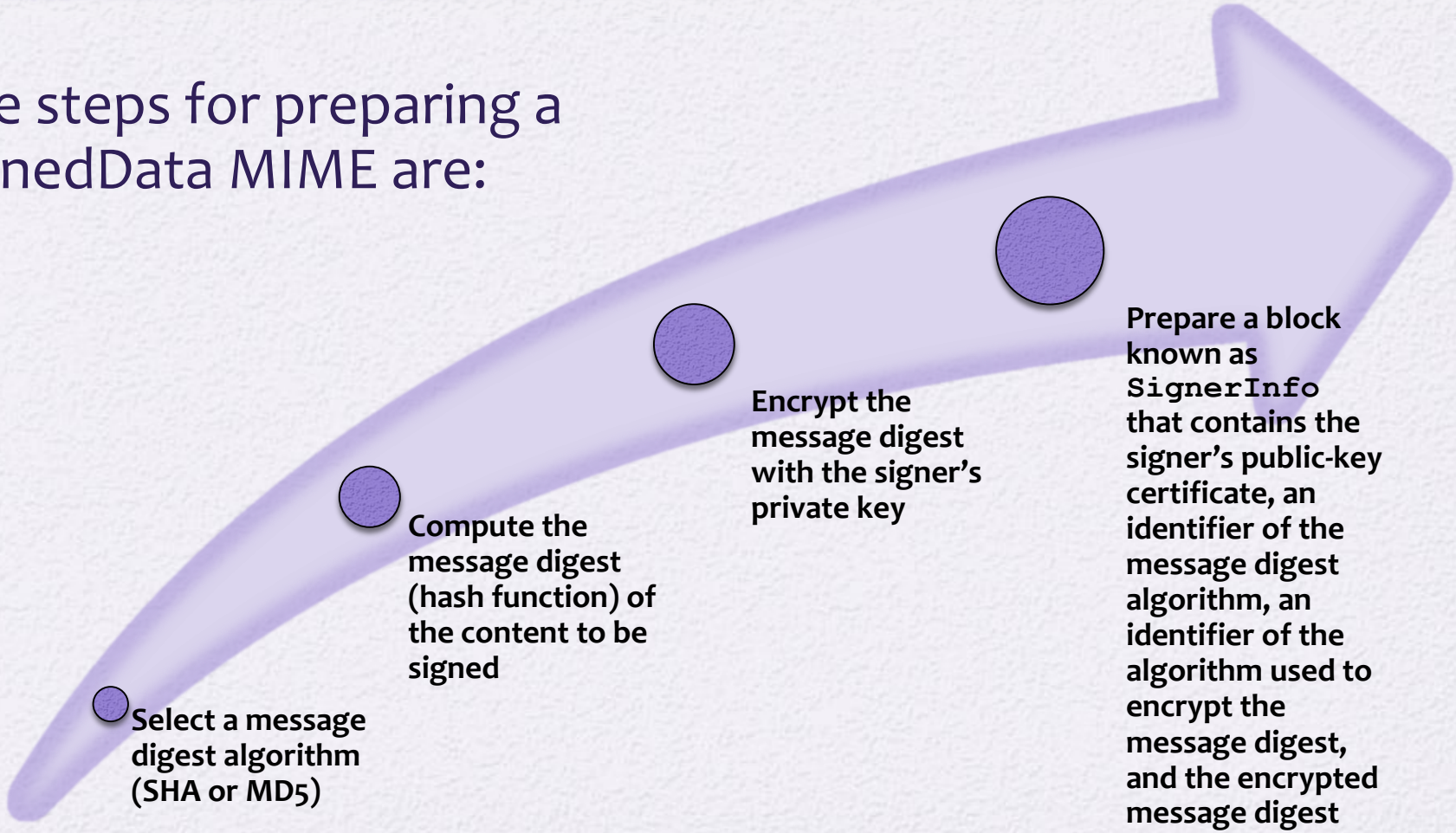
For each recipient, encrypt the session key with the recipient's public RSA key

For each recipient, prepare a block known as `RecipientInfo` that contains an identifier of the recipient's public-key certificate, an identifier of the algorithm used to encrypt the session key, and the encrypted session key

Encrypt the message content with the session key

SignedData

- The steps for preparing a signedData MIME are:



Clear Signing

- Achieved using the multipart content type with a signed subtype
- This signing process does not involve transforming the message to be signed
- Recipients with MIME capability but not S/MIME capability are able to read the incoming message

