CMPS 261 Server Management - Module 2: Getting Started with FreeBSD Server

Mark Voortman, Ph.D.

Course Modules

- *Module 0: Course Design*
- Module 1: Introduction to Servers and Server Operating Systems
- **Module 2: Getting Started with FreeBSD Server**
- Module 3: Software Maintenance
- Module 4: Tuning and Configuration
- Module 5: Storage Management
- Module 6: Networking
- Module 7: Shell Scripting
- Module 8: Building a WordPress Server

Module 2

- **Getting Started with FreeBSD Server**
- Part A
  - Shells
  - Shell commands
  - Piping and redirection
  - Processes, daemons and signals
  - Manual pages
  - Text editors
- Part B
  - The directory structure
  - Working with disks
  - Mounting and unmounting file systems
  - User accounts
  - File permissions

Objectives

- Upon successful completion of this module, you should be able to:
  - Start up and shut down the server
  - Manage users and accounts
  - Understand the directory structure
  - Understand and manage permissions
  - Understand disk organization
  - Mount and unmount file systems
  - Understand shells
  - Use the default text editor
  - Find and display manual pages

Readings

- FreeBSD basics
- FreeBSD reference

Install Xfce 4

- Xfce is a lightweight and open source desktop environment
- Install xorg: `pkg install xorg`
- Install xfce: `pkg install xfce`
- Run xfce: `startxfce4`

