CompTIA Linux+ (XK0-005)

- **Linux+**
  - *The new CompTIA Linux+ is for the IT pro who will use Linux to manage everything from cars and smartphones to servers and supercomputers, as a vast number of enterprises use Linux in cloud, cybersecurity, mobile and web administration applications. (CompTIA.org)*

- **Exam Description**
  - CompTIA Linux+ validates the skills of IT professionals with hands-on experience configuring, monitoring, and supporting servers running the Linux operating system. The new exam has an increased focus on the following topics: security, kernel modules, storage & virtualization, device management at an enterprise level, git & automation, networking & firewalls, server side & command line, server (vs. client-based) coverage, troubleshooting and SELinux.

- **Four Domains**
  - 32% System Management
  - 21% Security
  - 19% Scripting, Containers, and Automation
  - 28% Troubleshooting

- **Exam Details**
    - Up to 90 questions in 90 minutes
        - Multiple-choice
        - Performance-based/Simulations
        - Fill-in-the-Blank
            - Requires a 720 out of 900
            - Recommended Experience:
                - CompTIA A+, CompTIA Network+ and 12 months of Linux admin experience
    - Released: April 2, 2019 (XK0-004); July 12, 2022 (XK0-005)

- **Are You Ready?**
    - Take practice exams
    - Did you score at least 85% or higher?
    - If you need more practice, take additional practice exams to hone your skills before attempting the exam

- **What kind of jobs can I get?**

Basic Linux Task

Objectives:

- 1.1: Summarize Linux fundamentals.
- 1.7: Given a scenario, manage software configurations.
- 2.2: Given a scenario, implement identity management.

- **Linux Design Philosophy**
    - *Linux*
        - Family of open-source Unix-like operating systems
        - Linux is an open-source operating system that allows anyone to freely download, modify, and redistribute it
            - Ubuntu
            - Debian
            - Fedora Linux
    - *Open-Source*
        - Computer software in which the source code is readily available for public use or modified from the original design
    - *Proprietary*
        - Licensed software that has restrictions on what end users can do
            - General Public License
            - Apache License
            - MIT License
            - Creative Commons Zero

- Linux follows the Unix philosophy of simplicity and modularity

- Open-source software also comes with some caveats
  - Steep learning curve
  - Not well-supported
  - No definite/official version
  - No official vendor-provided support

- CentOS
  - Free software project and focuses on creating compatible open source and free versions of Red Hat and Enterprise Linux

- **CLI and The Shell**
  - *Command Line Interface (CLI)*
    - Text-based interface between the user and the operating system that accepts input in the form of typed commands

  - Shell
    - Contains the core portion of the operating system
    - The original Unix shell was called the Bourne shell (sh), and was replaced by Bourne-again (Bash)

  - *Syntax*
    - Proper way of structuring a command and any supporting information
      - Command

            - Options
            - Argument
        - Commands entered into Bash are case sensitive

    - Command line interface interacts directly with the shell. And the shell requires the user to use syntax when issuing commands

- **Bash Commands**
    - *Echo Command*
        - Repeats input back to the user on the screen

    - *LS Command*
        - Lists the content of a directory that can give options to view permissions and hidden files

    - *PWD Command*
        - Shows the current working directory

    - *CD Command*
        - Changes current working directory

    - *CP Command*
        - Copy file or directory to another location

    - *MKDIR Command*
        - Create new directory

- *Clear Command*
  - Used to clear the command line interface of all text
- *Cat Command*
  - Used to view the contents of a file without editing option
- *Less Command*
  - Used to view the contents of a file that won't fit on one screen
- Text Editors
  - *Vim*
    - Default text editor in Linux
  - *Nano*
    - Simple and user-friendly text editor that needs to be installed before using it
  - *Gedit*
    - GUI text editor that requires installation of a desktop environment like Gnome or KDE
- *Su (Substitute User) Command*
  - Allows to switch user credentials

- **Man Pages**
  - *Man Pages*
    - Contains the complete documentation for Linux commands
    - The most immediate source of help available
  - Man Page Features
    - Synopsis
      - Provides the syntax of the command with examples of its use
    - Bold text
      - Type exactly as shown
    - Italic text
      - Replace with appropriate argument
    - -abc
      - All commands in the brackets are optional
    - -a|-b
      - arguments on the left side of the pipe cannot be used together
    - Italic text with ellipsis (...)
      - the argument can be repeated
  - Man Command Options
    - -a
      - Find matching queries
    - -d
      - Display debugging information
    - -f
      - Show command description

        - -h
            - Display help options
        - -k
            - Same as the apropos command. It searches the short manual page descriptions for keywords and display any matches.
        - -K
            - Search for the specific string
        - -t
            - Format for printing

    - Man Page Sections
        - General commands
        - System calls
        - C library functions
        - Special files
        - File formats and conventions
        - Games and screensavers
        - Miscellaneous
        - System admin commands/daemons

    - Special Keys
        - Home Key
            - Moves to beginning page
        - End Key
            - Moves to end page

        - Page Up
            - Scrolls up one page
        - Page Down
            - Scrolls down one page
        - /
            - Begins a search
        - n
            - Moves to next occurrence
        - p
            - Moves to previous occurrence
        - q
            - Quits man page

- **Other Help Resources**
    - *Apropos*
        - Used to search the name section of all man pages
    - *Whatis*
        - Used to display a brief description of the given commands
    - *Info*
        - Used to display the information page of a command
        - Man pages contain all the information in a single page

- Ways to get help for commands
  - command name –h
  - command name –-help
- /usr/share/doc/
  - Directory of official documentation

- Internet Sources for help
  - https://help.ubuntu.com/
    - Linux Documentation Project
  - https://tldp.org
    - GNU Coreutils Manual
  - https://www.gnu.org/
    - software/coreutils/manual/

- Other Sources
  - Usenet Newsgroup
    - Online discussion repository
  - *Mailing Lists*
    - Threaded discussions in the form of email messages among members of a specific community
  - Q&A Website
    - Allows users to post questions that can be answered by other users
  - Stack Exchange
    - https://unix.stackexchange.com/

- Reddit
    - https://www.reddit.com/r/linux/
- *Internet Forum*
    - A social knowledge sharing platform that typically uses threaded posts
- Linux Questions
    - https://www.linuxquestions.org/

Users and Groups

Objective 2.2: Given a scenario, implement identity management.

- **Superuser**
  - Each account in the Linux system uses a UID or user ID
  - Types of accounts
    - Root user accounts
      - can do administrative tasks
      - Provides security for some applications and commands
      - A Linux root user account is more powerful than the local admin account in Windows
      - Logging on the system using the root user is a bad security practice
    - Standard user accounts
      - User that runs applications, configures databases, and creates websites
      - To ensure system security, user accounts should not be shared
      - *Least Privilege*
        - The practice of giving users only as much access as needed to perform certain job functions
    - Service accounts
      - Accounts that are specific to the service (HTTP for web service or mySQL for database service)
      - Service accounts run in the background and perform a single function

- Superuser (Root)
    - User with admin credentials
    - Always log into a system with a non-privileged user account
    - Superuser's account permissions may be needed to perform administrative functions such as:
        - Managing users
        - Configuring devices
        - Configuring network settings
    - By giving the user only the access needed, the system will remain secure

- *su command*
    - Allows to switch the user credentials to superuser
    - Options
        - -root
            - Switches the user credentials to root user

- *sudo command*
    - Enables the server admin to delegate commands to users
    - List the user in the /etc/sudoers file using the visudo editor to delegate the user account

- *sudoedit command*
    - Permits a user to edit a file with own credentials
    - Requires an entry in the sudoers file for anyone to use the sudoedit command

        - Do not edit /etc/sudoers with standard text editors like vi(m), nano, or gedit

- *visudo command*
    - Verifies /etc/sudoers syntax before committing changes
    - Options for visudo command
        - -c
            - (Check file errors)
        - -f
            - (Edit/check location)
        - -s
            - (Check file in strict mode)
        - -x
            - (Output in JSON)

- Administrative functions in some distributions are limited to the Wheel Group
    - Wheel group members exercise root privileges with less potential for damaging the system
    - Can use sudo and visudo commands

- *Polkit (PolicyKit)*
    - Controls system-wide privileges that allows non-privileged processes to communicate with privileged ones
    - *pkexec command*
        - Used to execute PolKit

            - Example of using pkexec to make a directory named Jason
                - pkexec mkdir /Jason
        - Sudo is easier to use, more flexible, and has better security than pkexec

- **Create, Modify, and Delete Users**
    - useradd command
        - Syntax
            - `useradd [options] [username]`
        - The account created
            - is stored in /etc/passwd file
            - is configured according to options set in the /etc/login.defs file
            - has a home directory is created in the /home/ <account name>
                - the home directory is populated using files from the /etc/skel directory
        - Useradd command does not set a password for the account
        - Options
            - **-c**
                - sets the comment field
            - **-e**
                - sets the expiration date
                    - example: `useradd -e 2021/12/31`

        - **-s**
            - sets the default shell of the user
                - example: `useradd -s /bin/ksh`
        - **-D**
            - used to view default configurations for new users

- *passwd command*
    - Used by the root user to set or reset a password

- /etc/passwd
    - Used to contain the passwords, but it posed a security problem
    - Administrators use this file to gain information about users on a system
    - Seven Fields in each line within /etc/passwd (separated by a colon)
        - Username
            - Contains the name the user use to log into the system
        - Password
            - Contains assigned password to the user
        - User ID
            - Unique number that represents the user to the system
        - Group ID
            - Unique number of a user's primary group membership
        - Comment
            - Often has the full name of the user
        - Home Directory
            - Path to the home directory of the user

- Login Shell
    - Shell that is launched when user logs in (/bin/bash or /bin/ksh)
        - useradd
        - usermod
        - userdel

- /etc/shadow
    - Modern storage location for hashed passwords and additional account information
    - Only root has the access to the content of /etc/shadow file
    - Fields of Information in the /etc/shadow
        - Username
        - Password (Hashed format)
        - Days since password was changed
        - Days before password must be changed
        - Days until user is warned to change password
        - Days after password expires that account gets disabled
        - Days the account has been disabled
        - Unused field that is reserved for future use

- **Create, Modify, and Delete Groups**
    - Users can be members of more than one group
    - /etc/group
        - Storage location for all groups
        - Fields within the /etc/group
            - Group Name
                - User-friendly name of the group
            - Password
                - Password required to enter the group
            - Group ID
                - Reference number on the system
            - Group List
                - Refers to members of the group
    - Commands used to edit the /etc/group file
        - groupadd command
            - creates a group
            - options
                - -g
                    - creates the group ID
                - -f
                    - exits with a success status if the group already exists
                - -o
                    - creates group with non-unique group ID

            - syntax
                - `groupadd [options]{group names}`
                - example, creates a new group called instructors
                    - `groupadd -g instructors`
        - groupmod command
            - Command to change the group's attributes
            - Options
                - -g
                    - Change group ID
                - -n
                    - Rename group
            - Syntax
                - `groupmod [options]{group names}`
        - groupdel command
            - delete groups
            - Groupdel will not delete user accounts within a group, but only delete the group itself
            - Syntax
                - `groupdel [options]{groupname}`

- **Query Users and Groups**
    - whoami command
        - Used to display the username currently logged in to the system
        - To verify the current username, enter the whoami command

- The command prompt will also show the level that a user is logged into in many distributions
    - # in the command prompt means you are logged in as the Root User
    - $ in the command prompt means you are logged in as a Standard User

- who command
    - Used to determine the details of the users currently logged in
    - It includes
        - Username
        - Name of the system
        - Date and time
    - Syntax
        - `who [options]`
    - Options
        - -u
            - time use has been Idle time
            - Results of the who -u command
                - .
                    - the user is active within 1 minute from the time the command was used
                - old
                    - the user has been inactive for more than 24hrs
        - am i (who am i)
            - (User information)

- w command
  - Used to display the details of users that are currently logged in to a system and their transactions
  - Output display
    - First Line
      - Displays the status of the system
    - Second Line
      - Displays a table column list of the users logged in to the system
    - Last Column
      - Indicates the current activities of the users
  - Syntax
    - `w [options] [username]`

- last command
  - Displays the history of user login and logout actions, and the actual time and date
  - Options allow you to filter users by using the number of the terminal
    - Example of filtering users of the first terminal
      - last 1
  - The last command retrieves information from the /var/log/wtmp file
  - Syntax
    - `last [options]`

- id command
  - Used to display user ID (UID) and group ID (GID) information
  - Entering no options displays information about the user who is currently logged in
  - Syntax
    - `id [options] [username]`

- **Account Profiles**
  - .bashrc File
    - Enables customization of the user's own environment
    - Can be customized to:
      - Adapt to specific needs and preferences
      - Create environment variable
      - Set default directories and file permissions
      - Change default command prompt

  - bash_profile File
    - Provides the shell configuration for the initial login environment
    - Sets the profile for all users, not just one
      - it changes the /etc/skel
    - When a new user account is created in /etc/skel/ directory, it is automatically copied into the new user's home directory
    - Files added to the /etc/skel/ directory after a user account is created will not be copied to existing users' home directories

- /etc/profile File
  - Provides system-wide environment variables that are used to apply certain settings to user accounts
  - pulls specification for the system-wide environmental variables and then goes to the variables found in the user accounts found in:
    - ~/.bash_profile
    - ~/.bash_login
    - ~/.profile

- /etc/profile.d Directory
  - Serves as a storage location for scripts that admins may use to set additional system-wide variables
  - Set the environment variables via scripts contained in /etc/profile.d

- /etc/bashrc
  - Provides system-wide configuration changes specific to Bash settings

Permissions and Ownership

Objectives 2.5: Given a scenario, apply the appropriate access controls.

- **File and Directory Permissions**
    - *Permission*
        - Access rights assigned to users that enable them to access or modify files and directories
    - ls -l command
        - gives a list of files and directories in the current working directory
        - Output is in 7 columns
            - Column 1 identifies the item
            - Column 2 displays the number of links
            - Column 3 displays the owner of the file or directory
            - Column 4 displays the group with granted access
            - Column 5 displays lists the size file or directory
            - Column 6 displays the date and time file was created/modified
            - Column 7 displays the name of the file
        - Directory signifiers
            - Directory
                - .
            - Parent Directory
                - . .

- Permissions define what users are allowed to do in a particular file or directory

- Permissions for Files
  - read (r)
    - can access and view the file
  - write (w)
    - can save changes
  - execute (x)
    - can run the script/program/software

- Permissions for Directories
  - read (r)
    - can list the directory content
  - write (w)
    - can create, rename, delete directories
  - execute (x)
    - can access directory, execute file, perform task on directories

- *Contexts*
  - users and entities that permissions are given to
  - Types of contexts
    - owner (u)
      - User
    - group (g)
      - File/directory's group

            - other (o)
                - All other users

    - The output of the ls -l command shows the permission string
        - Has 11 characters
            - 1st Character
                - Shows one of the following:
                    - d for directory
                    - - for file
            - 2nd 3rd 4th Characters
                - Owner permissions
            - 5th 6th 7th Characters
                - Group permissions
            - 8th 9th 10th Characters
                - Other permissions
            - 11th Character will have one of the following
                - . for SELinux security context
                - + for alternative access methods

    - *chmod command*
        - Enables the owner to modify the permissions of a file or directory
        - Syntax
            - `chmod [options] {mode} {file/directory name}`
        - Options
            - -c
                - Report changes

- -f
  - Hide error messages
- -v
  - Diagnostic file entry
- -R
  - Recursively modify permissions

- Chmod modes
  - Symbolic Mode
    - Enables to set permissions using three components
    - Permission Contexts
      - u/g/o/a
        - user/group/other/applies permissions to all three contexts
    - Permission Operators
      - +/-/=
        - +
          - Grants permission
        - -
          - Denies permission
        - =
          - Assigns permission
    - Permission Attributes
      - r/w/x
        - read/write/execute

        - Syntax for Symbolic mode
            - `chmod {access context} {operators} {permission attributes} {file/directory names}`

    - Absolute Mode
        - Uses octal (base-8) numbers to specify permissions
            - 4
                - Read
            - 2
                - Write
            - 1
                - Execute
        - The complete permission is a three-digit number that corresponds to the owner, the group, and others
        - Syntax for Absolute Mode
            - `chmod {number} {file/directory names}`
        - Example
            - 752
                - 7= User position
                    - Read, Write, and Execute permissions
                - 5= Group
                    - Read and Execute permissions
                - 2 = Others
                    - Write permission

            - 541
                - 5= Read and Execute permission
                - 4= Read permission
                - 1= Execute permission

    - umask command
        - Used to set the default permissions for newly created files and folders
        - options
            - -S
                - Current mask as symbolic value
            - -p
                - Current mask in numeric format
        - Syntax
            - `umask [mask]`

    - differences between umask and chmod
        - umask
            - Change default permission for newly created files and folders
        - chmod
            - Set permissions on files and folders that already exist

- **File and Directory Ownership**
    - *Ownership*
        - Refers to a property by which a user can apply and modify the permissions of a file or directory

- Only the superuser (root user) can change the permissions of an object owned by others
- *chown command*
  - Used to change the owner and/or the group of a file or directory
  - Syntax
    - Change the owner, but not the group for file or directory
      - `chown {username} {file/directory name}`
    - Change the owner and the group for file or directory
      - `chown {username}:{group name} {file/directory name}`
    - Change the owner and then assign the group to the new owner's login group
      - `chown {username}: {file/directory name}`
    - Change the group, but not the owner
      - `chown :{group name} {file/directory name}`
  - Option
    - –R
      - recursively change ownership throughout a directory structure

- *chgrp command*
  - Used to change the group ownership of a file or directory
  - Syntax
    - `chgrp {group name} {file/directory name}`

- **Special Permissions and Attributes**
  - Special Permissions
    - The less privileged users are allowed to execute a file by assuming the privileges of the file's owner or group
    - Main Special Permissions
      - *Set user ID (SUID)*
        - User is allowed to have similar permissions as the owner of the file
      - *Set group ID (SGID)*
        - User is allowed to have similar permissions as the group owner of the files and directories
    - Users in a shared environment don't need to change their group

  - To give special permission, use the chmod command in either symbolic mode or absolute mode

  - Determining SUID/SGID
    - ls -la

  - Configuring SUID
    - SUID (Symbolic Mode)
      - `chmod u+s {file names}`
    - SUID (Absolute Mode)
      - `chmod 4### {file names}`

- Configuring SGID
  - SGID (Symbolic Mode)
    - `chmod g+s {directory names}`
  - SGID (Absolute Mode)
    - `chmod 2### {directory names}`

- To remove the SUID or SGID, use the minus (-) operator in symbolic mode, or set to 0 in absolute mode

- *Sticky bit*
  - Special permission bit that protects files in a directory so only the owner or root user can delete the file
  - Setting the sticky bit
    - Symbolic Mode
      - `chmod +t {directory names}`
    - Absolute Mode
      - `chmod 1### {directory names}`
  - Files can have one or more attributes set that define how the system interacts with files

- *Immutable Flag*
  - Attribute of a file or directory that prevents from being modified
  - Immutable flag is useful for files that are highly sensitive and important

- *lsattr command (List Attribute)*
  - Used to list the attributes of a file or directory
  - Syntax
    - `lsattr [options] {file or directory names}`
  - Options
    - -R
      - Recursively lists attributes of directories and content
    - -a
      - Lists all files
    - -d
      - Lists directories
    - -v
      - Version number of the file

- *chattr command (Change Attribute)*
  - Used to change the attributes of a file or directory
  - Syntax
    - `chattr [-R] [-v {version}] [+-{attributes}] {file or directory names}`
  - Options
    - -R
      - Recursively change attributes of directories and content
    - -v
      - Version number of the file
    - +I
      - Read only and immutable

- -I
    - Removes read-only

- Access control list (ACL)
    - ACLs enable a more granular level of control than simply using file permissions
    - *getfacl command* (get file ACL)
        - Useful when retrieving the ACLs of files and directories
    - *setfacl command* (set file ACL)
        - Used to change the permissions associated with the ACL of a file or directory
        - Syntax
        - `setfacl [-bR] [-mx {acl_spec}] {file/directory names}`
        - Common commands
            - -r
                - Recursively set ACL options
            - -s
                - Set ACL
            - -m
                - Modify existing ACL
            - -x
                - Removes entries from existing ACL
            - -b
                - Removes all entries except standard permissions

- ACL (Users)
  - Syntax
    - `u:{user name}:{permissions}`
- ACL (Groups)
  - Syntax
    - `g:{group name}: {permissions}`

- **Troubleshooting Permissions Issues**
  - Troubleshooting
    - Begins with the identification of a problem and ends with service restored
    - Troubleshooting goal is to solve a problem efficiently with a minimal interruption of service
    - Process
      - Identify the problem
      - Establish theory of probable cause
      - Test the theory to determine the cause
      - Establish action plan
      - Implement the solution
      - Verify full system functionality
      - Document findings, actions, and outcomes

  - Use ls -al command to verify the user and group ownership of a file or directory

- *group command*
  - Displays the groups that a user belongs to
  - syntax
    - `groups {user name}`

- usermod command
  - changes group membership

- Some distributions have a command that allows the display of all members of a group
  - lid command
  - libuser-lid command

- *getent command*
  - Enables to retrieve group members of non-standard authentication methods

- When troubleshooting permissions
  - Follow overall Troubleshoot strategy
  - Verify permissions and ownership
  - Verify special permissions are set properly
  - Ensures proper owner and owning group set

Storage

Objectives:

- 1.1: Summarize Linux fundamentals.
- 1.2: Given a scenario, manage files and directories.
- 1.3: Given a scenario, configure and manage storage using the appropriate tools.
- 4.1: Given a scenario, analyze and troubleshoot storage issues.

