CT 320 9/2/14 Notes

Sean Wilson
Travis McDonald
Jacob Rede

Lecture 3:

Topics:

1) Access control principles
>Authorization, access control, and authentication
>Discretionary vs. mandatory vs. role-based access control

2) Ownership of files and processes
3) Superuser
4) Password
5) Becoming root
6) Pseudo-users

Authorization, Access Control & Authentication
-These work in conjunction with each other to keep the system safe-Authentication
>Who should be allowed to access protected resources and how to provide the protections for the various resources
>Passwords are the biggest means for authenticating users-Authorization

(Access Control Models)
>Who is actually trying to access they protected resources (Your identity on the system)
>Which objects are accessible by certain users

>Biggest thing, ensure you have the best policies and that they are enforced-Enforcement (Access Control Architectures)
>How the system enforces the authorization rules and design

Modern Access Control
-Discretionary Access Control>Traditional UNIX file system model (we use this in class)

-Mandatory Access control
>Military and SELinux model
>Model for specifying policies
>System level enforcement

~Can overturn owners set permissions if system determines the latter are risky

~System > owner
>SELinux
>System enforces certain types of permissions

-Role-based access control
>Latest industry standard
>Solaris & OpenSolaris, HP-UX, AIX
>Not yet popular at the OS level

Discretionary Access Control

-Owner of an object decides which other users are allowed to access the object>User's identity, objects's identity and permissions comprise the DAC policy>User's can be placed into groups and permission assigned to the groups

Mandatory Access Control-When a system mechanism controls access to an object and an individual user cannotalter that access, the control is mandatory access control

Why Mandatory Access Control?
-DIAGRAMS!!

~~~~~~~Diagram on Slide 8~~~~~~~
-Two files, A and Bisnt
-File A is created by user:Bob

>Sets access that Bob can rw (read and write) but Jill has no permissions-File B is created by user:Jill>Sets access that Jill can rw but Bob can only w (write)
-A new program from Dick is a fancy program with some malware, may be a Trojan Horse

>Because the Trojan Horse is embedded in the program it will have the same permissions as the original program which can allow it to read and write to various files by using its user's permissions
~~~~~~~Diagram on Slide 9 ~~~~~~~

-Two files, A and B
-File A is created by user:Bob
>Sets access that Bob can rw but Jill has no permissions

-File B is created by user:Jill

>Sets the access that Jill can rw and Bob can only w (write) the file

-The same new program from Dick is still fancy with some malware

>Using a high vs. low priority and permissions allow a user to categorize system rules across all users that prevents high and low users to break the given rules so that the Fancy Program's Trojan Horse can no longer both read and write but instead read only as dictated by the system

Traditional Unix access Control

-Non access control lists (ACL) based

>Rudimentary DAC policies but messy implementation

>Existence of root or super-user

>No single process/mechanism responsible for access control

~Certain systems calls restricted to root~Other systems calls involve both ownership evaluation and special
Provisions for root
>File system has separate access control based on access control bits