PART A

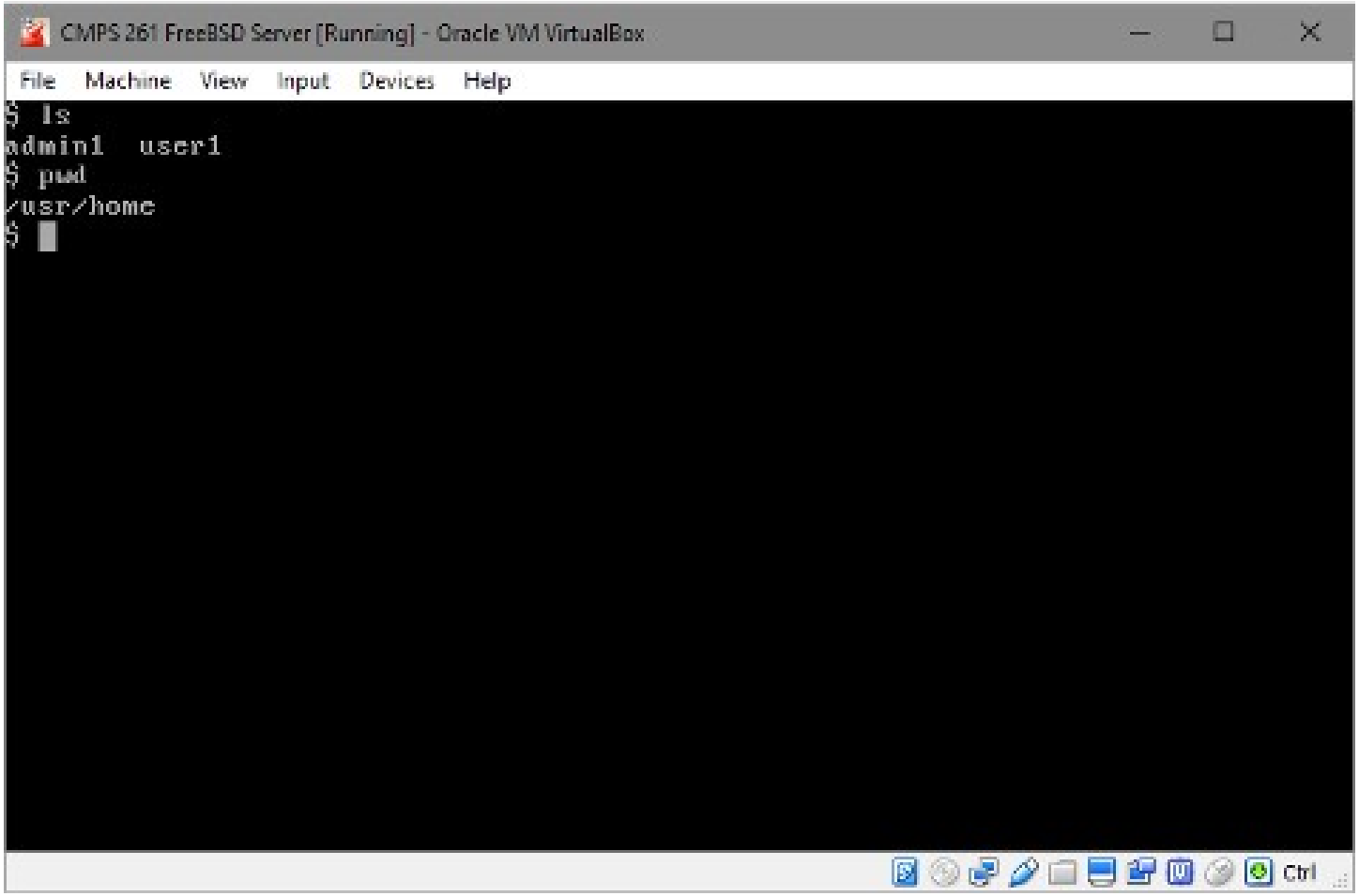
Overview

- Shells
- Shell commands
- Piping and redirection
- Processes, daemons and signals
- Manual pages
- Text editors

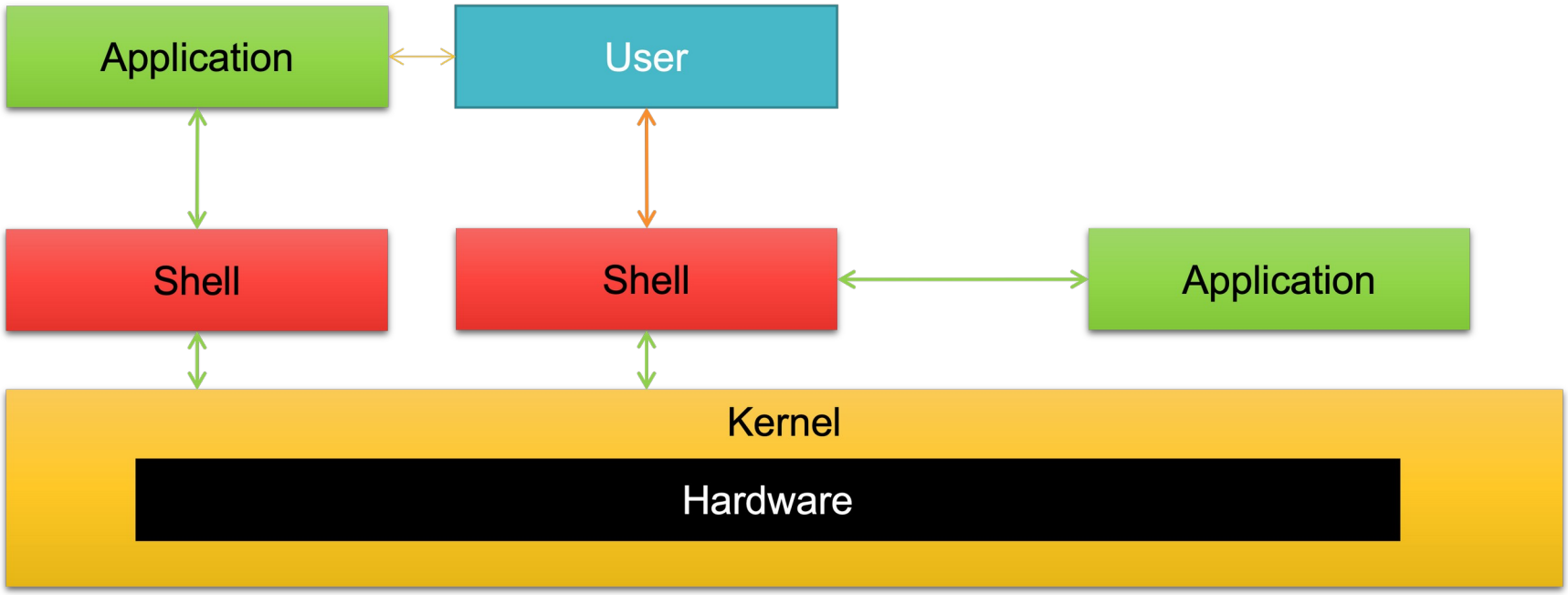
Shells

- The shell is simply the command line interface in use
- The shell accepts commands, parses them, and executes them
- The FreeBSD shell is the Bourne shell (sh)
- Shell loads upon start-up

