

# ISA 766

## Internet Security Protocols

Lecture #10 (Fall 05)  
Secure Email, Virus, Worm  
(modified by Saket Kaushik)  
(modifications in blue)

### Objectives of Lecture

- Understand how e-mail systems work.
- Discuss the threats to e-mail services.
- Security measures to mitigate threats
- What more is required?
  
- Slides are very detailed, we will skip a lot of them (skipped slides are marked in this color)

## Why Study E-mail Security?

- After web browsing, e-mail is the most widely used network-reliant application. (Billions of email addresses in use)
- Mail servers, after web servers, are the most often attacked Internet hosts. (Millions of mail servers are deployed)
- Basic e-mail (as proposed in RFCs on Email) offers little security, counter to public perception.
- Good technical solutions are available, but not widely used.
  - If we understand why this is so, we might understand something about why security is 'hard'.
- Examine some open problems related to E-mail

## What It Is and How It Works

- What is an e-mail?
  - RFC 2822 (later 822) and MIME (RFC 2045—)
  - Message  $\equiv$  Envelope + Content
  - Envelop  $\rightarrow$  Headers (From, To, Date, *etc.*)
  - Content  $\rightarrow$  Subject matter, MIME content
- How are e-mails transported, accessed and stored?
  - MUAs and MTAs
  - SMTP, POP3 and IMAP

## RFC 2822

- On the network, an email is a string of ASCII characters in a format specified by RFC 2822.
  - Reason why messages appear larger in size on the network, than when stored on a PC.
- Two parts:
  - The **envelope**: information needed for transmission and delivery
  - The **content**: object to be delivered to the recipient
- Use of ASCII causes problems for non-ASCII message bodies, e.g. attachments – more later.

## An Example RFC 822 Message

```
From: john@toys.com
To:   santa@wonderland.net
Cc:   angel@heaven.net
Subject: RFC 822 example
Date: Fri, 15 Nov 2002 13:58:49
```

```
Hi Santa! I've been a good boy all
year long. Won't you please get me
a bike this X-mas?
```

## Note

- Two distinct parts
  - a) A set of header fields like (sender: abc@gmu.edu), (to: xyz@umd.edu), *etc.*
  - b) A set of words (characters) forming the content
- First part contains information regarding
  - who is the message from,
  - who to send it to (multiple recipients possible),
  - what is the message about,
  - is it urgent, *etc.*
- Second part is the content of a message

## Type of content in a message

Not just text. MSWord, PDF, JPEG, *etc.*, can be sent  
Problem? Letting recipient know what is being sent  
(length, type and number of objects)

Solved by: MIME (Multipurpose Internet Mail Extensions)

- Explicitly indicate what objects are being sent
- Uses extra header fields in RFC 822 e-mails to specify form and content of extensions.
- Supports a variety of content types, but e-mail still ASCII-coded for compatibility.
- Specified in RFCs 2045-2049.

## MIME headers

MIME specifies 5 new e-mail header fields:

- `MIME-Version` (must be 1.0)
- `Content-Type`
- `Content-Transfer-Encoding`
- `Content-ID` -  
optional
- `Content-Description` -  
optional

## MIME Content-Type

- Seven major content types with 15 sub-types.
- `Multipart` content type has 4 subtypes.
- Most important is `Multipart/mixed`, indicating that the body contains multiple parts.
- Each part can be a separate MIME message – hence nesting of MIME messages to any level.
- Parts separated by a boundary string defined in `Content-Type` field.

## Content-Transfer-Encoding

- RFC 822 e-mails can contain only ASCII characters.
- MIME messages intended to transport arbitrary data.
- The Content-Transfer-Encoding field indicates how data was encoded from raw data to ASCII.
- base64 is a common encoding:
  - 24 data bits (3 bytes) at a time encoded to 4 ASCII characters.

