Email Security (2)
Secure E-mail Standards and Products

- We 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:
- 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.
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

Session Key K → Recipient’s Public Key

MIME entity → E → EnvelopedData PKCS object

RecipientInfo → EncryptedKey

EncryptedContent Info → EncryptedContent

Base64 encoding → S/MIME body: Base64 encoded PKCS object

S/MIME header
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-Disposion: attachment; filename=smime.p7m

rfvbnj756tbBghyHhHUujhJhjH77n8HHGT9HG4VQpfyF467GI7n8HHGghyHhHUujhJh4VQpfyF467GhIGfHfYGTfvpnjt6jHdF8HHGTrfvhJhjH776tb9HG4VQbnj7567GhIGfHfYT6ghtyHh6
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

MIME entity ➔ Hash ➔ Sign ➔ Sender’s Private Key ➔ SignedData PKCS object

SignedData PKCS object
- SignerInfo
- Signer’s Cert
- Sig and Hash alg
- Sig and Hash

MIME entity ➔ S/MIME body:
- Base64 encoding
- Base64 encoded PKCS object

S/MIME header
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
HUujhJh4VQpfyF467GhIGfHfYGTrfvbnjT6jH7756tbB9H7n8
**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
    ghyHhHUujhJhjH77n8HHGTrfvmnj756tbB9HG4VQpfyF4674VQpfyF4674GhIGfHYT6jH77n8HHGghyHhHUujhJh756tb6
--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”
- 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)
- 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.
- 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 (e.g., RSA); parameters (e.g., 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.

The whole hierarchy is commonly known as a Public Key Infrastructure (PKI).
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=GB, 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).
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 provide any protection against traffic analysis.
- Additional security measures will be needed to build a secure e-mail system.
Anti-virus and Content Filtering

- Supplement mail server (or client desktops) with content filtering software
  - Block e-mails 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!

- Significant load on mail server, may annoy end users (but whose e-mail is it anyway?)
Anti-spamming Protection

- Configure mail server to disallow mail relay feature.
- 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 MTA except on port 25 (SMTP).
- Configure firewall to block all internal traffic to/from MTA except on ports 25, 110 (POP3) and 143 (IMAP)
  - and other ports as needed – eg, SNMP management.
- 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.
  - See 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 (eg, sendmail, M’soft exchange):**
  - Use latest versions of software.
  - Choose appropriate configuration settings (eg, 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.
- Allow only console-based administration, or use SSH for remote administration.
- Take appropriate backups of mail server and user mail.
Client Side E-mail Security

Again, 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 publicize 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 your organisation through training.
Reading Assignment

- [Kaufman] Chapters 21, 22