- **Partitions**
    - Linux supports a variety of storage devices including
        - Hard disk drives
        - Solid-state devices
        - USB thumb drives
        - External storage drives
    - Linux refers to devices as either
        - Block Devices
            - Read/write in blocks of data
            - examples: hard drives, solid-state devices
        - Character Devices
            - Read/write in character streams of data
            - examples: keyboards, mice, serial ports
    - *File System*
        - A data structure is used by an operating system to store, retrieve, organize, and manage files and directories on storage devices
        - Types of file systems supported by Linux

- File Allocation Table (FAT)
  - An older file system compatible with different operating systems
- ext2
  - Used to be the native Linux file system of some older releases
- ext3
  - Much faster in recovering data and better ensures data integrity in abrupt system shutdowns
- ext4
  - Supports volumes up to one exabyte and files up to 16 terabytes in size
- XFS
  - A 64-bit, high-performance journaling file system that provides fast recovery and can handle large files efficiently
- BTRFS
  - Supports volumes of up to 16 exabytes in size and up to 18 quintillion files on each volume

▪ File systems that function as network protocols as well
  - enables the sharing of data over a network
  - Include
    - Server Message Block (SMB)
    - Common Internet File System (CIFS)
    - Network File System (NFS)

        - Windows supports SMB by default and doesn't offer NFS support by default

- *Index Node (Inode)*
    - Stores metadata about a file or directory on a file system
    - Journaling process includes
        - Changes to be made
        - Background processes
        - Pending changes after reboot
        - Incomplete entries

- Virtual File System
    - A software interface that sits between the kernel and the real file system
        - it translates details between the file system and the kernel
        - it allows for multiple file systems to be on a drive
    - File system labels are used for easy identification of file systems
        - may be up to 16 characters long
        - Types of labels
            - e2label
                - ext-based file systems
            - xfs_admin
                - XFS-based file systems

- *Partition*
    - A section of the storage drive that logically acts as a separate drive
    - Partition Types
        - *Primary*
            - Contains one file system or logical drive and is sometimes referred to as a volume
            - Swap file system and Boot partition are created in the primary partition
        - *Extended*
            - Contains several file systems, which are referred to as logical drives
        - *Logical*
            - Partitioned and allocated as an independent unit and functions as a separate drive

- *fdisk utility*
    - Used to create, modify, or delete partitions on a storage drive
    - Options
        - -b {sector size}
            - Specify number of drive sectors
        - -H {heads}
            - Specify number of drive heads
        - -S {sectors}
            - Specify number of sectors per track
        - -s {partition}
            - Print partition size in blocks

                - -l
                    - List partition tables for devices
            - Options in the in the fdisk menu
                - n
                    - Create new partition
                - d
                    - Remove partition
                - p
                    - List existing partitions
                - w
                    - Write drive changes and exit utility
                - q
                    - Cancel changes made and exit utility
        - *parted utility*
            - Used to create, destroy, and resize partitions and runs the GNU Parted utility
            - Menu options
                - select
                    - Choose device or partition to modify
                - mkpart
                    - Create partition with file system type specified
                - print
                    - List partition table

        - resizepart
            - Resize or modify a partition's end position
        - rm
            - Delete a partition
        - quit
            - Quit GNU Parted utility

- *partprobe command*
    - Used to update the kernel with changes that now exist within the partition table

- *mkfs command*
    - Used to build a Linux file system on a device, which is usually a drive partition
    - Options
        - -v
            - Produce verbose output that keeps changing as the program processes
        - -V
            - Produce verbose output, including all file system-specific commands executed
        - -t {fs type}
            - Specify type of file system to build
        - fs -options
            - Pass file system-specific options to the file system builder

        - -c
            - Check the device for bad blocks before building the file system
        - -l {filename}
            - Read the list of bad blocks from a specified file
    - Syntax possibilities
        - `mkfs [options] {device name}`
        - `mkfs {file system type} [options] {device name}`

- fstab File
    - Stores information about storage devices and partitions and where and how they should be mounted
    - Fields in each line
        - Device/Partition Name
            - Name of the device or file system to mount
        - Default Mount Point
            - Where the file system is to be mounted
        - File System Type
            - Type of file system used by the device or partition
        - Mount Options
            - Set of comma-separated options that will be activated when the file system is mounted
        - Dump Options
            - Indicates if the dump utility should back up the file system

        - fsck Options
            - Order in which the fsck utility should check file systems

- /etc/crypttab File
    - Stores information about encrypted devices and partitions that must be unlocked and mounted on system boot

- To set up a storage devices requires you to
    - Partition storage device
    - Format partition with a file system
    - Add formatted partition to fstab file

- */dev Directory*
    - A special file that contains details about all the files and subdirectories housed within it
    - Naming convention
        - Example:
            - /dev/sda1
                - sd= type of controller
                - a=first whole drive
                - 1=first partition
    - Persistent naming schemes to help identify devices
        - Naming scheme based on the device's hardware serial number
            - /dev/disk/by-id

        - Naming scheme based on the shortest physical path to the device
            - /dev/disk/by-path
        - Naming scheme based on the device's Universally unique identifier (UUID)
            - /dev/disk/by-uuid

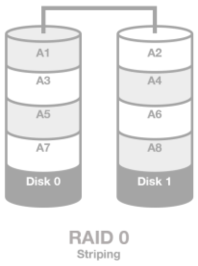
    - Special Character devices
        - /dev/null
            - A special type of virtual device that discards anything you send or redirect into it
            - It will disappear forever
        - /dev/zero
            - A special type of virtual device that returns a null character anytime you read from it
            - dev/zero will send back the ASCII null character of 0x00
            - Useful for sanitizing a drive
        - /dev/urandom
            - A special type of virtual device that returns a randomized series of pseudorandom numbers

- **Logical Volumes**
    - *Device Mapper*
        - Creates virtual device and passes data from that virtual device to one or more physical devices

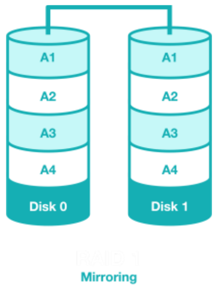
- *DM-Multipath*
  - Provides redundancy and improved performance for block storage devices
  - The configuration file for the multipath-tools package is found at /etc/multipath.conf

- *mdadm tool*
  - A tool used to create and manage software-based RAID arrays

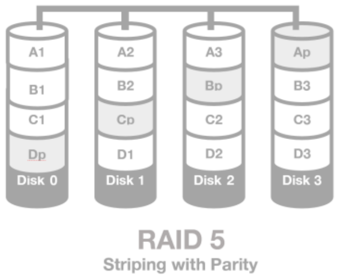
- *RAID*
  - Redundant Array of Independent or Inexpensive Disks
  - Key Terms
    - *Striping*
      - Combines multiple smaller physical disks to logically act as a single larger disk
    - *Mirroring*
      - Combines two physical hard drives into a single logical volume where an identical copy of everything is put on both drives
    - Parity
      - Used in RAID drive arrays for fault tolerance by calculating the data in two drives and storing the results on a different drive
  - Types
    - RAID 0
      - great for speed but provides no data redundancy



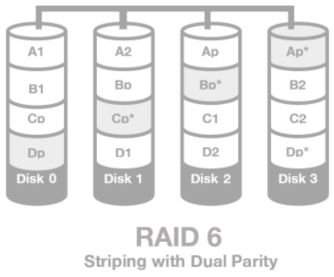
- RAID 1
    - full redundancy, but no increase in speed and loss of space



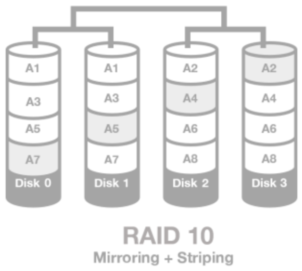
- RAID 5
    - parity, full redundancy, less loss of space



- RAID 6
    - double parity, full redundancy



- RAID 10
    - striping and mirroring, full redundancy and speed



- /proc/mdstat File
    - Contains a snapshot of the kernel's RAID/md state
    - to view state of the RAID
        - `cat /proc/mdstat`
        - Sample results:

```
Personalities : [raid6] [raid5] [raid4]
md0 : active raid5 sda1[0] sdd1[2] sdb1[1]
   1465151808 blocks level 5, 64k chunk,
   algorithm 2 [4/3] [UUU_]
    unused devices : <none>
```

- *Logical Volume Manager (LVM)*
  - Maps whole physical devices and partitions into one or more virtual containers called volume groups
  - Allows you to:
    - Dynamically create, delete, and resize volumes
    - Map logical volumes across physical devices
    - Create virtual snapshots of each logical volume
  - /dev/mapper/ Directory
    - contains all of the logical volumes on a system
    - Address:
      - `/dev/mapper/<volume group name >-<logical volume name>`

- Physical volume tools
  - *pvscan*
    - Scans for all physical devices being used as physical volumes
  - *pvcreate*
    - Initializes a drive or partition to use as a physical volume
  - *pvdisplay*
    - Lists attributes of physical volumes
  - *pvchange*
    - Changes attributes of a physical volume
  - *pvs*
    - Displays information about physical volumes
  - *pvck*
    - Checks the metadata of physical volumes

- *pvremove*
  - Removes physical volumes

- Volume Group Tools
  - *vgscan*
    - Scans all physical devices for volume groups
  - *vgcreate*
    - Creates volume groups
  - *vgdisplay*
    - Lists attribute of volume groups
  - *vgchange*
    - Changes attributes of volume groups
  - *vgs*
    - Displays information about volume groups
  - *vgck*
    - Checks the metadata of volume groups
  - *vgrename*
    - Renames a volume group
  - *vgreduce*
    - Removes physical volumes from a group to reduce its size
  - *vgextend*
    - Adds physical volumes to volume groups
  - *vgmerge*
    - Merges two volume groups
  - *vgsplit*
    - Splits a volume group into two

        - *vgremove*
            - Removes volume groups

- Logical Volume Tools
    - *lvscan*
        - Scans all physical devices for logical volumes
    - *lvcreate*
        - Creates logical volumes in a volume group
    - *lvdisplay*
        - Lists attributes of logical volumes
    - *lvchange*
        - Changes attributes of the volumes
    - *lvs*
        - Displays information about logical volumes
    - *lvrename*
        - Renames logical volumes
    - *lvreduce*
        - Reduces the size of logical volumes
    - *lvextend*
        - Extends the size of logical volumes
    - *lvresize*
        - Resizes logical volumes
    - *lvremove*
        - Removes logical volumes

- **Mounting File Systems**
    - *Mount Point*
        - An access point that is typically an empty directory where a file system is loaded or mounted to make it accessible to users

    - *mount command*
        - Loads a file system to a specified directory to make it accessible to users and applications
        - Syntax
            - `mount [options] {device name} {mount point}`
        - Options
            - auto
                - Device must be mounted automatically
            - noauto
                - Device should not be mounted automatically
            - nouser
                - Only the root user can mount a device or a file system
            - user
                - All users can mount a device or a file system
            - exec
                - Allow binaries in a file system to be executed
            - noexec
                - Prevent binaries in a file system from being executed
            - ro
                - Mount file system as read only

            - rw
                - Mount file system with read/write permissions
            - sync
                - Input and output operations should be done synchronously
            - async
                - Input and output operations should be done asynchronously

- *Binary*
    - A source code that is compiled into an executable program

- *umount command*
    - Disassociates a mounted file system from the directory
    - Syntax
        - `umount [options] {mount point}`
    - Options
        - -f
            - Force unmount a file system
        - -l
            - Perform a "lazy" unmount
        - -R
            - Recursively unmount specified directory mount points
        - -t {fs type}
            - Unmount only the file system types specified

            - -O
                - Unmount only the file systems with specified options in the /etc/fstab file
            - -fake
                - Test the unmounting procedure

    - *fstab command* (File System Table)
        - Lists file systems to be mounted, their mount points, and any options that might be needed for specific file systems

    - *systemd.mount*
        - can be used to create a new mount unit to mount the file system

    - *FUSE (Filesystem in USErspace)*
        - Lets non-privileged users create own file systems without editing the underlying kernel code

- **Managing File Systems**
    - */etc/mtab File*
        - Reports the status of currently mounted file systems
        - /proc/mounts
            - more accurate and includes more up-to-date information on file systems

    - */proc/partitions File*
        - Contains information about each partition attached to the system

        - Formatted in columns
            - major
                - Class of device
            - minor
                - separates partition into physical devices
            - #blocks
                - number of physical blocks the partition takes up
            - name
                - name of the partition

    - *lsblk command*
        - Displays information about block storage devices currently available on the system
        - Syntax
            - `lsblk [options] [device name]`
        - Options
            - -a
                - List empty devices
            - -r
                - List devices excluding provided output devices
            - -f
                - Display additional information
            - -l
                - Display results in list format
            - -m
                - Display device permission information

- *blkid command*
  - Prints each block device in a flat format and includes some additional information
  - Syntax
    - `blkid [options] [device name]`

- Some tools are designed to only work with specific file system types
  - Tools that work with any generation of the ext file system type
    - e2fsck
    - resize2fs
    - tune2fs
    - dumpe2fs

- *fsck command*
  - Used to check the correctness and validity of a file system
  - Syntax
    - `fsck [options] {device/file system name}`
  - Syntax to repair the file system
    - `fsck -r {device/file system name}`

- *resize2fs command*
  - Used to resize ext2, ext3, or ext4 file systems
  - Syntax
    - `resize2fs [options] {device/file system name} [desired size]`

- *tune2fs command*
  - Used to adjust various tunable parameters of the ext2/ext3 file systems
  - tune2fs can also add a journal to an existing ext2 or ext3 file system
  - Syntax
    - `tune2fs [options] {device/file system name}`
  - Options
    - -j
      - Used as an ext3 journal to the existing file system
    - -i {d|m|w}
      - Specify the maximum time interval
    - -c {maximum mounts count}
      - Specify the maximum number of mounts
    - -C {mount count}
      - Specify the number of possible mounts
    - -r {reserved blocks count}
      - Specify the number of reserved file system blocks
    - -e {continue|remountro|panic}
      - Specify the behavior of the kernel code
    - -l
      - List the contents within the superblock
    - -U
      - Set the specified UUID

- Superblock
    - Contains metadata about the file system, including its size, type, and status

- *dumpe2fs command*
    - Prints the superblock and block group information for the selected device
    - Syntax
        - `dumpe2fs [options] {device/file system name}`
    - Options
        - -x
            - Print a detailed report about block numbers
        - -b
            - Print the bad blocks
        - -f
            - Force display the file system status
        - -i
            - Display file system data from an image file created using the e2image command
        - xfs_info
            - Display details about the XFS file system
        - xfs_admin
            - Change the parameters of an XFS file system
        - xfs_metadump
            - Copy the superblock metadata of the XFS file system to a file

- xfs_growfs
  - Expand the XFS file system to fill the drive size
- xfs_copy
  - Copy the contents of the XFS file system to another location
- xfs_repair
  - Repair and recover a corrupt XFS file system
- xfs_db
  - Debug the XFS file system

- *lsscsi command*
  - Used to list information about SCSI devices connected to a Linux system

- *fcstat command*
  - Interacts with and displays statistics of Fibre Channel connected devices

- **Linux Directory Structure**
  - Types of Files
    - *Directories*
      - Containers for other files
    - *Special Files*
      - System files stored in the /dev directory
    - *Links*
      - Make a file accessible in multiple parts of the system's file tree

    - *Domain Sockets*
        - Provide inter-process networking that is protected by the file system's access control
    - *Named Pipes*
        - Enable processes to communicate with each other without using network sockets

- *Filesystem Hierarchy Standard (FHS)*
    - Specifies a set of guidelines for the names of files and directories and their locations on Linux systems
    - Standardized Subdirectories
        - /bin
            - Stores essential command-line utilities and binaries
        - /boot
            - Stores the files necessary to boot the Linux operating system
        - /dev
            - Stores hardware and software device drivers
        - /etc
            - Stores basic configuration files
        - /home
            - Stores users' home directories, including personal files
        - /lib
            - Stores shared program libraries required by the kernel, command-line utilities, and binaries

- /media
  - Stores mount points for removable media such as CD-ROMs
    and floppy disks
- /mnt
  - Refers to the mount point for temporary mounting file systems
- /opt
  - Stores optional files for large software packages
- /proc
  - Represents continually updated kernel information to the user in a typical file format
- /root
  - Refers to the home directory of the root user
- /sbin
  - Stores binaries used for completing the booting process which are also used by the root user
- /sys
  - Stores information about devices
- /tmp
  - Stores temporary files that may be lost on system shutdown
- /usr
  - A read-only directory that stores small programs and files accessible to all users

        - /var
            - Stores variable files, or files that are expected to constantly change as the system runs

- *Current Working Directory (CWD)*
    - The location on the system being accessed at any point in time
    - CWD is represented as a single period (.)

- *Parent Directory*
    - One level above the current working directory
    - Use the double period notation (..) to switch to the parent directory

- *Path*
    - Specifies a location in the file system
    - Path types
        - *Absolute Path*
            - The path to the specific location irrespective of the CWD or combined paths
        - *Relative Path*
            - The path relative to the current working directory

- File Navigation Commands
    - *cd command*
        - Traverse the directory structure using absolute or relative paths to change the current working directory

- *ls command*
    - List the files and directories in the current working directory or the relative/absolute path specified

- *pwd command*
    - Prints the current working directory to the console

Files and Directories

Objectives:

- 1.2: Given a scenario, manage files and directories.
- 3.1: Given a scenario, create simple shell scripts to automate common tasks.
- 4.1: Given a scenario, analyze and troubleshoot storage issues.

- **Create and Edit Text Files**
  - *Text Editor*
    - Application that enables users to view, create, or modify the contents of text files
    - Some text editors do not support formatting options that word processors do
    - Text editors
      - vi
        - Visual text editor originally created for Unix and was later cloned into FOSS versions
      - Vim
        - Default text editor in most Linux distributions
      - Emacs
        - Flexible, powerful, and popular text editor used in Unix and Linux
      - gVim
        - Graphical version of the Vim editor
      - gedit
        - Simple and powerful GUI-based text editor used in the GNOME desktop environment

        - GNU nano
            - Small and user-friendly text editor

- Vim, a contraction of Vi and improved, and an extended version of the vi editor
    - Features
        - Text completion
        - Syntax highlighting
        - Spell checking
        - Vim supports multiple files being opened simultaneously
    - Vim screen commands
        - Vertical Split Screen
            - ctrl+w+v
        - Horizontal Split Screen
            - ctrl+w+s
    - Modes
        - Insert Mode
            - Enables users to insert text by typing into the system
        - Execute Mode
            - Enables users to execute commands within the editor
        - Command Mode
            - Enables users to perform different editing actions using single keystrokes
        - Visual Mode
            - Enables users to highlight or select text for copying and deleting

- Vim commands
    - i
        - Insert text to the left of the cursor
    - A
        - Insert mode and add text at the end of a line
    - I
        - Insert mode and insert text at the beginning of a line
    - o
        - Insert mode and insert text on a new line below the cursor
    - O
        - Insert mode and insert text on a new line above the cursor
    - v
        - Visual mode to enable selection, one character at a time
    - V
        - Visual mode to enable selection, one line at a time
    - :
        - Execute mode to enable users to enter commands
    - Esc
        - Return to command mode

- Vim commands with the colon operator
  - When you enter the colon (:) operator, a small command prompt section appears at the bottom-left of the editor
  - :w {file name}
    - Save file with a file name if it's saved for the first time
  - :q
    - Quit when no changes are made after the last save
  - :q!
    - Quit while ignoring the changes made
  - :qa
    - Quit multiple files/ quit all
  - :wq
    - Write the file first and quit
  - :e!
    - Revert to last saved format without closing the file
  - :! {any Linux command}
    - Execute the command and display the result in the Vim interface
  - :help
    - Open Vim's built-in help documentation

- Vim Motions
  - *Motions*
    - are single-key shortcuts that are used to navigate through files in command mode
  - Useful motions
    - h
      - Move left one character
    - j
      - Move down one line
    - k
      - Move up one line
    - l
      - Move right one character
    - ^
      - Move to the beginning of the current line
    - $
      - Move to the end of the current line
    - w
      - Move to the next word
    - b
      - Move to the previous word
    - e
      - Move to the end of the current word
    - Shift+L
      - Move the cursor to the bottom of the screen

    - Shift+H
        - Move the cursor to the first line of the screen
    - (Line no.) Shift+G
        - Move cursor to specified line no.
    - gg
        - Move the cursor to the first line of the file
    - Shift+G
        - Move the cursor to the last line of the file

- Vim commands to edit more than one character at a time
    - x
        - Delete the character selected by the cursor
    - d
        - Delete text
    - dd
        - Delete the current line
    - p
        - Paste text on the line below the cursor
    - P
        - Paste text on the line above the cursor
    - / {text string}
        - Search through the document for specific text
    - ? {text string}
        - Search backward through document for specific text
    - y
        - Copy text

- yy
  - Copy the line directory above the cursor
- c{range of lines}c
  - Begins a change in the specific range
- u
  - Undo the latest change
- U
  - Undo all changes in the current line
- ZZ
  - Write a file only if changes are made, then quit editor

▪ Counts in Vim

- *Count*
  - Number that multiplies the effect of keystrokes in Vim
- When Count is used with a Motion
  - Multiplied according to count specified
- When Count is used with an Editing operator
  - Repeated the number of times specified

▪ Drawbacks to Vim

- GNU nano
    - Features
        - Visually helpful
        - Does not have different modes
        - Support multiple open files
    - nano command
        - opens a file for editing
            - if the file doesn't exist, nano will create the file using the name specified
        - Syntax
            - `nano {file name}`
    - shortcuts
        - The functions used to work with text files and the editor
        - Useful shortcuts
            - Ctrl+G
                - Open nano to the help screen
            - Ctrl+X
                - Exit nano or close the current "buffer"
            - Ctrl+O
                - Save currently open file
            - Ctrl+J
                - Justify the current paragraph
            - Ctrl+R
                - Insert another file into the current file
            - Ctrl+W
                - Search the file

