

## Introduction

- **Welcome**
  - **Five Domains**
    - The Linux Community and a Career in Open Source
    - Finding Your Way on a Linux System
    - The Power of the Command Line
    - The Linux Operating System
    - Security and File Permissions
- **About the Exam**
  - **Closed book**
  - **Up to 60 minutes**
  - **40 multiple-choice items**
  - **Minimum score of 500 points to pass**
  - **No prerequisites**
  - **Lifetime certification**



## Evolution of Linux

- **An Introduction to Linux**
  - **Linux**
    - A family of open source Unix-like operating system, typically packaged into a distribution
  - **Examples**
    - Ubuntu, Debian, Fedora
  - **Command Line Interface (CLI) or the terminal**
  - **Why study Linux?**
    - Over 96% of web servers are run on Linux
    - Low cost
    - Open source
    - Ease of scalability
- **Open Source Philosophy**
  - **Open Source**
    - Concept of making source codes available for public use or modification from its original design
  - **Examples**
    - Apache, WordPress
  - **Software License Types**
    - GNU General Public License
    - Apache License
    - MIT License
    - Unlicense
- **Linux Distributions**
  - **Distribution or Distro**
    - Linux kernel
    - Supporting software and libraries
    - Configuration files
  - **Kernel**
    - A low-level computer program which functions as the bridge between the user and the computer's resources
  - **Different distributions have different scripts and utilities to launch programs**
  - **Installation**
    - Image file on a CD
    - USB flash drive
    - Virtual server or cloud service

- **Distribution Life Cycle**
  - **Release Schedule**
    - Distribution life cycle that specifies when are new versions released to the public
  - **Pre-Release Versions**
    - Recommended to be used only for testing and debugging
    - Categories
      - Alpha – very new and contains a lot of bugs
      - Beta – where most testing are done
  - **Examples**
    - Disco Dingo (19.04), Cosmic Cuttlefish (18.10)
  - **Support Versions**
    - Short-term
    - Long-term
  - **Rolling Release Schedule** – ongoing updates
- **Embedded Systems**
  - **Embedded System**
    - A controller with a dedicated function within a larger mechanical or electrical system
  - **Example Applications**
    - Industrial automation
    - Navigation equipment
    - Medical devices
    - Wearable technology
    - Home appliances
  - **Examples**
    - Arduino
    - Raspberry Pi
    - Android
      - Highly-scalable
      - User-friendly
      - Open source
      - Free to use
- **Hardware Requirements**
  - **System requirements**
    - Prerequisites that are often used as a guide and not an absolute rule

- **Hardware Compatibility List (HCL)**
  - Lists tested, compatible, and sometimes incompatible hardware devices for a particular distribution
- **Minimum Requirements**
  - The lowest possible hardware specification that your computer should have to boot successfully and use with basic functionality
- **Recommended Requirements**
  - Hardware that one should have in order to fully maximize a system's potential
- **Operating System Differences**
  - **Windows**
    - 90% of home users
    - Widely-available and widely-supported
    - Most prone to malware
    - Requires a license
    - GUI-based
  - **macOS**
    - 7% of home users
    - Free but only works on Apple systems
    - GUI-based
  - **Linux**
    - Less than 2% of home users
    - More than 75% of enterprise server environments
    - Known as the OS for computer experts and hackers
    - Source code is available for modification
    - Open source
    - Can run from the command line only

### Open Source Applications

- **What is Open Source?**
  - **Open-source Software**
    - Software released under a license in which the copyright holder grants users the rights to study, change, and distribute it to anyone and for any purpose
  - **Advantages**
    - Security
    - Affordability
    - Transparency
    - Interoperability
    - Scalability
    - Localization
  - **Open-source Initiative**
    - Public benefit corporation that promotes the use of open-source software
  - **Open Source Definition (OSD)**
    - A document which determines whether a software license can be labeled with the open-source certification mark
- **Cost of Open Source**
  - **Technical support, training, administration, and maintenance**
  - **Public Domain**
    - No ownership, copyright, trademark, or patent
  - **Hacking Culture**
  - **Free License**
    - Private funding
    - Crowdfunding
    - Donations
    - Commercialization
- **Package Installs and Repositories**
  - **Linux applications generally don't run on every distribution**
  - **Linux users don't normally download and install applications from the applications' websites**
  - **Archives contain a list of files**
  - **Package Formats**
    - **.deb**

- Gets its name from Debian; also used by Debian-based distros such as Ubuntu
- **.rpm**
  - Originally stood for Red Hat Package Manager; used by Fedora and openSUSE
- **.tar**
  - Considered a universal package format and is used by distros such as ArchLinux and Slackware
  - Can be .tar, .tgz, or .tar.gz
- **Personal Package Archives (PPAs)**
  - Contains software compiled by individual users and teams that have newer features that are not in the official distributions

