Identities

Overview

- Files and objects
- Users and groups
- Certificates and names
- Hosts and domains
- State and cookies
- Anonymity

Identity

- *Identity*: specifies a principal (a unique entity)
  - Accountability and access control
- *Authentication*: binding of a principal to a representation of identity internal to the system

Files and Objects

Files and Objects

- Identity depends on system containing object
- Different names for one object
  - Human use file name
  - Process use file descriptor or handle
  - Kernel use file allocation table entry, inode

Users and Groups

Users

- Exact representation tied to system
- Example: UNIX systems
  - Login name: used to log in to system
    - Logging usually uses this name
  - User identification number (UID): unique integer assigned to user
    - Kernel uses UID to identify users
    - One UID per login name, but multiple login names may have a common UID

Multiple Identities

- UNIX systems
  - Real UID: user identity at login, but changeable
  - Effective UID: user identity used for access control
  - Saved UID: UID before last change of UID
  - Audit/Login UID: user identity used to track original UID

Groups

- Used to share access privileges
- Set of users
  - Group ownership of objects
  - All have the same access rights to the designated files and directories

Certificates and Names

Naming and Certificates

- Certificates issued to a principal
  - Principal uniquely identified to avoid confusion
- Problem: names may be ambiguous
  - Does the name "John Smith" refer to???

Disambiguating Identity

- Include ancillary information in names
  - Enough to identify principal uniquely
  - X.509v3 Distinguished Names do this
- Example: X.509v3 Distinguished Names
  - /O=University of Alabama
  - /OU=Huntsville
  - /OU=Department of Computer Science
  - /CN=John Smith/

  (CN is *common name*)
  (OU is *organizational unit*)
  (O is *organization*)

CAs and Policies

- CA's *authentication policy* says what type and strength of authentication
- CA's *issuance policy* says to which principals the CA will issue certificates

Internet Certification Hierarchy

- Tree structured arrangement of CAs
  - Root is *Internet Policy Registration Authority*, or IPRA
    - Sets policies all subordinate CAs must follow
    - Certifies subordinate CAs (called *policy certification authorities*, or PCAs), each of which has own authentication, issuance policies
    - Does not issue certificates to individuals or organizations other than subordinate CAs
  - PCAs issue certificates to ordinary CAs
    - Does not issue certificates to individuals or organizations other than subordinate CAs
  - CAs issue certificates to organizations or individuals

Avoid Naming Conflicts

- Assume CAs will prevent name conflicts as follows
- No two distinct CAs have the same Distinguished Name
- No two principals have certificates issued containing the same Distinguished Name by a single CA

Trust

- Goal of certificate: bind correct identity to DN
- Question: what is degree of assurance?
- X.509v3, certificate hierarchy
  - Depends on policy of CA issuing certificate
  - Depends on how well CA follows that policy
  - Depends on how easy the required authentication can be spoofed
- Really, estimate based on the above factors

Hosts and Domains

Identity on the Web

- Host identity
  - Static identifiers: do not change over time
  - Dynamic identifiers: changes as a result of an event or the passing of time

Dynamic Identifiers

- Assigned to principals for a limited time
  - Server maintains pool of identifiers
  - Client contacts server using *local identifier*
  - Example, NAT

Domain Name Server

- Maps transport identifiers (host names) to network identifiers (host addresses)
  - Forward records: host names → IP addresses
  - Reverse records: IP addresses → host names
- Weak authentication
  - Not cryptographically based
  - Various techniques used, such as reverse domain name lookup

Danger!

- Attacker spoofs identity of another host
  - Protocols at, above the identity being spoofed will fail
  - They rely on spoofed, and hence faulty, information
- Example: spoof IP address, mapping between host names and IP addresses

Attacks on DNS

- Associate an incorrect IP address with a host name
  - Attacker controls the name server
  - Intercept the query
- "Cache poisoning"
  - Add extra DNS records to answer a query

State and cookies

Cookies

- Token containing information about state of transaction on network
  - Usual use: refers to state of interaction between web browser, client
  - Idea is to minimize storage requirements of servers, and put information on clients
- Client sends cookies to server

Some Fields in Cookies

- *name*, *value*: name has given value
- *expires*: how long cookie valid
  - Expired cookies discarded, not sent to server
  - If omitted, cookie deleted at end of session
- *domain*: domain for which cookie intended
  - Consists of last *n* fields of domain name of server
  - *Must* have at least one "." in it
- *secure*: send only over secured (SSL, HTTPS) connection

Example

- Caroline puts 2 books in shopping cartcart at books.com
  - Cookie: *name* bought, *value* BK=234&BK=8753, *domain* .books.com
- Caroline looks at other books, but decides to buy only those
  - She goes to the purchase page to order them
- Server requests cookie, gets above
  - From cookie, determines books in shopping cart

Who Can Get the Cookies?