Shell Example



Shell Architecture



The shell has a strategic role

Shell Advantages

- Broad set of commands available
- Commands can be strung together to form complex operations
- Commands can be placed into scripts
- Fast in the hands of the power user

Shell Disadvantages

- All shells have steep learning curves
- Commands and their options can be cryptic
  - To copy, you use cp
  - To create a directory, you use mkdir
- Most users today are GUI oriented
- Burden is on user to learn and memorize
  - As opposed to GUI mode of discovery

Shell Concepts

- See handbook:
  - https://docs.freebsd.org/en/books/handbook/basics/#shells
- Environment variables
- Changing the shell
  - Implications
- Command piping and redirection
  - Examples

Common Shell Commands I

- clear: Clear screen contents
- ALT F1 to F5
  - Switch terminal session
- Command history
  - History difference in shells
- Type
  - Get information about a command

Common Shell Commands II

- Information commands
  - `w`: Display logged in accounts
  - `who`: Who is logged in to this session
  - `whoami`: User name only
  - `logname`: Logged in user name
  - `hostname`: Name of computer
  - `pwd`: Print working directory (shows where you are)
  - `df`: Display disk space status

Common Shell Commands III

- Display content commands
  - `cat`: Display content of a file
  - `echo`: Echo back command line parameter

Common Shell Commands IV

- Navigation commands
  - `cd`: Change directory
    - cd by itself goes home
    - Home also known as ~
  - `ls`: List directory contents

Common Shell Commands V

- File manipulation commands
  - `cp`: Copy file
  - `rm`: Remove file
  - `mv`: Move file (and rename)

Common Shell Commands VI

- Directory commands
  - `mkdir`: Create a directory
  - `rmdir`: Remove a directory

Piping and Redirection I

- Pipe operator: |
  - Example:
    - `ls –R | less`
  - less and more are commands that buffer content.
  - Multiple pipes allowed:
    - `ls –R | grep var | less`
  - grep searches for text.

Piping and Redirection II

- Redirection
  - > Send to
  - >> Send to append
  - < Input from
- Examples
  - `ls > directories.txt`
  - `cat directories.txt`
  - `ls > /dev/null`
  - `less < directories.txt`

Processes, daemons and signals

- FreeBSD documentation
- ps or ps auxww - list all processes
- top - interactively list all processes
- `kill XXX` - kill process XXX by sending a signal (`SIGTERM`)
- `kill -9 XXX` - force kill process XXX by sending a signal (`SIGKILL`)

Additional Commands

- See command references on Canvas
- https://en.wikipedia.org/wiki/List_of_POSIX_commands

Getting Help

- `man` command will display manual pages
- `apropos` command conducts searches
  - Equivalent to `man -k`
- `whatis` command synonym for `apropos`

Text Editors

- `vi` is default editor
- `ee` is generally considered much easier to use for most people