## The Linux Environment

- **Linux Desktop Environment**
  - **Kernel**
    - Translates software commands into hardware-specific requests
  - **Command Syntax**
    - Rules in which commands need to be run
  - **Desktop Environment**
    - Graphical user interface (GUI)
    - Linux desktop environments are modular
  - **Examples**
    - Ubuntu – Unity, Kubuntu, Xubuntu
- **Linux Shell and Commands**
  - **Linux Shell**
    - Program that takes commands from the keyboard and gives them to the kernel to execute
  - **Examples**
    - Bourne Again Shell (Bash)
      - Acts as the shell program within the command line interface
    - ksh (Korn shell)
    - tcsh (Tee See shell)
    - zsh (Zee shell)
  - **Terminal (terminal emulator)**
    - Provides access to a shell session
    - Examples
      - gnome-terminal
      - konsole
      - xterm
      - rxvt
      - kvt
      - nxterm
      - eterm
  - **Common commands**
    - **List (ls)**
      - Shows the files, folders, and directories within the system
    - **Change directory (cd)**
      - Switches between directories
    - **Move (mv)**
      - Moves a file from one folder directory to another

- **Manual (man)**
  - Shows all information about a command being used
- **Make directory (mkdir)**
  - Makes a new empty folder or directory inside the filesystem
- **Remove directory (rmdir)**
  - Removes an empty, existing folder or directory
- **Touch (touch)**
  - Makes a file using the command line
- **Remove (rm)**
  - Removes files and directories
- **Locate (locate)**
  - Finds a file within the system
- **Clear (clear)**
  - Clears the screen or terminal environment
- **Managing Software Packages**
  - **Package**
    - A compressed file archive containing all of the files that come with a particular application
    - Package prerequisites = dependencies
  - **Common types of packages**
    - .deb
      - Debian, Debian-derived distros
      - Package managers: DPKG, APT-GET, and APT
    - .rpm
      - Red Hat, Fedora, openSUSE
      - Package managers: RPM, YUM, and DNF
    - .tgz
      - the “universal” Linux format



## The Command Line

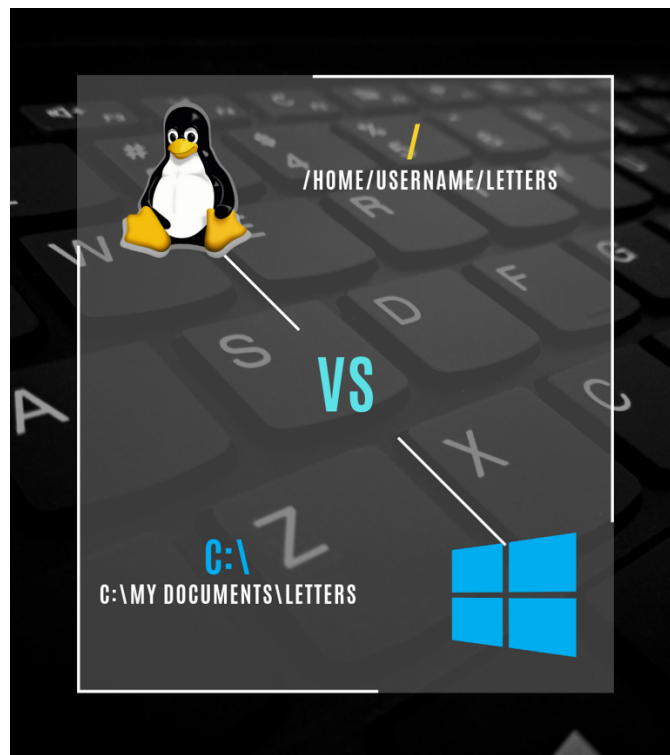
- **Basic Shell**
  - **Users are allowed to install various GUI terminal programs**
  - **Root User**
    - Administrator account in Linux
- **Command Line Syntax**
  - **Computer operation steps**
    - Computer waits for user input
    - User selects a command and enters it using a keyboard or a mouse
    - Computer executes the command
  - **In Linux, the shell displays a prompt**
  - **Command**
    - A sequence of characters in a line which ends by pressing the enter key and is subsequently evaluated by the shell
  - **Syntax - rules**
  - **The first word is usually the command name, while the other words are the parameters**
    - Types of parameters
      - Options (switches) – start with a dash and allow certain commands to be switched on or off
      - Argument – no leading dashes; often the names of files that the command should process
  - **General command structure**
    - Command (What to do?) > Options (How to do it?) > Arguments (What to do with it?)
- **Variables**
  - **Shell Script**
    - A file containing a series of commands
  - **Variables**
    - Areas of memory that can be used to store information and are referred to by a name
    - Act as a placeholder
  - **Naming rules**
    - Variable names must start with a letter
    - A name must not contain embedded spaces (Underscores are used instead)
    - Punctuation marks are not allowed