S/MIME Certificate Processing

- S/MIME uses public-key certificates that conform to version 3 of X.509
- The key-management scheme used by S/MIME is in some ways a hybrid between a strict X.509 certification hierarchy and PGP's web of trust
- S/MIME managers and/or users must configure each client with a list of trusted keys and with certificate revocation lists
  - The responsibility is local for maintaining the certificates needed to verify incoming signatures and to encrypt outgoing messages
- The certificates are signed by certification authorities

User Agent Role

- An S/MIME user has several key-management functions to perform:

Key generation

The user of some related administrative utility must be capable of generating separate Diffie-Hellman and DSS key pairs and should be capable of generating RSA key pairs

A user agent should generate RSA key pairs with a length in the range of 768 to 1024 bits and must not generate a length of less than 512 bits

Registration

A user's public key must be registered with a certification authority in order to receive an X.509 public-key certificate

Certificate storage and retrieval

A user requires access to a local list of certificates in order to verify incoming signatures and to encrypt outgoing messages

VeriSign Certificates

- VeriSign provides a certification authority (CA) service that is intended to be compatible with S/MIME and a variety of other applications
- Issues X.509 certificates with the product name VeriSign Digital ID
- At a minimum, each Digital ID contains:
  - Owner's public key
  - Owner's name or alias
  - Expiration date of the Digital ID
  - Serial number of the Digital ID
  - Name of the certification authority that issued the Digital ID
  - Digital signature of the certification authority that issued the Digital ID

Table 19.7
VeriSign Public-Key Certificate Classes

	Class 1	Class 2	Class 3
Summary of Confirmation of Identity	Automated unambiguous name and e-mail address search.	Same as Class 1, plus automated enrollment information check and automated address check.	Same as Class 1, plus personal presence and ID documents plus Class 2 automated ID check for individuals; business records (or filings) for organizations.
IA Private Key Protection	PCA: trustworthy hardware; CA: trust-worthy software or trustworthy hardware.	PCA and CA: trustworthy hardware.	PCA and CA: trustworthy hardware.
Certificate Applicant and Subscriber Private Key Protection	Encryption software (PIN protected) recommended but not required.	Encryption software (PIN protected) required.	Encryption software (PIN protected) required; hardware token recommended but not required.
Applications Implemented or Contemplated by Users	Web-browsing and certain e-mail usage.	Individual and intra- and inter-company e-mail, online subscriptions, password replacement, and software validation.	E-banking, corp. database access, personal banking, membership-based online services, content integrity services, e-commerce server, software validation; authentication of LRAAs; and strong encryption for certain servers.

IA	Issuing Authority
CA	Certification Authority
PCA	VeriSign public primary certification authority
PIN	Personal Identification Number
LRAA	Local Registration Authority Administrator

Enhanced Security Services

- Three enhanced security services have been proposed in an Internet draft:
  - Signed receipt
    - Returning a signed receipt provides proof of delivery to the originator of a message and allows the originator to demonstrate to a third party that the recipient received the message
  - Security labels
    - A set of security information regarding the sensitivity of the content that is protected by S/MIME encapsulation
  - Secure mailing lists
    - An S/MIME Mail List Agent (MLA) can take a single incoming message, perform the recipient-specific encryption for each recipient, and forward the message

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)