- Ctrl+K
  - Cut the currently selected line
- Ctrl+U
  - Paste the line that was cut
- Ctrl+C
  - Display the cursor's position
- Ctrl+V
  - Navigate to the next page
- Ctrl+Y
  - Navigate to the previous page

- Copying
  - "Mark" to copy the part of text on a line using the Ctrl+^ shortcut
  - Once a section is marked, use
    - Alt+^
      - Copy the marked/highlighted text
    - Ctrl+U
      - Paste text to another location

- Gedit text editor
  - has a GUI with a menu-based design that makes it easy to work with
    - Syntax highlighting
    - Spell checking
    - Customized plugins

- **Search for Files**
  - *locate command*
    - Performs a quick search for any specified file names and paths stored in the mlocate database
    - This database must be updated regularly for the search to be effective
    - Syntax
      - `locate [options] {string}`
    - Options
      - -r
        - Search file names using regular expressions
      - -c
        - Display the number of matching entries found
      - -e
        - Return only files that exist at the time of search
      - -I
        - Ignore the casing in file names or paths
      - --n{number of entries}
        - Return the first few matches up to the specified number

  - *updated command*
    - Used to build a database of files based on the /etc/updatedb.conf file
    - Updatedb used to update the /var/lib/mlocate/mlocate .db database file
      - open /etc/updatedb.conf File
      - Look for the *PRUNEPATH variable*

        - Used to specify a path that need not be included while building the database
    - Specify which paths should not be included
        - example:
            - `PRUNEPATH="/etc‟`
            - will not include the /etc files

- *find command*
    - Enables users to search specific location for files and directories that adhere some search criteria

- Options
  - -type
    - Specifies the type of object you are looking for
      - d
        - Directory
      - f
        - File
  - -name
    - Specifies the name of the object you are looking for
- Difference between locate and find commands
  - Locate command searches the database and retrieves information on files present on the system
  - Find command performs a live search of the file system and in a specific location
- Additional find command options
  - -print
    - Displays the location of the files found
  - -exec
    - Executes the command that follows
  - -ok
    - Executes the command that follows interactively
  - -delete
    - Deletes the files found
  - -fprint
    - Stores the results in the target file

- *which command*
  - Displays the complete path of a specified command by searching the directories assigned to the PATH variable
  - Syntax
    - `which [options] {program names}`

- *whereis command*
  - Used to display various details associated with a command
  - Syntax
    - `whereis [options] [directory name] {file name}`
  - Options
    - -b
      - Search only for binaries
    - -m
      - Search only for manual sections
    - -s
      - Search only for sources
    - -u
      - Search for unusual entries

- **Operations on Files and Directories**
    - *cat command*
        - Can display, combine, and create text files
        - Cat command does not have a screen scrolling capability
        - Options
            - -n
                - Precede the output with its respective line number
            - -b
                - Number the lines, excluding the blank lines
            - -s
                - Suppress output of repeated empty lines
            - -v
                - Display non-printing characters as visible characters
            - -e
                - Print a $ character at the end of each line, prior to the new line
            - -t
                - Print tabs as ^I and form feed as ^L
        - Syntax
            - `cat [options] {file names}`

    - Head and Tails Commands
        - *head command*
            - Displays the first 10 lines of each file
        - *tail command*
            - Displays the last 10 lines of each file

        - Syntax
            - head [options] {file names}
            - tail [options] {file names}
        - Option for tail
            - -f
                - Dynamically watches a file
        - Options for both head and tail
            - -n {number}
                - Shows specified number of lines

    - *less and more Commands*
        - Enable users to display the contents of a file and a page through the contents if extended beyond the screen
        - Less is used by most people, although they are similar
        - Syntax
            - `less [options] {file names}`
            - `more [options] {file names}`
        - less command Options
            - -e
                - Exit the program the second time it reaches the end of the file
            - -E
                - Exit the program the first time it reaches the end of the file
            - -I
                - Ignore the case in searches

            - -n
                - Suppress line numbers
        - less command Navigation
            - /
                - Search a file for a particular text string
            - n or N
                - Move to the next or previous instance of the searched string
            - q
                - Quit the program

- *cp command*
    - Enables users to copy and then paste a file or directory
    - Options
        - -R
            - Copy specified directory recursively
    - Syntax
        - `cp [options] {file/directory name you want to copy} {file/directory name for the destination}`

- *mv command*
    - Moves files and directories to other locations
    - mv is more like a cut and paste operation
    - can be used to rename files and directories

        - Syntax
            - `mv [options] {file/directory name to be moved} {file/directory name of destination}`

    - *touch command*
        - Tests the permissions or creates files that will be processed by some applications

    - *rm command*
        - Removes files and directories
        - Option
            - -R
                - Recursively remove files, subdirectories, and the parent directory
                - Example:
                    - `rm -R ~/myfiles`
                        - Remove files and directories
        - Syntax
            - `rm [options] {file/directory names}`

    - *unlink command*
        - Used to remove files but not directories

- *ls command*
  - Options
    - -l
      - Display permission list, number of hard links, owner, group, size, date, and file name
    - -F
      - Display the nature of a file
    - -a
      - Display all files present in the directory
    - -R
      - Recursively display all subdirectories
    - -d
      - Display information about symbolic links or directories
    - -L
      - Display all files in a directory, including symbolic links
  - Syntax
    - ls [options] {file/directory names}
  - Normal Text Default Colors for ls command results
    - Blue (Directory)
    - Skyblue (Symbolic link and audio file)
    - Green (Executable file)
    - Yellow with Black (Device)
    - Pink (Image file)
    - Red (Archive file)
    - Red with Black (Distinguishes broken link)

- *mkdir command*
  - Used to create (or make) a directory
- *rmdir command*
  - Removes empty directories
  - Option
    - rm -R
      - Delete directory with contents

- **Process Text Files**
  - *echo command*
    - Built-in Linux feature that prints out arguments as the standard output
    - Syntax
      - `echo {string}`
  - *printf command*
    - Provides the user with more control over how the output is formatted
    - Formatting character
      - \
        - Indicate when character are being used
  - *tr command*
    - Perform operations like removing repeated characters, converting uppercase to lowercase, and basic character replacement and removal
    - Syntax
      - `tr {character 1} {character 2}`

- wc command
    - Allows users to count the number of lines, words, characters, and bytes in file and print the result
    - Options
        - -c
            - Display byte count
        - -m
            - Display character count
        - -l
            - Display the newline count
        - -w
            - Display the word count

- *sort command*
    - Command line utility for sorting lines of text files
    - Options
        - -k {column numbers}
            - Specify filed values
        - –k2
            - Indicates second field
        - -n
            - Compares and sorts lines based on the string numerical value
        - -r
            - Sort fields in descending order

            - -t
                - Separate one field from another

    - *cut command*
        - Extracts the specified lines of text from a file
        - Options
            - -c
                - Specify the number of the character to cut from each line
            - -d {delimiter}
                - Separate one field from another
            - -f {field numbers}
                - Specify the field numbers to cut on as separated by the delimiter
            - -f2
                - Field between the first and second instances of the delimiter
            - -s
                - Suppress a line if the delimiter is not found

    - *paste command*
        - Used to merge lines from text files horizontally
        - Paste command uses a tab space delimiter to separate each column
        - Option
            - -d
                - Specify different delimiter

- *diff command*
  - Used to compare text files
  - Symbols to use with diff command
    - <
      - Line should be removed from the first file
    - >
      - Line should be added from the second file
  - Denotes the line numbers for each file that would be affected by deletion, addition, and change operations
  - Syntax
    - diff {file name 1} {file name 2}
  - Options
    - -b
      - Ignore spacing differences
    - -i
      - Ignore case differences
    - -t
      - Expand tab characters in output lines
    - -w
      - Ignore spacing differences and tabs
    - -c
      - Display a list of differences with three lines of context
    - -u
      - Output results in unified mode

- *grep command*
  - Used to search the contents of a file for a particular string of text
  - Syntax
    - grep [options] {search pattern} {file names}
  - Options
    - -E {pattern}
      - Match a pattern as an extended regular expression
    - -F {pattern}
      - Match a pattern as a list of fixed strings
    - –f {file name}
      - Match patterns contained in a specified file
    - -i
      - Ignore casing
    - -v
      - Output only lines that don't match the provided pattern
    - -c
      - Print only the number of matching lines
    - -l
      - Print only the files that have matching lines
    - -o
      - Print only the matching part of a line
  - Use grep to search a directory to locate a certain file

- *awk command*
  - Performs pattern matching on files
  - Awk keyword is followed by the pattern, the action to be performed, and the file name
  - Ways awk can process texts
    - Extracting text matching a certain pattern
    - Deleting text matching a certain pattern
    - Adding text matching a certain pattern
  - Awk scripts user can provide patterns with blocks of code
  - Syntax
    - awk [options] [‘patterns {actions}’] {file names}
  - Patterns that can be used in awk scripts
    - /regular_expression/
      - Retrieves all the records beginning with “a”, “b”, or “c”
    - relational_expression
      - Retrieves all the records containing the value “abc” in the first field
    - pattern_1 && pattern_2
      - Retrieves all the records that contain the value “abc” in the first field and the second field contains the value “01”
    - pattern_1 || pattern_2
      - Retrieves records that satisfy the condition that the first field contains or the second field contains or both

- pattern_1 ? pattern_2 : pattern_3
  - Evaluate and match pattern 1 to pattern 2 and pattern 3, then the record will print on the screen
- pattern_1, pattern_2
  - Prints a range of records from the record in the first field and goes in the second field

- *sed command*
  - stream editor
  - can use to modify text files according to various parameters
  - Options
    - d
      - (Delete the lines that match a specific pattern/line number)
    - -n,p
      - (Print only the lines that contain the pattern)
    - s
      - (Substitute the first occurrence of the string in the file)
    - s,g
      - (Globally substitute the original string with the replacement string for each occurrence)

- *ln command*
    - Used to create a link to a file
    - Any changes to the link will reflect in the target file
    - Options
        - -backup
            - Back up existing destination files
        - -f
            - Remove existing destination files
        - -s
            - Make symbolic links
        - -i
            - Prompt to remove destination files
        - -v
            - Print the name of a file before linking
    - Types of links
        - Hard Link
            - Reference to another file
            - If the original file or directory is deleted after a hard link is created, the contents are still available
        - Symbolic Link
            - Reference to a file/directory that can span multiple file systems
            - If the original file or directory is deleted after a symbolic link is created, the contents are lost

- **Manipulate File Output**
  - *Text Stream*
    - Sequence of lines of text that can be leveraged to read or write to a particular device or system component
  - *stdin command*
    - Standard Input
    - Acts as the source for command input
  - *stdout command*
    - Standard Output
    - Acts as the destination for command output
  - *stderr command*
    - Standard Error
    - Used as the destination for error messages
  - Direct the standard input, output, and error using
    - >
      - Redirect the standard output to a file
    - >>
      - Append standard output to the end of the destination file
    - 2>
      - Redirect the standard error message to a file

    - 2>>
        - Append standard error message to the end of the destination file
    - &>
        - Used to redirect standard output and the standard error message to a file
    - <
        - Used to read the input from a file rather than from the keyboard or mouse
    - <<
        - Used to provide input data from the current source and top when a line containing the provided string occurs
        - Here Document
            - Refers to a special block of code
    - |
        - pipe
        - Lets users use commands such that the output of one command serves as input to the next
        - Pipes help users to mash-up two or more commands at the same time and run them consecutively

- *xargs command*
    - Reads streams of data from standard input, then generates and executes command lines
    - Syntax

        - `command [options] [arguments] | xargs [options] {command}`
        - `example: find /foo -type f -name "*.pdf" | xargs rm`
            - The find command searches all files in /foo that have a .pdf extension, then pipes the result to the xargs command
    - Options
        - -l {replacement string}
            - Consider each line as a single argument
        - -L {number of lines}
            - Read specified number of lines and cat in one long string
        - -p
            - Prompt the user before each command
        - -n {number of arguments}
            - Read the maximum number of arguments and insert at the end of the command template
        - -E {end of string}
            - Represent the end of the standard input
        - -t
            - Write command to standard error output before executing the command
        - -s {max size}
            - Set maximum allowable size of an argument list

- *tee command*
  - Reads the standard input, sends the output to the default output device, and copies the output to each specified file
  - Options
    - -a
      - Append output
  - Syntax
    - `command [options] [arguments] | [options] {file names}`
    - `Example: ls -l | tee listing.txt`
  - Whatever is written in /dev/null will be discarded and forgotten

Kernel Modules

Objective 1.7: Given a scenario, manage software configurations.

- **The Linux Kernel**
  - Kernel Basics
    - *Kernel*
      - The core of an operating system
      - The kernel handles
        - System initialization
        - Process Scheduling
        - Memory and hardware management
      - The kernel manages
        - File system access
        - Memory
        - Processes
        - Devices
        - Resource allocation of a system
    - Two major divisions of the virtual memory to promote greater stability and security
      - *Kernel Space*
        - Where the kernel executes services
      - *User Space*
        - Area of memory outside the kernel space
    - Kernel Architecture Classification
      - *Monolithic Kernel*

            - All system modules, such as device drivers or file systems, run in kernel space
        - *Microkernel Architecture*
            - Kernel runs the minimum amount of resources necessary to implement a fully functional operating system
            - Has a small kernel space and much larger user spaces
    - *Device Driver*
        - Enables operating systems to identify the characteristics and functions of a hardware device

- *Linux Kernel*
    - Free and open-source monolithic kernel that manages all other resources on an operating system
    - Useful features
        - Virtual memory management
        - Support for TCP/IP networking
        - Shared libraries
        - Modularity
    - Version naming convention
        - For versions 2.6.39 and prior
            - w.x.y.z
                - w – major version
                - x – major revision to the version
                - y – minor revision
                - z – patch number

        - For versions 3.0 and after
            - Use x.xx rather than larger decimals to make things more readable
        - *uname*
            - Prints the name of the kernel
            - Options
                - -r
                    - View kernel version number of the current system
                - -i
                    - View the hardware platform
                - -a
                    - Print all information

- Kernel Layers
    - *System Call Interface (SCI)*
        - Handles system calls sent from user applications to the kernel
    - *Process Management*
        - Handles different processes by allocating separate execution space on the processor
    - *Memory Management*
        - Manages the computer's memory
    - *File System Management*
        - Manages the filesystem

        - *Virtual File System (VFS)*
            - Provides an abstract view of the underlying data that is organized under complex structures
        - *Device Management*
            - Manages devices by controlling device access and interfacing between user applications and hardware devices on the computer

- **Kernel Modules**
    - Linux kernel is loaded into memory by the boot loader
    - /usr/lib/modules/
        - Contains the modules of different kernel versions
        - Subdirectories
            - arch
                - Contains modules for the architecture-specific support
            - crypto
                - Contains modules for encryption and other cryptographic functions
            - drivers
                - Contains modules for various types of hardware
            - fs
                - Contains modules for various types of file systems
            - net
                - Contains modules for networking components

- Commands
  - *lsmod command*
    - Used to display the currently loaded kernel modules
  - *modinfo*
    - Used to display information about a particular kernel module
    - Syntax
      - `Modinfo [options] {module name}`
  - *insmod*
    - Used to install a module into the currently running kernel
    - Syntax
      - insmod [module name]
  - *rmmod*
    - Used to remove a module from the currently running kernel
    - Syntax
      - rmmod [module name]
  - *modprobe*
    - Used to add or remove modules from a kernel
    - Options
      - -a
        - Add a module
      - -r
        - Unload a module
      - -f
        - Force a module to be inserted or removed
      - -n
        - Conduct a dry run

        - -s
            - Print errors to the system log
        - -v
            - Enable verbose mode
    - Syntax
        - modprobe [options] [module names]
- *depmod*
    - Used to update database of dependencies
    - Depmod command searches the contents of /lib/modules/ for each module
    - Syntax
        - depmod [options]
- Modprobe command can add or remove modules
    - Configuration files will be located in the /etc/modprobe.d Directory
- *alias command*
    - Syntax
        - `alias {alternative name} {module name}`
- *blacklist command*
    - Syntax
        - `blacklist {module name}`
- *install command*
    - Syntax
        - `install {module name} {command}`

- /proc/sys/
    - Lists the parameters to configure on a system
    - Subdirectories
        - crypto
            - contains parameters related to encryption and other cryptographic services
        - debug
            - contains parameters for debugging the kernel
        - dev
            - contains parameters for hardware devices
        - fs
            - contains parameters for file system data
        - kernel
            - includes parameters related to kernel functions
        - net
            - includes parameters related to networking functions
        - user
            - includes parameters related to user space limitations
        - vm
            - includes parameters related to virtual memory management
    - sysctl
        - Powerful Linux command which acts as an interface to dynamically change the kernel parameters
        - The parameters available for modification can be found under /proc/sys directory

- Options
    - -a
        - Display all parameters and current values
    - -w {parameter}={value}
        - Set a parameter value
    - -p[file name]
        - Load sysctl settings from the specified file
    - -e
        - Ignore errors
    - -r {pattern}
        - Apply command to parameters matching a given pattern
- Syntax
    - `sysctl [options]`
- /etc/sysctl.conf
    - Enables configuration changes to a running Linux kernel

- **Monitoring Kernel Modules**
    - /proc/
        - Virtual file system (VFS) that provides information about the kernel's running process
        - Key files
            - /proc/cmdline
                - Contains options passed to the kernel by the boot loader

- /proc/cpuinfo
    - Contains CPU information
- /proc/devices
    - Contains a list of character and block device drivers loaded into the currently running kernel
- /proc/filesystems
    - Contains a list of file systems types that are supported by the kernel
- /proc/meminfo
    - Contains information about RAM usage
- /proc/modules
    - Contains information about modules currently installed on the system
- /proc/stat
    - Contains various statistics about the system's last reboot
- /proc/version
    - Specifies several points of information about the Linux kernel

- *GNU Compiler Collection (GCC)*
    - Used to compile the kernel, the user name, and the time the kernel was compiled

- *dmesg command*
    - Used to print messages that have been sent to the kernel's message during and after system boot
    - Drivers can also send diagnostic messages to the kernel in case they encounter errors
    - Syntax
        - `dmesg [options]`
    - Options
        - -c
            - Clear the kernel buffer after printing
        - –f {facility list}
            - Restrict output to the specified comma-separated list of facilities
        - -l {level list}
            - Restrict output to the specified comma-separated list of levels
        - -e
            - Display a human-readable version of the time messages
        - -L
            - Color-code messages for easier readability
        - -H
            - Output in a human-friendly format
        - -h
            - List the available options

The Linux Boot Process

Objective 1.1: Summarize Linux fundamentals.

- Linux Boot Components
    - *Booting*
        - Process of starting or restarting a computer and loading an operating system
    - *Boot Loader*
        - Small program stored in ROM that loads the kernel from a storage device
        - Boot loaders protect the boot process with a password
    - *Boot Sector Program*
        - Loads the second boot loader on startup
    - *Second Stage Boot Loader*
        - Loads the operating system and contains a kernel loader
    - *Boot Loader Installer*
        - Controls the installation of drive sectors and runs only when booting
    - *BIOS*
        - Enables to test the various hardware components in a computer as well as run a boot loader

- *UEFI*
  - Operates with a greater amount of memory, accesses storage drives and hardware types, and has improved security protections

- *Preboot eXecution Environment (PXE)*
  - Enables a client to retrieve the necessary boot loader and system files from a server over the network
    - Basic input/output system (BIOS)
    - Unified Extensible Firmware Interface (UEFI)
    - Booting from ISO
    - Booting from Network File System (NFS)

- *Master Boot Record (MBR)*
  - Sector that the BIOS reads in and starts when the machine is first booted

- *GUID Partition Table (GPT)*
  - Partition structure with a more modern design and is part of the UEFI standard

- *Raw Partition*
  - Enables users and applications to read from and write to a block storage without using the system cache

- *Initial RAM Disk (initrd)*
    - Root file system that is temporarily loaded into memory upon system boot
    - Phases
        - System is booted with minimal set of modules
        - Main root file system is mounted
    - *initrd image*
        - Archive file that contains all the essential files that are required for booting the operating system
            - Initrd image is stored in the /boot directory
    - *mkinitrd command*
        - Used to create the initrd image for preloading the kernel modules
        - Options
            - --preload={module name}
                - Load a module in the initrd image before the loading of other modules
            - --with={module name}
                - Load a module in the initrd image after the loading of other modules
            - -f
                - Overwrite an existing initrd image file
            - –nocompress
                - Disable the compression of the initrd image

            - Syntax
                - `mkinitrd [options] {initrd image name} {kernel version}`

- /boot
    - Contains files that are used to facilitate the Linux boot process
    - Subdiretories
        - GRUB
            - /boot/grub
        - GRUB 2
            - /boot/grub2
        - ESP
            - /boot/efi
        - initramfs image
            - /boot/initramfs-<kernel version>.img
        - Linux kernel
            - /boot/vmlinuz-<kernel version>

- *Dracut command*
    - Used to generate an initramfs image
    - Syntax
        - `Dracut /boot/initramfs-$(uname -r) .img $(uname -r)`

- Steps into the boot process
  - The processor checks the BIOS/UEFI firmware
  - The BIOS/UEFI checks for bootable media
  - The BIOS/UEFI loads the primary boot loader
  - GRUB 2 selects the operating system
  - The boot loader determines the kernel and locates the kernel binary
  - The kernel configures the available hardware drivers
  - The kernel mounts the main root partition and releases unused memory
  - The systemd program searches for the default.target file
  - The system authenticates the user
  - The system is ready to use

- *Kernel Panic*
  - mechanism by which the system detects fatal errors and responds to them

- **GRUB 2**
  - *GNU Grand Unified Bootloader (GNU GRUB)*
    - Enables users to choose which operating system or kernel version to boot
  - GRUB 2 offers more control over the boot process, boot devices, and boot behavior

- Features
  - Non-86 architecture platforms
  - Boot OS from storage media
  - Partition UUIDs and loading modules
  - Configure boot loader through scripts
  - Customization features
- GRUB 2 became the default boot loader on almost all modern Linux distributions

- *grub2-install command*
  - Copies over the GRUB2 files into the /boot/GRUB2 directory
  - applies to BIOS systems, not UEFI
    - to install to UEFI, use a package manager
  - Syntax
    - `grub2-install [options] [device name]`
  - `Options`
    - --modules {module names}
      - Reload specified kernel modules
    - --install-modules {module names}
      - Install only the specified modules and dependencies
    - --directory {directory name}
      - Install files from the specified directory
    - --target {target platform}
      - Specify the target platform

        - --boot-directory {directory name}
            - Specify the boot directory
        - –force
            - Install GRUB 2

- Important Directories
    - grub.cfg
        - Main configuration file for GRUB 2
        - Locations
            - BIOS
                - /boot/grub2/
            - UEFI
                - /boot/efi/EFI//

- /etc/grub.d/
    - Contains scripts that are used to build the main grub.cfg file
    - To add a custom script in the directory, use ##_ file name prefix to name the script
    - Add the script to the existing 40_custom file to execute last by default

- /etc/grub.d/40_custom
    - Enables customization of the menu presented to the user during the boot process

- /etc/default/grub
  - Contains GRUB 2 display menu settings that are read by the /etc/grub.d/ scripts

- *grub2-mkconfig command*
  - Generates a new grub.cfg configuration file and is used to update the existing grub.cfg file
  - Syntax
    - `grub2-mkconfig [-o {file name}]`
  - *update-grub2 command*
    - used to update grub2 configuration file

System Components

Objectives:

- 1.1: Summarize Linux fundamentals.
- 1.4: Given a scenario, configure and use the appropriate processes and services.
- 1.7: Given a scenario, manage software configurations.
- 4.3: Given a scenario, analyze and troubleshoot central processing unit (CPU) and memory issues.
- 4.5: Given a scenario, use systemd to diagnose and resolve common problems with a Linux system.

