Feb 25, 2005 -- Lecture 16

22C:169
Computer Security

Douglas W. Jones

Department of Computer Science

Amplification

Access-rights amplification problem

Anita K. Jones, Protection in Programmed Systems, 1973

Problem:

User U has capability C for object O
C gives U no rights to O representation

U passes C to method M of class of O
M must gain access to O representation

Example:

O is an open file
M is the read method of the file system

Solution to the amplification problem

The Unix SETUID bit

Object O is a file with owner P

Application U runs in domain Q

U has limited or no access to O

Application M has owner P, SETUID

U runs M with parameter O

M runs in domain P

M gains owner access to O!

System V and successors break it

Solution to the amplification problem

Enter rights

generalization of trap handler

Continuation object O

Domain D plus execution context C

Enter operation on continuation object O

make D current domain
begin execution of C
pass parameters (capabilities)
pass R, return continuation for caller

Solution to the amplification problem

Cambridge CAP sealed objects

Wilkes and Needham, 1979

Morris, Protection in Programming Languages, CACM, 1973

Capability C for object O is sealed with K

K is a capability not in domain of U

Domain of U contains capability to enter M

Domain of M contains K

U calls M, passing C

M may unseal C using K

M gains owner access to O!

Authentication

Inside one closed system

Security = domain enforcement

When system is open

Authentication problem arises

Central issue

How to bind external users to domains

How does the system know
you are who you say you are?

Password authentication

Traditional, widely used

If users have multiple passwords

Passwords are easily forgotten

Users are tempted to write them down

If users have only one password each

No containment of failures

Minimize use of passwords

Unix `/etc/passwd`, a classic error

When users enter passwords

password is immediately encrypted
trapdoor function used
specific trapdoor function is well known

Plaintext of password erased immediately

File `/etc/passwd` contains one line per user

`name:passwd:uid:gid:class:change:expire:gecos:home:shell`

File is world readable!

Unix /etc/passwd risks

Dictionary attack

Encrypt entire dictionary using trapdoor
Compare result with `/etc/passwd`

Name attack

Take user names from `/etc/passwd`
Convert names to passwords

`jones` becomes `j0ne5`, etc.

encrypted passwords
should not have been exposed.

A Better Model

Each user's authentication information

Belongs in that user's domain

Global user-list

Has authenticate rights to user domains

Authenticator

Enters user domain

Exits on authentication failure

Launches user's application on success

Customize authenticator per user

Alternatives to Passwords

Passphrases

these are just long passwords

Challenge-response models

system outputs a random number n

user replies with pass-function of n

Difficult for humans

Biometrics

are these really constant?

Physical tokens

smartcards, USB keys, etc.

Can be lost or stolen

Diebold AccuVote TS voting machine

Smartcard used to authenticate voter

Voter inserts card in machine
*Machine to card "*password is XXXX*"*
*Card to machine "*OK*"*
*Machine to card "*are you valid?*"*
*Card to machine "*Yes*"*
*Machine to card "*invalidate yourself*"*
*Card to machine "*Done*".*

Card replies all constants!