- **Print Environment (printenv)**
  - Displays all the environmental variables (variables in ALL CAPS)
- **Quoting**
  - **Shell meta-characters**
    - Treated in a special way in the command line
  - **Examples of meta-characters**
    - Blank or space character ( " " )
      - Tells a shell to use separate arguments between commands
      - Any number of blanks or spaces are treated as one
    - Dollar sign ( \$ )
    - Star ( \* )
    - Semi-colon ( ; )
    - Greater-than symbol ( > )
    - Question mark ( ? )
    - Ampersand ( & )
    - Pipe ( | )
  - **Quoting**
    - The act of protecting shell meta-characters from being treated specially by the shell
    - Prevents the shell from acting on and expanding the meta-characters
    - Done in double quotes ( " " ), single quotes ( ' ' ), or backslash ( \ )
    - Backslashes can be used to quote or turn off a special character's ability
- **Man and Info Pages**
  - **Manual (Man) Pages**
    - A set of pages explaining every command available on the system
    - `man <command>`
    - `man -k <search term>`
    - `/<search term>`
  - **Commands**
    - Long hand
      - More human-readable
      - Lets users remember what the command is doing
      - Example: `--all`
    - Short hand
      - Quicker to type out
      - Allows for multiple options to be chained together and is quicker to type out
      - Example: `-a`

- **Info Pages**
  - More detailed than man pages; divided into different nodes or pages and works like a web browser
  - Info <command>
  - p for previous
  - n for next
  - q for exit

## Directories and Files

- **Introduction to the Linux Filesystem**
  - **Multuser system**
    - Any number of users can simultaneously work on one single machine
  - **Types of users**
    - Normal user
    - Superuser/administrator (root)
      - A superuser can access all parts of the system to execute admin tasks
  - **Home directory**
    - This is where all private data is stored
  - **System directory**
    - Holds central configuration and executable files that can only be modified by the superuser
  - **Root of the filesystem**
    - Topmost directory represented by a single slash ( / )
    - Example:



- **Linux Filesystem Key Features**
  - In Linux, data from unmounted partition or device is inaccessible
  - Linux distinguishes between uppercase and lowercase letters in the filesystem
    - Example: test.txt ≠ TeSt.txt ≠ Test.txt
  - Files in Linux may have a file extension, but do not need to have one
  - Files can be hidden by adding a dot ( . ) at the beginning of the filename
  - Only the owner of a file or directory (plus the root user) can grant access
  - Access permissions
    - Write
    - Read
    - Execute
  - Access permissions to files and folders can be changed through the shell or the file manager
  - Filesystem Hierarchy Standard (FHS)
    - Distinguishes between shareable and unshareable files, and between static and variable files
    - Shareable files – user data files, program binary files
    - Unshareable files – configuration files
    - Static files – program executables
    - Variable files – any files stored in the home directory like documents, photos
- **Navigating Files and Directories**
  - List command (ls)
    - ls -a – displays all the files in the directory
    - ls -l – displays an output which includes permission strings, ownership, file sizes, file creation dates
  - Change directory (cd)
    - Linux uses a forward slash as a directory separator
    - A backslash serves as a “quote” or “escape” character
  - Present Working Directory (pwd)
    - Tells what directory you are in
  - File references
    - Absolute
      - Relative to the root directory
      - Begins with a forward slash
      - Example: /home/user1/file.txt
    - Home directory
      - A tilde (~) replaces the path’s home directory
      - Example: ~/file.txt

- Relative
  - Relative to the current directory
  - Subdirectories
    - . refers to “this” directory
    - .. refers to the parent directory
  - Subdirectory replaces the parent directory
  - Example: ../dir2/afile.txt
- **File Creation and Management**
  - **Touch (touch)**
    - Creates files on the command line
    - Example: touch newfile.txt
    - Empty files can act as “scratch files”
  - **Copy (cp)**
    - Copies files
  - **Move (mv)**
    - Moves and renames files
    - The mv command’s effects are much like cp, except the new file will replace the original file
    - Copies and deletes the original file
- **Creating Links**
  - **Hard Link**
    - A duplicate directory entry where both entries point to the same file
    - In origname linkname
  - **Symbolic Link**
    - A file that refers to another file by name
    - In -s origname linkname
- **Wildcards and Case Sensitivity**
  - **Wildcard**
    - A symbol or set of symbols that stands in for other characters
  - **Types of wildcards**
    - Question mark ( ? )
      - Stands for a single character
      - Example: b??l can match bowl, ball, bull, etc.
    - Asterisk ( \* )
      - Matches any character or set of characters, including no characters
      - Example: b\*l can match bowl, ball, bull, bl, blll, brawl, etc.

