NYIT
NEW YORK INSTITUTE OF TECHNOLOGY

CSCI 620 – Operating Systems Security

Chapter 4

Chapter 4

Security in Ordinary Operating Systems

Unix (including MAC OS X) and Windows

Chapter Overview

- For Unix and Windows, we will cover:
  - History
  - Protection System
  - Authorization
  - Security Analysis
  - Vulnerabilities

Early Unix History

- Developed by Dennis Ritchie and Ken Thompson who had been in the Multics project
- Started as a game-playing OS on a PDP-7
- Ported to a PDP-11/20
- Rewritten in C in the 70's, made it the first machine-independent OS.
- Unix has an application program interface equal to none which allows for high productivity.
- Though it had some Multics ideas, Unix was much simpler. It also was smaller and faster.

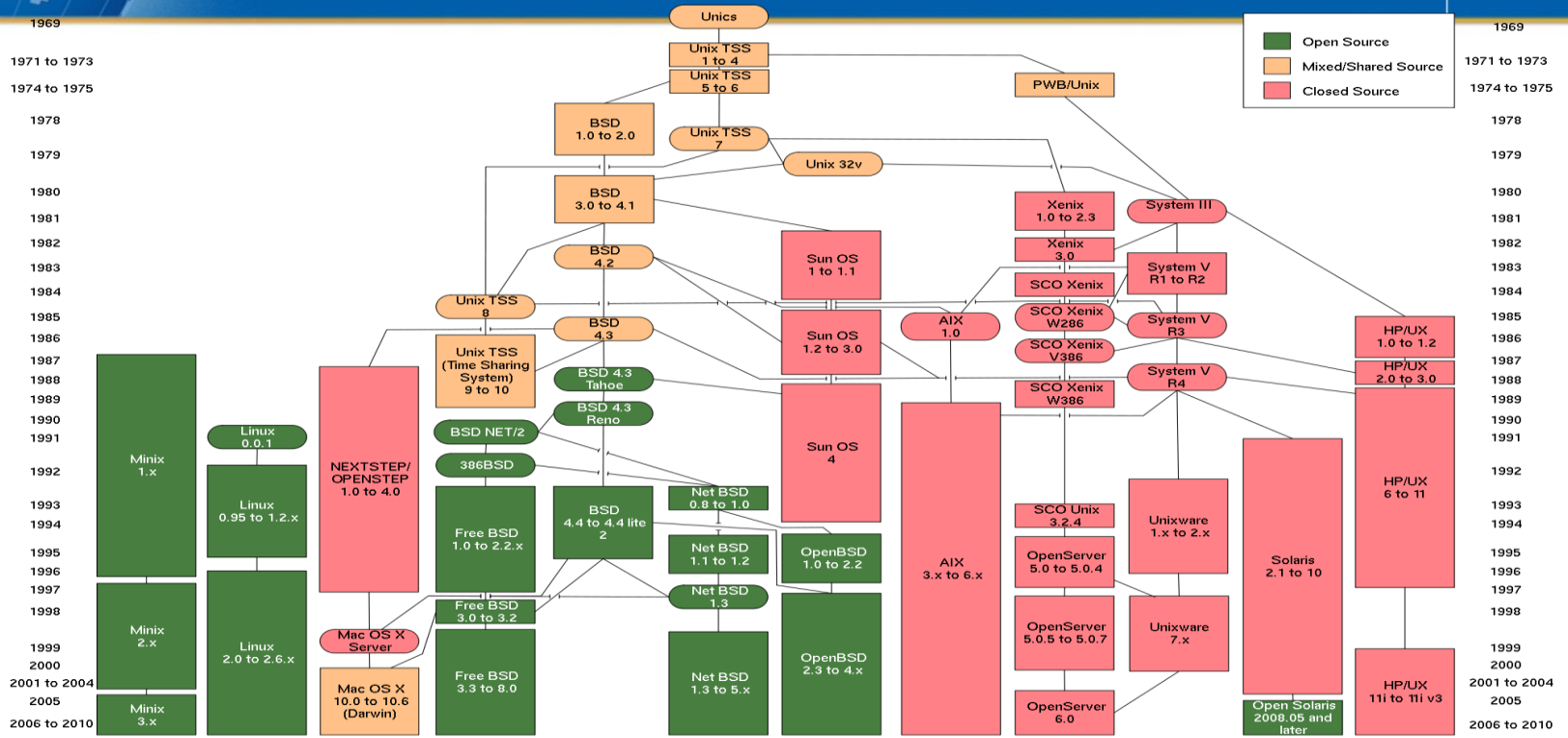
Ideas common to Multics and Unix

- Protection ideas (streamlined)
  - Password Storage
  - Protection ring usage (albeit only two rings)
  - Access control lists
- Other common ideas:
  - Command shell
  - Processes and multiprogramming
  - Hierarchical file structure
  - Standard I/O devices.
  - Segmentation
  - I/O redirection (simpler syntax, however)
  - Virtual memory
  - “Community oriented” OS

New ideas in Unix and other differences with Multics

- Small, basic kernel
- Pipes
- Separation of fork and exec
- B and C
- Concept of a “project”
- Porting an operating system.
- Less dynamic linking.
- Originally, all data stored as plain text
- All devices pretend to be files
- Originally in assembly language.
- Protection emphasis is different.