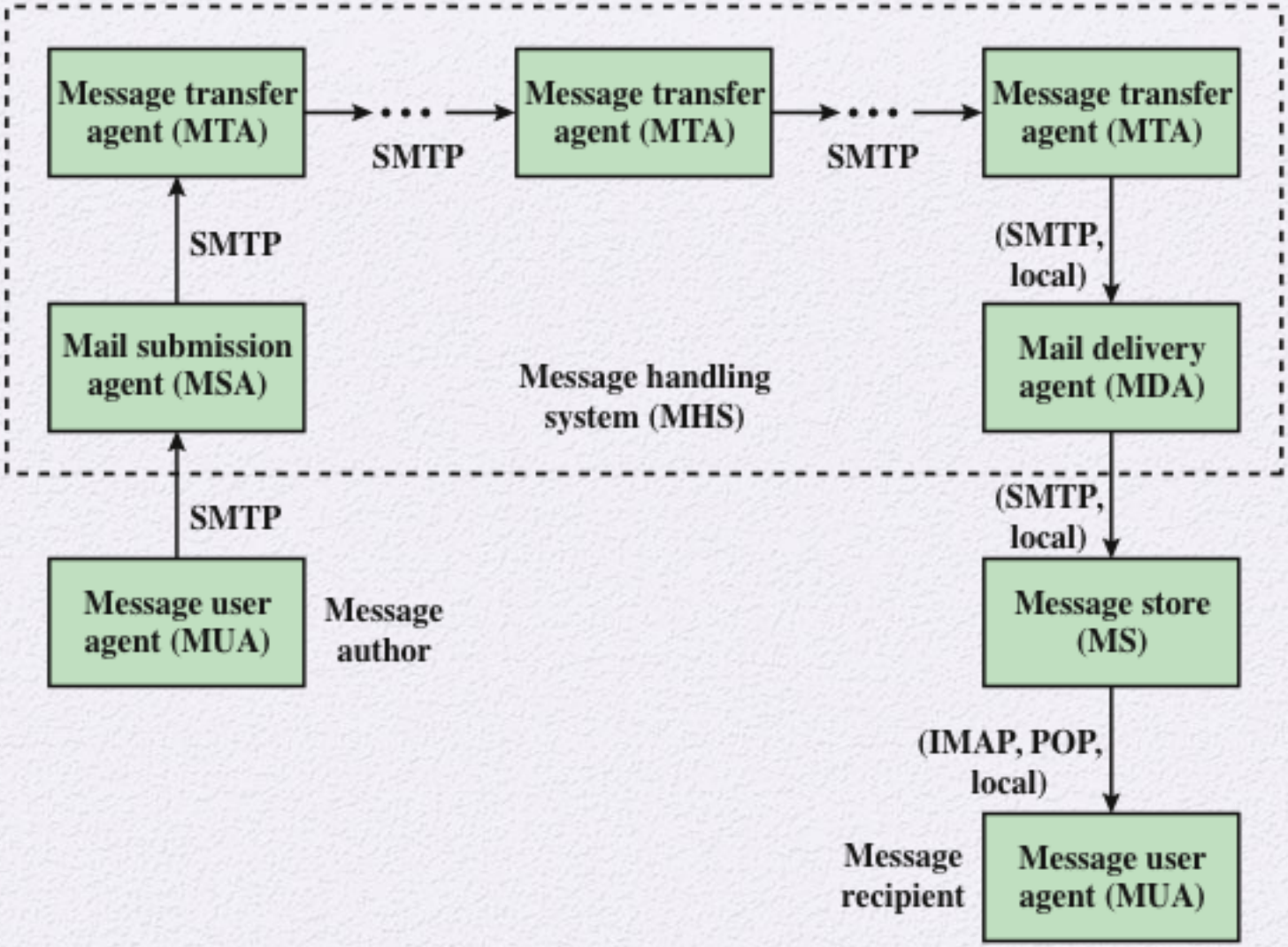
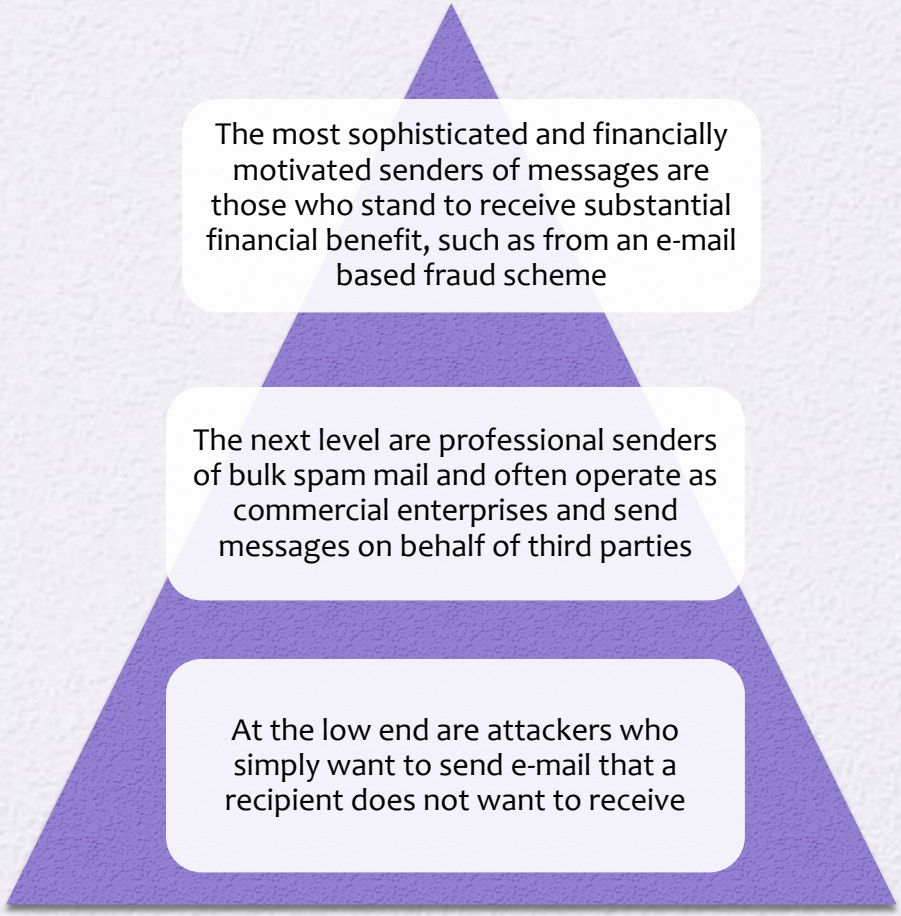