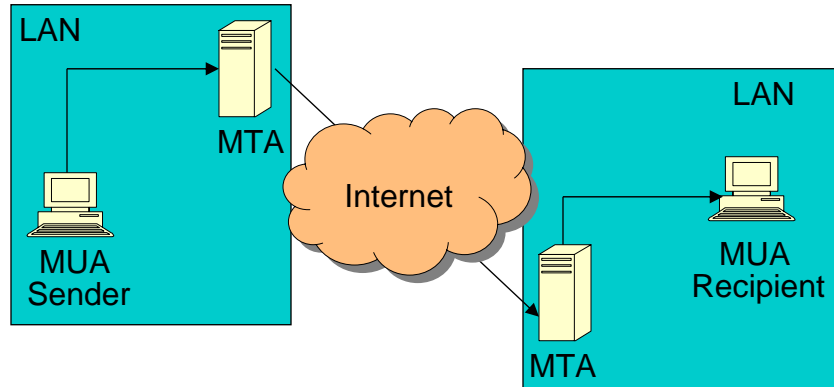
## An Example MIME Message

```
From: santa@wonderland.net
To: somebody@gmu.edu
Subject: That document
Date: Wed, 3 Nov 2004 19:55:47 -0000
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="----next part"
-----next part
Content-Type: text/plain; charset="iso-8859-1"
Content-Transfer-Encoding: 7bit

Hello, here's that document I said I'd send. Regards, Santa
-----next part
Content-Type: application/x-zip-compressed; name="report.zip"
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename="report.zip"

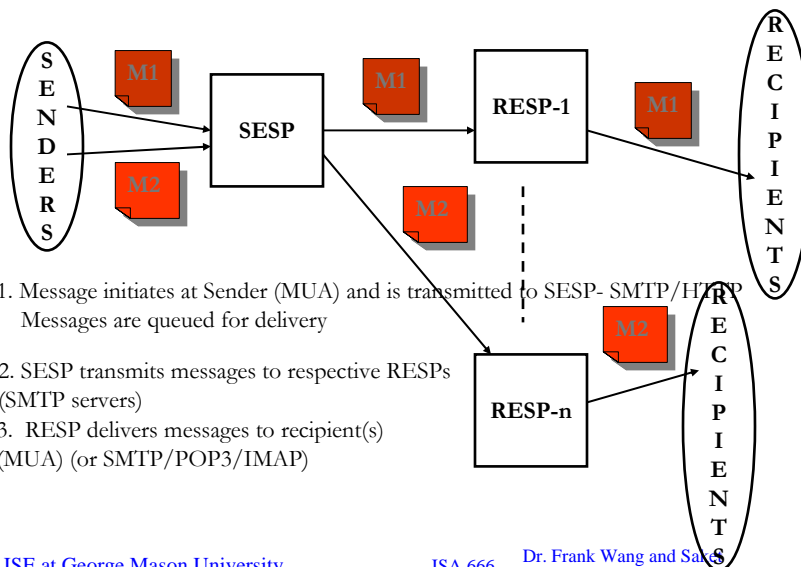
rfvbnj756tbGHUSISyuhssia9982372SHHS3717277vsgGJ77JS77HFyt6GS8
-----next part--
```

## How Are E-mails Transported?



- MUA= Mail User Agent, *aka* Mail Client
- MTA=Mail Transport Agent, *aka* Mail Server

## Message Transmission



1. Message initiates at Sender (MUA) and is transmitted to SESP- SMTP/HTTP  
Messages are queued for delivery
2. SESP transmits messages to respective RESPs (SMTP servers)
3. RESP delivers messages to recipient(s) (MUA) (or SMTP/POP3/IMAP)

## Composition

- MUA = Mail client is a program running on Sender's machine, e.g. Microsoft Outlook or IBM iPlanet
- Sender supplies To: and Subject: fields and message body.
- MUA translates into RFC 822 message and connects across LAN to MTA = Mail server.
- MUA instructs MTA using a protocol called SMTP (or a proprietary alternative) and sends RFC 822 message.

## Delivery (across email domains)

1. Sender's MTA (SESP) 'discovers' address of the Recipient's MTA (RESP) using DNS {MX (– Mail Exchange) records}
2. Sender's MTA opens connection to Recipient's MTA and uses SMTP (Simple Mail Transfer Protocol)
3. Recipient's MTA may deliver to Recipient's MUA or may store message locally for later retrieval across LAN.
4. Intermediate MTAs may be involved.

## Simple Mail Transfer Protocol

- Basic SMTP is defined in [RFC 2821](#), widely used for MUA-MTA and MTA-MTA conversations.
- SMTP uses TCP on port 25 for connections, so SMTP traffic carried over LAN and Internet and is (largely) unprotected.
- ‘Skilled’ user can talk SMTP directly over a telnet connection to remote MTA, supplying `From:` field of choice.
- Often mail servers don’t verify the authenticity of supplied `From:` field
  - Forgery is often possible (avoided by reverse MX/ RR checks)

## Where’s The E-mail?

- UNIX systems often transfer e-mail from MTA to files in local client file system.
  - Use elm, pine, xmail to read e-mail on client machine.
  - UNIX username and password controls access to client mailbox.
  - Thus security of mail system partly relies on user account security.

