Email Security

Outline

- Email basics
- What security services are needed for email?
- How?

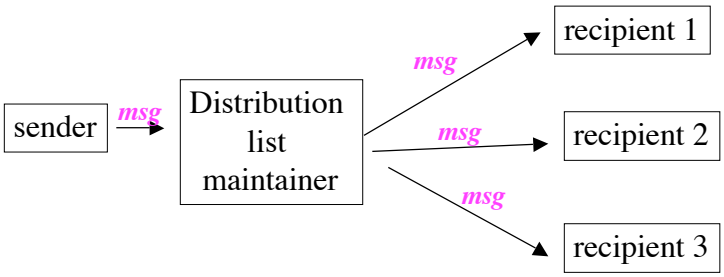
Email Basics

- Distribution Lists
- Mail infrastructure

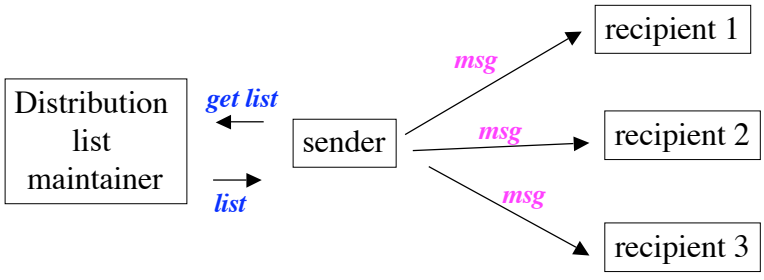
Distribution Lists

- Remote Exploder
- Local Exploder

Remote Exploder



Local Exploder



Advantages w/ Remote Exploder

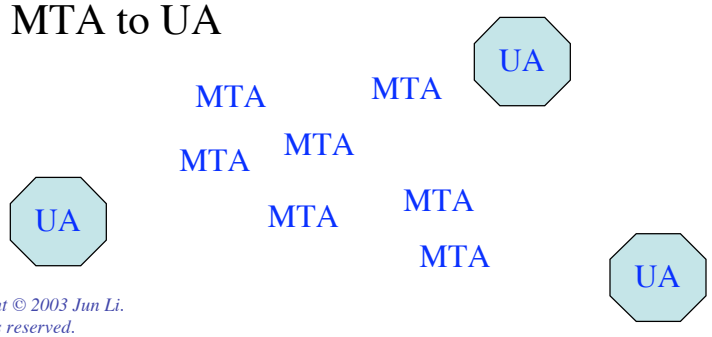
- The mailing list can stay anonymous to the sender
- Maybe good for bandwidth (imagine all members of a mailing list is in Mars)
- Save bandwidth if the mailing list is very long
- Can be in parallel when multiple mailing lists

Advantages w/ Local Exploder

- Easier to prevent mail forwarding loops
- Duplicate copy prevention
- Can estimate bandwidth cost before sending out emails

Email Infrastructure

- MTA: Mail Transfer Agents
- UA: User Agents
- Mail is forwarded from UA to MTA to … to MTA to UA



What Security Services are Needed?

- Privacy
- Authentication
- Integrity
- Non-repudiation
- Proof of submission
- Proof of delivery
- Message flow confidentiality
- Anonymity
- Containment
- Audit
- Accounting
- Self destruct
- Message sequence integrity

Establishing Keys

- Establishing public keys
  - Out of band mechanism
  - PKI
  - Piggybacking certificates on emails
- Establishing secret keys
  - Alice phones Bob . . . (we knows this is bad)
  - Kerberos

Privacy

- Why?
  - Eavesdropper
  - Relay nodes (routers or MTAs)
- End-to-end privacy
- Privacy with distribution list exploders

End-to-end Privacy

- Alice sends Bob an email that is encrypted with Bob's public key
- Well, not ideal, because
  - Multiple recipients
  - Public key crypto is far less efficient than secret key crypto
  - Better not to use long term key unless really the only way to do so

A Public Key Based E2E Privacy Solution

- Alice picks up a secret key and then sends out the following:

  Bob's name; $K_{Bob}\{S\}$

  Carols's name; $K_{carol}\{S\}$

  Ted's name; $K_{Ted}\{S\}$

  $S\{m\}$

Authentication of the Source

- Source authentication based on public key technology
  - Sign the message using the sender's private key
- Source authentication based on secret keys
  - A message must carry a MAC (*message authentication code*)
  - MAC can be:
    - CBC residue of the message computed with the shared secret key
    - Message digest of the shared secret append to the message
    - Encrypted message digest (preferred when multiple recipients)
- Source authentication with distribution lists

Message Integrity

- Source authentication often must come with the message integrity
  - Otherwise, why care the source authentication?
- But how about message integrity w/o source authentication?
  - Can be done if the message is encrypted with the recipient's public key
  - Perhaps needed by a kidnapper

Non-Repudiation

- Non-repudiation based on public key technology
  - Relatively easy
  - Require the message to be signed by the sender using its private key
    - Remember nobody else knows the private key, so . . .
- Non-repudiation with secret keys
  - Relatively difficult
  - The message is signed using a shared secret key
    - But nobody else knows the secret key (what's the difference here from above?)

Plausible Deniability Based on Public Key Technology

- Alice picks a secret key S
- $\{S\}_{\text{Bob}}$ (encrypted with Bob's public key)
- $[\{S\}_{\text{Bob}}]_{\text{Alice}}$ (signed with Alice's private key)
- *MAC* of $m = f(S, m)$
- Alice sends the following to Bob:

  m, *MAC*, $[\{S\}_{\text{Bob}}]_{\text{Alice}}$

- Bob can know that m is from Alice, but he can't prove to anyone else that m is from Alice

Bob can know S

Non-Repudiation w/ Secret Keys

Proof of Submission

Proof of Delivery

Message Flow Confidentiality

Anonymity

Containment

Verifying WHEN a message was really sent

- Preventing Backdating
- Preventing Postdating

Quiz 4

Assume secret key crypto. If Bob wants to verify that an email is indeed from Alice, he will check a piece of data that comes with the message:

(1) What's that piece of data called?

(2) Who calculated this piece of data?

(3) List three different ways to calculate this piece of data.

PEM - Privacy Enhanced Mail

Intro to PEM

- Developed in the late 80's
- For ordinary messages
- Four main RFCs:
  - RFC 1421: message formats
  - RFC 1422: CA hierarchy
  - RFC 1423: crypto algorithms
  - RFC 1424: certificate exchange format

Main Goals of PEM

- Privacy
- Integrity
- Source authentication

- PEM uses the similar methods we talked earlier

PEM Model

- Smart PEM software sitting at the source and the destination
- User keys are used to sign or encrypt
  - One key per message
- User keys are based on either secret key or public key technology

PEM Message Structure

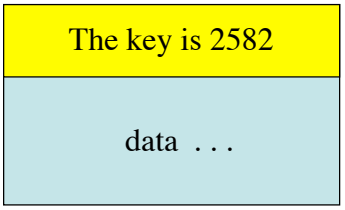
- A PEM message can contain several parts
- And each part treated differently
  - Clear text
  - Integrity protected
  - Or encrypted
- With markers around each block

Types of Message Pieces

- Ordinary, unsecured data
- MIC-CLEAR
  - Clear text + MIC
- MIC-ONLY
  - Encoded text + MIC
- ENCRYPTED
  - Encoded (Encrypted (clear text) + encrypted(MIC))
- Note: MIC here is the PEM's term for MAC

Establishing Keys

- One key per message
  - Randomly chosen by the sender
- The per-message key is established through **interchange key**
  - Which can be either a secret key
    - PEM does not specify how to establish this
  - A public key
    - PEM defines certification hierarchy



