IC3 - Network Security

M.Sc. in Information Security
Royal Holloway, University of London

IC3 - Network Security

Lecture 5
E-mail Security

Objectives of Lecture

- Understand how e-mail systems operate over networks.
- Classify the threats to the security of e-mail.
- Study two schemes that can be used to add security to e-mail systems.
  - S/MIME (Secure Multipurpose Internet Email Extensions).
  - PGP (Pretty Good Privacy).
- Examine what other security measures are needed to ensure security for e-mail systems.
- Bring together diverse elements to understand the security of a key application.

Contents

1 Why Study E-mail Security?

- After web browsing, e-mail is the most widely used network-reliant application.
- Yet basic e-mail 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'.
- E-mail security makes a good case study for IC3.
  - A single, well-defined network application whose security we can evaluate.

2 What Email Is and How It Works

- What is an e-mail?
  - IETF Standards define the message format
    - RFC 822
    - MIME (Multipurpose Internet Email Extensions)
- How are e-mails transported, accessed and stored?
  - MUA (Mail User Agents)
  - MTA (Mail Transfer Agents)
  - More IETF Standards
    - SMTP (Simple Mail Transfer Protocol)
    - POP3 (Post Office Protocol)
    - IMAP (Internet Message Access Protocol)

RFC 822

- An e-mail is a message made up of a string of **ASCII characters** in a format specified by RFC 822 (dating from 1982).
  - Latest version is RFC 2822 (2001)
- Two parts, **separated by blank** line:
  - The **header**: sender, recipient, date, subject, delivery path,…
  - The **body**: containing the actual message content.
- Use of ASCII causes problems for non-ASCII message bodies, e.g. attachments – more later.

An Example RFC 822 Message

```
From: Allan.Tomlinson@rhul.ac.uk
To:   Kenny.Paterson@rhul.ac.uk
Cc:   kennypaterson@hotmail.com
Subject: RFC 822 example
Date: Fri, 18 Nov 2005 13:58:49

This is just a test message to
illustrate RFC 822. It's not very long
and it's not very exciting. But you get
the point.
```


MIME

- MIME specifies a standard format for encapsulating multiple pieces of data into a single Internet message.
- Extends RFC 822 to allow e-mail to carry non-textual content, non-ASCII character sets, long messages.
- Uses extra header fields in RFC 822 e-mails to specify form and content of extensions.
- Supports various content types, but e-mail still ASCII-coded for compatibility with RFC 822.
- Specified in IETF standards RFC 2045-2049.

MIME headers

MIME specifies 5 new RFC 822 header fields:

1. `MIME-Version` (must be 1.0)
2. `Content-Type`
3. `Content-Transfer-Encoding`
4. `Content-ID` - optional
5. `Content-Description` - optional

N.B. An additional *optional* header field, `Content-Disposition`, is also widely used to handle *attachments* and their presentation. This MIME field is specified in RFC 2183

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 only allows ASCII characters.
- MIME messages 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.
  - Results in data expansion.