## Where's The E-mail?

- Can also store e-mail on mail server rather than on client machine.
- Two common protocols for mail client-mail server interaction:
  - POP=Post Office Protocol (RFC 1939, v3)
  - IMAP=Internet Message Access Protocol (RFC 2060, v4rev1).
  - Other proprietary protocols also exist.
- Username and password required before mail can be accessed
  - often sent over network in clear.
- Secure extensions to POP and IMAP also exist:  
challenge/response based on user password.

## Web-based Access

- Useful for users with web browser but no mail client, e.g. user on the road.
- Username/password combination to control access.
- Now entire client-server interaction over HTTP instead of POP/IMAP.
  - What happens to passwords in cybercafe? Keyboard sniffers?
  - Does history on browser reveal mail messages read and sent?
- Possibly protected using SSL.

## E-mail Security Threats

We distinguish two kinds of threats to the security of e-mail:

### Threats to the security of e-mail itself

- Confidentiality, Integrity, Availability
- Availability – DDoS (Email Bombs), Spam

### Threats that are enabled by the use of e-mail.

- Almost all firewalls let e-mails pass through!
- An attractive vehicle for attackers to launch attack

Other classifications are possible!

Not an exhaustive list of threats!

## Threats to E-mail

- Loss of confidentiality.
  - E-mails are (*usually*) sent in clear over open networks.
  - E-mails stored on potentially insecure clients and mail servers.
    - Stored in File system as plain text
- Loss of integrity.
  - No integrity protection on e-mails; message body can be altered in transit or on mail server.

## Threats to E-mail

- Lack of data origin authentication.
  - *Is this e-mail really from the person named in the From: field?*
  - *Identity spoofing is easy*
  - *E-mail could also be altered in transit.*
- Lack of non-repudiation.
  - *Can I rely and act on the content? (integrity)*
  - *If so, can the sender later deny having sent it? Who is liable if I have acted?*

## Threats to E-mail

- Lack of notification of receipt.
  - *Has the intended recipient received my e-mail and acted on it?*
    - *Certified Email functionality is not provided by SMTP*
  - *A message locally marked as 'sent' may not have been delivered.*
    - *SMTP ensures best effort delivery, generates error messages for undelivered messages*
    - *Can be used as an means to send SPAM or launch DoS attacks*
    - *Many SMTP servers 'turn off' error message reporting and delivery*

## Threats to E-mail

- Denial of service (DoS, DDos) attacks.
  - *Email bombs and excessive spam.*
    - *Bombs originating from a single source are easy to detect, and countermeasures can be taken. [Hence not a big concern for now]*
    - *Distributed email-bomb attacks can be more dangerous (none reported so far).*
  - *Spam is an annoyance; can cause decreased productivity, loss of communications due to usage of memory space*

## Threats to E-mail

- Spam and Phishing attacks
  - *Identity spoofing for sending unsolicited mail*
  - *Email relays are exploited for sending spam. (largely fixed now using ORDBs)*
    - *ORDBs can be used as reputation servers to block all e-mail originating from that servers on the list*
  - *Zombies or unsuspecting compromised machines are used for routing spam.*
  - *Fraudulent mail (phishing attacks) very common*
  - *Level of attack sophistication is rising everyday*
  - *Latest reports -- 80% of all e-mail is now spam.*

## Threats Enabled by E-mail

- **Deliberate/unintentional information leakage**
  - *It's much easier to distribute information by e-mail than it is by paper and snail mail.*
  - *Disclosure may be of inappropriate, sensitive or proprietary information.*
  - *Lack of non-repudiation may help offenders get away*
- **Exposure of systems to malicious code**
  - *Social engineering to launch malicious code on receiving host*
  - *Malicious code can be virus/worm (harmful to the host) or trojan/zombie (for carrying out attacks on other hosts)*

## Threats Enabled by E-mail

- **Exposure of systems to denial of service attacks.**
  - *E-mail server attached to network, may be vulnerable to DoS attacks.*
  - *More relevant with increasing dependence on e-mail as the communications tool.*
  - *DoS on mail server may compromise other network services too.*

## Threats Enabled by E-mail

- Unauthorized access to systems.
  - *Mail servers (OS and application) can have many security vulnerabilities; they are also attached to external networks.*
  - *Malicious code can open up entry points into a network.*
    - *E.g.: A zombie on a machine can be used to launch network attacks even though attacker outside the network has no connectivity to intranet.*
    - *Easily bypass perimeter security*

## Threats Enabled by E-mail

- Any more threats?