- Web browser can send *any* cookie to a web server
  - Even if the cookie's domain does not match that of the web server
  - Usually controlled by browser settings
- Web server can *only* request cookies for its domain

Where Did the Visitor Go?

- Server books.com sends Caroline 2 cookies
  - First described earlier
  - Second has *name* "id", *value* "books.com", *domain* "adv.com"
- Advertisements at books.com include some from site adv.com
  - When drawing page, Caroline's browser requests content for ads from server "adv.com"
  - Server requests cookies from Caroline's browser
  - By looking at *value*, server can tell Caroline visited "books.com"

Anonymity

Anonymity on the Web

- Recipients can determine origin of incoming packet
  - Sometimes not desirable
- Anonymizer: a site that hides origins of connections
  - Usually a proxy server
    - User connects to anonymizer, tells it destination
    - Anonymizer makes connection, sends traffic in both directions
  - Destination host sees only anonymizer

Example: *anon.penet.fi*

Example: *anon.penet.fi*

- Offered anonymous email service
  - Sender sends letter to it, naming another destination
  - Anonymizer strips headers, forwards message
    - Assigns an ID (say, 1234) to sender, records real sender and ID in database
    - Letter delivered as if from anon1234@anon.penet.fi
  - Recipient replies to that address
    - Anonymizer strips headers, forwards message as indicated by database entry

Problem

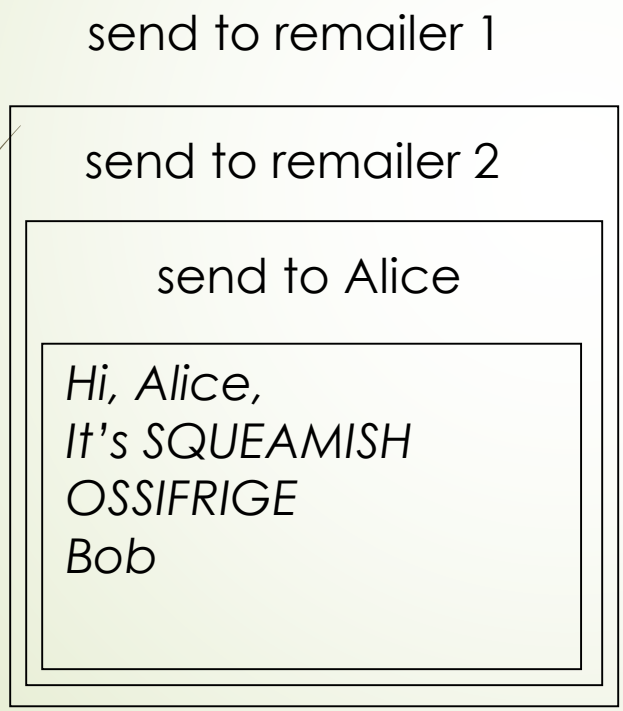
- Anonymizer knows who sender, recipient *really* are
- Called *pseudo-anonymous remailer* or *pseudonymous remailer*
  - Keeps mappings of anonymous identities and associated identities
- If you can get the mappings, you can figure out who sent what

Example: Cypherpunk Remailer

Cypherpunk Remailer

- Remailer that deletes header of incoming message, forwards body to destination
- Also called *Type I Remailer*
- No record kept of association between sender address, remailer's user name
  - Prevents tracing, as happened with *anon.penet.fi*
- Usually used in a chain, to obfuscate trail
  - For privacy, body of message may be enciphered

Cypherpunk Remailer Message



- Encipher message
- Add destination header
- Add header for remailer *n*

  …
- Add header for remailer 2

Weaknesses

- Attacker monitoring entire network
  - Observes in, out flows of remailers
  - Goal is to associate incoming, outgoing messages
- If messages are cleartext, trivial
  - So assume all messages enciphered
- So use traffic analysis!
  - Used to determine information based simply on movement of messages (traffic) around the network