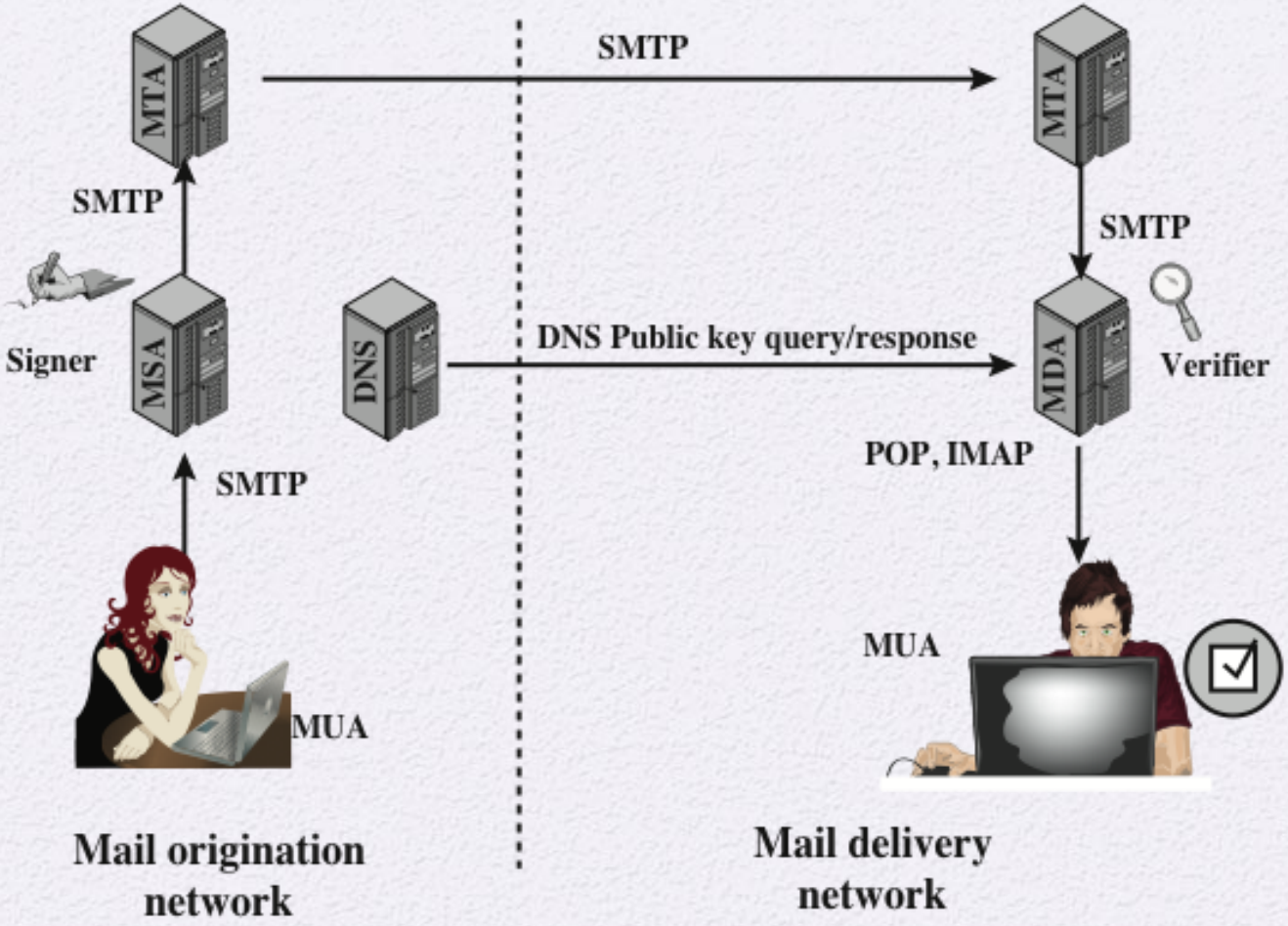
Figure 19.4 Function Modules and Standardized Protocols Used Between Them