- **Localization Options**
    - *Localization*
        - Adapting system components for use within a distinct culture
        - Will adapt to
            - the language of the region you are in
            - adjust to measurements used in your region
            - configure the keyboard layout to your language
            - sets date and time

    - *Cron daemon*
        - Uses the system's time zone for executing cron jobs
        - *cron jobs*
            - automated tasks that are executed at certain times of the day

- /usr/share/zoneinfo Directory
  - A container for all the regional time zones the system can use
  - To change the timezone, create a symbolic link to one of the individual time zone files to the /etc/localtime file
  - In some Debian distros, this is found in /etc/timezone
    - Lists the time zone by the region structure seen in the /usr/share/zoneinfo directory

- *date command*
  - Prints the date in a specified format based on the /etc/localtime file
  - Syntax
    - `date [options] [format]`
  - Options
    - %A
      - Print the full weekday name
    - %B
      - Print full month name
    - %F
      - Print the date in yyyy-mm-dd format
    - %H
      - Print the hour in 24-hour format
    - %I
      - Print the hour in 12-hour format
    - %j
      - Print the day of the year

        - %S
            - Print seconds
        - %V
            - Print the week of the year
        - %x
            - Print the date representation based on the locale
        - %X
            - Print the time representation based on the locale
        - %Y
            - Print the year

- *timedatectl command*
    - Sets system date and time information
    - Subcommands
        - status
            - Show the current date and time information
        - set-time
            - Set the system's time to the time provided
        - set-timezone
            - Set the system's time zone to the time zone provided
        - list-timezones
            - List all available time zones in the format specified
        - set-ntp {0|1}
            - Enable or disable synchronization with an NTP server

- Syntax
  - `timedatectl [options] [subcommand]`
- Options
  - -H {remote host}
    - Execute the operation on the remote host specified by IP address or hostname
  - --no-ask-password
    - Prevent the user from being asked to authenticate when performing a privileged task
  - --adjust-system-clock
    - Synchronize the local (system) clock based on the hardware clock when setting the hardware clock
  - -M {local container}
    - Execute the operation on a local container
- Clocks exposed by the timedatectl command
  - Local Clock
    - Local current time
  - Universal Time Clock
    - UTC/GMT
  - Hardware Clock
    - Hardware level

- *hwclock command*
  - Allows for the viewing and setting of the hardware clock
  - *Systematic Drift*
    - The predictable amount of time that the hardware clock gains or loses each day
  - /etc/adjtime File
    - Records information about when and by how much the hardware clock is changed
  - Syntax
    - hwclock [options]
  - Options
    - –set
      - Set the hardware clock to the provided date and time
    - -u
      - Set the hardware clock to UTC
    - -s
      - Set the system time from the hardware clock
    - –adjust
      - Add or subtract time from the hardware clock to account for systematic drift

- *localectl command*
  - Views and configures the system locale and keyboard layout settings
  - localectl Subcommands

- status
  - Show the current locale and keyboard layout
- set-locale
  - Set the system locale to the locale provided
- list-locales
  - List all available locales on the system
- set-keymap
  - Set the keyboard layout to the provided layout
- list-keymaps
  - List all available keyboard layouts on the system

▪ Syntax
  - `localectl [options] [subcommand]`

▪ Options
  - -H {remote host}
    - Execute the operation on the remote host specified by IP address or hostname
  - --no-ask-password
    - Prevent the user from being asked to authenticate when performing a privileged task
  - --no-pager
    - Prevent the output from being piped into a paging utility
  - --no-convert
    - Prevent a keymap change for the console from also being applied to the X display server, and vice versa

- Coding and Decoding
    - *Character Encoding*
        - Converts text into bytes
    - *Character Decoding*
        - Converts bytes into text
    - Default encoding is generally UTF-8 using the Unicode character set
    - C is associated with the positional number U+0043 in Unicode

- **Graphical User Interface**
    - *Graphical User Interface (GUI)*
        - Enables users to interact with a system or application through visual design elements

    - *Display Server*
        - Constructs and manages the windowing system and other visual elements that can be drawn on the screen
        - X Window System is the GUI system for Windows
        - Wayland was meant to improve the X Window System
            - Wayland did have improvements, but Windows X is still used because it provides
                - Better screen sharing
                - Better remote desktop connection
                - Easier recovery from crashes
        - Linux has a wide variety of desktop environments
            - Allows users to choose the one they like best
            - Most common

        - GNOME
            - Most popular
            - Free and open source
            - Powerful search tool
        - KDE Plasma
            - Has widgets
            - Lots of GUI apps
            - Supports both X and Wayland
        - Cinnamon
            - Default for Linux Mint distro
            - Uses a desktop metaphor
            - Developed in response to GNOME3
        - MATE
            - Extension of GNOME2
            - Has some default applications
            - Also used in Linux Mint

- Linux offers many remote desktops
    - *Virtual Network Computing (VNC)*
        - A cross-platform remote desktop service that enables full remote control of a desktop environment
        - Most popular
    - *xrdp*
        - Free and open-source utility that constructs a Remote Desktop Protocol (RDP)-like server for non-Windows systems
    - *NoMachine (NX)*

        - Cross-platform proprietary remote desktop software that offers support for multi-session environments and account management
    - *Simple Protocol for Independent Computing Environments (SPICE)*
        - Free and open-source protocol designed specifically for use in virtual environments
    - *Console Redirection*
        - The process of forwarding input and output through a serial connection rather than through any I/O peripherals that are directly attached to the system
        - Console redirection enables administrators to remotely configure systems in a pre-boot environment like BIOS/UEFI
    - *Secure Shell*
        - A remote access protocol that encrypts transmissions over a network
        - *SSH Port Forwarding*
            - The process of tunneling an application through SSH to secure it in the transmission
        - *local forwarding*
            - local client listens for connections and then tunnels any active connection on a remote server using SSH
        - *remote forwarding*
            - the SSH server forwards inbound traffic to another system on a different port

- Accessibility services vary from distro to distro

- **Services**
  - *Services*
    - Running programs or processes that provide support for requests and monitoring from other processes or external clients
    - Daemons
      - programs that run in the background without human intervention
      - lay dormant until called on to act
      - used by most services
  - *init*
    - The backend service that controls when and how services are started
    - *init daemon*
      - a configuration file that initiates the processes listed in it
  - The system can be initialized with
    - SysVinit
    - Systemd suite
  - *systemd suite*
    - Provides an init method for initializing a system
    - is now the dominant init method in modern Linux distributions
    - supports parallelization
    - tracks processes instead of using PIDs
    - uses unit files to determine how units are handled

        - *unit files*
            - system resources that systemd can manage
            - include things like block storage devices, peripheral devices
            - can also be used to set system environment variables and parameters
    - *targets*
        - a method of grouping unit configuration files together to represent specific modes of operations

- *systemctl command*
    - Enables the control of the systemd init daemon
    - Subcommands
        - status {service}
            - Retrieve the current status of a service
        - enable {service}
            - Enable a service to be started on boot
        - disable {service}
            - Disable a service so that it is no longer started on boot
        - start {service}
            - Activate a service immediately
        - stop {service}
            - Deactivate a service immediately
        - restart {service}
            - Restart a service immediately

            - set-default {target}
                - Set the default target for the system to use on boot
            - isolate {target}
                - Force the system to immediately change to the provided target
            - mask {unit file}
                - Prevent the provided unit file from being enabled or activated
            - daemon-reload
                - Reload the systemd init daemon, including all unit files
        - Syntax
            - `systemctl [options] [subcommand] [arguments]`
        - Options
            - -t {unit file type}
                - Specify the unit file types to perform the operation on
            - -a
                - List all unit files or properties, regardless of state
            - no-reload
                - Prevent the reloading of configuration changes when enabling or disabling a service
            - --no-ask-password
                - Prevent users from being asked to authenticate when performing privileged operations
            - --runtime
                - Make changes temporary so that they will not be present after a reboot

        - -H {remote host}
            - Execute the operation on the remote host specified by IP address or hostname
        - --no-pager
            - Prevent the output from being piped into a paging utility
    - Systemctl can also be used to change targets

- *hostnamectl command*
    - Shows the system's network hostname and other information about the system's hardware and the Linux kernel it is running
    - Syntax
        - `hostnamectl [options] [subcommand] [arguments]`

- *SysVinit*
    - An older init method that has been largely replaced by system
    - make use of runlevels which determine what types of daemons should be running
    - SysVinit Runlevels
        - 0
            - Shuts down the system
        - 1
            - Starts single-user mode
        - 2
            - Starts multi-user mode without remote networking

        - 3
            - Starts multi-user mode with remote networking
        - 4
            - Unused
        - 5
            - Starts multi-user mode with networking and GUI capabilities
        - 6
            - Reboots the system
    - *telinit command*
        - Switches the current runlevel of the system
        - Syntax
            - telinit [options] {runlevel}
    - *runlevel command*
        - Prints the previous and current runlevels of the system, each separated by a space
    - /etc/inittab File
        - Stores details of various processes related to system initialization on SysVinit
        - Format within file
            - id:rstate:action:process
                - id – unique identifier
                - rstate – runlevel
                - action:process – how actions will be handled
    - /etc/init.d
        - Stores initialization scripts for services

- /etc/rc.local File
  - Executed at the end of the init boot process, typically used to start custom services
- *chkconfig command*
  - Controls services in each runlevel and can also start or stop services during system startup
  - Subcommands/Options
    - {service} on
      - Enable a service to be started on boot
    - {service} off
      - Disable a service to keep from starting on boot
    - {service} reset
      - Reset the status of a service
    - --level {runlevel}
      - Specify the runlevel in which to enable or disable a service
  - Syntax
    - `chkconfig [options] [service] [subcommand]`
- *service command*
  - Controls SysVinit services through SysVinit scripts
  - Subcommands/Options
    - {service} status
      - Print the current state of a service
    - {service} start
      - Start a service immediately

        - {service} stop
            - Stop a service immediately
        - {service} restart
            - Restart a service immediately
        - {service} reload
            - Re-read a service's configuration files while the service runs
    - Syntax
        - `service [options] [service] [subcommand]`

- **Process Issues**
    - Common Issues
        - Instability when processes hang
        - Inefficient resource consumption
        - System sluggishness
    - Process Life Cycle
        - Running state
            - The process is currently executing in user space or kernel space
        - Interruptible sleep state
            - The process relinquishes access to the CPU and waits to be reactivated by the scheduler
        - Uninterruptible sleep state

            - The process will only wake when the resource it's waiting for is made available to it
        - Zombie state
            - A process was terminated but not yet released by its parent process so it cannot accept a kill signal
        - Stopped state
            - The process was stopped by a debugger or through a kill signal

    - *Process ID (PID)*
        - A non-negative integer used to identify a process
        - Numbering
            - Starts with 1
                - increases for each new process started
            - init daemon always has a PID of one because it is the first process to start

    - *pgrep command*
        - Identifies a process based on multiple factors when the exact PID is not known
        - Syntax
            - `pgrep [options] {pattern}`

    - *pidof command*
        - Finds the process ID of a named running program
        - Syntax
            - `pidof {name of the program}`

- *ps command*
  - Displays the process table which summarizes the current running processes on a system
    - Will show
      - PID
      - Terminal
      - CPU time
      - Command that started the process
  - Options
    - a
      - List all user-triggered processes
    - -e
      - List all processes
    - -l
      - List processes using a long-listing format
    - u
      - List processes along with the username and start time
    - r
      - Exclude processes that are not running currently
    - x
      - Include processes without a terminal
    - T
      - Exclude processes started by other terminals
    - -U
      - Display the processes based on the specified user

                - -p {PID}
                    - Display only the process associated with the specified PID
                - -C {command}
                    - Display all processes by command name
                - --tty {terminal number}
                    - Display all processes running on the specified terminal
            - Syntax
                - ps [options]

        - *top command*
            - Acts as a process management tool that allows for interactive process prioritization, sorting, or termination
            - Options
                - Enter key
                    - Refresh the status of all processes
                - Shift+N
                    - Sort processes in the decreasing order of their PID
                - M
                    - Sort processes by memory usage
                - P
                    - Sort processes by CPU usage
                - u
                    - Display processes belonging to the user specified
                - k
                    - Terminate the process specified

- r
  - Alter the scheduling priority of the process specified
- q
  - Exit the process list

▪ Syntax

- `top [options]`

- htop
  - ▪ A newer version of an interactive system monitor, process viewer, and process manager
  - ▪ htop is not installed by default on all Linux distributions
  - ▪ Key difference is the interface
    - top - Text-only black and white interface
    - htop - Text-based graphics with colorful interface

- *systemd-analyze command*
  - ▪ Retrieves performance statistics for boot operations
  - ▪ Subcommand
    - blame
      - identifies services and other units that slow the bootup process down
      - provides a list of all the systemd units that were executed at boot and the time it took to execute
  - ▪ Syntax
    - `systemd-analyze [options] [subcommand]`

- *lsof command*
  - prints a list of all files currently opened to all active processes
  - can be used to terminate a process
  - shows
    - Command/Process
    - PID
    - Invoking user
    - File descriptor
    - File permissions
    - File type
    - File name
  - Syntax
    - `lsof [options]`

- Scheduling
  - The scheduler provides each process with CPU time
  - Usually effective, but can be manually changed
  - Nice or niceness value
    - a number between -20 and +19
    - represents a priority for a process
    - higher the number, the higher the priority it has for CPU time
      - higher the number the more CPU time it has

- *nice command*
  - Runs a command with a different nice value than the default
  - must have root user authority to run this command

        - Option
            - -n
                - Increments the nice value by the given integer

- *renice command*
    - Alters the scheduling priority of an already running process
    - Options
        - -n option
            - Specifies the new nice value for a running process
        - -g option
            - Alters the nice value of the processes in a process group
        - -u option
            - Alters the nice value of all processes owned by the user

- *fg command*
    - moves the process into view (foreground)
    - Options
        - fg %{job ID}
            - Brings a job to the foreground
        - Ctrl+Z
            - Halts a job temporarily to allow the use of the bg command

- bg command
  - moves the process out of view (background)
  - Options
    - bg %{job ID}
      - Pushes a job to the background
    - &
      - Starts a command running in the background when added to the end of a command

- *jobs command*
  - Lists out all jobs either in the foreground or in the background
  - Commands
    - Ctrl+Z
      - Stops a foreground job and places it in the background
    - Ctrl+C
      - Force quits a running program via the command line environment
    - Ctrl+D
      - Logs out the current user session

- *nohup command*
  - Prevents a process from ending when the user logs off
  - stands for no "hangup"
  - Syntax
    - `nohup {command or script name}`

- *kill command*
    - Sends any specified signal (termination) to one or more processes
    - Syntax
        - `kill [options] PID`

- *pkill command*
    - Sends any specified signal (termination) to processes based on a matching pattern
    - Syntax
        - `pkill [options] {pattern}`

- *killall command*
    - Sends any specified signal (termination) to all processes matching the name specified
    - Syntax
        - `killall [options] {process name}`
    - Whether you can kill a process depends on permissions
        - user
            - only kill processes you own
        - root
            - can kill any processes

- Kill Signals
  - Used to provide additional information about terminating the process
  - Types
    - SIGHUP (1)
      - Send to a process when its controlling terminal is closed
    - SIGINT (2)
      - Interrupt a process from the terminal
    - SIGKILL (9)
      - Kill the process immediately
    - SIGTERM (15)
      - Terminate a process
    - SIGSTOP (17, 19, 23)
      - Pause a process
    - SIGSTP (18, 20, 24)
      - Pause a process from the terminal

- **CPU and Memory Issues**
  - Problems
    - Underperforming CPU
    - Overloaded CPU
    - Non-functional cores

  - /proc/cpuinfo File
    - Identifies characteristics about the CPU that might indicate issues related to performance or lack of support for features

    - Useful information
        - Processor
        - Vendor_id
        - Model name
        - CPU MHz
        - Cache size
        - Flags
        - Supported features

- Commands to use for diagnosing CPU problems
    - *sysctl command*
        - Enables the viewing of kernel parameters at runtime
    - *uptime command*
        - Displays the time from when a system started running
        - The load average field is the most relevant in CPU troubleshooting
    - *sar command*
        - Displays system usage reports based on data collected from system activity
        - Syntax
            - `sar [options]`
    - sysctl also retrieves CPU-based kernel parameters at runtime
        - usual format of parameters
            - `kernel.sched_domain.cpu#.domain#.param`
    - *lscpu command*
        - Displays information about the CPU architecture

- Memory Issues
    - Not enough total memory for all processes
    - Not enough free memory for new processes
    - Processes unable to access memory
    - Processes accessing too much memory
    - Leaving other processes without memory
    - System cannot access files from cache/buffer
    - RAM performs to specification
    - Memory consumption is at expected rate
    - System has enough available memory

- /proc/meminfo Files
  - Can show if
    - RAM is performing to specifications
    - Memory consumption is at the expected rate
    - The system has enough available memory
  - Useful fields
    - MemTotal
      - Total amount of physical memory in the system
    - MemFree
      - Total amount of physical memory currently used
    - Cached
      - Total amount of physical memory that is being used as cache memory
    - SwapTotal
      - Total amount of swap space on the system

        - SwapFree
            - Total amount of swap space that is currently unused
        - Dirty
            - Total amount of memory that is waiting to be written to storage
        - Writeback
            - Total amount of memory currently being written to storage

- *free command*
    - Parses the /proc/meminfo file for easier analysis of memory usage statistics
    - Options
        - -b, -k, -m, -g, -tera
            - Display memory in bytes, kilobytes, megabytes, gigabytes, and terabytes
        - -s {seconds}
            - Update memory statistics at a delay of the specified seconds
        - -o
            - Disable the display of the buffered or cached information
        - -t
            - Display total line that combines physical RAM with swap space
        - -h

            - Make the output more human-readable
        - Buffers field in /proc/meminfo file
            - indicates memory assigned to a specific block device
            - Caches file system metadata
        - Cached
            - Caches actual file contents, not metadata

    - *lsmem command*
        - Lists the ranges of available memory with their online status

    - *vmstat command*
        - Displays various statistics about virtual memory, as well as process, CPU, and I/O statistics
        - vmstat Statistics
            - Memory-based
                - Total virtual memory available
                - Total virtual memory that is free for use
                - Total memory used in buffers and cache
                - Total memory used in swap space
            - CPU-based
                - Time spent running user space
                - Time spent running in kernel space
                - Time spent idle
                - Time spent waiting for I/O
        - Syntax
            - vmstat [options] [delay [count]]

- *Out-of-Memory (OOM) Killer*
  - Determines processes to kill when the system is extremely low on memory
  - OOM killer leverages an algorithm that assigns each process an OOM score
    - The higher the score, the more likely to be killed
  - You can mount the OOM control group at the desired mount point

- Configuration of Swap Space
  - Types
    - *Device Swap Space*
      - Used to run large applications
    - *File System Swap Space*
      - An emergency resource when the available swap space runs out
    - *Pseudo Swap Space*
      - Enables large applications to run on computers with limited RAM
  - *Swap Partition*
    - An area of virtual memory on a storage device to complement the physical RAM in the computer
  - *mkswap command*
    - Creates swap space on a storage partition

        - Options
            - -c
                - Verify the device is free from bad sectors before mounting the swap space
            - -p
                - Set the page size to be used by the mkswap command
            - -L
                - Activate the swap space using labels applied to partitions or file systems
    - *swapon command*
        - Activates a swap partition
        - Options
            - –e
                - Skip devices that do not exist
            - –a
                - Activate all of the swap space
    - *swapoff command*
        - Deactivates the swap space
        - Options
            - –a
                - Deactivate all of the swap space

Devices

Objectives:

- 1.1: Summarize Linux fundamentals.
- 1.3: Given a scenario, configure and manage storage using the appropriate tools.
- 4.1: Given a scenario, analyze and troubleshoot storage issues.