- Bracketed value ( [ ] )
  - Matches any character inside the bracket
  - Example: b[ao][lw]l can match bowl, ball, etc., but not bull
- **Careless usage of wildcards can lead to undesired consequences**
- **Case sensitivity is a function of the Linux filesystem and not the Linux operating system itself**
- **Manipulating Directories**
  - **Make Directory (mkdir)**
    - Creates a new directory
    - Example: mkdir newfolder
  - **Remove Directory (rmdir)**
    - Deletes an empty directory
    - Example: rmdir newfolder
    - To delete a set of nested empty directories, -p can be used
      - Example: rmdir -p newfolder
    - To delete directories that contain files within a directory tree, use:  
rm -r newfolder
  - **Remember that directory names are case sensitive**
  - **Directories are just special files as far as the Linux filesystem is concerned**
  - **Touch (touch)**
    - When used with a directory, it will only update the date/time stamp
  - **Copy (cp)**
    - Can be used to copy a directory if used with the -r or -a switches
  - **Move (mv)**
    - Can be used for directories or files
  - **Link (ln)**
    - ln -s can create symbolic links to a directory
    - directories don't have support but hard link, only symbolic links

### Searching and Extracting Data from Files and Archiving

- **Command Line Pipes**
  - **Command line pipe or pipeline ( | )**
    - A standard output from one program or command that is used as input for the second program or command
    - Example: `cat sample | less`, `cat sample | pg`, `cat sample | more`
    - Pipes allow for the quick execution of commands performing complex tasks
  - **Grep**
    - Searches for keywords in the output
    - Example: `grep <optional set of options> <regular expression> <optional filename specification>`
- **Input/Output Redirection**
  - **Redirect to an output file if you need to save it for future reference**
  - **Redirect from an input file if you have a program that needs it**
  - **A program's output can be used as an input via the process of piping**
  - **Xargs**
    - Enables a user to generate command line options from files or other programs' output
  - **Common redirector operations**
    - Greater than symbol ( > )
      - Creates a new file containing standard output
    - 2 Greater than symbols ( >> )
      - Appends the existing output to an existing file
    - The number 2 + Greater than symbol ( 2> )
      - Creates a new file containing any standard errors
    - The number 2 + 2 Greater then symbol ( 2>> )
      - Appends standard error to an existing file
    - Ampersand + Greater then symbol ( &> )
      - Creates a new file that contains the standard output and the standard error
    - Less than symbol ( < )
      - Sends the content of a specified file as input back into the standard input
    - 2 Less than symbols ( << )
      - Accepts text on the following lines as standard input
    - Less than + Greater than symbols ( <> )
      - Specified file can be used for standard input and standard output



- **Standard Output**
  - Normal program messages
- **Standard Error**
  - Any error messages
- **Basic Regular Expressions**
  - **Regular Expressions**
    - A way to describe patterns that a user might want to look for in data files
    - Regular expressions are similar to wildcards
  - **Forms of Regular Expressions**
    - Basic
    - Extended
  - **Types of Regular Expressions**
    - Alphanumeric/Alphabetic string
    - Non-alphanumeric string
  - **Special Characters**
    - Bracket Expression ( [ ] )
      - Example: b[aeiou]g can match bag, beg, big, bog, bug
    - Range Expression ( - )
      - Example: a[2-4]z can match a2z, a3z, a4z
    - Single Characters ( . )
      - Example: a.z can match all words that starts with a and ends in z that has one letter in between
    - Caret ( ^ )
      - Shows the start of a line
    - Dollar sign ( \$ )
      - Shows the end of a line
    - Repetition
      - Example: .\*
    - Backslash ( \ )
      - Used to escape a special character
      - Example: filename\.txt
- **Archiving Files**
  - **File-Archiving Tool**
    - Collects a group of files into a single “package” file to easily move around on a single system
  - **Common archiving commands**
    - Tar (Tape archiver)

- Used to archive various data files into a single file (archive file) while the original files remain on the disk
- Tarball
  - Compressed archive file
- ZIP
  - GZIP (.gz; .tgz for tarballs)
    - gunzip
  - BZIP2 (.bz2; .tbz, .tbz2, .tb2 for tarballs)
    - bunzip2
  - XZ (.xz; .txz for tarballs)
    - unxz