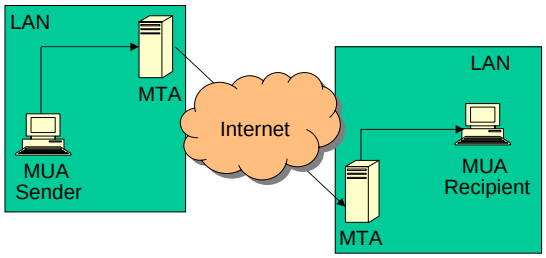
An Example MIME Message

```
From: Allan.Tomlinson@rhul.ac.uk
To: Kenny.Paterson@rhul.ac.uk
Subject: That document
Date: Wed, 16 Nov 2005 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

Kenny, here's that document I said I'd send. Regards, Allan
--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

Composition and Delivery – 1

- MUA = Mail client is a program running on Sender's machine, e.g. Microsoft Outlook or Netscape Messenger.
- 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 **Simple Mail Transfer Protocol** (SMTP), or a proprietary alternative, and sends RFC 822 message.

Composition and Delivery – 2

- Sender's MTA uses DNS (Domain Name Service) to find IP address of recipient's MTA (could be local) based on `To:` field.
- Sender's MTA opens connection to Recipient's MTA and uses SMTP to instruct remote MTA and transfer RFC 822 message
  - often across public Internet.
- Intermediate MTAs may be involved.
- Recipient's MTA may deliver to their MUA or may store message locally for later retrieval across LAN.

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.
- So forging e-mail is nearly trivial (though mail headers usually give away source IP address).

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:
  - POP3 (RFC 1939, v3).
  - IMAP (RFC 3501, v4rev1).
- Username and (hashed) password required before mail can be accessed.
  - often sent over network in clear.
  - as used at RHUL: Microsoft Outlook mail client, and Microsoft Exchange mail server.
- Secure extensions to POP and IMAP also exist.

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.

3 E-mail Security Threats

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

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

Threats to **an organisation** that are **enabled by the use of e-mail.**

Other classifications are possible!
Not an exhaustive list of threats!

Threats to E-mail

- Loss of **confidentiality**.
  - *E-mail is sent in clear over open networks.*
  - *E-mail stored on potentially insecure clients and mail servers.*
  - *Ensuring confidentiality may be important for e-mail sent within an organisation.*
- Loss of **integrity**.
  - *No integrity protection on e-mails; 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?*
    - *How many* `Allan.Tomlinson`*'s are there?*
    - *Recall SMTP directly over telnet allows forgery of all e-mail fields!*
  - *E-mail could also be altered in transit.*
  - *Even if the* `From:` *field looks fine, who was logged in as* `Allan.Tomlinson` *when the e-mail was composed?*
    - *Sharing of e-mail passwords common.*

Threats to E-mail

- Lack of **non-repudiation.**
  - *Can I rely on, and act on, the content?*
  - *If so, can the sender later deny having sent it? Who is liable if I have acted?*
  - *Example of stock-trading via e-mail.*
- Lack of **notification of receipt.**
  - *Has the intended recipient received my e-mail and acted on it?*
  - *A message locally marked as 'sent' may not have been delivered.*

Threats *enabled* by E-mail

- Disclosure of sensitive information.
  - *It's easier to distribute information by e-mail than it is by paper and snail mail.*
  - *Disclosure may be deliberate (and malicious) or unintentional.*
  - *Disclosure may be internal or external (e-mail crosses LANs as well as the Internet).*
  - *Disclosure may be of personal, inappropriate, commercially sensitive, or proprietary information.*
  - *Can lead to loss of reputation and possibly to dismissal of staff.*

Threats *enabled* by E-mail

- Exposure of systems to malicious code.
  - *Today, e-mail is the main vector by which computer viruses spread.*
  - *Self-replicating code embedded in e-mail, exploits features/vulnerabilities of e-mail client.*
    - *Visual basic script;*
    - *Javascript in html formatted e-mail;*
    - *.exe attachments.*
  - *Often requires user interaction to propagate an e-mail virus (but not always) .*
  - *Virus outbreak can result in Denial of Service.*

Threats *enabled* by E-mail