- **Linux Devices**
    - *Device Drivers*
        - Act as an interface between the operating system and hardware devices
    - Client Devices
        - *Thin Client*
            - Any lightweight computing device that connects to a more powerful server
            - A thin client has fundamental I/O devices like a keyboard, mouse, and monitor connected to it
            - Server
                - Processing and storing data
            - Thin Client device
                - Acts as a user interface
    - *Universal Serial Bus (USB)*
        - De facto standard for connecting input devices, external storage devices, and mobile devices

        - Linux registers USB storage devices attached to the system in /dev/sd# format

- *Wireless Devices*
    - Transmit and receive signals over the air

- *Wi-Fi*
    - Technology used primarily in establishing wireless local area connections (WLAN)

- *Bluetooth*
    - Technology used primarily for establishing a personal area network (PAN)
    - Bluetooth enables users to listen to audio without the need for cables

- *Near Field Communication (NFC)*
    - Communications protocol used by mobile devices and peripherals

- Video and Audio Devices
    - Input or output peripherals that are attached to client systems
        - Microphone
            - Common audio input
        - Speakers or headphones
            - Common audio output
    - When connecting video or audio to a system, check the connection type

        - Printers
            - You may need to install drivers manually
            - Most modern printers offer local connection support through a USB interface

- *Network Adapter*
    - Acts as an interface that allows computer devices to have access to a network

- *Network Interface Card*
    - Device that provides an interface with which hosts exchange data over a network

- *General-Purpose Input/Output (GPIO)*
    - Pins on a circuit board that have no designated purpose
    - GPIO is controlled through software

- *Serial AT Attachment (SATA)*
    - Computer bus interface standard for attaching storage devices to traditional computers

- *PCI Express*
    - Supports raw data rates of up to 16 Gb/s

- *Small Computer System Interface (SCSI)*
  - Computer bus interface for connecting devices to computers
  - SCSI is used for storage
- *Serial Attached SCSI (SAS)*
  - Developed to apply a serial interface to SCSI technology
  - SAS4 offers speeds up to 24 Gb/s and supports higher-capacity drives
- *Host Bus Adapter (HBA)*
  - Hardware component that connects a host system to a storage device
  - HBAs may be built into the motherboard or a separate expansion card
- *Peripheral Component Interconnect (PCI)*
  - Used as an expansion bus for attaching peripheral devices
- *PCI Express (PCIe)*
  - Supports greater transfer speeds, more reliable error detection, and is physically smaller than PCI
  - PCIe is the dominant expansion bus technology

- **Configure Devices**
  - Device Files
    - Represent information about hardware devices and settings

  - Directories that contain configuration information
    - /proc/
      - Contains various files that represent system information reported by the kernel
      - /proc/devices
        - Contains list of device drivers that the kernel is currently running
    - /sys/
      - Virtual file system that focuses on creating a hierarchical view of device information
      - /sys/devices/
        - Includes files that shows details about specific devices
    - /dev/
      - Enables the system and users to access devices
        - /dev/sda1
          - Storage device
        - /dev/mapper/
          - Logical and encrypted volumes
    - /etc/
      - Contains configuration files for components that interface with devices

- /etc/X11/
  - Contains configuration files for I/O devices affecting X.Org server

- *Hot-Pluggable*
  - Can be physically added or removed from the system without requiring a reboot
    - Hot-Pluggable
      - Detected by the system
    - Cold-Pluggable
      - Not detected by the system

- *udev utility*
  - Handles module loading for cold-pluggable and hot-pluggable devices
  - Udev manages the automatic detection and configuration of hardware devices
  - Directories for using udev to configure devices
    - /etc/udev/rules.d/
      - Used to configure rules for udev functions
      - /etc/udev/rules.d/ directory is used for local administration of udev
    - /usr/lib/udev/rules.d/
      - Contains rules generated by the system
      - shouldn't be customized

- *udevadm command*
    - Used to manage udev
    - Subcommands
        - Info
            - Used to view the device's vendor ID, product ID, and serial number
        - Control
            - Modifies the running state of udev
            - The --reload-rules option ensures udev is reading from the newly added rules files
        - Trigger
            - Executes rules that apply to any device that is currently plugged in
            - The -c option is used to add, remove, or change rules
        - Monitor
            - Used to watch for events sent by the kernel or by a udev rule
        - Test
            - Used to simulate udev events and present results as output
    - Syntax
        - `udevadm [options] [subcommand] [arguments]`

- Printer
    - Bundled with software utilities that enable users to configure settings

        - *Common Unix Printing System (CUPS)*
            - Print management system for Linux that enables a computer to function as a print server
            - Changes made in CUPS modify the /etc/cups/cupsd.conf and /etc/cups/cups-files.conf files

    - *lpr command*
        - Submits files for printing
        - Options
            - -E
                - Force encryption
            - -P {destination}
                - Send print job
            - -# {copies}
                - Set number of copies to print
            - -T {name}
                - Set job name
            - -l
                - Print formatted file
            - -o
                - Set job option
            - -p
                - Print specified files
            - -r
                - Delete file after printing

        - Syntax
            - `lpr [options] [file names]`

- **Monitor Devices**
    - *lsdev command*
        - Displays hardware information from the interrupts, ioports, and dma files in the /proc directory
    - /proc/interrupts
        - Lists each logical CPU core and its associated interrupt requests (IRQ)
        - *IQR*
            - Signal sent by a device to the processor
            - The IRQ address lists the signals that were sent to each CPU core
    - /proc/ioports
        - Lists input/output ports and the mapped hardware devices
    - /proc/dma
        - Lists all Industry Standard Architecture (ISA) direct memory access (DMA) channels on the system
        - *ISA DMA*
            - Hardware controller that supports legacy technology like floppy disks

- *lsusb command*
    - Used to display information about devices connected to the system's USB buses
    - scans the /dev/bus/usb directory
    - by default it prints
        - the number of the bus
        - the connected device
        - the ID of the device
        - the name of the vendor
    - Options
        - -v
            - Device information
        - -s
            - Filter result by bus
        - -d
            - filter by Vendor or product
    - Syntax
        - lsusb [options}

- *lspci command*
    - Used to display information about devices connected to the system's PCI/PCIe buses
    - By default shows devices in the format Bus:Device.Function

- *lpq command*
    - Shows the status of the print queue
    - Also shows
        - who owns the job
        - job number
        - files in the job
        - size of the job
    - Option
        - +interval
            - Reports update at every second until the queue is empty
    - Syntax
        - `lpq [options]`
- *lsblk command*
    - Identify block storage devices
- *dmesg command*
    - Print all messages sent to kernel's buffer
    - Use dmesg output to monitor issues related to device drivers and hardware

- **Troubleshooting Hardware Issues**
    - Keyboard mapping issues
        - Identify the layout of the physical keyboard
        - Use localectl status to verify the layout
        - Identify the correct layout and set it up on the system

- SSH client (PuTTY) enables users to change the effects of keystrokes on the environment

- Communication port issues
  - Ensure that the device is correctly slotted into the port
  - Ensure that power is being supplied to the bus adapter
  - Ensure that drivers are installed and loaded into the kernel
  - Linux will assign the port an interface at /dev/ ttyS#

- Printer Issues
  - Consult the printer's manual and/ or the manufacturer's website
  - Ensure the printer is supported y Linux-compatible drivers
  - Use network diagnostic tools (ping)
  - lpq command
    - Check status of print job
  - lprm command
    - Stop a job

- Memory Issues
  - Use memory monitoring tools (free) and the process monitoring tools (top) to identify the problem
  - If the message contains error-correcting code (ECC) errors, one of the memory modules has failed

- Video Issues
    - Ensure the monitor and other devices are properly connected and compatible
    - GPU Driver
        - Video-intensive application

- Storage adapter issues
    - The problem might be with the physical HBA or HBA used like SCSI or SATA
        1. Ensure that the Host Bus Adapter (HBA) is powered on
        2. Ensure the device connecting to the HBA uses the right interface
        3. Ensure that all devices are properly slid and that all cables are connected and damage-free
    - RAID Arrays
        - *mdadm command*
            - used to manage RAID arrays
            - Options
                - -F
                    - Activates monitor mode
                - -f
                    - Mark specified device
                - -r
                    - remove specified device
                - --re-add
                    - Add removed device

            - -a
                - Add device as hot-spare
- *lshw command*
    - Lists detected hardware components on the system and provides device details
    - Options
        - -c network
            - Output details about network-class devices
        - -short | sort -k2
            - Display list of classes currently in use in the system
    - Syntax
        - `lshw [options]`
- *dmidecode command*
    - Dumps the system's Desktop Management Interface (DMI) table and presents in a readable format
    - Syntax
        - `dmidecode [options]`
    - *DMI Table*
        - Industry standard for tracking information about hardware components
        - Do not rely on DMI tables as the sole source of hardware information

- Automatic Bug Reporting Tool (ABRT)
    - Analyzes and reports problems detected during system runtime
    - ABRT runs as the abrtd daemon and can be configured using abrt-cli or abrt-gui

- General Steps for troubleshooting hardware devices
    1. Ensure that hardware devices are supported by robust drivers
    2. Ensure that necessary drivers are installed and loaded in the kernel
    3. Ensure that hardware devices are compatible with the Linux software
    4. Verify that the system has the correct keyboard layout and language set
    5. Verify that a network-enabled printer is identifiable
    6. Use the lprm command to stop large or numerous print jobs
    7. Check the mcelog for memory errors
    8. Run a utility like memtester to stress test RAM modules
    9. Download the latest GPU drivers from the vendor's website
    10. Ensure that storage and peripheral devices are properly slotted into the correct buses
    11. Ensure the connected cables are not loose or damaged
    12. Use a command like lshw to identify connected hardware
    13. Be aware that dmidecode may produce inaccurate results
    14. Review crash data compiled by the ABRT utility

- Troubleshooting requires knowledge and familiarity with the command line and system messages

Networking

Objectives:

- 1.5: Given a scenario, use the appropriate networking tools or configuration files.
- 3.5: Summarize container, cloud, and orchestration concepts.
- 4.2: Given a scenario, analyze and troubleshoot network resource issues.

- **TCP/IP Fundamentals**
  - *TCP/IP*
    - Family of network protocols offers various services
  - Open Systems
    - Interconnection (OSI)
    - Standardizes networking functions
    - 7 Layers of OSI Model
      - 7
        - Application supports applications and end-users
      - 6
        - Presentation formats data for use
      - 5
        - Session establishes, maintains, and tears down a connection
      - 4
        - Transport enables reliable transmission of information
      - 3
        - Network enables logical addressing

                - 2
                    - Data link enables physical addressing
                - 1
                    - Physical enables physical network connectivity

- TCP/IP is used to govern network communications and the internet

- Networking Terms
    - *Node*
        - Devices with an identity on the network
    - MAC Address
        - Physical address
    - IP Address
        - Logical address
    - *Hostname*
        - Human-readable name of the device

- Networking Hardware
    - *Switch*
        - Acts as a concentrator, centralizing all network connection
    - *Router*
        - Acts as a control point for communications between network segments
        - Routers work with IP addresses at Layer 3 of the OSI model

        - *Media*
            - Actual path of an electrical signal travels from one component to another
            - Network Cable
                - Twisted pair Ethernet cable
                - Twisted pair may come shielded (STP) or unshielded (UTP)

    - Data Movement
        - *Packet*
            - What data is referred to when it is at the Network layer (Layer 3)
        - *Frame*
            - What data is referred to when it is at the Data link layer (Layer 2)
        - *Bit*
            - What data is referred to when it is at the Physical Layer (Layer 1)

    - Network Services
        - *Domain Name System (DNS)*
            - Service provides name resolution
            - DNS is implemented as a database hosted on one or more servers

- Configuration types
  - Static Configuration
    - For servers and network devices
  - Dynamic configuration
    - End-user workstations
- *Dynamic Host Configuration Protocol (DHCP)*
  - Service provides dynamic configuration

- Identifiers
  - *IP Address*
    - Provide an addressing system for managing network identities
  - *Network Identifier*
    - Defines network host segment
  - *Host Identifier*
    - Uniquely identify the host in segment

- IP Address Classes
  - Each class provides a specified number of networks and the number of hosts available
    - Class A
      - 0.0.0.0 127.0.0.0
    - Class B
      - 128.0.0.0 191.255.00
    - Class C
      - 192.0.0.0 223.255.255.255

- Class D
  - 224.0.0.0 239.255.255.255
- Class E
  - 240.0.0.0 255.255.255.255

▪ Due to the depletion of IPv4, three IP address ranges are reserved for internal use only
- Class A Reserved
  - 10.0.0.0 – 10.255.255.255
- Class B Reserved
  - 172.16.0.0 – 172.13.255.255
- Class C Reserved
  - 192.168.0.0 – 192.168.255.255

▪ Loopback Address
- Used for diagnostics and to allow the system to network to itself

▪ Link-local Range
- Used for zero-configuration LANs or when the DHCP lease generation process fails (APIPA)

▪ IPv6
- Addresses IPv4's weaknesses, has a larger address space, built-in encryption, and more efficient routing

o *Network Port Numbers*

▪ Numeric values assigned to application-layer protocols

        - Humans work with HTTP, while computers need to work by port number
            - 22
                - Secure Shell
            - 25
                - Simple Mail Transfer Protocol
            - 80
                - Hypertext Transfer Protocol
            - 110
                - Post Office Protocol version 3
            - 443
                - Hypertext Transfer Protocol Secure

    - Network administrators divide the network into segments to manage network traffic
        - *Subnet*
            - Logical divisions of the network
        - *Network ID*
            - Part of the IP address each node is using
        - All nodes in a subnet have the same network ID

- **Linux Server Roles**
    - *Network Time Protocol (NTP)*
        - Enables the synchronization of nodes time with a designated and definitive time source

        - *Crony*
            - Designed to utilize NTP and perform in a large range of conditions

    - *Secure Shell (SSH)*
        - Provides an authenticated and encrypted method of connecting to a remote or local system

    - *Web Servers*
        - Host the files and images on the websites
        - Types
            - Insecure
                - HTTP
                - TCP port 80
            - Secure
                - HTTPS
                - TCP port 443
        - The web services on Linux are hosted through Apache or Nginx

    - *Certificate Server*
        - Provide a means of identity guarantee
        - Certificate authority (CA)
            - manage the enrollment, approval, expiration, and revocation of certificates

- *Domain Name System (DNS) Server*
  - Performs name resolution for easy-to-remember hostnames
  - DNS server may contain records for a company's internal network

- *Dynamic Host Control Protocol (DHCP) Server*
  - Provides configurations including IP addresses, subnet masks, and default gateways

- *Simple Network Management Protocol (SNMP) Server*
  - Capable of passing information of performance and workloads to a central management database

- *Centralized Authentication Server*
  - Holds information about user identities in a directory store

- *Proxy Server*
  - With direct access to the Internet and an internal network connection

- *Logging Server*
  - Used to centralize log files from the Linux servers

- *Monitoring Services*
  - Monitor specific applications
  - ApacheTop
    - Provides log file analysis for Apache and connection response time

      - Monit
         - Simple monitoring utility for Linux

- *Load Balancing Servers*
   - Used to distribute inbound connection requests across multiple servers
   - Multiple web servers create a load balance service needed to ensure connections to servers
   - *Node*
      - a server inside a cluster and can accept client connections

- *File and Print Servers*
   - Allows file storage and printing

- *Samba*
   - Windows compatible file sharing system that runs on SMB
   - Server Message Block (SMB) compatible file sharing protocol

- *NFS*
   - Used to provide access to directories stored on a server

- *Database Server*
   - Used to store large quantities of data and make easy queries
   - SQL
      - Uses relational tables

        - NoSQL
            - Unorganized relational tables

    - *Virtual Private Network (VPN) Server*
        - Enables remote users to connect to the internal company network and access internal resources

    - *Email Server*
        - Responsible for the distribution of electronic mail
        - Sending
            - Sendmail
        - Receiving
            - Postfix
        - Protocols
            - Simple Mail Transfer Protocol (SMTP)
            - Post Office Protocol (POP3)
            - Internet Message Access Protocol (IMAP)

- **Connecting to a Network**
    - *Computer Network*
        - Two or more computers connected through network media

    - Device/System Hostname
        - Used to easily recognize a machine within a network
        - For a computer to participate in a network, it must have a valid identity

        - Errors or misconfigurations in values will result in not participating in the network

    - *Network Manager Utilities*
        - Aids in the proper configuration of the IP information
        - nmcli
            - Command line Network Manager
            - Contains subcommands to view and configure network information
            - Subcommands
                - general status
                    - View summary of network connectivity data
                - connection show
                    - View identification information for each NIC
                - con up {device ID}
                    - Enable specified NIC
                - con down {device ID}
                    - Disable specified NIC
                - con edit {device ID}
                    - Enter interactive mode to configure specified NIC
                - device status
                    - Display current status of each NIC
            - Syntax
                - `nmcli [options] [subcommand] [arguments]`

- nmtui
  - Network Manager with a text-based user interface (TUI)
  - Moves the cursor from field to field
  - Make selections within the field
  - Activate setting (OK or Quit)
  - Check or uncheck a check box
- nmgui
  - Network Manager for GUI systems
  - Enables changes to IPv4 and IPv6 configuration

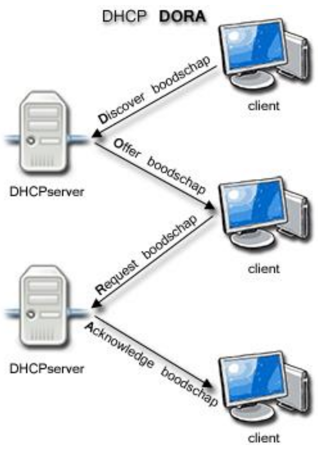
- Network commands
  - *ifconfig command*
    - Shows IP address, subnet mask, broadcast ID, MAC address, basic performance information, and NIC name
    - deprecated
    - Syntax
      - `ifconfig [options] [interface]`
  - *ip command*
    - updated version of ifconfig
    - Subcommands
      - ip addr show
        - Shows the IP address information
      - ip link
        - Shows the status of each interface
      - ip link set eth1 up
        - Enables the interface identified as eth1

            - ip link set eth1 down
                - Disables the interface identified as eth1
        - Syntax
            - `ip [options] {object} [subcommand]`
    - *iwconfig command*
        - Used to provide wireless NIC configurations
        - Syntax
            - iwconfig [options] [interface]
        - Options
            - nick {name}
                - Set a nickname
            - mode {mode}
                - Set the operating mode
            - freq {number}
                - Set the Wi-Fi frequency
            - channel {number}
                - Set the Wi-Fi frequency
            - retry {number}
                - Set the maximum number of MAC retransmissions

- **Configure DHCP and DNS Client Services**
    - Ways to configure IP addresses
        - *Static IP address configuration*
            - The settings are implemented manually by an administrator
        - *Dynamic IP address configuration*

    - The settings are retrieved from a server

- DHCP
    - The DHCP service must be installed on the server and allow client machines to lease configurations
    - DORA
        - Discover Offer Request Acknowledge



    - /etc/dhcp/dhclient.conf File
        - Enables the configuration of DHCP client settings
    - NetworkManager
        - Serves as a network configuration service for multiple network settings

- DNS
    - TCP/IP data packets must include a source IP address and a destination IP address

        - Methods to associate domain names to IP addresses
            - Static text files
                - /etc/hosts
                    - Used in special case situations where a particular system
            - Dynamic database
                - /etc/resolv.conf
                    - Informing the system of the IP address of one or more DNS servers
                - /etc/nsswitch.conf
                    - Includes several configuration options
                - /etc/hosts → DNS
        - Network Manager
            - Command-line
            - Text-based
            - Graphical interface utilities

    - *dig command*
        - look up domain names and IP addresses

    - *nslookup command*
        - Used to test name resolution

    - *host command*
        - Used to query DNS server

- *whois command*
  - Host information about the owner of the domain name

- **Cloud Technologies**
  - Cloud computing is a relatively new and rapidly changing aspect of the IT industry

  - Essential Characteristics of cloud computing
    - On-demand self-service
    - Broad network access
    - Resource pooling
    - Rapid elasticity
    - Measured service

  - Cloud services indicate flexibility in terms of deployment, scale, support, and fault tolerance
    - *Software as a Service (SaaS)*
      - Provides applications to the end-users
      - In SaaS, work and storage of the data is done by the cloud services, not on the installed application
    - *Platform as a Service (PaaS)*
      - Virtualization of the environment's operating system layer
      - The PaaS model provides services to developers and database administrators

        - *Infrastructure as a Service (IaaS)*
            - Includes physical devices that are virtualized and owned by a cloud service provider

    - Cloud Deployment models
        - *Public Clouds*
            - Hardware resources that can be shared by multiple customers
        - *Private Clouds*
            - The use of cloud technologies as an on-premise solution
        - *Hybrid Clouds*
            - Enabling more effective cost management combined with strict security management

    - Cloud Service Providers
        - Amazon Web Services (AWS)
            - Supports deployment options of a variety of cloud-based services
            - Amazon Linux AMI solution is free and open-source
        - Microsoft Azure (Azure)
            - Supports deployment options and services like AWS
            - Microsoft Azure does support both Windows and Linux servers
        - Google Cloud Platform (GCP)
            - Allows for the deployment of SaaS, PaaS, and IaaS services on Linux and Windows servers

- Red Hat
  - Provides Linux-based cloud solution, designed as a full-featured private cloud for organizations

- **Virtualization Technologies**
  - *Virtualization*
    - Enables the use of hardware and provides fault tolerance, disaster recovery, and scalability

  - *Hypervisor*
    - Software layer that provides control between the virtual machines and the physical hardware
      - *Type 1 Hypervisors*
        - Run directly on the hardware called bare metal deployment
      - *Type 2 Hypervisors*
        - Run as a service on top of Linux, Windows, or OS X

  - *Kernel-Based Virtual Machine (KVM)*
    - Enables Linux virtual machines with the attributes of type 1 and type 2 hypervisors