## Scripting Basics

- **Text Files and Text Editors**
  - **Text Files**
    - Configuration files and shell scripts
  - **Text Editor**
    - Lets you edit documents that are stored in plain-text format
  - **File Types**
    - Human language files
    - Programming language files
    - Formatted text files
    - Program and system configuration files
    - Program log files
  - **Popular Text Editors**
    - vi
      - very small and installed by default
      - a little more difficult to use
    - emacs
      - big editor with a lot more features
      - less likely to be installed by default on lightweight distributions
      - also has a GUI version
    - pico
      - modeled after emacs
      - more lightweight and better suited for lightweight distributions
      - often installed by default
    - nano
      - clone of pico but has extra features
      - much more lightweight than emacs
      - commonly found in small and lightweight distributions
- **Beginning a Shell Script**
  - **Script**
    - A program written in an interpreted language, typically associated with a shell or a compiled program
  - **GUI-based Editors**
    - KWrite
    - Gedit
  - **Text-based Editors**
    - vi
    - emacs

- pico
- nano
- **Commands**
  - **Most commands entered into a shell prompt are external commands**
  - **The ampersand (&) lets users run many programs simultaneously**
  - **Commonly-used Commands**
    - **ls, mv, cp, rm**
    - **Grep**
      - Locates files containing specific strings and display them into a file or screen
    - **Find**
      - Locates files by filenames, ownerships, and other permissions at the filesystem level
    - **Cut**
      - Extracts text from a field within a file
    - **Echo**
      - Displays something as a message to the user in the command line terminal
- **Arguments**
  - **Values of variables can be:**
    - Passed as parameters to a script
    - Generated internally to a script
    - Extracted from a script's environment
  - **Arguments**
    - Variables that are passed to the script. These are also called parameters
    - Represented in the script as a dollar sign ( \$ ) followed by a number from 0 onwards
    - Examples: \$0, \$1, \$2
- **Variables**
  - **Common Commands**
    - **Hostname**
      - Shows the current hostname where the command is being run on
    - **Date**
      - Gives the current date and time
    - **Uptime**
      - Total time the computer has been up and running

- df
  - Shows the free disk space
- **Conditional Expressions**
  - **Conditional Expressions**
    - Enable a script to perform one of several actions relying on a particular condition or value of a variable
    - Example: if [-f file], if [-s file]
  - **Conditionals may be combined together with the logical and ( && ) or logical or ( || ) operators**
  - **You can nest several if / then /else clauses**
- **Loops and Functions**
  - **Loops**
    - Tell the script to perform the same task repeatedly
  - **Seq**
    - Generates a list of numbers starting from its first argument and continuing to its last one
    - Example: seq 1 10, seq 1 2 10
  - **While**
    - Executes for as long as its condition is true
  - **Until**
    - Similar to the while loop but continues to execute as long as its condition is false or until the condition becomes true
  - **Function**
    - A specific subtask to be called by name from other parts of the script
    - The keyword function may optionally precede the function name
- **Exit Value**
  - **A script's return value is the same as the last command the script called**
  - **Exit**
    - Causes immediate termination of the script

## Processes and Process Data

- **Package Management Principles**
  - **Process**
    - Any running program
  - **Package management varies between distros**
  - **Package Management Principles**
    - Each package is a single file
    - Packages rely on other programs to do the work of installing the software
    - Packages contain dependency information
    - Packages contain version information
    - Packages contain architecture information
    - Binary packages are built from source packages
  - **Package software maintains a database of information about all installed packages, which includes:**
    - Names and version numbers
    - Locations of all installed files
- **Package Management Systems**
  - **Examples:**
    - RPM
    - Debian
  - **Always use native packages on your system**
  - **Software Repositories**
    - Area where tools can be automatically downloaded as packages and then get installed
  - **Software Installation Process**
    - A command is issued to install a program
    - The software locates dependencies of the specified program
    - The user issues a final approval for software installation
    - The software downloads all of the necessary packages
    - The software installs all the packages
- **Process Hierarchy**
  - **Init Process**
    - Responsible for starting up all basic programs that Linux needs to run
    - **Children**
      - Program launched by init
    - **Parent**
      - Process that launched a program