E-mail Threats

- RFC 4684 (*Analysis of Threats Motivating DomainKeys Identified Mail*)
  - Describes the threats being addressed by DKIM in terms of the characteristics, capabilities, and location of potential attackers

- Characterized on three levels of threat:





DNS = domain name system
MDA = mail delivery agent
MSA = mail submission agent
MTA = message transfer agent
MUA = message user agent

Figure 19.5 Simple Example of DKIM Deployment

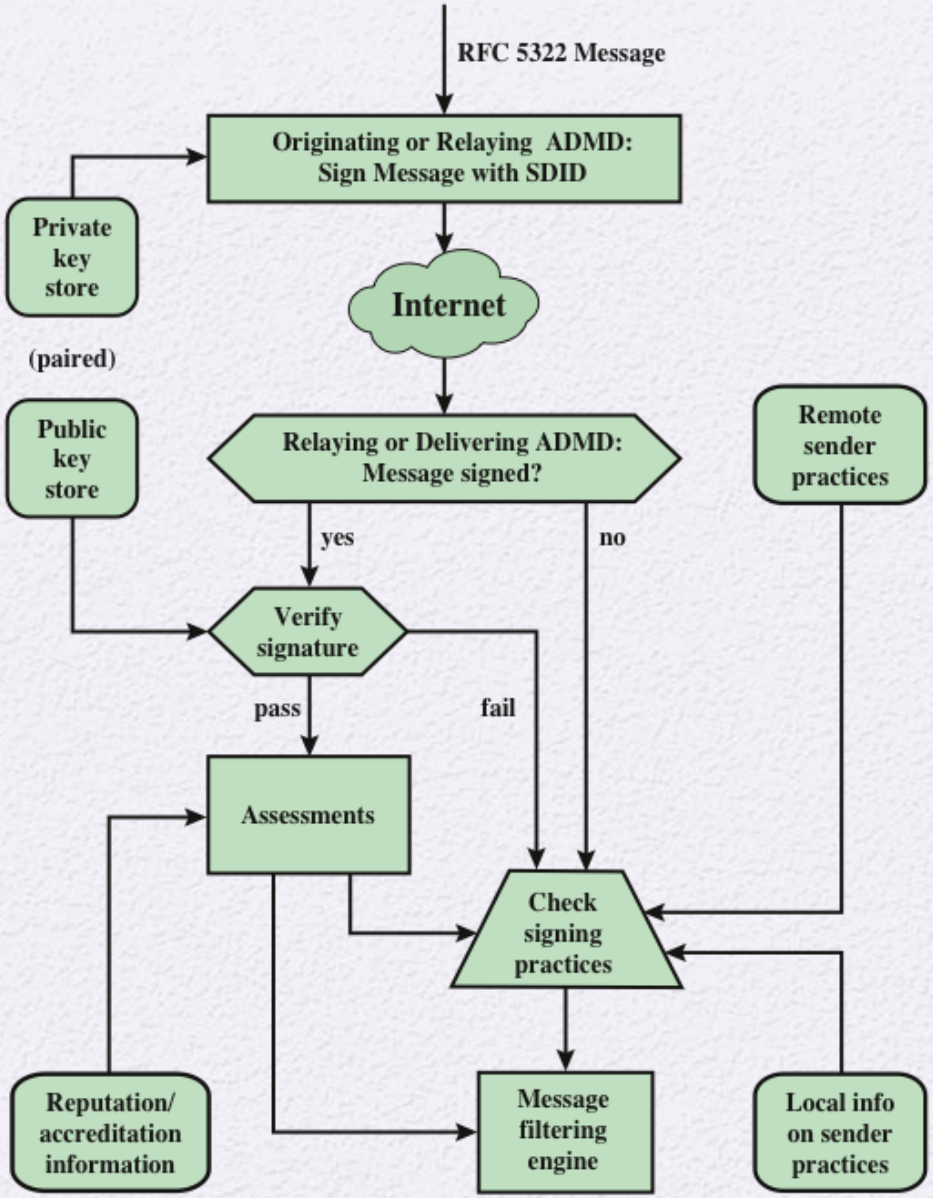


Figure 19.6 DKIM Functional Flow

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