  - Template files and formats
    - *Open Virtualization Format (OVF)*
      - Format contains configuration files, packages, and settings for virtual machines and network devices

        - JSON
            - JavaScript Object Notation
            - Used by most programming languages to store information
        - YAML
            - YAML Ain't Markup Language
            - Used to store configuration information on the newly deployed virtual machines
        - Container Image
            - Used by a specialized type of virtual machine called container

    - Bootstrapping
        - Basics
            - Refers to the adage "pulling yourself up by the bootstraps"
            - Operating system starts loading with simple layers and then moves upward to more complex layers
            - Handled by the virtualization layer in virtual machines
        - Methods of handling bootstrap management
            - Cloud-init
                - Cloud-based Linux mechanism to customize a virtual machine during the first boot up
            - Anaconda
                - Used by Linux distributions to manage deployments
            - Kickstart
                - Used to customize the installation and provide an automated and unattended installation of new virtual machines

- Storage
    - Storage space is used to store virtual machines and to process data
    - Virtual Storage
        - File that resides on the physical drive
        - Virtual machines treats this like a physical device
    - *Thin Storage (Provisioning)*
        - Virtual storage device file that will grow on demand up to a maximum size
    - *Thick Storage (Provisioning)*
        - Reserves the allocated space for the virtual device
        - The benefit of thick provisioning is the guaranteed space available
    - Ways to store data in a cloud
        - traditional SQL database
            - saves data in a structured manner
        - Blob
            - Name used by Microsoft Azure for large amounts of unstructured storage of data
        - Bucket
            - Name used by AWS for large amounts of unstructured data
        - Blobs and buckets can be used to store audio, video, other multimedia, and text files
    - Blocks
        - small chunks of data written to the storage device (physical or virtual)

- Networking and Virtualization
    - Can use virtual NICs
        - are connected to virtual switches to communicate with the physical NIC
    - Virtualization Hypervisor
        - Configured to provide access to networking services
        - Configuration options
            - *No networking*
                - simulates a computer that doesn't have a NIC card at all
            - *Internal option*
                - virtual switch allows communication with other VMs on that switch
            - *Private option*
                - virtual switch allows communication with other VMs on that switch and with the host operating system
            - *Public option*
                - virtual switch allows communication with other VMs on that switch, with the host operating system, and with the physical NIC
    - Network Address Translation (NAT)
        - Provide virtualized network functionality in physical networks
        - Virtualized networks may be thought of as "overlay networks"
        - *"Headless" Mode*

    - virtualization host servers that runs Linux without a GUI
- *virsh command*
    - Interactive shell to control the virtual machines
    - Stands for virtual shell
    - Subcommands
        - help
            - Get help
        - list
            - Get VM list
        - shutdown {VM}
            - Shutdown
        - start {VM}
            - Start
        - reboot {VM}
            - Reboot
        - create {XML file name}
            - Create
        - save {VM} {file name}
            - Save the state
        - console {VM}
            - Open console
- *libvirt*
    - an API that provides software building blocks to write virtualization solutions

- VMM
  - Virtual Machine Manager
  - A virtual machine manager utility used with GNOME
  - Used to manage connectivity to virtual machines
  - Install the virt-manager to begin using VMM

- **Troubleshooting Network Issues**
  - General steps to begin troubleshooting
    - Check if the device is powered or plugged in
    - Verify and configure the network interface
    - Check if network interface is detected by Linux
  - Name resolution issues
    - Ping a destination by hostname and by IP address
    - Use host and nslookup to test the system's ability to perform DNS
  - Network traffic
    - Use the netstat command to check if the latency is high or saturation is occurring
  - Failing Network Interface Card
    - check the other side connection
    - Replace the network card

- Application performance
    - The localhost method creates a full network connection
        - completes full TCP error checking
    - Use Unix sockets (Unix domain sockets)
        - faster, but the error-checking is less detailed

- Unrecognized network adapters
    - Verify the appropriate driver has been installed

- Utilities and commands
    - *ping command*
        - Generate a response request from the sending computer
        - Possible replies
            - <host>
                - Connection successful
            - Destination unreachable
                - No path destination
            - timeout
                - Request reached the destination but a response did not return to the source computer
        - Ping only shows that something is wrong, not what is wrong
        - Syntax
            - `ping [options] {destination}`

        - Options
            - -c
                - Send number of pinging attempts
            - -v
                - Specify verbose output

- View network paths
    - *traceroute command*
        - Used to report the network path between the source and destination
        - *hop*
            - Process of a packet traveling from one router to another
            - traceroute shows each hop
        - Syntax
            - `traceroute [options] {destination}`
    - tracepath command
        - Similar to traceroute
        - Syntax
            - `tracepath [options] {destination}`
    - Traceroute and tracepath allow you to see routing loops in which traffic is routed back and forth and never reaches its destination

- *netstat utility*
  - Used to gather information about TCP connections to the system
  - Options
    - -v
      - Activate verbose mode
    - –i [interface]
      - Display interface information
    - -c
      - Print information
    - -l
      - Show port/s being listened
  - Syntax
    - `netstat [options]`

- *ss utility*
  - Stands for socket state
  - Replacement for netstat which has been deprecated
  - Information gathering utility with simpler output and syntax
  - Symptoms
    - Missing socket
      - Service is not running
    - Closed socket
      - Premature termination of connection

- Options
  - -l
    - Show currently listening sockets
  - dst {host}
    - Show host statistic connection
  - -i
    - Show ports that being listened
- Syntax
  - ss [options]

- Name resolution
  - one of the most important network services so failures can cause many problems
  - *dig command*
    - Used for gathering information and testing name resolution
    - Syntax
      - `dig {domain name}`
        - used to determine what the IP address is for a domain name
      - `dig @ {IP address} {domain name}`
        - will resolve the domain name against a DNS server
        - will help determine if the issue is with your DNS server or the DNS at large

        - *nslookup command*
            - Tool for gathering name resolution information and testing name resolution
            - Syntax for using nslookup interactive mode
                - `nslookup`
        - *host*
            - Capable of gathering information and testing name resolution
            - Syntax
                - `host {domain name} {IP address}`

    - *ip command*
        - Replaced the ifconfig command for interacting with the NIC
        - Verify all settings are correct
        - Use ip addr subcommand to ensure the configuration is accurate

    - *route command*
        - Used to view the routing table
        - Syntax
            - To see the current routing table on the system
                - `route [options]`
            - To see the default gateway by IP address
                - `route add default gw {IP address}`
            - To add another host
                - `route add -host {IP address}`
        - Reject subcommand
            - Used to reject or filter traffic

- *nmap utility*
  - Stands for Network Mapper
  - Tool for exploring a network environment
  - Syntax
    - `nmap [options] {target}`

- *Wireshark*
  - Common packet sniffer and network analyzer
  - Wireshark has the ability to see moving or not moving packets through an NIC
  - Requires a GUI

- *tcpdump utility*
  - One of the most popular packet sniffers available
  - Determines traffic type and content
  - Options
    - -i
      - Specify the interface to use
    - -n
      - Not resolve hostnames
    - -v
      - Specify verbose mode
  - Syntax
    - Tcpdump [options] [-i {interface}] [host {IP address}]

- *netcat command*
  - Can be abbreviated as nc depending on the distribution
  - Used to test connectivity and send data across network connections
  - Syntax
    - `netcat [options]`
    - `nc [options]`
- *iftop command*
  - Displays bandwidth usage information for the system
  - Syntax
    - `iftop [options] [-i {interface}]`
- *iperf command*
  - Used to test the maximum throughput of an interface
  - Syntax
    - Client
      - `iperf -c [options]`
    - Server
      - `iperf -s [options]`
  - Difference between Bandwidth and Throughput
    - Bandwidth
      - Potential amount of data
    - Throughput
      - Actual amount of data

- *mtr command*
  - Combination of ping and traceroute that enables testing the quality of a network connection
  - Lost packets is a strong indicator of a network issue along the path
  - Syntax
    - `mtr [options] [hostname]`

- *arp command*
  - *Address Resolution Protocol (ARP)*
    - Used to relate IP and MAC addresses
  - Syntax
    - `arp [options]`

- *whois command*
  - Provides information on Internet DNS registrations
  - Syntax
    - `whois [options] {domain name}`
- Use ping, traceroute, and iftop when experiencing slow network performance

Packages and Software

Objective 1.6: Given a scenario, build and install software.

- **Package Managers**
    - Linux distributions rely on two different methods for managing software throughout its lifecycle
        - *Package Managers*
            - Install, update, inventory, and uninstall packaged software
        - *Compiling software*
            - Compiling code is more common for Linux administrators than for Windows or macOS users

    - Many Linux applications are modular and have dependencies which must be installed along with the applications in order for the applications to work
        - Package managers check for dependencies and download them automatically
        - Using compiling software means you have to find the dependencies and download and install them on your own
            - Compiling the software is the traditional method of managing software

- RPM
  - Red Hat package manager
  - Software packages that are prepared for RPM use the .rpm file extension
    - Inventory Software
      - One of RPM's most useful features
  - Main RPM package managers
    - Yellowdog Updater, Modified (YUM)
      - Offers a more elegant set of commands and greater flexibility for using software repositories and handling dependencies
    - Dandified YUM (DNF)
      - Uses fewer resources while still maintaining support for RPM
      - Syntax to install
        - `dnf install {package name}`
      - Syntax to uninstall
        - `dnf remove {package name}`
    - Zypper
      - An openSUSE package manager that supports .rpm packages
      - Syntax to install
        - `zypper in {package name}`
      - Syntax to uninstall
        - `zypper rm {package name}`

- *dpkg*
    - Debian package manager
    - Software packages with the .deb file extension can be managed using dpkg command
    - *Advanced Package Tool (APT)*
        - Preferred package management method in Debian-derivatives

- **RPM Packages and YUM**
    - *rpm command*
        - Manages RPM packages on Red Hat-derived distributions
        - Options
            - -i {package name}
                - Install the specified software
            - -e {package name}
                - Erase or uninstall the package
            - -v
                - Enable verbose mode to provide more detail
            - -h
                - Enable verbose mode, providing more detail
            - -V {package name}
                - Verify the software components of the package exist
        - Syntax
            - `rpm [options] [package name]`
        - Options to query the rpm database
            - -qa
                - List all installed software

            - -qi {package name}
                - List information about a particular package
            - -qc {package name}
                - List the configuration files for a particular package
        - Options for upgrading
            - -U
                - Upgrade or install a package
            - -F
                - Freshen installed package

    - *yum command*
        - Improves on the functionality of rpm while still using .rpm packages and maintaining an RPM database
            - Automatic handling of dependencies
            - Use of repositories
        - Subcommands
            - install {package name}
                - Install the package from any configured repository
            - localinstall {package name}
                - Install the package from the local repository
            - remove {package name}
                - Uninstall the package
            - update [package name]
                - Update the package
            - update
                - Update all installed packages

            - info {package name}
                - Report information about the package
            - provides {file name}
                - Report what package provides specified files/libraries
        - Syntax
            - `yum [options] [subcommand] [package name]`
        - Options
            - -y
                - Automatically answer yes to installing additional software dependencies

    - dnf subcommand
        - Syntax
            - Syntax to install
                - `dnf install [package name]`
            - Syntax to uninstall
                - `dnf remove [package name]`

- **Debian Packages and APT**
    - *dpkg command*
        - Debian's main package management program
        - Options
            - -i {package name}
                - Install the package

- -r {package name}
  - Remove or uninstall the package
- -l {package name}
  - List information about the specified package
- -l
  - List all installed packages
- -s {package name}
  - Report whether the package is installed

▪ Syntax
  - `dpkg [options] [package name]`

▪ dpkg ensures all the necessary components and dependencies are installed

- *APT*
  - ▪ A front-end manager to the dpkg system
  - ▪ Newer version of dpkg
  - ▪ Older apt commands
    - apt-get
    - apt-cache

- *apt command*
    - Subcommands
        - install {package name}
            - Install the package
        - remove {package name}
            - Uninstall the package and keep its configuration files
        - purge {package name}
            - Uninstall the package and remove its configuration files
        - show {package name}
            - Report information about the package
        - version {package name}
            - Display version information about the package
        - update
            - Update APT database of available packages
        - upgrade [package name]
            - Upgrade the package
        - upgrade
            - Upgrade all packages

            - Syntax
                - `apt [options] [subcommand] [package name]`
        - The apt-get and apt-cache commands are still functional and have more specific controls
            - apt command is updated and better than apt-get and apt cache

    - Run the apt update command prior to running apt upgrade
        - apt Update
            - Does not install any software
        - apt Upgrade
            - Upgrades all installed software

- **Repositories**
    - *Repositories*
        - Storage locations for available software packages
        - also called Repos
            - *Local Repository*
                - Stored on the system's local storage drive
            - *Centralized Internal Repository*
                - Stored on one or more systems within the internal LAN
            - *Vendor Repository*
                - Maintained on the Internet, often by the distribution vendor

- *createrepo command*
  - Updates the XML files used to reference the repository location
  - Creates a .repo Configuration File
    - Provides additional information about the repository and is stored in the /etc/yum.repos.d/ directory
    - .repo File Components
      - [repo-name]
        - Repository name
      - name=Repository Name
        - Human-friendly name of the repository
      - baseurl=
        - Path to the repository
      - enabled=1
        - Enables the repository
      - gpgcheck=0
        - Disables GPG checking

- yum Subcommands related to repositories
    - repolist
        - See all available repositories
    - makecache
        - Locally cache information about available repositories
    - clean all
        - Clear out-of-date cache information

- *Mirroring*
    - Enables the synchronization of an online repository to a local storage location
    - *reposync*
        - Manages the mirroring process

- **Acquire Software**
    - wget and curl commands
        - Can be written into scripts to automate the process of downloading package files
        - wget syntax
            - wget {web address}
        - Comparison
            - wget
                - Command line utility only
                - Can download files recursively
                - Supports HTTP/S and FTP
                - Download files

          - curl
            - Cross-platform
            - Cannot download files recursively
            - Supports more network protocols
            - Builds/manages complex requests

- *tar command*
  - Stands for Tape Archiver
  - Bundles together multiple files into a single tarball with a .tar extension
  - Options
    - -c
      - Creates the tarball
    - -x
      - Extract the tarball
    - -v
      - Enable verbose mode
    - -r
      - Append more files to an existing tarball
    - -t
      - Test the tarball or see what files are included in the tarball
    - -f
      - Specify the name of the tarball in the next argument
  - Syntax
    - `tar [options] {file name}`

- *gzip command*
  - Compression utility that produces files with the .gz extension
  - It is common to compress a tarball to create the .tar.gz or .tgz extension
  - Syntax
    - `gzip {file name}`
      - Compress the file and appends the .gz extension
    - `gzip -d {file name}`
      - Decompress the file
- .tar.bz2 indicates the tarball was compressed with the bzip2 utility

- **Building Software from Source Code**
  - *Compiler*
    - Translates source code written in a human-friendly programming language into machine-readable binaries
    - Software developers provide a list of all necessary libraries or dependencies for the software
      - Will be stored in either
        - Header files (.h file extension)
        - Library files (.a file extension)

  - *Program Libraries*
    - Chunks of compiled code that can be used in programs to accomplish common tasks

- *Shared Libraries*
  - Enable more modular program builds and reduce time when compiling the software
  - Found in the following directories
    - /usr/lib/
      - General access
    - /lib/
      - Essential binary access

- *ldd command*
  - Enables a user to view shared library dependencies

- Software Compilation Process
  - Decrypt the downloaded program using tar and gzip commands and change into the directory that gets created
  - Run the ./configure command to gather system information needed by the application
  - Use the make command to compile the application using the information stored in the makefile
  - Use the make install command to install the resulting binaries

- *makefile command*
  - Contains instructions used by a compiler to build a program from source code

- **Troubleshooting Software Dependency Issues**
  - If you are using a package manager, first check the configuration files for your specific package manager
    - APT
      - /etc/apt/sources.list.d /etc/apt.conf
    - YUM
      - /etc/yum.repo.d
      - /etc/yum.conf
    - DNF
      - /etc/dnf/dnf.conf
  - Use the verify command to check if the software and its dependencies were properly installed
  - Run the apt update command first to get the latest versions of the dependency packages
  - Check the documentation for the compiler for issues with the compiling software itself
  - When necessary, run the make install command using sudo to assume root privileges to run certain programs or code
  - Always ensure a solid backup and restoration plan when updating and upgrading software
  - If things go wrong, check the installation logs immediately

Securing Linux Systems

Objectives:

- 1.2: Given a scenario, manage files and directories.
- 2.1: Summarize the purpose and use of security best practices in a Linux environment.
- 2.3: Given a scenario, implement and configure firewalls.
- 2.4: Given a scenario, configure and execute remote connectivity for system management.
- 2.5: Given a scenario, apply the appropriate access controls.

- **Cybersecurity Best Practices**
    - Linux is the operating system used on most network devices and security appliances

    - Cybersecurity
        - Protection of computer systems and digital information resources from unauthorized access, attack, theft, or data damage
        - CIA Triad
            - Confidentiality
                - Keeps the information and communications private and protected from unauthorized access
                - Confidentiality is controlled through encryption and access controls

                - Integrity
                    - Keeps the organizational information accurate, error-free, and without unauthorized modifications
                    - Integrity is controlled through hashing, digital signatures, certificates, and change control
                - Availability
                    - Ensures that computer systems run continuously, and authorized users can access data
                    - Availability is controlled through redundancy, fault tolerance, and patching

- Authentication
    - Enables an organization to trust the users
    - Ways to authenticate users
        - Knowledge factors
            - PINs
            - Passwords
            - Passphrases
        - Possession factors
            - *Token*
                - Any unique object (physical or digital) used to verify identity
        - Inheritance factor
            - *Biometrics*
                - Authentication scheme that verifies a user's identity based on physical characteristics

        - Authentication protocols
            - *Remote Authentication Dial-In User Service (RADIUS)*
                - Internet standard protocol that provides authentication, authorization, and accounting (AAA) services
            - *Terminal Access Controller Access-Control System (TACACS)*
                - Provides AAA services for remote users
                - TACACS+
                    - More secure and scalable than RADIUS
            - Lightweight Directory Access Protocol (LDAP)
                - TCP/IP-based directory service protocol

    - *Kerberos*
        - Authentication service based on a time-sensitive ticket-granting system
        - *kinit command*
            - Authenticate Kerberos ticket if successful
            - subcommands
                - kpassword
                    - Change the user's Kerberos password
                - klist
                    - List the user's ticket cache
                - kdestroy
                    - Clear the user's ticket cache
                - klist –v
                    - Verify ticket

- *Privilege Escalation*
    - User is given access additional resources or functionality
    - While changing a permission (SUID/SGID), consider using the lowest permissions needed for the task
    - *Chroot Jail*
        - Way to isolate a process and its children from the rest of the system

- *Encryption*
    - Cryptographic technique that converts data from plaintext form into coded or ciphertext
    - *Decryption*
        - Converts ciphertext back to plaintext
    - Algorithm (Cipher)
        - Responsible for the conversion process
    - Encryption is one of the most fundamental cybersecurity techniques for upholding the confidentiality of data
    - Data at rest encryption
        - *Full Drive/Disk Encryption (FDE)*
            - Encrypts a storage drive, partition, or volume using hardware/software utilities
        - *File Encryption*
            - Encrypts individual files and folders using software utilities

- *Linux Unified Key Setup (LUKS)*
    - Used to encrypt storage devices in a Linux
    - LUKS standardizes the format of encrypted devices
    - *shred command*
        - Used to securely wipe a storage device
    - *cryptsetup command*
        - Used as the front-end to LUKS and dm-crypt
        - LUKS extensions
            - isLuks
                - Identify if a device is a LUKS device
            - luksOpen
                - Open a LUKS storage device
            - luksClose
                - Remove a LUKS storage device
            - lucksAddKey
                - Associate new key with a LUKS device
            - luksDelKey
                - Remove key material from a LUKS device
        - Syntax
            - `cryptsetup [options] {action} [action arguments]`

- *Hashing*
    - Transforms plaintext input into an indecipherable, fixed-length output

- Best Practices in Network Configurations
  - Enable SSL/TLS
  - Configure SSH
  - Change service defaults (SSH and HTTP/S)

- Security through Obscurity
- Setting up SSL and TLS on an Apache Server
  - Generate OpenSSL
  - Download and install mod_ssl package (/etc/httpd/conf.d/ssl.conf)
  - Point SSLCertificateFile
  - Point SSLCertificateKeyFile
  - Restart Apache
  - Open the browser and verify certificate

- Best practices for managing user access
  - Protect the boot loader configuration with a password
  - Enable password protection in the system's BIOS/UEFI
    - *lsmod command*
      - Search for USB storage an any dependent modules
    - *modprobe -r command*
      - Unload relevant modules from kernel
    - /etc/modprobe.d/
      - Create a block list file
  - Ensure user IDs are not being shared
  - Establish a public key infrastructure
  - Restrict access to cron (Linux job scheduler)

        - Disable the use of Ctrl+Alt+Del
        - Enable the auditd service
        - Add a banner message to /etc/issue
        - Separate operating system data and other types of data
        - Monitor regularly the Common Vulnerabilities and Exposures (CVE) database
        - Harden the system by disabling or uninstalling unused and/or insecure services

- **Identity and Access Management**
    - IAM
        - Identity and Access Management
        - Security process that provides identity, authentication, and authorization mechanisms

    - SSH Protocol
        - Supports many authentications method
        - *Public Key Authentication*
            - Used for interactive and automated connections
            - Improves security
            - Usability benefits such as single sign-on
            - Automated password less login
        - Files used to configure