Assignment

- See Canvas

PART B

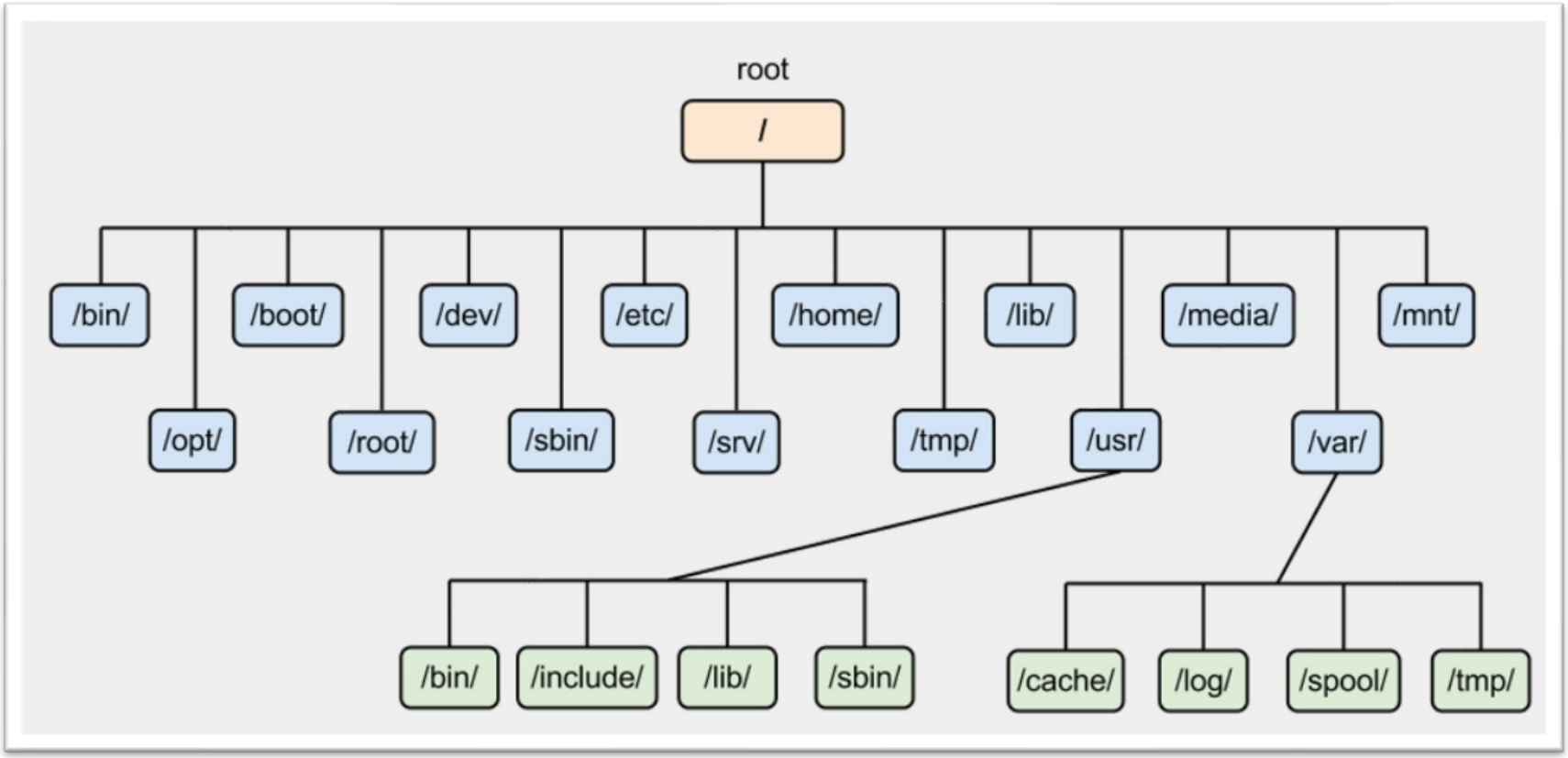
Overview

- The directory structure
- Working with disks
- Mounting and unmounting file systems
- User accounts
- File permissions

The FreeBSD Directory Structure

- Every computer system needs to store files
- UNIX systems have a well-known, standardized structure
- FreeBSD documentation

Overall Schematic



/

- It all starts at the root
- This is the base directory
  - Contents are mainly other directories
  - A couple of visible files
  - Several hidden files
  - A couple of symbolic links

/bin/

- User utilities fundamental to both single-user and multi-user environments
- All of the built-in commands that we enter at the shell can be found here

/boot/

- Programs and configuration files used during operating system bootstrap
- The subdirectory /defaults/ contains default boot configuration files

/dev/

- This directory contains device nodes
- Device nodes can be found for all devices on your system, such as your hard drive
- Note that FreeBSD does not use the same nomenclature for drives as Windows does

/etc/

- System configuration files and scripts
- Several subdirectories, including
  - /defaults/ Default system configuration files
  - /mail/ Configuration files for mail transport agents such as sendmail
  - /periodic/ Scripts that run daily, weekly, and monthly, via cron

/home/

- This is a symbolic link to home directories

/mnt/

- Empty directory used by system administrators as a temporary mount point

/proc/

- Process virtual file system
- A runtime system information location
  - System memory, devices mounted, hardware configuration
- A control and information center for the kernel
- Any files seen here likely have a size of zero bytes