A Plethora of Unix Versions



A Summary of Unix

- Operating System kernel.
- Many processes, each running a program.
- Protection ring boundary isolates the kernel from the processes.
- Each process has its own address space and an associated identity.
- A *file* is any persistent data object, including devices, network, etc.

Unix Security

- Access to files is limited by the user's identity.
- The aim of Unix security is to protect users from each other and protect the TCB from all users.
- The TCB consists of the kernel and a variety of processes that run with the identity of the system administrator. Of course, the authentication program is part of the TCB.

Introductory Discussion of Unix Protection System

- The Unix protection system is a classical, discretionary system.
- There are states and operations on that protection state.
- There could be talk of a labeling mechanism and label protection system, but it would be very ad-hoc.
- It turns out that the mechanisms in Unix are insufficient to provide the secure OS system requirements.

Unix Protection

- Model is quite classical.
- Identity of user consists of a UID and a collection of GID's.
- Each file has a UID (the owner's) and GID associated with it.
- The protection for each file is specified as rwx for user, group and others.
- Since it is discretionary, the security goals cannot be met.

Some Extras on Unix Protection

- Some variants allow the owner (and always the administrator) to change the owner of a file and/or the group.
- Protection mode transitions are handled with the SETUID bit and setxuid syscalls; the SETUID bit is also discretionary, i.e. dangerous.

Unix Authorization

- Unix authorization occurs when files are opened and the operations allowed on the file are verified on each file access.
- Open creates a *file descriptor*, stored in the kernel; only the index is returned to the process.
- Once opened, less mediating is done; in particular, non-files can be non-mediated, also ioctl and fcntl can do many things. Network communications are not mediated.

Unix authorization mechanism

- login/sshd
- Other processes have to run as root also, therefor part of TCB.
- There are privilege restrictions, nobody and chroot; however, they are restricted to root.

Unix Security Analysis: Complete Mediation

- Are all security-sensitive operations mediated correctly?
  - No
- Does the reference monitor mediate security-sensitive operations on all system resources?
  - No; for example, network communications.
- How do we verify that the reference monitor interface provides complete mediation?
  - It is difficult to tell whether all sensitive operations have been identified. No specific approach has been used to verify complete mediation.

Unix Security Analysis: Tamperproof

- How does the system protect the reference monitor, including the protection system, from modification?
  - Stored in the kernel, but
    - Protection system is discretionary
    - NO specific gates to enter the trusted ring.
    - Many interfaces to modify the kernel:
      - Installing kernel modules
      - /proc or sysfs
      - /dev/kmem
- Is the TCB protected?
  - The Unix TCB consists of all root processes; protecting them against compromise is impossible.

Unix Security Analysis: Verifiable

- What is the basis for the correctness of the system's TCB?
  - Since the TCB is unbounded (we can add another process to the TCB at any time!) verifying the TCB formally is impossible.
- Does the protection system enforce the system's security goals?
  - Impossible because of the lack of complete mediation and lack of tamperproofing. Also, cannot the policy clearly.

Some Unix Vulnerabilities

- Network-facing daemons (sshd, ftpd, telnetd, sendmail,NFS)
- Rootkits
- Environment Variables: LIBPATH, config files.
- Shared resources, e.g. /tmp
- TOCTTOU

Windows History

- CP/M
- QDOS (Tim Patterson)
- MS-DOS
- Windows 1, 2, 3, based on the Lisa, MAC
- Windows 95, 97
- Windows NT
- Windows 2000, XP, VISTA, 7