- Exposure of systems to Denial of Service (DoS) attacks.
  - *E-mail server attached to network, may be vulnerable to DoS attacks.*
  - *More relevant with increasing dependence on e-mail as a communications tool.*
  - *For example,*
    a virulent worm using large percentage of network capacity to spread
    will prevent efficient use of e-mail as well as slowing down web browsing.

Threats *enabled* by E-mail

- Exposure of *individuals* to denial of service attacks.
  - *Mail bombing and excessive spam.*
  - *Individuals get so swamped by incoming e-mail that they stop reading it and switch to other communication methods.*

Threats *enabled* by E-mail

- Spamming.
  - *Bulk distribution of unsolicited e-mail.*
  - *50% and more of all e-mail traffic is now spam.*
  - *Hotmail and other free e-mail systems are particularly victimised by spammers.*
  - *Anti-spam legislation in development or on the statute books in many countries.*
    - *Federal CAN-SPAM act in force in US since 1/1/2004.*
    - *Does not outlaw spamming, but controls use.*
    - *First conviction in September 2004, Nicholas Tombros.*
      - *Spamming + war-driving.*
    - *See http://www.spamlaws.com/ for details of laws.*
    - *Effectiveness of the US CAN-SPAM act and similar legislation still in question.*

Threats *enabled* by E-mail

- Relaying and blacklisting.
  - *Misconfiguration of relaying capability allows mail server to be exploited for spamming.*
  - *Guilty server can end up being placed on Open Relay Blacklist (www.ordb.org).*
  - *Result is that* **all** *e-mail from that server gets blocked by mail servers using blacklist.*

Threats *enabled* by E-mail

- Unauthorized access to systems.
  - *Mail servers can have many security vulnerabilities.*
    - *Operating system and application.*
  - *They are also attached to external networks.*
    - *Perfect target for hacker.*
  - *Lead to your mail server being used as attack platform on other systems.*
    - *your own and other peoples'.*
  - *Consequent loss of reputation and potential damages claim.*

Threats *enabled* by E-mail

- Any more threats?

4 Secure E-mail Standards and Products

- We will focus on S/MIME and PGP.
  - Other now defunct standards: PEM (privacy enhanced mail), X.400.
  - Parts of these persist: PEM introduced `base64` encoding, X.400 led to X.509 certificate standards.
- Lots of commercial products:
  - Hushmail (www.hushmail.com),
  - XenoMail,
  - Identity-based secure e-mail (www.voltagesecurity.com),…

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.
- Version 3 specified in RFCs 2630-2634.
- Version 3.1 updates in RFCs 3370, 3850-3852.
- Allows flexible client-client security through encryption and signatures.
- Widely supported, e.g. in Microsoft Outlook, Netscape Messenger, Lotus Notes.

S/MIME Message Formats

- Adds security features by extending MIME.
- Adds 5 new content type/subtype combinations, including:

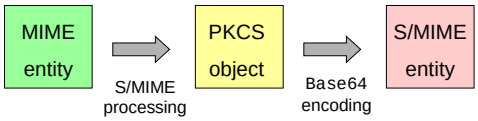
1. `application/pkcs7-mime;smime-type=`**`enveloped-data`**
2. `application/pkcs7-mime;smime-type=`**`signed-data`**
3. **`multipart/signed`**

  – Remaining types for key management messages.

S/MIME Processing

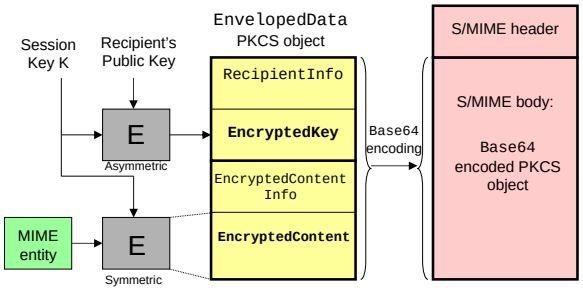
- S/MIME processing can be applied to ***any*** MIME entity:
  - One part of a MIME multipart message, perhaps one that is itself of S/MIME `Content-Type`.
  - Hence encryption and signature can be *applied one after another* (and in either order).
  - End result of S/MIME processing is always another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

S/MIME Processing – Sender



- Initial S/MIME processing produces a ***PKCS object***.
  - PKCS=Public Key Cryptography Standard, a set of specifications developed by RSA (www.ietf.org).
  - 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 3851):

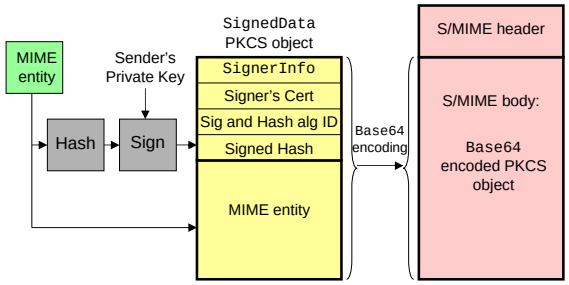
```
Content-Type: application/pkcs7-mime;
  smime-type=enveloped-data; name=smime.p7m
Content-Transfer-Encoding: base64
Content-Disposition:attachment;filename=smime.p7m

