Topic 12
Lecture 12
Disk Fundamentals

CSCI 150

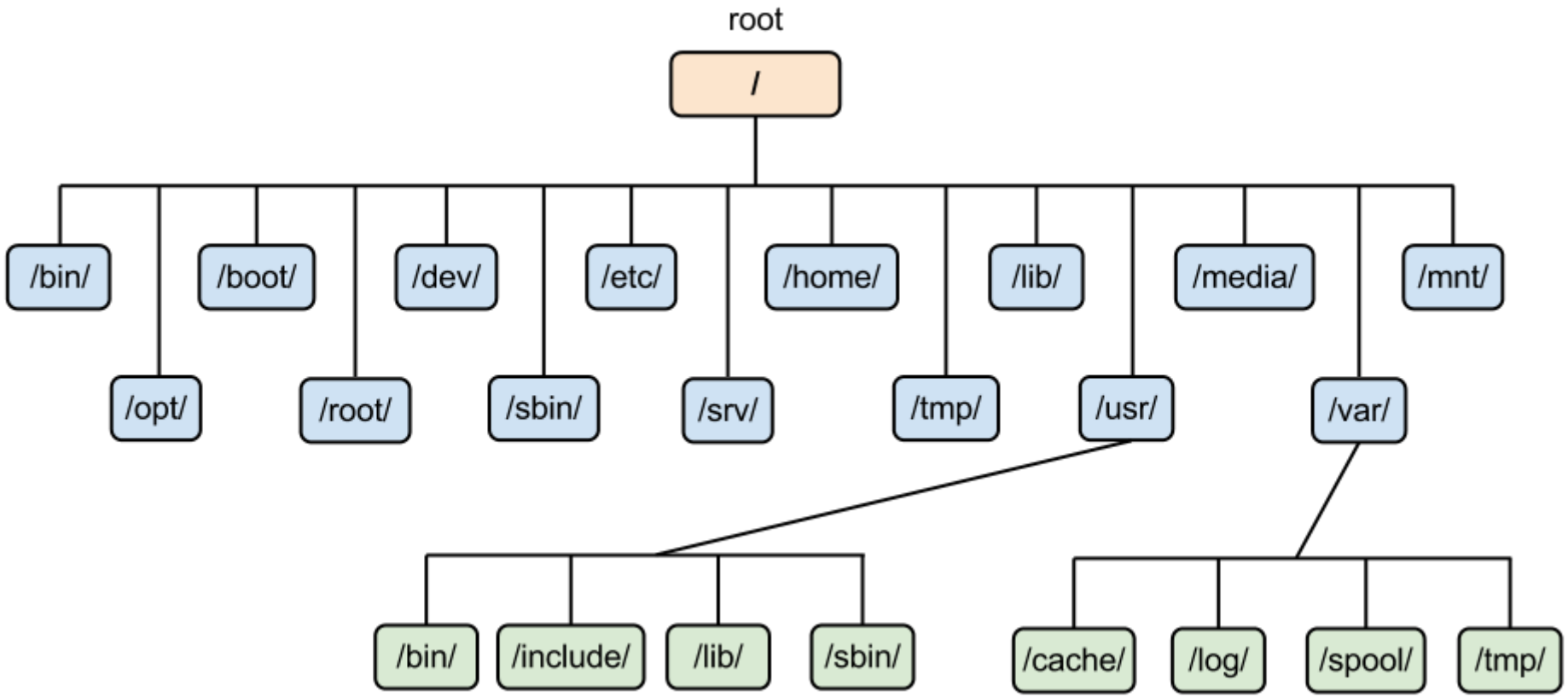
Assembly Language / Machine Architecture

Prof. Dominick Atanasio

The File System

- All resources on a Linux/Unix operating system are available through the file system
  - All devices
  - CPU
  - Running processes
  - etc
- The file system is hierarchical
- The filesystem is best represented as a tree structure (an acyclic graph starting at a single vertex)
- Directories are special files that act as containers for other directories and files
- The root directory (/) is the directory to which all other filesystem objects are mounted/linked

File System


root
/
/bin/
/boot/
/dev/
/etc/
/home/
/lib/
/media/
/mnt/
/opt/
/root/
/sbin/
/srv/
/tmp/
/usr/
/var/
/bin/
/include/
/lib/
/sbin/
/cache/
/log/
/spool/
/tmp/

File System Interaction

- We interact with the file system through system calls to operating system
- We have been doing this all along when printing to the terminal or taking input from the keyboard
- We follow the same process for reading and writing files

File Permissions (file modes)