/root/

- Home directory for the root account

/tmp/

- Temporary files which not preserved across a system reboot
- A memory-based file system is often mounted at /tmp/

/usr/

- User utilities and applications
- Subdirectories include
  - /bin/ Common utilities, programming tools, and applications
  - /include/ Standard C include files
  - /lib/ Archive libraries
  - /libdata/ Miscellaneous utility data files
  - /ports/ The FreeBSD Ports collection
  - /src/ BSD and/or local source files

/var/

- Multi-purpose log, temporary, transient, and spool files
- Subdirectories include
  - /log/ Miscellaneous system log files
  - /mail/ User mailbox files
  - /spool/ Miscellaneous printer and mail system spooling directories
  - /tmp/ Temporary files which are usually preserved across a system reboot

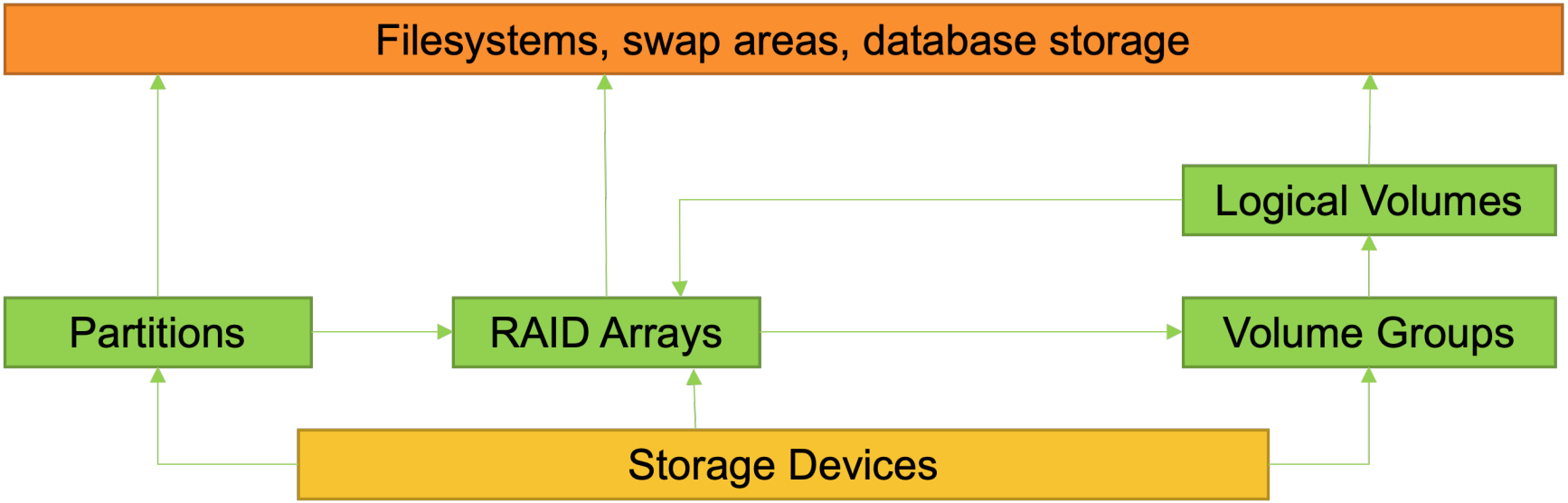
Storage Management

- All computers require access to a file system
- We have just examined the directory structure
- Now let’s look at disks
- We will come back to storage management later in the course
- FreeBSD documentation

Disk Organization

- At the base is the physical disk or device
- On top of this is the logical file system structure
- The first structural definition is the partition (or slice in FreeBSD)
  - A physical disk may be divided into multiple partitions
- The partition must then be formatted
  - Several options for file structure are available
- We encountered these concepts when we created our server
- FreeBSD documentation

Storage Management Layers



Server storage can be a complex configuration

Looking at Disk Free Space

- `df -h` Present output in human readable format

Looking at Disk Usage

- `du -sh` Present output for every entry, human readable
- `du -a /var | sort -nr | head -n 10` Display out for each entry in /var, pipe it to sort, sort numeric in reverse order, pipe to head, display first 10.
- `du -hc * | grep “[0-9]M” | tail | sort -nr` Display human readable, grand total, pipe to grep, search for all lines with a number followed by “M”, pipe to tail (which will display the bottom lines, pipe to sort, sort in numeric reverse order.