rfvbnj756tbBghyHhHUujhJhjH77n8HHGT9HG4VQpfyF467GI
7n8HHGghyHhHUujhJh4VQpfyF467GhIGfHfYGTrfvbnjT6jHd
f8HHGTrfvhJhjH776tbB9HG4VQbnj7567GhIGfHfYT6ghyHh6
```


S/MIME `enveloped-data`

- S/MIME `enveloped-data` type provides a *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.
- Asymmetric encryption using recipient's public key to protect session key.
- Recipient reverses steps: obtain session key using private key, then use this 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 3851):

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

567GhIGfHfYT6ghyHhHUujpfyF4f8HHGTrfvhJhjH776tbB97
7n8HHGT9HG4VQpfyF467GhIGfHfYT6rfvbnj756tbBghyHhHU
HUujhJh4VQpfyF467GhIGfHfYGTrfvbnjT6jH7756tbB9H7n8
```

S/MIME `signed-data`

- S/MIME `signed-data` type gives *data integrity*, *data origin authenticity* and *non-repudiation* services using sender signatures.
- Multiple signers supported
  - prepare a `SignerInfo` block for each one.
- Recipient checks signature using
  - S/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

  ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB9HG4VQpfyF467
  4VQpfyF467GhIGfHfYT6jH77n8HHGghyHhHUujhJh756tb6
--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 (Pretty Good Privacy)

- First released in 1991
  - developed by Phil Zimmerman
  - provoked export control and patent infringement controversy.
- OpenPGP (RFC 2440)
  - defined by IETF OpenPGP working group.
  - www.ietf.org/html.charters/openpgp-charter.html
- Freeware: OpenPGP and variants:
  - www.openpgp.org, www.gnupg.org
- Commercial:
  - PGP Corporation at www.pgp.com
- Available as plug-in for popular e-mail clients, can also be used as stand-alone software.

PGP (Pretty Good Privacy)

- 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 are 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!

PGP and S/MIME Key Management

- 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?
- 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:

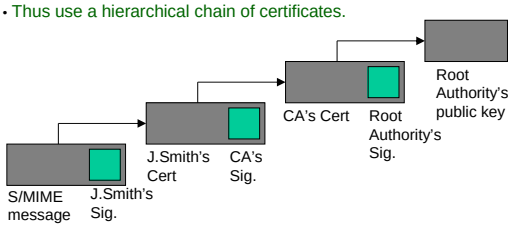
- Version number (1, 2, 3 or 4).
- Serial number of certificate.
- Issuer name.
- Validity period.
- Subject name – a “Distinguished Name”.
- Subject’s public key info: algorithms (eg RSA); parameters (eg size); the public key itself.
- Extension fields.
- The Issuer’s signature on all the above fields.

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 either:
  - a) Verification b) Encryption
  - specified in an X.509 extension field.
- X.509 Subject name must be a distinguished name,
  - e.g. "c=GB, o=company, ou=sales, cn=John Smith".
  - So X.509 ***does not directly support use of e-mail addresses***.
- 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.
- Compute a ***trust level*** for each public key in key ring
  - Formula used is based on:
    - The number of signatures obtained for the public key, and
    - User-assigned trust levels for the public keys corresponding to those signatures.
- 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.
  - Later versions of PGP support X.509 certs.
- How should security-unaware users assign and interpret trust levels?
- PGP fine for small groups combined with out-of-band public key distribution (eg floppy).