## Secure E-mail Standards and Products

- Several standards/protocols: SMTP AUTH, STARTTLS, Certified Email, S/MIME, *etc.*
- SMTPAUTH/STARTTLS are extensions to SMTP for allowing PKI based authentication of client and server SMTP
- Certified Email provides non-repudiation.
  - *Requires an additional protocol on top of SMTP*
- S/MIME and PGP.
  - Provide security (confidentiality/integrity) for objects in the content
- Several mechanisms/products to solving availability (spam/phishing) problem
  - Bayesian filters, message bonds, human initiation, *etc.*

## SMTP AUTH – RFC 2554

- SMTP service extension for authentication
  - SMTP client indicates authentication mechanism to SMTP server
  - Server and client can negotiate on the security layer for subsequent protocol interactions
  - SASL (simple authentication and security layer) profile
  - New command/reply: AUTH/235, AUTH/504 *etc.*  
(authentication successful or not)

## SMTP AUTH

- What does it really provide?
  - Password based authentication
    - Requires prior knowledge
  - PKI based authentication
    - May not scale to individual senders
    - Could be used to authenticate client and server SMTP

## SMTP AUTH -- Example

```
S: 220 smtp.example.com ESMTP server ready
C: EHLO jgm.example.com
S: 250-smtp.example.com
S: 250 AUTH CRAM-MD5 DIGEST-MD5
C: AUTH FOOBAR
S: 504 Unrecognized authentication type.
C: AUTH CRAM-MD5
S: 334
  PENCeUxFREJoU0NnbmhNWitOMjNGNndAZWx3b29kLm
  lubm9zb2Z0LmNvbT4=
C: ZnJlZCA5ZTk1YWVlMDljNDBhZjJiODRhMGMyYjNiYmFl
  Nzg2ZQ==
S: 235 Authentication successful.
```

## Secure SMTP over TLS – RFC 2487

- **SMTP service extension for confidential, authenticated communication over TLS**
  - Protection against eavesdroppers/attackers
    - A malicious party cannot monitor or alter communications
  - Helpful if communication goes through routers not trusted by the SMTP client and server
  - New command: **STARTTLS**
  - Followed by TLS handshake
  - After handshake protocol is set to initial state

## Secure SMTP over TLS

- **Confidentiality and integrity problems solved?**
  - Not necessarily
- **Note**
  - Messages delivered in multiple hops.
    - Even though data is secure on the network, it is stored in plain text at the end machine
    - An insider can monitor or alter messages
- **What is needed to rectify this problem?**

## Certified Email

- The protocol
  - Sender sends a message to a recipient and asks for a receipt
  - Recipient must issue a receipt before reading the receipt
  - An independent trusted third party is involved in ensuring that the protocol is followed
- Commercial products are available
  - Authentica, Certifiedmail.com, Zixit, Postx, ZeroGravity, Sigaba, *etc.*

## Certified Email – Example protocol

- Taken from Abadi, Glew, Horne, Pinkas (WWW 02)
  1. Sender encrypts the message using a freshly generated symmetric key ( $K$ ), encrypts  $K$  with third party's public key  $T$ ; sends  $(M)_K$  and  $(K)_T$  to the recipient
  2. Recipient receives  $(M)_K$  and  $(K)_T$ , but can't read it unless  $K$  is known; sends a request to third party to decrypt  $K$
  3. Third party decrypts  $K$ , sends  $K$  to recipient, and a receipt to sender

## Certified Email

- Ensures:
  - Fair exchange
  - Sender gets the evidence of sending a message
  - Non-repudiation of origin
  - Non-repudiation of receipt
  - Authenticity of sender and recipient
  - Integrity of message
  - Confidentiality– only sender and recipient can extract the message content
  - Timeliness – duration of the protocol is finite

## Certified Email

- Difficulties
  - Requires a third party to be involved
  - Requires a protocol other than or on top of SMTP
  - Guarantees given based on Public-Private key pairs
    - Consequently, requires more sophisticated senders and recipients
    - Scalability issues?