Attacks

- If remailer forwards message before next message arrives, attacker can match them up
  - Hold messages for some period of time, greater than the message interarrival time
  - Randomize order of sending messages, waiting until at least *n* messages are ready to be forwarded
    - Note: attacker can force this by sending *n*–1 messages into queue

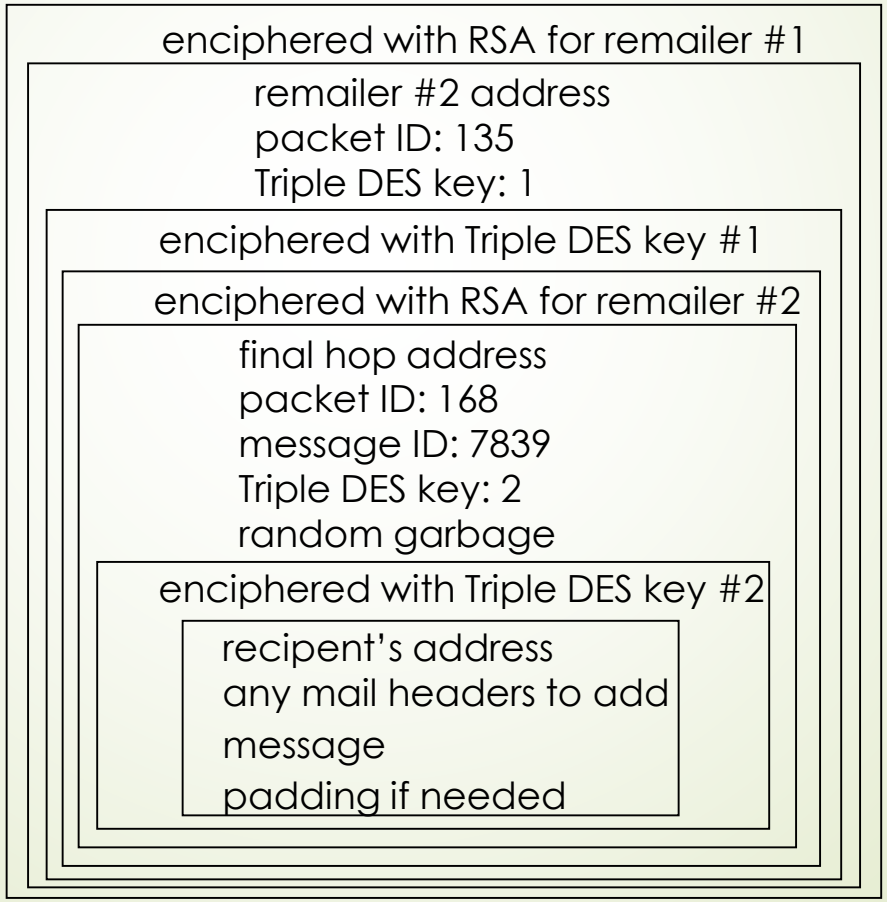
Attacks

- As messages forwarded, headers stripped so message size decreases
  - Pad message with garbage at each step, instructing next remailer to discard it
- Replay message, watch for spikes in outgoing traffic
  - Remailer can't forward same message more than once

Mixmaster Remailer

- Cypherpunk remailer that handles only enciphered mail and pads (or fragments) messages to fixed size before sending them
  - Also called Type II Remailer
  - Designed to hinder attacks on Cypherpunk remailers
    - Messages uniquely numbered
    - Fragments reassembled *only* at last remailer for sending to recipient

Cypherpunk Remailer Message



Anonymity and Privacy

Anonymity Itself

- Some purposes for anonymity
  - Removes personalities from debate
  - With appropriate choice of pseudonym, shapes course of debate by implication
  - Prevents retaliation
- Are these benefits or drawbacks?
  - Depends on society, and who is involved

Privacy

- Anonymity protects privacy by obstructing amalgamation of individual records
- Important, because amalgamation poses 3 risks:
  - Incorrect conclusions from misinterpreted data
  - Harm from erroneous information
  - Not being let alone
- Also hinders monitoring to deter or prevent crime
- Conclusion: anonymity can be used for good or ill
  - Right to remain anonymous entails responsibility to use that right wisely

Key Points

Key Points

- Identity specifies a principal
  - Files and objects
  - Users and groups
  - Certificates and names
  - Hosts and domains
  - State and cookies
- Unique naming a difficult problem
- Anonymity possible; may or may not be desirable