Finding Files

- Find is a complex statement with many options. This command will search for files starting at the root that are larger than 1,024 bytes and display the top 25.
- `find / -type f -size +1024k -exec ls -al {} \; | head -25`
  - `-type f` – this means include only regular files
  - `-size +1024k` – this means include only files larger than 1,024K
  - `-exec ls -al {} \;` – execute the ls command with the -al options.
    - The `{}` is a placeholder for the files found by the file command.
    - Think of it as each output from find is passed to the ls command.

Looking at Partition Information

- The gpart command
- gpart status
  - Provides brief view of partition status
- gpart list
  - More details for all partitions

fstab Contents

- /etc/fstab configuration file controls automatic mounting at boot time
- Contents for our systems show the two partitions we have
- FreeBSD documentation

Mounting and Unmounting File Systems

- Mounting attaches a file system (device) to a directory location
  - Directory must exist and be empty
- Mounting only permitted after super user elevation
- FreeBSD documentation

Mount Command

- mount *device* mountpoint
  - Device is the device name, such as cd0
  - Mountpoint is the directory location

Unmount Command

- unmount *mountpoint*
  - Disconnects the directory from the device

fstab File

- Within the /etc directory, the fstab file contains a line for each mount
- This file controls what get automatically mounted when system boots
- During OS install, attached drives were detected and set up
- In the future, if you add a new drive, you will need edit this file

fstab Contents

```
# / was on /dev/sda1 during installation
UUID=93d62c8f-9cfe-4caf-8011-140ca3a64489 /               ext4    errors=remount-ro 0       1
# swap was on /dev/sda5 during installation
UUID=8997f5aa-6515-44bb-bb24-889bc1972da7 none            swap    sw              0       0
# 2 GB second hard drive
/dev/sdb1 /media/wdc_2tb ext4 defaults 0 0
# Buffalo Linksys NAS device
//192.168.1.200/driveimage /media/lssysfam_driveimage cifs guest,rw,uid=1000,nounix,iocharset=utf8,file_mode=0777,dir_mode=0777 0 0
# LaCie NAS device
//192.168.1.201/OpenShare/LinuxBackup /media/lacie_linux cifs guest,rw,uid=1000,nounix,iocharset=utf8,file_mode=0777,dir_mode=0777 0 0
```

Here is an example of a more complex fstab file, taken from a Linux system with two hard drives and two network connections to NAS boxes

Class Activity: Adding a Hard Drive

- We will use the facilities of VirtualBox to add a hard drive to our computers
- FreeBSD documentation

Miscellaneous Commands I

- cal
  - Display calendar
  - cal 03 2019
- stat filename
  - Display file status
- wc filename
  - Get word, line, character, byte count
- file filename
  - Get file type

Miscellaneous Commands II

- cmp file1 file2
  - Compare two files
- comm file1 file2
  - Compare two files, three column output
- diff file1 file2
  - Get file differences
- md5 filename
  - Generate MD5 file hash
- sha256 filename
  - Generate SHA256 file hash

Accounts I

- Access to a system is controlled by accounts
- Two types of accounts:
  - System.
    - Used to run services
  - User.
    - Used by humans to log in
- List all accounts in system:
  - cat /etc/passwd | more
- FreeBSD documentation

Accounts II

- Fields in passwd file:
  - admin1:*:1001:1001:CMPS 261 Admin:/home/admin1:/bin/sh
  - user1:*:1002:1002:CMPS 261 User:/home/user1:/bin/sh
  - user2:*:1003:1003:user2:/home/user2:/bin/sh
- username:password:user-id:group-id:user-id-info:homedir:command-shell

Passwords

- Account passwords stored in a file
  - Passwords are hashed
  - Readable only by root
  - /etc/master.passwd

The Super User

- The super user, or root, account, is all powerful
- Direct usage of root is discouraged
- su command will provide root access when needed
  - Can only be executed by users in wheel group

Command Summary

- adduser: Command-line application for adding new users
- rmuser: Command-line application for removing users
- chpass: Change user database information
- passwd: Command-line tool to change user passwords
- pw: General tool for modifying all aspects of user accounts
- id: Find out information about an id