## S/MIME

- Originated from RSA Data Security Inc. in 1995.
- Further development by IETF S/MIME working group at:  
[www.ietf.org/html.charters/smime-charter.html](http://www.ietf.org/html.charters/smime-charter.html).
- Version 3 specified in RFCs 2630-2634.
- Allows flexible client-client security through encryption and signatures.
- Widely supported, e.g. in Microsoft Outlook, Netscape Messenger, Lotus Notes.

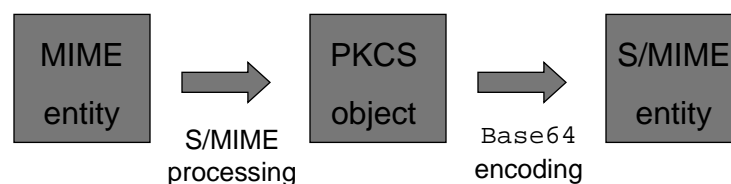
## S/MIME Message Formats

- As the name suggests, S/MIME adds security features by extending MIME.
- S/MIME adds 5 new content type/subtype combinations, including:
  - application/pkcs7-mime;  
smime-type=enveloped-data
  - application/pkcs7-mime;  
smime-type=signed-data
  - multipart/signed

## S/MIME Processing

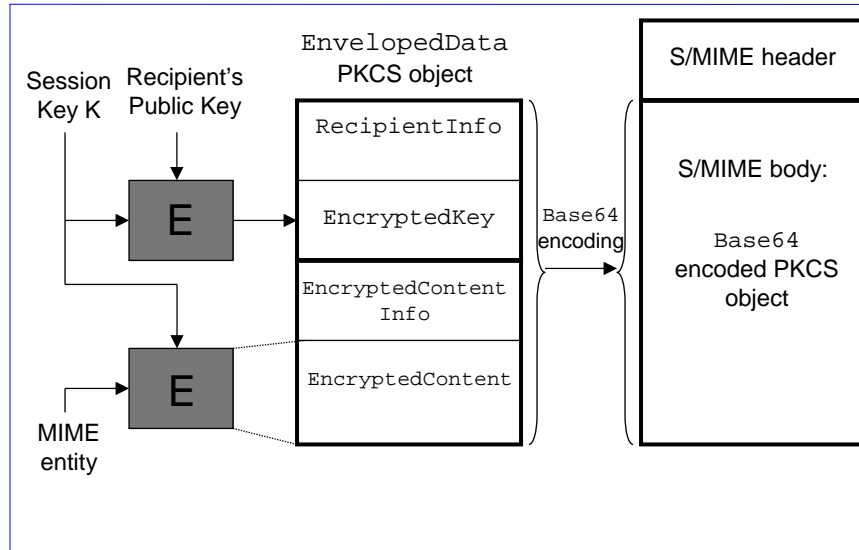
- S/MIME processing can be applied to any MIME entity:
  - One part of a MIME multipart message, perhaps one that is itself of S/MIME Content-Type.
  - End result of S/MIME processing is always another MIME entity, of S/MIME Content-Type.
  - Hence encryption and signature can be applied one after another, and in either order.

## S/MIME Processing – Sender



- Initial S/MIME processing produces a PKCS object.
- PKCS=Public Key Cryptography Standard.
- PKCS object includes information needed for processing by recipient as well as the original content.
- But PKCS objects are in binary format, hence need for further base64 encoding to produce final result MIME object of S/MIME content-type.
- Recipient performs steps in reverse.

## S/MIME enveloped-data



## S/MIME enveloped-data

An example message (from RFC 2633):

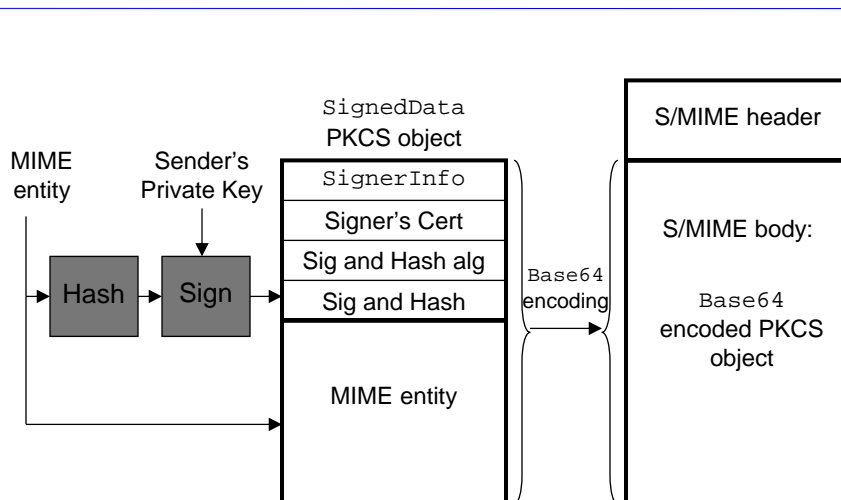
```
Content-Type: application/pkcs7-mime;
  smime-type=enveloped-data; name=smime.p7m
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7m

rfvbnj756tbBghyHhHUujhJhjH77n8HHGT9HG4VQpfyF467GI
7n8HHGghyHhHUujhJh4VQpfyF467GhIGfHfYGTTrfvbnjT6jHd
f8HHGTrfvhJhjH776tbB9HG4VQbnj7567GhIGfHfYT6ghyHh6
```

