

Protection

- **Mechanisms to control access**
 - Processes, files, etc.
 - System shared by untrustworthy users
- **Access specification**
- **Access enforcement**



Goals of Protection

- **Prevent intentional violations**
 - Attempt to circumvent restrictions
- **Avoid accidental damage**
 - E.g., mistyped “delete” command
- **Enforce system policies**
 - Mechanism supports multiple policies

Protection Domain (1)

- **OS as collection of objects**
 - Each object has defined operations
- **Process accesses certain objects**
 - Using certain operations
- **“Need to know” principle**
 - Limit access to minimum needed

Protection Domain (2)

- **Defined for each process**
- **Collection of access rights**
 - <object, rights>
- **Domain scope**
 - User, process, procedure, ...
 - Static versus dynamic association

UNIX Protection



- **Domain == user**
 - Each process has user (& group)
 - Determines object access rights
- **Process may change “user”**
 - Effective user ID
 - “setuid” file attribute

UNIX “setuid”



- **File attribute**
- **If set, process assumes ID of file owner**
 - Not actual owner of process
 - E.g., “setuid root”
- **Actual user needs “x” access**

UNIX "setuid"

- **Privileged program**
 - Privileges from file owner
- **Must validate access to objects**
 - Using "real owner" information
- **Trusted programs only!**
 - Potential security weakness

Access Matrix

		Objects			
		F ₁	F ₂	F ₃	Pr ₁
Domains	D ₁	R	-	R	-
	D ₂	-	-	-	P
	D ₃	-	R	X	-
	D ₄	RW	-	RW	-

access(i,j) :: D_i ⇒ O_j

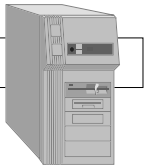
Access Variations 

- **Domain switching**
- **Access right copy/transfer**
 - With/without propagation right
- **Owner/control rights**
 - Change rights to owned objects or controlled domains (discretionary??)

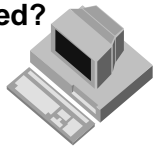
Access Implementation

- **Actual matrix?**
 - Large matrix, usually sparse
- **Global access rights table**
- **Object access lists**
 - One for each object
- **Domain capability lists**
 - One for each domain

Complex Capabilities



- **Example: client & server**
 - Client has data rights
 - Server has database access
- **Trusted, privileged server?**
- **“Two-key” access required?**
 - Combination of
 - Client rights
 - Server rights



Security



- **Protection an internal issue**
- **Security a broader concern**
 - Ensure proper use of system
 - Difficult to accomplish
- **Trade-off**
 - Security vs convenience, etc.

Authentication

- **Identify user with confidence**
- **Mechanisms**
 - Possession: key, access card, etc.
 - Knowledge: password, challenge
 - Attribute: fingerprint, voice
- **Combination & multi-level**



Passwords

- **Easy to guess?**
 - Username, spouse, hobby, etc.
 - Dictionary word
- **Password disclosure**
 - Shared between users
 - Posted near keyboard?

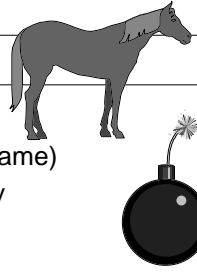


Password Control

- **Required changes**
 - Aging, history file
- **Password screening**
 - Dictionary, length, etc.
 - Word combinations OK?
- **Passwords unique by system?**



Program Threats



•Trojan Horse

- A “gift” program (e.g., game)
- Has hidden functionality

•Logic bomb

•Trap door

- Access bypass in trusted program

System Threats

•Worms

- Stand-alone program
- Propagates across system or network

•Viruses

- Modifies existing system objects
- Propagates to other objects (files?)

Types of Attacks

•Data access

- Destruction
- Interception
- Modification
- Fabrication

•Unauthorized use

•Denial of service

Internet Worm 1988 November 2-11, 1988

- **Robert Tappan Morris, Jr.**
 - Cornell graduate student
- **Targeted systems**
 - VAX 4.3 BSD UNIX systems
 - Sun 3 workstations
- **Injected into net from MIT**

Internet Worm Attack Methods

- **Sendmail debug mode (SunOS)**
- **“finger” daemon (VAX)**
- **Password guessing**
- **Remote execution system**
 - rexec, rsh

Sendmail Debug Mode 

- **Internal “security defeat”**
 - Initially built in for testing?
- **Enabled by default!**
- **Permitted program execution**
 - At request of incoming mail message
 - Transferred “grappling hook” code

Finger Daemon Bug

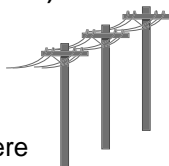
- **Daemon included “gets” call**
- **Attack by overrunning buffer**
 - Replaced actual stack frame
- **Executed Bourne shell**
 - Input from incoming network link
 - Used to transfer worm code

Password Guessing

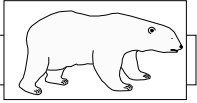
- **Once a host is penetrated**
- **Tried common passwords**
 - None, username, usernameusername, nickname, last name, name reversed
 - Internal “worm” dictionary
 - /usr/dict/words

Remote Execution

- **Trusted host file (other machines)**
 - /etc/hosts.equiv
 - ~/.rhosts
- **Accesses remote host**
 - If local user has account there
 - Transfers code via “rsh”



Camouflage

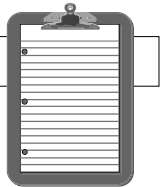


- **Grappling hook forks child**
 - Parent exits, removing evidence
- **Erases text of command args**
- **Deleted executing binary ("sh")**
- **Check for other instances**
 - Errors in these routines

Epilogue

- **Morris convicted**
 - Federal felony
- **Three years probation**
- **400 hours community service**
- **\$10,000 fine**
- **>\$100,000 legal fees**

Security Principles

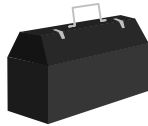


- **Least privilege**
- **Economy of mechanisms**
- **Acceptability**
- **Complete mediation**
- **Open design**

Saltzer, *Proc IEEE*, September 1975

Security Tools

- **Break-in attempt detection**
 - Failed logins
 - Activate “break-in” mode?
- **Intrusion detection**
 - Detect behavior anomalies
- **System audit logging**



Encryption

- **Limit access to data**
- **Transformation**
 - Clear text to cipher text: transmission
 - Reverse on receipt
- **Secret vs public key systems**
 - DES, RSA, PGP, etc.

Firewalls



- **Network security mechanism**
- **Special node**
 - Filters network traffic
 - Passes only authorized messages
- **Requires careful design**
 - Ongoing maintenance and auditing

References

- ⇒ CERT Coordination Center (aka Computer Emergency Response Team), Software Engineering Institute, Carnegie Mellon University. www.cert.org, comp.security.announce.
- ⇒ Stoll, C., Stalking the wily hacker. *Comm. ACM*, May 1988.
- ⇒ Stoll, C. *The Cuckoo's Egg*. Doubleday, 1989.
- ⇒ Eichin, M. and Rochlis, J. *With Microscope and Tweezers: An Analysis of the Internet Virus of November 1988*. <http://web.mit.edu/user/e/eichin/www/virus/intro.html>