Add User Command

```
# adduser
Username: jru
Full name: J. Random User
Uid (Leave empty for default):
Login group [jru]:
Login group is jru. Invite jru into other groups? []: wheel
Login class [default]:
Shell (sh csh tcsh zsh nologin) [sh]: zsh
Home directory [/home/jru]:
Home directory permissions (Leave empty for default):
Use password-based authentication? [yes]:
Use an empty password? (yes/no) [no]:
Use a random password? (yes/no) [no]:
Enter password:
Enter password again:
Lock out the account after creation? [no]:
Username : jru
Password : ****
Full Name : J. Random User
Uid : 1001
Class :
Groups : jru wheel
Home : /home/jru
Shell : /usr/local/bin/zsh
Locked : no
OK? (yes/no): yes
adduser: INFO: Successfully added (jru) to the user database. Add another user? (yes/no): no
Goodbye!
#
```

Remove User Command

```
# rmuser jru
Matching password entry:
jru:*:1001:1001::0:0:J. Random User:/home/jru:/usr/local/bin/zsh
Is this the entry you wish to remove? y
Remove user's home directory (/home/jru)? y
Removing user (jru): mailspool home passwd.
#
```

Change User Information Command I

```
--- As regular user:
#chpass jru
Shell: /usr/local/bin/zsh
Full Name: J. Random User
Office Location:
Office Phone:
Home Phone:
Other information:
```

Change User Information Command II

```
--- As super user.
#chpass jru
Login: jru
Password: *
Uid [#]: 1001
Gid [# or name]: 1001
Change [month day year]:
Expire [month day year]:
Class:
Home directory: /home/jru
Shell: /usr/local/bin/zsh
Full Name: J. Random User
Office Location:
Office Phone:
Home Phone:
Other information:
```

Changing Passwords

```
$ passwd
Changing local password for jru.
Old password:
New password:
Retype new password:
passwd: updating the database...
passwd: done
```

Group Membership Changes

```
$ pw groupadd teamtwo
$ pw groupshow teamtwo
teamtwo:*:1100:

$ pw groupmod teamtwo -M jru  (Replaces group membership)
$ pw groupshow teamtwo
teamtwo:*:1100:jru

$ pw groupmod teamtwo -m db  (Appends to group membership)
$ pw groupshow teamtwo
teamtwo:*:1100:jru,db
```

User Information

```
$ id jru
uid=1001(jru) gid=1001(jru) groups=1001(jru), 1100(teamtwo)
```

Command Summary

- adduser: Command-line application for adding new users
- rmuser: Command-line application for removing users
- chpass: Change user database information
- passwd Command-line tool to change user passwords
- pw: General tool for modifying all aspects of user accounts
- id: Find out information about an id

FreeBSD File Permissions I

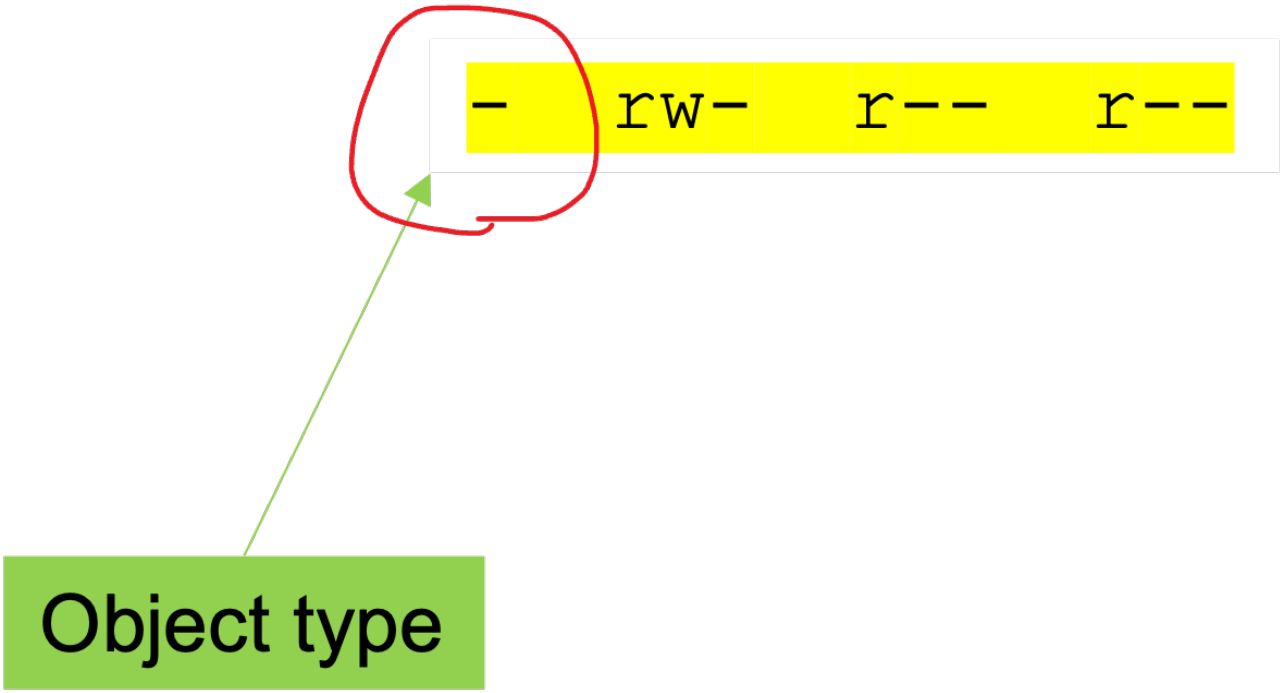
- Files and directory access and usage controlled by file permissions
- Permissions set for the object owner, the group the owner belongs to, and everyone else