## S/MIME enveloped-data

- S/MIME enveloped-data type gives data confidentiality service through encryption.
- S/MIME header contains original To:, From: and Subject: fields, so protection not complete.
- Symmetric algorithm with session key for efficient bulk encryption and asymmetric encryption using recipient's public key to protect session key.
- Recipient reverses steps: obtain K using private key, then use K to decrypt EncryptedContent.
  - Algorithms needed are specified in RecipientInfo and EncryptedContentInfo blocks.

## S/MIME signed-data



## S/MIME signed-data

An example message (from RFC 2633):

```
Content-Type: application/pkcs7-mime;  
    smime-type=signed-data; name=smime.p7m  
Content-Transfer-Encoding: base64  
Content-Disposition: attachment; filename=smime.p7m
```

```
567GhIGfHfYT6ghyHhHUujpFyF4f8HHGTrfvhJhjH776tbB97  
7n8HHGT9HG4VQpFyF467GhIGfHfYT6rfvbnj756tbBgHyHhHU  
HUujhJh4VQpFyF467GhIGfHfYGTTrfvbnjT6jH7756tbB9H7n8
```

## S/MIME signed-data

- S/MIME signed-data type gives integrity, authenticity and non-repudiation services using sender signatures.
- Multiple signers supported – prepare a `SignerInfo` block for each one.
- Recipient checks signature using MIME entity embedded in PKCS object and public (verification) key of sender.
- Recipient without S/MIME capability cannot read the original message (even if he doesn't care about signatures).

## S/MIME Clear Signing

- Uses MIME multipart/signed content type.
- First part contains MIME entity to be signed.
- Second part contains S/MIME application/pkcs7-signature entity, created as for signed-data type.
- Recipients who have MIME but not S/MIME capability can still read message contents.
- Recipients who have S/MIME capability use first part as MIME object in S/MIME signature verification.

## S/MIME Clear Signing

```
Content-Type: multipart/signed;  
  protocol="application/pkcs7-signature";  
  micalg=sha1; boundary=boundary42  
--boundary42  
Content-Type: text/plain  
  
This is a clear-signed message.  
--boundary42  
Content-Type: application/pkcs7-signature;  
  name=smime.p7s  
Content-Transfer-Encoding: base64  
Content-Disposition: attachment; filename=smime.p7s  
  ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB9HG4VQpfyF4674  
  VQpfyF467GhIGfhfYT6jH77n8HHGghyHhHUujhJh756tb6  
--boundary42--
```

## S/MIME Algorithms

- Symmetric encryption:
  - DES, 3DES, RC2 with 40 and 64 bit keys.
- Public key encryption:
  - RSA, ElGamal.
- Hashing:
  - SHA-1, MD5.
- Signature:
  - RSA, Digital Signature Standard (DSS).

## PGP

- PGP=“Pretty Good Privacy”
- **Interactive protocol**
- First released in 1991, developed by Phil Zimmerman, provoked export control and patent infringement controversy.
- Freeware: OpenPGP and variants:
  - [www.openpgp.org](http://www.openpgp.org), [www.gnupg.org](http://www.gnupg.org)
- Commercial: formerly Network Associates International, now PGP Corporation at [www.pgp.com](http://www.pgp.com)
- OpenPGP specified in RFC 2440 and defined by IETF OpenPGP working group.
  - [www.ietf.org/html.charters/openpgp-charter.html](http://www.ietf.org/html.charters/openpgp-charter.html)
- Available as plug-in for popular e-mail clients, can also be used as stand-alone software.