Encrypted w/ interchange key

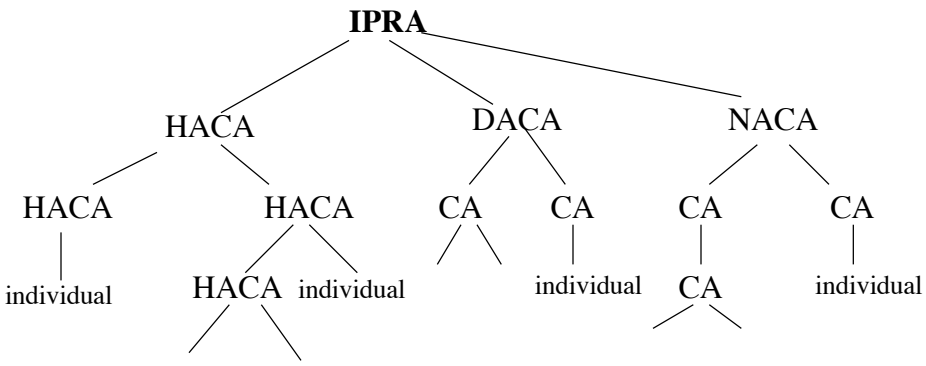
Encrypted with message key (2582)

PEM Certificate Hierarchy

- A hierarchy of CAs in a tree form
  - The root is called **IPRA** (Internet Policy Registration Authority)
  - CAs certified by IPRA are called **PCA** (Policy Certificate Authority)
  - Then other CAs
- Policy: each CA has a policy on issuing certificates
  - Three different policies

CA Types

- High Assurance (HA) CA
  - Super secure
  - Very strict on deciding to issue a certificate to somebody
- Discretionary Assurance (DA) CA
  - Well managed, but no guarantee
- No Assurance (NA) CA
  - No constraints as long as no duplications



Certificate Revocation Lists (CRLs)

- A certificate may expire
  - Or broken
- Must be revoked
- CRL service
- Message types
  - CRL-RETRIEVAL-REQUEST
  - CRL

S/MIME

MIME

- MIME - Multipurpose Internet Mail Extensions (RFC 2045)
  - It specifies how to encode non-text data and type labels
    - Pictures, rich text, video, binary files . . .
  - So it will look like a text message to MTAs
- But remember PEM is only intended to handle ordinary text
- S/MIME
  - RFC 2633
  - Took design principles from PEM for security

S/MIME Certificate Hierarchy

- S/MIME does not try to define a particular PKI
  - Easy to deploy
  - With less security (compared to PEM's)
- But instead assumes a number of parallel independent hierarchies
  - Each user simply trusts a subset of them

(cont'd)

- S/MIME w/ a public certifier
  - Verisign, Thawte
- S/MIME w/ an organization certifier
  - Your employer helps
- S/MIME w/ certificates from any old CA

PGP

PGP Overview

- PGP is not just for mail
  - It can be used for file encryption
  - Then mail the encrypted files to recipients
  - PGP source code can be integrated with common mail systems
- There are many versions of PGP
  - We focus on **PGP Classic**
  - The ideas are the same among different versions

Key Distribution

- PGP uses public key crypto for personal keys
- Certificates are optional in PGP
- People can publish their **PGP fingerprints**
  - Cryptographic hashes of public keys
  - E.g. 29 6F 4B E2 56 FF 36 2F AB 49 DF DF B9 4C BE E1
  - Then send emails containing the public key (and fingerprints)

When PGP Uses Certificates

- Differences from PEM and S/MIME
  - PGP assumes anarchy
    - Anyone can issue a certificate for anyone!
    - Remember PEM assumes a strict hierarchy and S/MIME assumes several hierarchies
  - PGP is different in verifying certificates
    - Need to search for a chain of trust

Chain of Trust

- Carol's public key is P1, signed by Alice
- Alice's public key is P2, signed by Bob
- Carol's public key is P1, signed by Jason

Issues of Chain of Trust

- With a disorganized mass of certificates, how to find a chain of certificates that can lead to Alice's public key?
- What if there are multiple chains, but lead to different keys for the same person?
- If a chain is found, do you trust it?

Private Key

- Needed when
  - Signing your own message
  - Decrypting a message delivered to you that is encrypted using your public key
- PGP can generate a private key for you
  - Then store it in an encrypted form

Midterm

- 7-10 problems
  - 2-3 essay questions
- 80 minutes
  - 2 - 3:20 p.m. Nov 18th
- First couple weeks are covered in course reserve materials
- Use the lecture slides as the guidance
  - Textbook and course reserves as reference
- The level of materials details to remember
  - To the level that slides have
  - But not to the level of textbooks
- Know steps of Kerberos, SSL, . . .