- Most files are found in the ~/.ssh/ Directory
  - Contains files related to SSH keys
    - id_rsa
      - Contains the user's private key
    - id_rsa.pub
      - Contains the user's public key
    - authorized_keys
      - Lists the public keys that server accepts
    - known_hosts
      - Contains the lists the public keys that the client accepts
    - config
      - Configures SSH connection settings
- Commands used when working with ssh files
  - *ssh-keygen command*
    - Generate public/private key pair
  - *ssh-copy-id command*
    - Append user's public keys to remote server's authorized_keys file
  - *ssh-add command*
    - Add private key identities to the SSH key agent

- /etc/ssh/sshd_config File
  - Used to configure an SSH server
  - Settings that can be configured in this file

- *PasswordAuthentication*
    - Used to enable or disable password-based authentication
- *PubkeyAuthentication*
    - Used to enable or disable public key-based authentication
- *UsePAM*
    - Enables or disables support for Pluggable Authentication Modules (PAM)
- *Port*
    - Used to change the port number to bind the SSH service
- *SyslogFacility*
    - Used to change the logging level of SSH events
- *ChrootDirectory*
    - Used to reference a chroot jail path for a user
- *AllowUsers/AllowGroups*
    - Used to enable user-specific access by allowing the specified users or groups access over SSH
- *DenyUsers/DenyGroups*
    - Used to restrict the specified users or groups from accessing the server over SSH
- *PermitRootLogin*
    - Used to enable or disable the ability for the root user to log in over SSH

- *TCP Wrapper*
  - Checks the allowed and denied hosts before permitting the host to connect with the SSH service
  - Configure in
    - /etc/hosts.allow File
      - Allow host
    - /etc/hosts.deny File
      - Deny host

- *Pluggable Authentication Modules (PAM)*
  - Used to help applications make proper use of user accounts in Linux
  - PAM configuration files are located in /etc/pam.d/ directory
  - Each file has a directive which includes
    - *Module Interface*
      - Defines functions of the authentication and authorization process contained within a module
    - *Control Flag*
      - Indicates what should be done upon a success or failure of the module
    - Module Name
    - *Module Argument*
      - Additional options that can pass into the module
    - Example: password required pam_cracklib.so retry=5
  - Module interfaces
    - *Account Module*
      - Checks user accessibility

        - *Auth*
            - Used to verify passwords and set credentials (Kerberos tickets)
        - *Password*
            - Used to change and verify passwords
        - *Session*
            - Configures and manages user sessions
    - Control flags tell PAM what to do with the result
        - *Optional*
            - Module result is ignored
        - *Required*
            - Module result must be successful for authentication to continue
        - *Requisite*
            - Notifies the user of the first failed required/requisite module
        - *Sufficient*
            - Module result is ignored upon failure
    - Examples of Pam policies
        - password requisite pam_pwquality.so local_users_only
            - Requires the user to entera strong password
        - password requisite pam_pwhistory.so remember=90
            - Enforce a password history for 90 days
        - password sufficient pam_unix.so sha512 use_authtok
            - Allow the module not to do any password checks

        - Pam modules
            - pam_faillock
                - Recommended as it is newer and improves upon pam_tally2
            - pam_tally2
                - Supports user lockout when authentication is done
                - Place user lockout directives in
                    - /etc/pam.d/password-auth
                    - /etc/pam.d/system-auth
        - pam_ldap module
            - Specifies other directives (log in/access resources)
            - add into to the /etc/pam.d/common file
        - /etc/securetty
            - Determines the controlling terminals the root user has access to
            - Configure this to determine what terminals a root user can use to access PAM

    - PKI can be publicly available or maintained privately by an organization
        - Components of the PKI system
            - *Digital Signature*
                - Encrypted message digest with a user's private key
            - *Digital Certificate*
                - Electronic document that associates credentials with a public key

            - *Certificate Authority*
                - Issues digital certificates for entities and maintains the associated private/public key pair
            - *Certificate Signing Request (CSR)*
                - Message sent to the certificate authority in which an entity applies for a certificate
        - *openssl command*
            - Open-source implementation of the SSL/TLS protocol for securing data in transit using cryptography
            - OpenSSL is one of the most common tools for generating and managing components of a PKI
            - Syntax
                - openssl [subcommand] [options]

    - VPNs
        - *Internet Protocol Security (IPSec)*
            - Used to secure data traveling across the network or the Internet
            - Creates VPNs in two modes
                - Transport Mode
                    - Packet contents are encrypted, whereas the header is not
                - Tunnel Mode
                    - Both the packet contents and header are encrypted

- *StrongSwan Utility*
    - Can set up username and password authentication and generate digital certificates
    - /etc/strongswan/ipsec.conf
        - Contains the main configuration file for StrongSwan
    - /etc/strongswan/ipsec.secrets
        - File where user accounts are configurable

- SSL/TLS
    - Used as a VPN authentication and encryption protocol
- *OpenVPN*
    - Supports password-based, certificate-based, and smart card-based authentication mechanisms for clients
    - Configuration files are stored in the /etc/openvpn/ directory

- Datagram Transport Layer Security (DTLS)
    - Implements SSL/TLS over datagrams

- Troubleshooting
    - Users must have set up proper credentials and transmit them to SSH or VPN
    - Check if user remote connection attempts are triggering a policy violation
    - Sign on with the local account (service issue or networking issue)
    - Ensure the user identities are correctly configured

        - Lax PAM policies are leading to unauthorized users accessing resources they shouldn't
        - Privileged access should be granted on an as-needed basis

- SELinux or AppArmor
    - Access Control Models
        - *Mandatory Access Control (MAC)*
            - System-enforced access control based on subject clearance and object labels
        - *Context-Based Permissions*
            - Permission scheme that defines various properties for a file or process
            - SELinux and AppArmor are two types of Context-Based Permissions
        - *Discretionary Access Control (DAC)*
            - Each object has a list of entities that are allowed to access it

    - *SELinux*
        - Default context-based permissions scheme provided with CentOS and Red Hat Enterprise Linux
        - Enforces MAC
        - 3 main contexts for each file and process
            - *User*
                - Defines what users can access the object

            - Most common user types
                - unconfined_u
                    - All users
                - user_u
                    - Unprivileged users
                - sysadm_u
                    - System administrators
                - Root
                    - Root user
        - *Role*
            - Permits or denies users access to domains
            - object_r
                - the role that applies to files and directories
        - *Type*
            - Groups objects together that has similar security requirements or characteristics
        - *Level*
            - Optional
            - Describes the sensitivity level called “Multi-level security”
    - Modes
        - Disabled
            - SELinux is turned off and the DAC method will be prevalent
        - Enforcing
            - SELinux security policies are enforced

- Permissive
  - SELinux is enabled but the security policies are not enforced
- Types of policies
  - Targeted Policy
    - Default SELinux policy used in Red Hat Enterprise Linux and CentOS
  - Strict Policy
    - Every system subject and object is enforced to operate on MAC
    - Higher level of security
- Commands for SELinux
  - *semanage command*
    - Configure SELinux policies
  - *sestatus command*
    - Get SELinux status
  - *getenforce command*
    - Display SELinux mode
  - *setenforce command*
    - Change SELinux mode
    - Examples
      - setenforce 1
        - Enable enforcing mode
      - setenforce 0
        - Enable permissive mode
  - *getsebool command*

- Display the on or off status of Boolean values

- *setsebool command*
    - Change the on or off status of a SELinux Boolean value
- *ls -Z command*
    - List directory contents
    - example
        - `ls -Z {file or directory name}`
        - List directory contents along with each object's security context
- *ps -Z command*
    - List running process
    - Check the context of a specific process
        - `ps –Z {PID}`
- *chcon command*
    - Change the security context of a file
    - Basic Syntax
        - `chcon {-u|-r|-t} {context value} {file or directory name}`
- *restorecon*
    - Restore the default security context
    - Restore Objects
    - Syntax
        - `restorecon {file or directory name}`

- Violations
  - Violation occurs when an attempt to access an object or an action goes against an existing policy
  - *sealert command*
    - makes sure all alert messages are sent to the logs
  - *audit2why command*
    - Allows you to see the violations in the logs
    - Translates violation into more human readable form
  - *audit2allow command*
    - Used to gather information from the denied operations log
    - Will generate SELinux policy allow rules for denied operations
    - `audit2allow -w -a`
      - Read the audit log and display the human-readable description of the blocked activity
    - `audit2allow -a -M [RuleName]`
      - Generate a loadable module to allow the activity to occur
      - Creates two files in the directory
        - RuleName.pp
          - Policy package file
        - RuleName.te
          - Type enforcement file

            - semodule -i {rulename.pp}
                - Loads the two files to install the rule into SELinux

- *AppArmor*
    - Alternative context-based permissions scheme and MAC implementation for Linux
    - Difference between AppArmor and SELinux
        - AppArmor
            - Works with file system objects
            - Easier to work with
        - SELinux
            - References inodes directly
            - More difficult to configure
    - /etc/apparmor.d Directory
        - contains profiles for AppArmor
    - Main rules that can be configured
        - *Capabilities*
            - Provide the executable in question access to system functionality
        - *Path Entries*
            - Enable the executable to access a specific file on the file system
    - Modes
        - Complain Mode
            - Profile violations are logged but not prevented

- Enforce Mode
  - Profile violations are both logged and prevented

▪ Tunables
- Mechanism for tuning configuration in AppArmour without profile adjustments
- Stored in
- /etc/apparmor.d/ tunables/home
  - Contains the most common tunable to adjust

▪ Commands
- *apparmor_status*
  - Display the current status
- *aa-complain*
  - Place a profile in complain mode
- *aa-enforce*
  - Place a profile in enforce mode
- *aa-disable*
  - Disable profile
- *aa-unconfined*
  - List processes with open network sockets

▪ AppArmor is configured to reduce the potential attack surface and provide greater in-depth defense

▪ AppArmor can only do so much to protect against exploits in application codes

- **Firewalls**
    - *Firewall*
        - Program interface between a private network and the Internet

    - Main Generations
        - *Packet Filters*
            - Make decisions based on rules that correspond to network packet attributes
            - Packet filtering firewalls are also called stateless firewalls
        - *Stateful*
            - Identifies past traffic related to a packet
        - *Application Layer Firewall*
            - Inspects the contents of application layer traffic

    - *Stateless Firewall's ACL*
        - Allows or denies packets based on various factors
        - Used by all of the main generations of firewalls
        - Actions taken when a packet matches a rule
            - *Accept*
                - Traffic is allowed through the firewall and sent to its destination
            - *Reject*
                - Traffic is blocked at the firewall and the firewall notifies the sender

        - *Drop*
            - Traffic is blocked at the firewall and does not notify the sender

- Tools
    - *iptables*
        - Applies to a certain context and consists of rule sets (chains)
            - packets are checked against rule sets one by one
                - if it doesn't match the first rule set, it is then checked against the next rule set, like links in a chain
                - Continues until it passes through all of the rules or matches one of them
            - If the packet matches a rule, it can be accepted, rejected, or dropped
        - Syntax for running iptables
            - `iptables [option] [-t table] [commands] {chain/rule specification}`
        - Have 5 tables that can be activated by default in your kernel
            - *Filter table*
                - Default table used for typical packet filtering functionality
            - *Nat table*
                - Used to implement Network Address Translation rules

- *Mangle table*
  - Used to alter the packets' TCP/IP header
- *Raw table*
  - Used to configure exceptions involved in connection tracking
- *Security table*
  - Used to mark packets with SELinux security contexts

- Different distros allow for persistence of iptables when rebooting
  - *CentOS/RHEL Distros*
    - Install the iptables-services package and issue the service iptables save command
  - *Debian-based Distros*
    - Install the iptables persistent package
    - Iptables-persistent service will automatically run at boot and load rules after installation
- Enabling logging Syntax
  - Create a new chain
    - `iptables -N LOGCHN`
  - Ensure all incoming packets not processed by any prior rules will jump to the LOGCHN chain
    - `iptables -I INPUT -j LOGCHN`
  - Performs the actual dropping of packets
    - `iptables -I LOGCHN -j DROP`

        - Events for iptables are written to the /var/log/messages or /var/log/kern.log files

- *Uncomplicated Firewall (UFW)*
    - iptables management tool
    - Makes the iptables service easier to configure
    - Examples of rules/commands
        - ufw allow http/tcp
            - Allow rule for HTTP
        - ufw logging
            - Turn on logging
        - ufw enable
            - Enable firewall
    - Syntax
        - `ufw [options] {actions}`
    - For more detailed control, edit one of the following files
        - /etc/default/ufw
            - Configure high-level settings like policy defaults and kernel module usage
        - /etc/ufw/ directory
            - Contains more granular configuration files

- Firewall Daemon (firewalld)
    - Used to dynamically manage a firewall without requiring a restart
    - Firewall zones are the rule sets that apply to network interfaces
    - Default zones have different levels of trust

- Example
  - drop
    - Zone with the lowest level of trust

- Commands
  - firewall-cmd
    - Configure firewalld by querying, adding, modifying, and deleting zones and services as desired
    - Syntax
      - `firewall-cmd [options]`
    - Options
      - --get-zones
        - Lists Available firewalld Zones
      - --zone=dmz --list-all
        - Lists Details dmz Zone
      - --zone=dmz – change-interface=<device ID>
        - Add Specified Interface to the dmz Zone
      - --zone=dmz --add-service=http
        - Add HTTP Service to the dmz Zone
      - --zone=dmz --add-port=21/tcp
        - Add TCP Port 21 (FTP) to the dmz Zone
      - --zone=dmz --remove-service=http
        - Remove HTTP Service from the dmz Zone
      - --zone=dmz --remove-port=21/tcp
        - Remove TCP Port 21 (FTP) from the dmz Zone

            - --reload
                - Reloads Zone's Configuration
            - --permanent
                - Persist Change
    - *Netfilter*
        - Handles packets that traverse a network interface
        - Netfilter (nftables) was designed as a replacement for iptables and is installed by default on Debian

- IP Forwarding
    - Enables incoming traffic on one network interface to another

- IP Set
    - Stored collection of IP and MAC addresses, network ranges, port numbers, and network interface names
    - iptables tool leverages IP sets for more efficient rule matching
    - *ipset command*
        - Create and modify IP sets
        - syntax
            - ipset [options] {command}
        - test subcommand
            - used to test the entry exists
            - often linked to ports
                - Internet Assigned Numbers Authority (IANA)
                    - Trusted Ports/Privileged Ports
                        - 0–1023

        - The test subcommand is used to test the entry exists
        - Check the firewall rule set to ensure there are no overtly blocked ports in the system
        - ACL can be configured to only block specific source ports

- *Intrusion Prevention System (IPS)*
    - Security appliance
    - Monitors and evaluates a system for attack signs and blocks traffic that it determines malicious
    - second layer of defense that monitors traffic
    - Common third party IPSs
        - *DenyHosts*
            - Protects SSH servers from brute force password cracking attacks
            - /etc/denyhosts.conf
                - Primary configuration file for DenyHosts
                - Settings that can be adjusted
                    - ADMIN_EMAIL
                        - Define email address to send alert
                    - BLOCK_SERVICE
                        - Define services to be blocked from unauthorized users
                    - DENY_THRESHOLD_VALID
                        - Defines number of times a user can attempt to log in

- *Fail2ban*
  - Monitors log files with an authentication component
  - Primary Configuration file
    - /etc/fail2ban/jail.conf
    - Copy file to /etc/fail2ban/jail.local or make a custom .conf file within /etc/fail2ban/jail.d/
  - Settings that can be configured
    - *Bantime*
      - Defines a host being blocked from accessing a resource
    - *Maxretry*
      - Defines the number of times a host can fail before being blocked
    - *ignoreip*
      - Defines a whitelist of accepted hosts

- **Logging Services**
  - *Operating System Log*
    - Provides a wealth of diagnostic information about a computer
  - *System Log*
    - Records of system activities and events
  - *Remote Logging*
    - Centralized logging server that receives and processes syslog data
  - /var/log/ Directory
    - location for System Logs storage
    - Subdirectories and their contents

        - Debian based
            - /var/log/syslog
                - All types of system events
            - /var/log/auth.log
                - Authentication messages
        - Red Hat/CentOS based
            - /var/log/messages
                - General non-critical system events
            - /var/log/secure
                - Authentication messages
        - Both Debian and CentOS based
            - /var/log/kern.log
                - Kernel messages
            - /var/log/ [application]
                - Miscellaneous applications (cron, firewalld, mailog) messages

- *Log Rotation*
    - Practice of creating new versions of a log file
    - *logrotate command*
        - Used to perform automatic rotation of logs
    - Can also be configured using
        - /etc/logrotate.d/
            - Log Rotation Behavior

- To change configurations on the rsyslogd Service edit/etc/rsyslog.conf
  - /etc/rsyslog.conf is laid out in columns
    - The first column lists message facilities and/or severities
      - severities are defined in word format, not numbers
    - The second column defines actions for the messages

- syslog-ng
  - Replacement for syslogd

- Centralization of log data
  - The syslog standard is not universally supported on all platforms
    - Third party Agents must be used to facilitate actions between syslog and non-syslog platforms
    - *Agent*
      - A software program that acts on behalf of some other program or service

- Logging Commands
  - *journalctl command*
    - Enables the viewing and querying of log files
    - Prints a journal log
    - Queries journald log data used in syslogd or rsyslogd
      - configure journald log data by altering etc/systemd/journald.conf
    - Syntax
      - journalctl [options] [matches]

        - Options
            - -n {number of lines}
                - Specify number of lines of journal logs to display
            - -o {output format}
                - Specify format of the output
            - -f
                - Display most recent journal entries
            - -p
                - Filter journal log output by severity
            - -u
                - Filter journal log output by the name of service
            - -b [boot ID]
                - Show log message from Boot ID specified
        - systemd Journal
            - Stores logs in memory
            - cleared out on reboot
                - To make them persistent, create a /var/log/journal/ directory
    - *last command*
        - Displays the user's history of login and logout events
        - Syntax
            - `last [options`
        - Example
            - last 1
                - will show the history of login and log out events for terminal 1

        - *lastlog command*
            - Lists all users and the last time a user logged in
            - Retrieves Information from /var/log/lastlog

- **Backup, Restore, and Verify Data**
    - *Backup Strategy*
        - Data protection that directs data backup and recovery policy actions
        - *Backup*
            - Copy of data that exists in another logical or physical location
        - Types of backups
            - *Full Backup*
                - All selected files are backed up
            - *Differential Backup*
                - Focuses on the files that have changed since the last full backup
            - *Incremental Backup*
                - Only backs up the changed data found in files
            - *Snapshot*
                - Records the state of a storage drive at a certain point in time
            - *Image-Based Backup*
                - Saves the state of an operating system in an image file format
            - *Cloning*
                - Copies all the contents of a storage drive to another storage medium

- *tar command*
  - Tape archive
  - Enables the creation of data archives
  - Syntax
    - `tar [options] {file names}`
  - Example
    - `tar -xvf`
      - Restores the contents of a source file

- *dar command*
  - disk archiver
  - Offers more backup and archiving functionality
  - Examples
    - `dar -R mydata -c full.bak`
      - Full Backup
    - `dar -R mydata -c diff1. bak -A full.bak`
      - Differential Backup
  - Options
    - -x extract
      - extract
      - Recover a Backup
    - -w
      - Overwrites Changes

- *cpio command*
  - Copies files to and from archives

- Modes
  - Copy-out
    - Used to copy files into an archive
  - Copy-in
    - Used to copy files from an archive
  - Copy-pass
    - Used to copy files from one directory tree to another
- Examples
  - `ls | cpio -o > dir_archive`
    - Archive Directory Content
  - `cpio -i < dir_archive`
    - Extract Archive (copy-in)

- *dd command*
  - disk duplicate
  - Copies and converts files to be transferred from one type of media to another
  - Options
    - if={file name}
      - Specify file to be read
    - of={file name}
      - Specify file to be written
    - bs={bytes}
      - Block size to read and write in bytes
    - count={count}
      - Specify number of blocks to be written

- status={level}
  - Specify information to print to standard error
- Syntax
  - `dd [options] [operands]`
  - Examples
    - `dd if=/dev/sda of=/dev/sdb`
      - Copy Full Backup of Storage
    - `dd if=/dev/sda2 of=/backup/full.dd`
      - Copies full backup to a file

- *mirrorvg command*
  - Creates copies of logical volumes in a specified logical volume group
  - Syntax
    - `mirrorvg [options] {volume group}`

- *mklvcopy command*
  - Mirrors individual logical volumes in a volume group

- *lvcreate command*
  - creates mirrors of logical volumes
  - Option
    - -m#
  - Example
    - `lvcreate -L 10G -ml -n mirrorlv volgr`
      - creates one 10 gigabyte mirror called mirrorlv

- off-site
  - Physical location outside of the main site that stores copies of data

- Tools for transporting data safely
  - scp tool
    - Used to copy data to or from a remote host over SSH
  - Secure File Transport Protocol (SFTP)
    - Uses an SSH tunnel as a transportation mechanism to encrypt data
  - rsync
    - Used to copy files locally and to remote systems

Bash Scripting

Objective 3.1: Given a scenario, create simple shell scripts to automate common tasks.