## PGP

- Functionality similar to S/MIME:
  - encryption for confidentiality.
  - signature for non-repudiation/authenticity.
- One level of processing only, so less flexible than S/MIME.
- Sign before encrypt, so signatures on unencrypted data.
  - Sigs can be detached and stored separately.
- PGP-processed data is base64 encoded and carried inside RFC822 message body.

## PGP Algorithms

Broad range of algorithms supported:

- Symmetric encryption:
  - DES, 3DES, AES and others.
- Public key encryption of session keys:
  - RSA or ElGamal.
- Hashing:
  - SHA-1, MD-5 and others.
- Signature:
  - RSA, DSS, ECDSA and others.

## PGP Key Rings

- PGP supports multiple public/private keys pairs per sender/recipient.
- Keys stored locally in a *PGP Key Ring* – essentially a database of keys.
- Private keys stored in encrypted form; decryption key determined by user-entered passphrase.
- So security once again depends on users remembering passwords!

## Key Management for PGP and S/MIME

- PGP and S/MIME use
  - public keys for encrypting session keys / verifying signatures.
  - private keys for decrypting session keys / creating signatures.
- Where do these keys come from and on what basis can they be trusted?

## S/MIME Key Management

- S/MIME uses public-key certificates and certificate chains to validate public keys.
- Certificates comply with ISO/ITU-T X.509v3 public key certificate standard.
- Same standard as used to define certificates in SSL/TLS and IPsec.

## X.509 Certificate Format

An X.509 certificate is a data structure including the following fields:

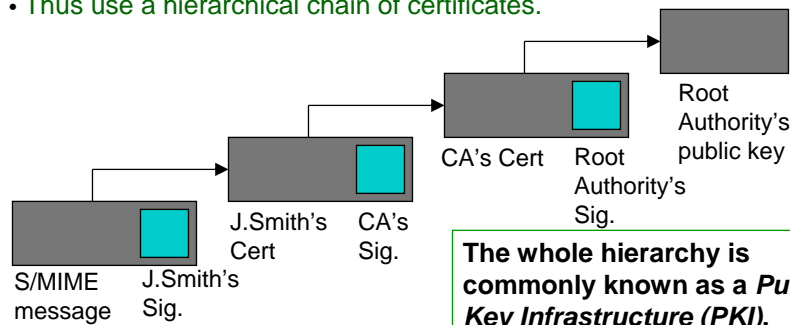
- Version number (1, 2, 3 or 4).
- Serial number of certificate.
- Issuer name.
- Validity period.
- Subject name – a “Distinguished Name”.
- Subject’s public key info: algorithms (eg RSA); parameters (eg size); the public key itself.
- Extension fields.
- The Issuer’s signature on all the above fields.

## Use of X.509 Certificates

- Issuer commonly called a Certification Authority (CA).
- Third party can check validity of Issuer's signature in certificate.
- Certificate can therefore vouch that subject is in possession of the private key corresponding to the public key in the certificate.
- But first need authentic copy of Issuer's public key!

## X.509 Certificate Chains

- Repeat the checking process on Issuer's certificate,... until *root of trust* is reached
  - a certificate embedded in browser or e-mail client from a root authority whose public key is implicitly trusted.
- Thus use a hierarchical chain of certificates.



## X.509 and S/MIME

- Subject's public key can be for signature verification or for encryption – specified in an X.509 extension field.
- X.509 Subject name must be a distinguished name, e.g. “c=US, o=company, ou=sales, cn=John Smith”
- So use another X.509 extension field “Alternative Name” to include e-mail address in certificate.

## S/MIME Key Management Issues

Some issues:

- **Interpretation:** End-user is asked: “Do you trust this certificate?” How should a security-unaware user interpret this?
- **Scale:** How to manage large populations of users?
- **Revocation:** How to communicate to all users that a certificate is no longer valid?
- **Liability:** How much liability (if any) does the Issuer accept? Maybe OK if Issuer is your employer.
- **Private key storage:** End-user's desktop most likely, maybe password protected.
- **Certificate issuance procedures** (aka registration): Is this really J. Smith? OK, which J. Smith?

## PGP Key Management

- PGP adopts a completely different trust model – the *web of trust*.
- No centralised authority like a root of trust in X.509.
- Individuals sign one another's public keys, these "certificates" are stored along with keys in key rings.
- PGP computes a *trust level* for each public key in key ring
  - Complex formula based on number of signatures on that key, and level of trust in each signature.
- Users interpret trust level for themselves.