FreeBSD File Permissions II

```
$ ls -l
total 530
-rw-r--r--  1 root  wheel     512 Sep  5 12:31 myfile
-rw-r--r--  1 root  wheel     512 Sep  5 12:31 otherfile
-rw-r--r--  1 root  wheel    7680 Sep  5 12:31 email.txt
```

The output of a long directory listing

FreeBSD File Permissions III



Meaning of the permission display

FreeBSD File Permissions IV

```
-  rw-  r--  r--
```

Owner permissions:
First position: read permission (yes)
Second position: write permission (yes)
Third position: execute permission (no)

Meaning of the permission display

FreeBSD File Permissions V

```
- rw- r-- r--
```

Group permissions:
First position: read permission (yes)
Second position: write permission (no)
Third position: execute permission (no)

Meaning of the permission display

FreeBSD File Permissions VI

```
-   rw-   r--   r--
```

Everyone permissions:
First position: read permission (yes)
Second position: write permission (no)
Third position: execute permission (no)

Meaning of the permission display

File Permissions Encoding I

Octal	Binary	Permission	Octal	Binary	Permission
0	000	- - -	4	100	r - -
1	001	- - x	5	101	r - x
2	010	- w -	6	110	r w -
3	011	- w x	7	111	r w x

Three octal digits are used to represent permissions

File Permissions Encoding II

```
$ chmod 644 myfile
```

Three octal digits are used to represent permissions

File Permissions Calculator

- Easy tool to use to compute values to use
- http://permissions-calculator.org/

Symbolic Permissions

Option	Letter	Meaning
Who	u	User
	g	Group
	o	Other
	a	All (“world”)
Action	+	Add
	-	Remove
	=	Explicit set
Permissions	r	Read
	w	Write
	x	Execute

Characters can be used instead of numbers

chmod: Change Permissions

- The chmod command changes permissions on a file or directory
- Can be used with either the numeric or symbolic notations
- Examples:
  - chmod 711 myprog – Gives all permissions to the owner and execute only permission to everyone else
  - chmod ug=rw,o=r – Gives read/write permission to owner and group, read permission to others

chown: Change Ownership

- The chown command changes the owner of a file
- You must be owner or file, or elevate to superuser

chgrp: Change Group

- The chgrp command changes the group ownership of a file
- You must belong to the group you are changing to, or elevate to superuser

Access Control Lists I

- User, group and world are very high level
- Not sufficient for modern system usage
- Access control lists supported

Access Control Lists II

```
setfacl -m u:lisa:r filename
```

Some Thoughts on Permissions

- We have been discussing local control of permissions
- This would be very tedious to manage in a large organization
- Instead, a directory service is used
- Most commonly, LDAP
  - Lightweight directory access protocol
  - OpenLDAP is the traditional open source LDAP server

Assignment

- TODO

THANK YOU!