- **Process ID (PID)**
- **Parent Process ID (PPID)**
- **Identifying Running Processes**
  - **Utilities to identify processes**
    - **ps**
      - Provides information at only a single moment in time
    - **top**
      - An interactive version of ps
    - **free**
      - Tells how much memory is being used
  - **Load Average**
    - Measure of the demand of CPU time by different applications
    - Useful in detecting “runaway processes”
- **Measuring Memory Use**
  - **Pressing the M key within top sorts processes by memory use**
  - **A program with a memory leak consumes increasing amounts of memory**
  - **Mem**
    - This line reveals total RAM statistics
  - **Swap**
    - This line reveals how much swap space Linux is using (disk space that’s set aside as an adjunct to memory)
- **Log Files**
  - **Daemons**
    - Programs that run in the background
  - **Log Files**
    - Serve as a record or notes
    - Mostly stored in /var/log
  - **Common Log Files**
    - **boot.log**
      - Summary of services that started up late in the boot process through the `rc` startup scripts
    - **cups/**
      - Related to Linux printing system
    - **gdm/**
      - Related to the gnome display manager or gdm environment
      - Handles graphical-based user logins

- messages or syslog
  - General purpose log file containing messages from many different daemons that don't have dedicated log files
- Secure
  - Security-related messages
- Xorg.0.log
  - Info about the most recent startup of an X windows system
- **Log files are frequently rotated**
- **Log files are mostly plaintext files**
- **Syslog/Syslogd**
  - General standard for logging system and program messages
- **Klog/Klogd**
  - Handles logging messages from the kernel separately from ordinary programs
- **System Messaging**
  - A technique wherein a log daemon accepts messages from other processes
- **Kernel Ring Buffer**
  - Returned by entering `$ dmesg`, `$ dmesg | less`
  - Kernel ring buffer messages are invaluable in diagnosing hardware and driver problems
  - To manually create a kernel ring buffer log file, edit `/etc/rc.d/rc.local` by adding this at the end: `dmesg > /var/log/dmesg`



## Networking Basics

- **Network Features**
  - **Common Networking Terms and Definitions**
    - Domain Name System (DNS)
      - Global network of servers that translates between hostnames and IP addresses
      - “Internet phonebook”
    - Dynamic Host Configuration Protocol (DHCP)
      - A way for computers on a network to be able to obtain configuration information from another computer on the network
    - Ethernet
      - Wired network hardware used by most computers today
    - Hostname
      - The name of a computer that is easier to read and remember
    - Internet
      - Globe-spanning network of interconnected computers that we use TCP-IP to communicate over
    - Internet Protocol (IP) Address
      - A number assigned to computer for network addressing purposes
      - “Phone number” for a computer
      - IPv4, IPv6
    - Network Mask (Netmask)
      - Distinguishes between the network and the machine portions of an IP address
    - Router
      - A device that connects two or more networks together
      - Serves as a gateway between these two networks
    - Transmission Control Protocol-Internet Protocol (TCP-IP)
      - A set of standards that underly most modern network connections at the software level
      - Backbone of the Internet
    - Wi-Fi
      - Common name for wireless networking IEEE 802.11 standard
  - **Four Things Needed for a Valid Internet Connection**
    - IP Address
    - Netmask
    - Router’s IP Address
    - DNS Server’s IP Address

- **Configuring a Connection**
  - **Automatic configuration is handled via DHCP**
  - **Fixed DHCP**
    - Each computer receives the exact same IP address every time it boots up
  - **Dynamic DHCP**
    - A computer receives possibly different IP addresses from the DHCP server every time it connects
  - **Establishing a Wi-Fi connection is easiest using the GUI method**
  - **Wireless Configuration Tools**
    - iwlist
      - Can identify nearby wireless networks
    - iwconfig
      - Can connect to and disconnect from specific wireless networks
  - **Wired Configuration Tools**
    - ifconfig
      - Brings up or shut down a specific network connection and associate a netmask to a particular piece of network hardware such as a network adapter
    - route
      - Adjusts the computer's routing table
    - /etc/resolv.conf
      - Configuration file that contains the IP addresses of up to 3 DNS servers
    - DHCP client
      - Can configure a network connection automatically
      - Example: dhclient, dhcpcd
    - Distribution-Specific Network Scripts
- **Network Testing**
  - **Common Network Connectivity Tests**
    - Routing table
    - PING
      - If you can ping your local systems but not your remote systems, you must be having a router problem
      - If you can ping the IP address but not the name of the device, you must be having a DNS problem
      - If you cannot ping at all, you must be having a fundamental configuration problem
    - Traceroute
      - Tests the connectivity for breaks

- Domain Name Servers (DNS)
  - Host, dig, nslookup
- Netstat
  - “Swiss army knife” of networking tools
- **Network Protection**
  - **Basic Tips**
    - Shut down unused servers
    - Enable a firewall
    - Use good passwords
    - Be suspicious
    - Keep your software up-to-date

