Authentication

- password-based authentication
- address-based authentication
- cryptographic protocols
- passwords as keys
- eavesdropping
- trusted intermediaries
- session key establishment
- delegation

Password

- proof by knowledge, sharing
- eavesdropping
- needed for dumb end systems
- cellular phone cloning
- single password across multiple hosts

Password Guessing

on-line: limit tries, delay, alarm

off-line: dictionary attack ➠ capture $f(p)$

1. Your first, last, or kid’s name
2. ”secret”
3. stress-related words (”deadline”, ”work”)
4. sports teams or terms (”bulls”, ”golfer”)
5. ”payday”
6. ”bonkers”
7. The current season (”winter”, ”spring”)
8. Your ethnic group

9. repeated characters (”aaaaa”, ”bbbbb”)
10. obscenities, sexual terms

Storing Passwords

per-node: /etc/passwd

server: authentication storage server, retrieved by node (yp/NIS)

facilitator: server says yes/no

➠ need to authenticate node asking

- store hash only
- store encrypted with good, protected key
- but: needs to be in non-volatile memory (ROM?)

Address-Based Authentication

- rcp, rsh: `.rhosts` ➠ node, user name
- per user
- reverse-lookup on IP address (in-addr.arpa)
- can use different login names
- `/etc/hosts.equiv`: trusted hosts

Address-Based Authentication: Threats

- break in one, break in all
- often: A trusts B, B trusts A
- address spoofing; not easy for connections, but “blind” sending
- easy to listen/send on broadcast network
- MAC address spoofing prevention: filter on port, scramble

Source routing to have T spoof A: $\langle A, T, D \rangle$ ➠ $\langle D, T, A \rangle$

Humans and Computers

humans: short, memorable key (8 characters, 48 bits)
directly or as key for longer key (PGP, Netscape)

computers: hidden key, directly

Passwords as Keys

- directly as 56-bit key (e.g., use words)
- can't use for RSA p, q:
  - use as seed for rng
  - "simulation-style" rng, until primes found
  - do once, then give offset hints to user

Eavesdropping

- public key: need to secure Alice’s private key
- use random challenge with signing
- difficult to protect against eavesdropping and disclosure ➠ Lamport, S/Key

Trusted Intermediaries

- can’t do pairwise authentication with secret keys: key explosion!
- ➠ Key Distribution Center (KDC)
  - KDC knows all secrets
  - α asks KDC for secret (securely) to talk to any other node β
  - hand out session key $R_{\alpha\beta}$: *ticket*
  - single point of failure
  - bottleneck

Trusted Intermediaries: CA

CA: ensure validity of public keys

- small number, preconfigured
- CA: single PoF
- CA: typically off-line, protected
- certificates are not sensitive
- compromised CA cannot eavesdrop
- need revocation list (CRL) ➠ must be signed and recent

Multiple KDC Domains

Secret keys:

- KDCs share pairwise key
- topology of KDC: tree with shortcuts

Public keys:

- cross-certification of CAs
- example: Alice with CA_A, Boris CA_B
  - Alice gets CA_B's certificate signed by CA_A
  - Alice gets Boris' certificate signed by CA_B

Session Key Establishment

- use public keys to authenticate, generate private key
- trade-off: processing, exposure
- limit lifetime ➠ limit replay attacks
- only need to expose short-term key to semi-trusted software

Authorization

- authentication: *identity* (who)
- authorization: *capability* (what)
- may be implied (physical access)
- network: authentication ➠ access control list (ACL)
- groups: central server, signed certificate
- certificate: unwieldy, CRLs
- hierarchical groups
- typical: hiearchy (DH, director, . . .) and organization

Solaris ACLs

- `setfacl -r -m user:czen:r-- file`
- default entries per directory
- `getfacl`:

```
# file: papers
# owner: hgs
# group: faculty
user::rwx
group::r-x              #effective:r-x
group:irt:r-x           #effective:r-x
mask:r-x
other:---
```

Delegation

- short-term authorization for principals
- sign “letter of authority” (delegation)
- limit time, scope