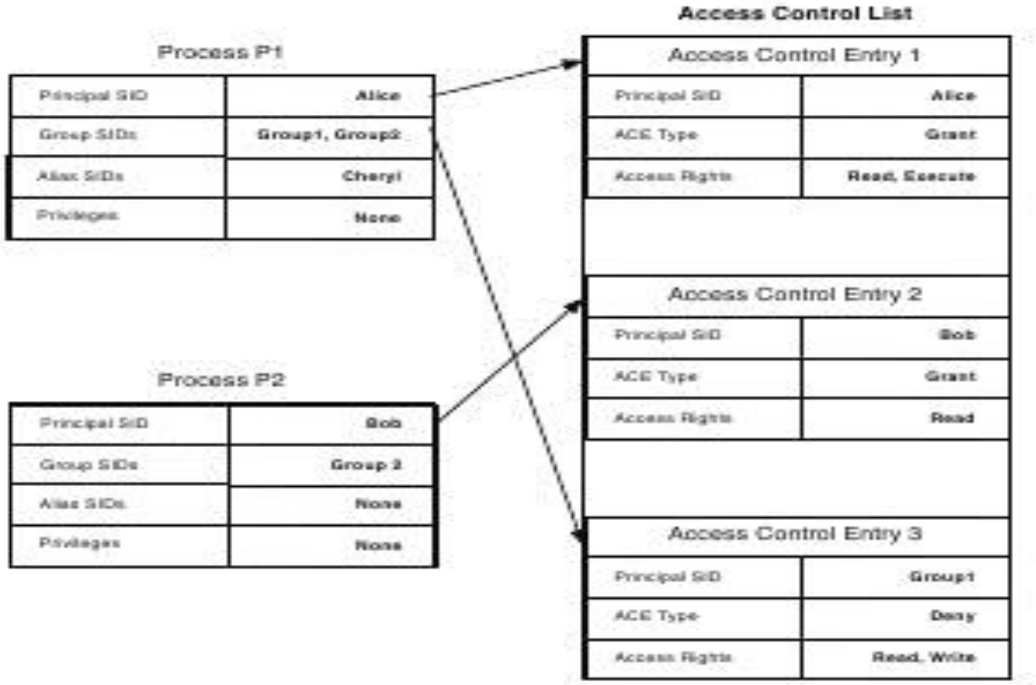
Windows initial “Security” emphasis

- Single user microcomputer platform
- When added to the network, problems arose.
- Ubiquity of windows only made things worse.
- Microsoft was not security oriented until late.
- Modern security enhancements are ineffective and clumsy. (Very invasive).
- The Windows TCB is huge!

Windows Protection System

- Discretionary system.
- More flexible than Unix: extensible, more expressive power, more complicated.
- Subjects in Windows are similar to subjects in Unix:
  - Identity = User SID + set of group SID's + a set of alias SID's + set of privileges
- Objects can be files, or any data types added to the *active directory*; these new objects are defined by the set of operations defined on them (up to 30).
- Operations on files include “modify”, write, read, execute, access file attributes, synchronize file operations.
- Access control lists, with access control entries giving both positive and negative permissions.(grant/deny)

An ACL


Access Control List
Process P1
Principal SID
Alice
Group SIDs
Group1, Group2
Alias SIDs
Cheryl
Privileges
None
Process P2
Principal SID
Bob
Group SIDs
Group 2
Alias SIDs
None
Privileges
None
Access Control Entry 1
Principal SID
Alice
ACE Type
Grant
Access Rights
Read, Execute
Access Control Entry 2
Principal SID
Bob
ACE Type
Grant
Access Rights
Read
Access Control Entry 3
Principal SID
Group1
ACE Type
Deny
Access Rights
Read, Write

Windows Authorization 1

- Handled by a *Security Reference Monitor,* a kernel component, takes a process token, object ID, set of operations and answers yes/no.
- For ACL's, the SRM reads the ACE's in order until it either: satisfies all the requests or finds a denial.
- Mediation is determined by object managers, one for each object type, independent entites.
- Windows has a SETUID-like mechanism. It can be used to invoke *services*.

Windows Authorization 2

- TCB consists of all system services running with administrator privileges.
- Complexities of software installation and accessn control model lead many users to run as administrator, defeating security.
- In Vista, there is some (a little) protection against automatic software installation. It does not work too well. Though it prevents low integrity processes to writing to high integrity files, it does not stop system files from executing low integrity code.

Windows Authorization 3

- Windows does allow for “restricted contexts”.
- Difficult to define correctly.
- Not used very often.
- Not at all by general users.

Windows Security Analysis: 1 Complete Mediation

- How does the reference monitor interface ensure that all security-sensitive operations are mediated correctly?
  - No source code, so difficult to know where mediation is performed.
- Does the reference monitor interface mediate security-sensitive operations on all system resources?
  - Not all operations are mediated; in addition, mediation is done by object managers, which may be extended, without any formalism
- How do we verify that the reference monitor provides complete mediation?
  - No specific approach has been used to verify complete mediation.

Windows Security Slide 2: Tamperproof

- How does the system protect the reference monitor, including its protection system, from modification?
  - Discretionary, so subject to abuse.
  - Users often run as administrator, no protection.
  - Kernel easily modifiable through kernel modules. Signatures are required, but useless.
  - System calls are not checked (like in Unix)
- Does the protection system protect the trusted computing base programs?
  - Any program can be part of the TCB, corrupting it.
  - Also Windows has special APIs to tamper with processes: CreateRemoteThread, WriteProcessMemory, AdjustTokenPrivileges..

Windows Security Analysis: Verifiable

- What is the basis for the correctness of the system's TCB?
  - TCB is very big and unbounded; only informal basis.
- Does the protection system enforce the system's security goals?
  - Very complicated system; Windows policy not specified, extensible; verification very difficult, if at all.

Some Windows Vulnerabilities

- The Windows Registry
- Administrator Users
- Enabled by default

Summary

- Unix and Windows protection is lacking because the security assumptions changed.
- Access controls should be mandatory, comprehensive and verifiable.
- The security enforcement mechanism should work even in the presence of malicious software and be tamper-proof. This means the TCB cannot be extensible.

- Questions?