### User Accounts and Groups

- **Understanding User Accounts**
  - **User accounts**
    - Enable multiple users to share a single, physical computer
    - Allow for tracking of who's using system resources
    - Most account features are defined in `/etc/passwd`
  - **Username**
    - Most relevant feature which consists mainly of lowercase letters and some numbers
  - **Password**
    - Usually contains an "x" which means it is stored in a different file known as the shadow file
  - **User ID (UID)**
    - Used to track the accounts
  - **Group ID (GID)**
  - **Groups**
    - Collections of accounts that can be given special permissions within the Linux system
  - **Command field**
    - Holds the user's full name
  - **Home directory**
    - States where the home directory is
  - **Default shell**
    - Associated with every single account
- **Account Security**
  - **Passwords today are stored in `/etc/shadow`**
  - **Salted hash**
    - Uses a one-way mathematical process with additional random input to produce a non-readable password
  - **Username**
  - **Password**
    - Stored as a salted hash
  - **Last password change**
  - **Days until a change is allowed**
  - **Days before a change is required**
  - **Days of warning before password expiration**
  - **Days between expiration and deactivation**
  - **Expiration date**

- **Special flag**
- **Understanding Groups**
  - **Groups**
    - Collections of accounts that are defined in the `/etc/group` file
  - **Group name**
  - **Password**
  - **GID**
  - **User list**
  - **Group membership**
    - By specifying the group's GID in users' individual `/etc/passwd` entries
    - By specifying usernames in the user list in the `/etc/group` file
- **Using Account Tools**
  - **whoami**
    - Tells the user with multiple accounts which one they're logged in as
  - **id**
    - Gives more information about the user
  - **who**
    - Gives information about who's currently using the computer
      - Username
      - Terminal identifier
      - Login date and time
      - Remote host
  - **w**
    - Similar to `who` but produces more verbose output
      - Idle time
      - JCPU – total amount of CPU time for a given session
      - PCPU – total amount of CPU time for the processes running inside a given session
      - WHAT – tells what the session is running and doing
- **Working as Root**
  - **Root**
    - Also called superuser or administrator
    - Has access to system files
  - **Root user privilege**
    - Log in as root user
    - Use the switch user (`su`) command
    - Use the `sudo` command

- Similar to the su command but only works for one command at a time
- **Precautions**
  - Confirm the need for root access
  - Verify the command
  - Do not run a suspicious program as root
  - Favor the use of sudo over su
  - Do not leave a root shell open
  - Do not share root password with others

# Managing File Ownership and Permission

- **Setting Ownership**
  - **Ownership**
    - Every file has an associated owner or account with it
    - Also applies to running programs and processes
  - **Permission**
    - Defines what a file's owner, the member of the file's group, and other users can do with the file
  - **Sets of permissions**
    - User
    - Group
    - Other
  - **Ownership association**
    - User ID number
    - Group ID number
  - **Change ownership (chown)**
    - Changes a file's ownership inside the terminal
    - Can only be done by a root user
  - **Change group (chgrp)**
    - Changes a file's group
    - Can also be done by an ordinary user if you're the file owner and you belong to both groups
- **Understanding Permissions**
  - **Permissions**
    - Shows the file permissions
  - **Number of links**
    - Shows the number of unique file names that may be used to access a file
  - **Username**
    - Identifies the file's owner
  - **Group name**
    - Identifies the group that owns a file
  - **File size**
  - **Time stamp**
    - Tells when a file was last modified
  - **Filename**
  - **Permission string**
    - File Type Code

- This type character represents the file's type and is often omitted from descriptions when the file type is not relevant
  - Normal data file ( - )
  - Directory disk ( d )
  - Symbolic link ( l )
  - Named pipe ( p )
  - Socket ( s )
  - Block device ( b )
  - Character device ( c )
- Owner Permissions
  - Determine what the file's owner can do with the file
- Group Permissions
  - Determine what members of the file's group who are not its owner can do with the file
- World or Other Permissions
  - Determine what users who are not the file's owner or members of its group can do with the file
- **Permission Strings and Setting Umask**
  - **Access types**
    - Read
    - Write
    - Execute
      - The file can be run as a program
    - Example: -rwxr-xr-x





Permission String	3-digit Code	Permissions
<code>rw-rw-rw-</code>	<code>777</code>	Read, write, and execute permissions for all users
<code>rw-r--r--</code>	<code>750</code>	Read and execute permission for the owner and group. The file's owner also has write permission. Other users have no access to the file
<code>rw-r--r--</code>	<code>644</code>	Read and write permissions for the owner. Read-only permission for all others
<code>r-----</code>	<code>400</code>	Read permission for the owner. No permission for anybody else