## PGP Key Management Issues

- Original intention was that all e-mail users would contribute to web of trust.
- Reality is that this web is sparsely populated.
- How should security-unaware users assign and interpret trust levels?
- Later versions of PGP support X.509 certs.
- PGP fine for small groups and out-of-band public key distribution (eg floppy).

## Open Problems in E-mail Security

- PGP and S/MIME counter the basic threats to confidentiality, integrity and authenticity of e-mail quite well (assuming good key management).
- They don't protect against other threats
  - Virus, virus hoax
  - Worm
  - Trojan horse
  - Spam
  - DDoS
  - Traffic analysis
  - Unauthorized use

## Computer Virus

- **What is a virus?**

A computer virus is a small program written to alter the way a computer operates, without the permission or knowledge of the user. A virus must meet two criteria:

  - **It must execute itself.** It will often place its own code in the path of execution of another program.
  - **It must replicate itself.** For example, it may replace other executable files with a copy of the virus infected file. Viruses can infect desktop computers and network servers alike.

## Computer Virus (Cont'd)

- **Five recognized types of viruses** (according to Symantec)
  - **File infector viruses**
    - These viruses normally infect executable code, such as .com and .exe files
  - **Boot sector viruses**
    - Boot sector viruses infect the system area of a disk--that is, the boot record on floppy disks and hard disks.
  - **Master boot record viruses**
    - Master boot record viruses are memory resident viruses that infect disks in the same manner as boot sector viruses.
  - **Multi-partite viruses**
    - Multi-partite (also known as polypartite) viruses infect both boot records and program files. These are particularly difficult to repair.
  - **Macro viruses**
    - These types of viruses infect data files.

## Worm

- **What is worm?**
  - Computer program that **replicates itself** from system to system **without the use of a host file**.
  - This is in contrast to viruses, which requires the spreading of an infected host file.
    - Although worms generally exist inside of other files, often Word or Excel documents, there is a difference between how worms and viruses use the host file.
  - **Worm propagates extremely fast**
    - It is possible for a worm to infect 90% of the susceptible hosts in minutes
    - Worm has growth pattern similar to real-world biological virus.
    - The propagation of worm fits well with simple epidemiological mathematical models

## Worm (Cont'd)

- Notorious worms
  - CodeRed (2001)
    - MS-Windows NT/2000
  - Nimda (2001)
    - MS-Windows 95/98/ME/NT/2000
  - Slammer (2003)
    - MS SQL Server 2000
    - MS Desktop Engine (MSDE) 2000
  - Blaster (2003)
    - MS-Windows NT/2000/XP/Server 2003
  - Mydoom (2004)

## Trojan Horse

- **What is a Trojan horse?**

Trojan Horses are impostors--files that claim to be something desirable but, in fact, are malicious.

  - Trojan horse does not replicate itself.
  - For a Trojan horse to spread, you must, invite these programs onto your computers
    - for example, by opening an email attachment or downloading and running a file from the Internet.
- Trojan horse example
  - PWSteal.Trojan

## Detection of Computer Virus and Worms

- Based on predefined attack signatures
  - Attack signatures won't be available before the attack happens and causes damage
  - Signature updates can be overwhelming
- Based on contact tracing and transmission chain identification
  - “borrow” the idea of classical epidemiological method that uses contact investigation to identify the transmission chain
  - Do not need attack signatures, and could potentially detect new (previously-unknown) virus or worm

## Decidability Issue on E-mail Virus and Worm Detection

- Two important theoretical results
  - Fred Cohen has demonstrated that **there is no algorithm that could detect all computer viruses**
    - F. Cohen. Computer Viruses: Theory and Experiments. *Computers & Security*, Vol. 6(1), Pages 22–35, 1987
  - David Chess and Steve White have added the bad news and pointed out that **there exist computer viruses that no algorithm can detect**
    - D. M. Chess and S. R. White. An Undetectable Computer Virus. In *Virus Bulletin Conference 2000*.  
<http://www.research.ibm.com/antivirus/SciPapers/VB2000DC.pdf>

## Current Defences against Virus and Worm

- Install Anti-virus software
  - Overwhelming the customer with frequent signature updates
- Install patches (as fast/much as you can)
  - Overwhelming the customer with
    - Compatibility issue
    - Regression test effort
- Automatic patching
  - How about letting Microsoft automatically updating your Windows OS?
- Active Immunization (just like vaccine in real-world)
  - Anti-worm (new method proposed in WORM'04)
  - What else?