- **The Bash Shell Environment**
  - Basic Terms
    - *Shell Environment*
      - Mechanism by which Bash maintains settings and other behavioral details
    - *Shell Spawning*
      - Process of creating a new session
    - *Script*
      - Any computer program that automates the execution of tasks
  - *Variables*
    - Entity whose values change from time to time
    - Set variables syntax
      - `VAR=value`
      - Example: create a variable called MYVARIABLE
        - MYVARIABLE=123
      - To view the variable syntax
        - `${VARIABLE NAME}`
      - To view contents of MYVARIABLE syntax
        - `echo ${MYVARIABLE}`
    - *Environment Variable*
      - Variable that is inherited from parent shell processes and passed to the child processes

- Default Variables
    - HOSTNAME={hostname}
        - Specifies the system hostname
    - SHELL={shell path}
        - Specifies the system shell path
    - MAIL={mail path}
        - Specifies the mail path storage
    - HOME={home directory}
        - Specifies user's home directory
    - PATH={user path}
        - Specifies the search path
    - HISTSIZE={number}
        - Specifies the command history
    - USER={username}
        - Specifies the user's name
- Edit the /etc/locale.conf File to configure environmental variables and assign the locale to the variable
    - Examples
        - LC_*={locale}
            - Collection of Localization Environment Variables
        - LANG={locale}
            - Locale for LC_* variables

        - LC_ALL={locale}
            - Locale for All Options
        - TZ={time zone}
            - System Time Zone

- Commands
    - *export command*
        - used to effectively changes a shell variable into an environmental variable
        - Example:
            - `export SHELL_VAR`
                - makes SHELL_VAR an environmental variable
            - `export SHELL_VAR=”New value”`
                - makes SHELL_VAR an environmental variable and gives it a new value
        - Syntax
            - `export [options] [NAME[=value]]`
    - *env command*
        - Used to run a command with modified environment variables
        - changes it for the session only
        - Syntax
            - `env [options] [NAME=value] [command]`

            - *set command*
                - Use without arguments to print all shell variables, environment variables, and shell functions
                - Syntax
                    - set
            - Differences
                - export
                    - Change the variable value for child processes
                - env
                    - View or change variables for a specific command
                - set
                    - View or change the value of a shell command
        - *Search Path*
            - Sequence of various directory paths to locate files
            - Can be assigned to the PATH environmental variable
                - has a list of all the directory names separated by colons
        - HISTFILESIZE Variable
            - Sets the maximum number of lines in the command history file
            - Default history file value is 1000

    - *alias command*
        - Used to customize the shell environment by generating command-line aliases
        - Can create shortened versions of commands

- *time command*
    - Used to gather information about how long to execute a command
    - gives three pieces of information by default
        - Elapsed real time between invocation and termination
        - User CPU time
        - System CPU time
    - Syntax
        - `time [options] [command]`

- Troubleshooting
    - When adding an alias, check the syntax
    - When executing scripts, add their location to the PATH variable
    - Use the export command to set a variable for all shell child processes
    - Configure environment variables in the ~/.bash_profile file
    - Ensure values are set for any environment variables that a software package has a dependency on

- **Scripting and Programming Fundamentals**
    - Bash
        - Powerful scripting language
        - Bash scripts support modern programming elements (loops and conditional statements)
        - Bash script syntax is similar to CLI
        - #!/bin/bash
            - Instructs the operating system to use the Bash shell interpreter

- Assigning Variable
  - Symbolically associate a piece of information with a name
  - All Bash variables are treated as strings
  - Syntax
    - `variablename='value'`
    - make sure there are no spaces before or after the =

- Substitution or Parameter Expansion
  - Act of referencing/retrieving the value of a variable
  - Takes place after it has been assigned
  - Syntax
    - `$variablename`

- Comparisons
  - Operators
    - Objects that can evaluate expressions in different ways
  - Operands
    - Values being operated on
  - Types of Operators
    - Arithmetic operators
      - Includes addition, subtraction, multiplication, division, and other operations
      - Example, adding variables syntax
        - `$((var1 + var2))`

        - *Comparison operators*
            - Includes checking if operands are equal
            - use bracket [], not parentheses ()
            - Example of comparison syntax
                - `[$var1 -ge $var2]`
        - *Logical operators*
            - Connect multiple values (AND, OR, and NOT)
            - Example of testing two variables
                - `'AND' , [$var1 -ge $var2] && [$var3 -le $var4]`
            - OR symbol is ||
            - NOT symbol is !=
        - *String operators*
            - Used in operations that manipulate strings
            - Example: concatenate two variables syntax
                - `$var1$var2`

- String Literal
    - Any fixed value that represents a string of text within source code
    - Can use single or double quotes
        - When using variables, use double quotes
        - Always put quotes around strings being assigned to a variable

- Reserved Characters
    - *Escape Character*
        - Used to remove special meaning from special characters

            - Escape character in Bash is a single backlash (\)
                - Also us single quotes
        - *array*
            - Enables to store multiple values in a single variable
            - Compound assignment in Bash arrays uses parentheses with a value separated by a space
        - *function*
            - Block of code that can reuse to perform a specific task
            - put code between { }
        - #
            - Every character after it is part of a comment

    - Metacharacters
        - Special characters that the Bash shell will interpret in a certain way
        - >
            - Output redirection
        - >>
            - Output redirection (in different manner)
        - <
            - Input redirection
        - <<
            - Input redirection (here documents)
        - |
            - Piping
        - "
            - Weak string literals

- ‘
  - Strong string literals
- `
  - Breaking out a string literal
- \
  - Escaping characters
- =
  - Variable assignment
- $
  - Variable substitution and other types of shell expansion
- #
  - Commenting
- ||
  - Logical OR operations
- &&
  - Logical AND operations
- *
  - Wildcard matching
- ?
  - Wildcard matching applied in a single character matched
- [ ]
  - Wildcard matching applied any characters between brackets matched
- { }
  - Parameter substitution and arrays

    - ( )
        - Grouping commands
    - &
        - Running a process in background
    - ;
        - Separating multiple commands on the same line
    - !
        - Referencing command history

- exit Code/Exit Status
    - Programs can pass a value to a parent process while terminating
        - Status code of 0
            - Successful
        - Exit code 1 or higher
            - Error
    - Redirection and Piping (CLI)
        - Determine where you want stdout/stderr/ stdin to go

- Shell Expansion
    - Process by which the shell identifies special tokens and substitutes values for them
    - Variable Substitution
        - Identifies the $ special character and expands into its actual value
        - A type of shell expansion

- Expands in a specific order
    1. Brace expansion
    2. Tilde expansion
    3. Parameter and variable expansion
    4. Arithmetic expansion
    5. Command substitution
    6. Word splitting
    7. Filename expansion

- Expansion allows for the use of output from a command to become input for the next command
    - increases the complexity and functionality of commands

- Globbing
    - Used for matching or expanding specific types of patterns
    - Used for searches
    - Uses wildcard charachters
        - Asterisk (*)
            - Used to match any number of characters
        - Question mark (?)
            - Used to match a single character
        - Square brackets ([ ])
            - Used to match any of the characters listed

- Positional Parameter
  - Variable within a shell script that is assigned to an argument when the script is invoked
  - Example:
    - `dion.sh arg1 arg2 arg3`

- exec
  - Replaces the bash with the command to be executed
  - prevents a user from returning to a parent process inside your script

- source
  - Used to execute another command within the current shell process

- File extensions in Linux are optional
  - Adding .sh as an extension to a shell script does not imbue the script with any special meaning

- Use two permissions for each user that needs to run the script
  - Execute (x) bit on the script
  - Write (w) and execute (x) bits on the directory containing the script

Task Automation

Objectives:

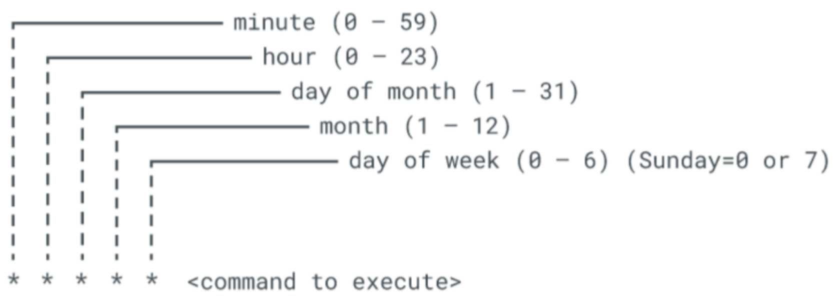
- 1.4: Given a scenario, configure and use the appropriate processes and services.
- 1.6: Given a scenario, build and install software.
- 3.2: Given a scenario, perform basic container operations.
- 3.3: Given a scenario, perform basic version control using Git.
- 3.4: Summarize common infrastructure as code technologies.
- 3.5: Summarize container, cloud, and orchestration concepts.

- **Schedule Jobs**
    - Bash Scripting
        - Powerful part of system administration and development
        - Bash contains sets of commands which are used to automate the execution of tasks
        - Utilities that allow the scheduling
            - Cron command
                - Repetitive task
            - At service
                - One-time task
        - *at command*
            - Schedules a command to run once at a particular time
            - Syntax
                - `at [options] {time}`
            - Options
                - -m
                    - Send mail to the user

        - -M
            - Prevent sending mail to the user
        - -f {file name}
            - Read a job from a file
        - -t {time}
            - Run the job at the specified time value
        - -v
            - Time the job will be executed
                - Noon
                    - 12 P.M.
                - Teatime
                    - 4 P.M.
                - Midnight
                    - 12 A.M.
                - 3 minutes from now
                    - now + 3 minutes
                - 1 hr from now
                    - now + 1 hour
    - Additional time commands
        - *atq command*
            - Used to view the current queue of tasks scheduled by at command
        - *atrm command*
            - Used to delete a scheduled task

- *Cron Daemon*
  - Used to manage scheduled tasks called cronjobs
  - *Crontab command*
    - can create, view, and delete crontab files
    - Options
      - -e
        - Edit crontab for current user
      - -l
        - View crontab for current user
      - -r
        - Delete current crontab file
      - -u
        - Create crontab file for specified user
    - Syntax
      - `crontab [options] [file/user]`
  - Crontab also used by system administrators to do tasks at routine intervals inside Linux
    - * * * * * /path/to/command
      - 45 23 * * 6 /home/user/scripts/exportdump.sh

Each line in this file represents a job, and is formatted as follows:



```
┌───────────── minute (0 - 59)
│ ┌───────────── hour (0 - 23)
│ │ ┌───────────── day of month (1 - 31)
│ │ │ ┌───────────── month (1 - 12)
│ │ │ │ ┌───────────── day of week (0 - 6) (Sunday=0 or 7)
│ │ │ │ │
│ │ │ │ │
│ │ │ │ │
* * * * *  <command to execute>
```

Cron Job

- Examples
  - * 20 * * 1-5 /path/to/command
    - 8 P.M., Monday through Friday
  - 15 2 * * * /path/to/command
    - 2:15 A.M., daily
  - 30 4 1 * * /path/to/command
    - 4:30 A.M. on the first day of each month
- Scheduled crontab Files found in
  - /etc/cron.d/ directory
  - /var/spool/cron/ directory
- Crontab files are user specific
  - The Root User and Services can use the /etc/cron.d/directories file to schedule system-wide tasks
  - Regular users are not allowed to populate the /etc/cron directories
  - Standard users can schedule tasks in a personal directory located at /var/ spool/cron
    - /etc/cron.hourly
    - /etc/cron.daily
    - /etc/cron.weekly
    - /etc/cron.monthly

- **Version Control using Git**
  - *Git*
    - Mature, actively maintained open-source project
    - Syntax
      - git [options] {subscommand}
  - *Git Repository*
    - Storage area where versions of code and related files are stored
  - Subcommands
    - *config*
      - Set options for repository or Git users
    - *init*
      - Create Git repository or reinitialize an existing one
    - *clone*
      - Create a copy of an existing repository
    - *add*
      - Add files to be tracked by Git repository
    - *commit*
      - Update the Git repository with changes (snapshot)
    - *status*
      - Display status of the repository
    - *branch*
      - Manage branches (after changes)
    - *merge*
      - Integrate changes into a master branch

- *pull*
  - Acquire and merge changes
- *push*
  - Upload local working copy of a repository to a remote repository
- *log*
  - Display the changes made (local repository)
- *checkout*
  - Switch to a specific branch
- *tag*
  - Add a tag to your git repository
- *rebase*
  - one of two utilities that specialize in integrating changes from one branch to another
  - moves or combines a sequence of commits to a new base
  - rewrites historical features
  - maintains the lineage history
- *merge*
  - one of two utilities that specialize in integrating changes from one branch to another
  - merges two bases into one
    - the new one will be the base used going forward

- Process for making changes
    - Configure global settings including username
        - Example:
            - `git config –global user.name 'User'`
    - Create a directory where the project will reside
        - Example:
            - `mkdir /dev-project`
        - Use cd command to change into /dev-project
    - Change into the created directory and then initialize it with Git to designate it as a Git repository
        - Example:
            - `git init /dev-project`
    - Add project files to the repository
        - use clone command to create a working copy
        - Example:
            - `git clone /dev-project`
    - Add project files into the Git tracking system, specifically, myfile
        - Example:
            - git add myfile
    - Commit the changes to take a snapshot of the project
        - enter a note that indicates what has been done
        - Example:
            - `git commit -m 'Initial commit'`

            - See the status of the Git repository
                - Example:
                    - `git status`

    - Branching
        - Feature available in most modern version control systems
        - A pointer to the snapshot of the different changes you have made
        - Spawn a new branch to encapsulate your changes
            - Do this so you can work on the master branch without affecting all of the other developers
            - Syntax to create Branch Master Copy
                - `git branch newbranch`
        - When you are done, you need to merge the changes back into the master branch
            - Syntax to change and save to master branch
                - `git merge newbranch`
        - Allows collaboration
            - Requires a process flow
                - To pull in another developer's code and changes and put it in a remote repository
                    - `git pull other branch`
                - Then push your changes into the remote repository
                    - `git push <remote repository> mybranch`
            - Changes made locally can be uploaded to the central repository using the Git push command

                - *git log command*
                    - shows all of the changes that have happened over a specific amount of time
                    - used for troubleshooting
            - *checkout subcommand*
                - allows navigation and switching between branches of a project
                - syntax
                    - `git checkout specificbranch`

    - *.gitignore File*
        - Identifies files that should be ignored during a commit action

    - **.git/ Directory*
        - Contains all files Git uses to manage version control for a project

- **Orchestration Processes and Concepts**
    - Basics
        - *Orchestration*
            - Automation of multiple steps in a deployment process
        - *Automation*
            - Process of accomplishing a configuration task without human intervention
        - Difference between the two:
            - Automation
                - Single task

        - Orchestration
            - Series of tasks
            - Orchestration is the automation of the automations
    - To complete automation and orchestration, you need to break down the process so you know what the workflow looks like

- Rapid elasticity computing would not be possible without orchestration
    - *Resource Orchestration*
        - provisions and allocates resources
    - *Workload Orchestration*
        - management of applications and other cloud work
    - *Service Orchestration*
        - deploys services in the cloud

- Third-party orchestration platform is protection from vendor lock-in
    - *vendor lock-in*
        - having to have all parts of the orchestration from the same vendor

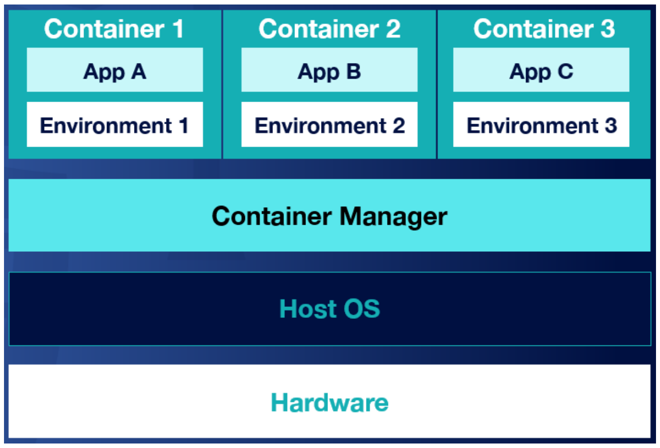
- Tools for Orchestration
    - *Chef*
        - Uses "cookbooks" to deliver configuration declarations to cloud and on-premises managed systems
    - *Puppet*
        - Uses manifest files to define infrastructure as code for application, cloud, and infrastructure orchestration

        - *Ansible*
            - Uses YAML files to create repeatable “playbooks”
        - *Docker*
            - Open platform for developing, shipping, running, and deploying applications using container- based virtualization
        - *Kubernetes*
            - Provides container deployment and application orchestration for cloud and on-premises container environments
        - *OpenStack*
            - Deployed as an IaaS solution to manage cloud resources
        - *GitHub*
            - Service that allows the developers to share code

    - Agent-Based vs. Agentless Orchestration
        - *Agent-Based*
            - Requires a software component to reside on the managed device
        - *Agentless-Based*
            - Does not require additional software to exist on the managed system
    - Orchestration tools
        - Inventory Management of Hardware
        - Virtual machines
        - Operating systems
        - Applications
        - Configurations

- Benefits of Configuration management
  - Consistent configured system
  - Enforced security
  - Service-level agreements
  - Efficient change management

- **Containerization**
  - *Containerization*
    - A type of virtualization applied by a host OS to provision an isolated execution environment for an application



What Containerization looks like

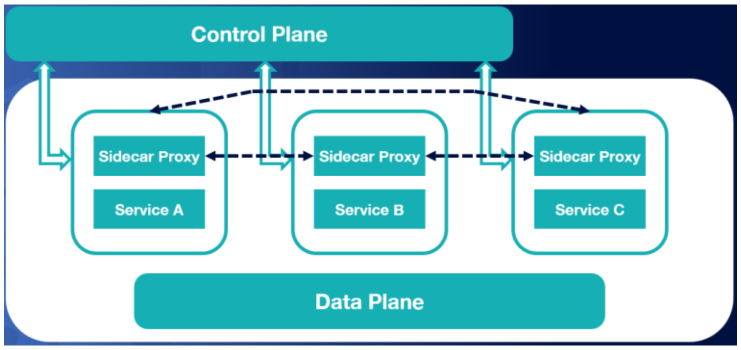
  - Virtualization software
    - *Kubernetes*
      - An open-source system for the automated deployment, scaling, and management of containerized applications
      - *Node*
        - An object that runs the containerized applications

- *Pod*
    - One or more containers that have shared storage and network resources
- *Sidecar*
    - Designed to run alongside the main container or pod
    - Sidecars enhance and extend the functionalities of the main container without modifying its codebase
    - *Ambassador Container*
        - A special type of sidecar container that simplifies the process of accessing data and services outside of a given pod
            - PersistentVolume
            - PersistentVolumeClaim

- Storage
    - Needs a persistent storage device because containers are non-persistent by default
        - To create a persistent storage device use
            - PersistentVolume
            - PersistentVolumeClaim that is bound to that storage

- Container Registry
    - A place to store, manage, and secure container images

- Service Mesh
  - Manages the network traffic between different services, containers, and pods



- **Container Operations**
  - Commands for podman
    - *podman build*
      - Build a container image
    - *podman push*
      - Push a container image to a specified destination
    - *podman pull*
      - Pull a container image from a container registry
    - *podman images*
      - List out the container images available on the local system
    - *podman rmi {Image ID}*
      - Remove a container image

- Podman uses the exact same syntax as Docker
  - replace the word podman with docker and all of the commands above will work in the same way with Docker

- Container Operations
  - Starting/stopping containers
    - `podman start {container image name}`
  - Inspecting containers
    - `podman inspect {container image name}`
  - Listing containers
    - `podman pod list`
  - Deploying existing images
    - use a template to make this easier
    - create a template
  - Connecting to containers
    - `podman attach {container image name}`
  - Logging actions in containers
    - `podman logs {container image name}`
  - Exposing ports for containers
    - `podman create --expose=port`
    - `podman run --expose=port`

- **Sandboxed Applications**
  - Benefits of Sandboxing
    - Improves security
    - Increases application integrity

- Tools to sandbox application
  - *Snap*
    - A bundle that contains an app and its dependencies that work without modification across all Linux distributions
    - To manage and maintain Snaps or applications, run the snapd
    - snap command
      - Used to find a snap or application to install
      - By default, applications are installed under the /snap/bin directory
    - *snapd*
      - Snap daemon
      - The backend daemon that runs the Snaps on a system
  - *Flatpak*
    - Runs in a sandbox that contains everything needed for the programs to operate
    - Syntax to run
      - flatpack [options] {command}
  - *AppImage*
    - A universal package manager where the apps are installed without modifying system libraries or system preferences
    - Create a directory under the home directory to put all the AppImage applications

- **Infrastructure as Code (IaC)**
  - *Infrastructure as Code (IaC)*
    - A provisioning architecture in which deployment of resources is performed by scripted automation and orchestration
    - Infrastructure as Code allows for the use of scripted approaches to provisioning infrastructure in the cloud
  - Robust orchestration can lower overall IT costs, speed up deployments, and increase security
  - *Snowflake Systems*
    - Any system that is different in its configuration compared to a standard template within an IaC architecture
    - Lack of consistency leads to security issues and inefficiencies in support
  - *Idempotence*
    - A property of IaC where an automation or orchestration action always produces the same result, regardless of the component's previous state
    - IaC uses carefully developed and tested scripts and orchestration runbooks to generate consistent builds

- IaC Methodologies
    - *Terraform*
        - A modern method used to provision, change, and version resources on any cloud-based environment using automation and orchestration
        - Commonly used for
            - IaC and multi-cloud deployments
            - Kubernetes management
            - Network infrastructure
            - Virtual machine images
            - Policy as code enforcement
    - *SaltStack (Salt)*
        - A configuration management and orchestration tool commonly used with IaC deployments
        - SaltStack eliminates the manual processes used by legacy IT operations

- **CI/CD**
    - In the past development and deployment was done in a linear fashion
        - Steps
            - Development
            - Testing/Integration
            - Staging
            - Production

        - Drawbacks
            - Slow
            - Internal conflict

    - *CI/CD attempts to speed up development*
        - Common source repository holds the code that is developed
        - Integrated through a continuous integration server
            - Builds
            - Tests
            - Provides results (success or failure of the code)
        - Goes back to developers for the next step

    - *Continuous Integration*
        - A software development method where code updates are tested and committed to a development or build server/code repository rapidly
        - The process of creating, testing, and committing updates can be done multiple times per day
        - Focused on detecting and resolving development conflicts early and often

    - *Continuous Delivery*
        - If you are doing continuous delivery, you must be doing continuous integration

        - A software development method where application and platform requirements are frequently tested and validated for immediate availability
        - Ready for release

    - *Continuous Deployment*
        - A software development method where application and platform updates are committed to production rapidly