- **Special Cases**
  - Directory execute bit
    - Grants permission to search the directory
  - Directory write permission
    - Allows a user to create, delete, rename files within a directory, even if the user does not own the file
  - Symbolic link
    - Permissions on symbolic links are always `777`
  - Root user
    - Many of the permission rules don't apply to root
- **User Mask (Umask)**
  - Determines the default permissions for new files
- **Using Sticky Bits**
  - **Sticky Bit**
    - A special filesystem flag that alters behaviors and limits a user to only delete own files or files stored in own directory
  - **Octal Code**
    - 1 – on, 0 – off
  - **Symbolic Code**
    - `+t` – on, `-t` – off
- **Special Execute Permissions**
  - **Set User ID (SUID)**
    - Tells Linux to run the program with the permissions of whoever owns the file rather than the user who runs the program
  - **Set Group ID (SGID)**
    - Sets the group of the running program to the group of the file
  - **Octal Codes**
    - 4 to set the SUID bit

- 2 to set the SGID bit
  - 6 to set both bits
  - **Symbolic Codes**
    - u+s to set the SUID bit
    - g+s to set the SGID bit
    - ug to set both bits
- **Hiding Files and Directories**
  - **Hidden bit**
    - Hides a file in the file manager
    - Linux uses a file naming convention to hide files by way of a dot ( . ) at the beginning of the file name
    - Example: .myfile.txt
  - **Hidden directories**
    - Current directory ( . )
    - Parent directory ( .. )
  - **Renaming a file will also make the file inaccessible to programs that uses it**
  - **The -d option will get information from the subdirectories rather than the content**

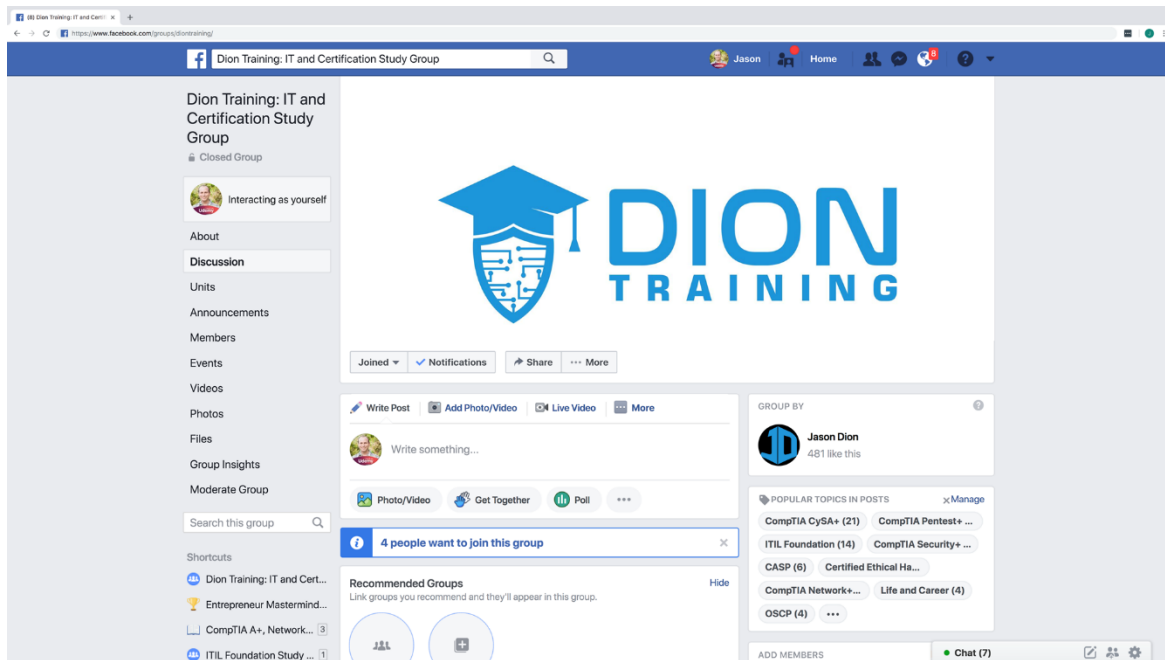
## Conclusion

- **Conclusion**
  - Five Domains
    - The Linux Community and a Career in Open Source
    - Finding Your Way on a Linux System
    - The Power of the Command Line
    - The Linux Operating System
    - Security and File Permissions
  - Take the two practice exams that are included in this course
  - Keep studying, take the two practice exams that are included in this course, and let me know when you pass!
  - Visit [DionTraining.com](https://www.DionTraining.com) to get your discounted exam voucher and use the coupon code LINKEDIN to save even more!





## Linux Essentials (Study Notes)



**Let's get you certified!**