5 E-mail Security: Beyond PGP and S/MIME

- 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, DoS, Disclosure, Unauthorized use,…
- They don't protect against traffic analysis.
- Additional security measures are needed to build a secure e-mail system.

Anti-virus and Content Filtering

- Supplement mail server (or client) with content/spam filtering software
  - Block e-mail with active content or specific attachment types.
  - Reject or mark suspected spam e-mail.
  - Scan incoming and outgoing e-mail for viruses and inappropriate content.
  - Add legal disclaimers.
- Server cannot apply content filter to encrypted e-mail
  - unless it has the relevant keys.
- Significant load on mail server, may annoy end users.

Anti-spamming Protection

- Configure mail server to disable mail relay.
  - Prevents server being used as an agent to forward e-mail for third party spammers.
- Discard all e-mail from servers on Open Relay Blacklist (ORB).
- Control who can run an e-mail server in your organisation
  - Through appropriate policy setting and enforcement.

Firewalls and Mail Servers

- Place mail server behind a firewall in network.
- Configure firewall to block all external traffic to/from mail server except on port 25 (SMTP).
  - Limits attack possibilities on mail server
  - But successful attack may give access to internal systems.
  - Need additional security measures on server.
- Better to use a perimeter network.
  - Fully isolate mail server from internal and external network using firewall.
  - Configure firewall to block all internal traffic to/from mail server except on ports 25, 110 (POP3),143 (IMAP) and 53 (DNS).
  - More details in Lecture 10.

Mail Server Hardening

Take additional measures on mail server:

- Harden OS:
  - Remove unnecessary accounts, applications, and network services.
  - Apply latest OS vulnerability patches.
- Harden mail server application (e.g. sendmail, Microsoft exchange):
  - Use latest versions of software.
  - Choose appropriate configuration settings
    - (e.g. limit attachment sizes, mail relay features and file permissions).
  - Keep up-to-date with vendor patches.

Mail Server Administration

- Log mail server data and review log files regularly.
  - consider automated analysis.
- Keep up-to-date with latest patches and vulnerability alerts.
- Consider allowing only console-based administration or using SSH for remote administration.
- Take appropriate backups of mail server and user mail.

Client Side E-mail Security

Proper configuration and patching are essential:

- Disable automatic message preview.
- Disable active content processing
  - macros, ActiveX, Java, Javascript,…
- Disable POP/IMAP “remember this password?” dialogue boxes if possible.
- Consider strengthened POP and IMAP protocols.
- Be aware of extra risks of web-based access:
  - Key stroke logging and user credential capture.
  - Content over http may bypass content filters.
  - Client e-mails may be left in browser history and temporary files.

E-mail Policy and Training

- Develop and publicise an e-mail policy for users.
  - Rules of use, definitions of abuse of service, clarify ownership of e-mail.
- Ensure users sign-up to policy before use.
- Raise awareness of security issues in organisation through training.
- Enforce the policy!

Lecture Summary

- E-mail is routed across internal LANs and the public Internet.
- E-mail is ***subject*** *to* many threats.
- E-mail also ***enables*** many threats.

- PGP and S/MIME can address part of the problem through encryption and signature mechanisms.
- Addressing the remaining issues requires a careful blend of computer security, network security and security management countermeasures.

Some Resources

- NIST Special Publication 800-45:
  Guidelines on Electronic Mail Security by S. Bisker, M. Tracy and W. Jansen. Available from:
  http://csrc.nist.gov/publications/nistpubs/index.html
- W. Stallings “Network Security Essentials”, Chapter 5: more on PGP and S/MIME.
- http://www.spamlaws.com/: details on anti-spam legislation.
- Open PGP: www.openpgp.org
- PGPv7 on ISG lab machines.
- S/MIME: www.ietf.org/html.charters/smime-charter.html
- All the RFCs are at www.ietf.org.
  - *http://www.apps.ietf.org/rfc/index.html*