- Linux/Unix file permissions are organized into three categories (user types):
  - Owner: The user that owns the file. This is the user that most likely created the file.
    - Ownership can be reassigned at any time using the command *chown*.
  - Group Owner: users can be organized into groups. This groups of users can be given privileges to a filesystem object.
    - The group is usually assigned based on the primary group of the owner or the group of the parent directory
    - Group ownership can be reassigned using the *chown* or *chgrp*.
  - Others: Privileges can be assigned to all other users who do not fall into the first two categories
- There are three possible permissions that can be assigned to each of these categories
  - Read: permission to read a file or list the contents of a directory
  - Write: permission to change a file or alter the contents of a directory (create delete files in the directory.
  - Execute: permission to execute a file or change into a directory

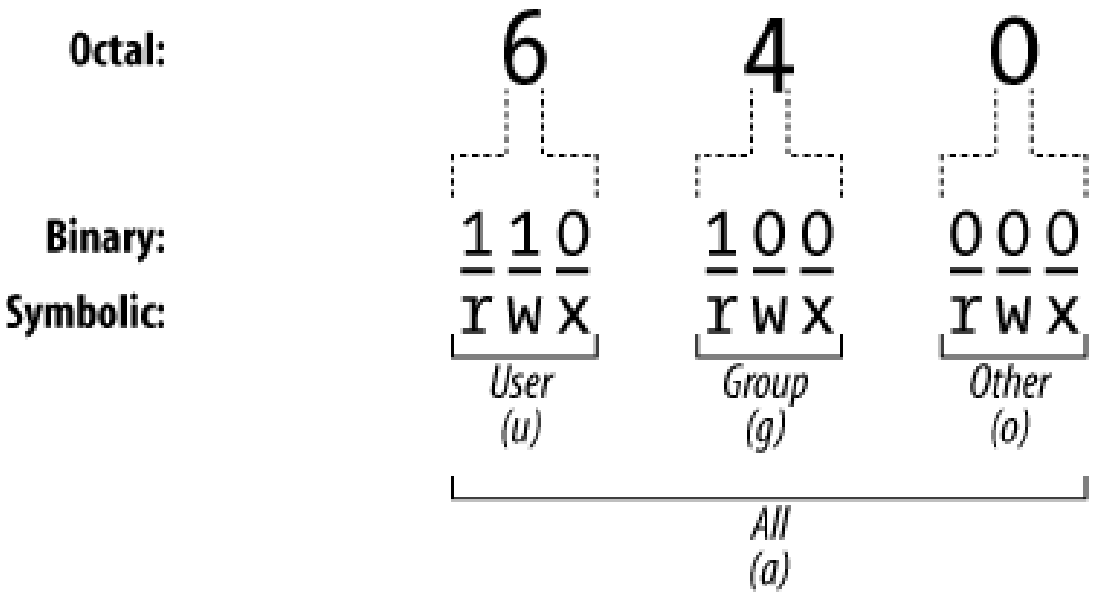
Changing a File or Directory's Mode

- *chmod* is the utility used to change the mode (permissions) of a file
- The three permission categories are referred to as U, G, O for user, group, and other respectively.
- The permissions are referred to as R, W, X for read, write, and execute, respectively.
- There are three operators that can be used with *chmod* to assign or remove privilege to a filesystem object.
  - The '+' operator adds one or more specified privileges to one or more specified categories
  - The '-' operator removes one or more specified privileges from one or more specified categories
  - The '=' operator explicitly sets privilege
- Example (assume there is a file called, "test" in the current working directory:

```
chmod ug+wx    # assigned write and execute for user and group owner
chmod o=r      # sets read permission to Other and removes all others
chmod g-x      # removes execute permission from the group owner
```

Changing a File or Directory's Mode

- *chmod* can also explicitly set permissions sing three octal values
- We use this technique for assigning permissions to a newly created file when performing file I/O in assembly language



System Call Codes for File I/O

- Recall that we print a string to *stdout* using the following instructions

```
mov     eax, 4 ; set code for write
mov     ebx, 1 ; set output stream descriptor (stdout)
mov     ecx, string ; address of the string (buffer)
mov     edx, str_sz ; the size of the string (buffer)
```

- To take input from *stdin* using the following instructions:

```
mov     eax, 3 ; set code for read
mov     ebx, 0 ; set input stream descriptor (stdin)
mov     ecx, string ; address of buffer to read into
mov     edx, str_sz ; the size of the buffer
```

System Call Codes

eax	Name	ebx	ecx	edx	
3	sys_read	unsigned int	byte *	size	returns the qty of bytes read on eax
4	sys_write	unsigned int	byte *	size	
5	sys_open	const char *	access mode	perm	returns file descriptor on eax
6	sys_close	unsigned int	-	-	
8	sys_creat	const char *	perm	-	returns file descriptor on eax

Access Modes:

```
0      read only
1      write only
2      read and write
```

In-class Exercise: File Copy Utility

- We will now write a file copy utility.
- This utility will copy any type and size of file.
- We will use a buffer of 4096 bytes (4KB)