Introduction

Your Instructor

- Sander van Vugt, mail@sandervanvugt.nl
- Teaching 15+ live Linux and Open Source courses at this platform
  - Kubernetes Fundamentals
  - Ansible in 4 Hours
  - RHCSA Crash Course
- Author of 60+ books
- Founder of the Living Open Source Foundation
  - livingopensource.net

Poll Question 1

Which of the following reasons to learn Linux best applies to your motives to be here?

- To support my work as a developer
- To support my work as a network engineer / DBA
- To build on my generic skills as IT professional
- As a preparation for a career/certification in Linux
- Other

Poll Question 2

Are you planning to take any Linux Certification?

- No
- RHCSA
- LPI
- Linux+
- LFCS
- Other

Poll Question 3

Is there a specific distribution that you are most interested in?

- Red Hat and related
- Oracle Linux
- Ubuntu and related
- SUSE
- Others
- No preference yet

Poll Question 4

Would you consider yourself an IT professional?

- yes
- no

Poll Question 5

Which part of the world are you from?

- North America
- South America
- India
- Asia (not India)
- Netherlands
- Europe (not Netherlands)
- Africa
- Australia / Pacific

Question Handling Policy

- If you want me to answer your question, post in Q&A please. I will answer it if it meets the requirements, but you may have to wait a bit
- Questions about personal issues "when I do this, I get error message xyz" only in group chat please
- If you just missed something, please watch the recording. I cannot repeat topics I just explained as that wastes valuable time
- Questions about topics that you should have known in group chat only please
- Off-topic questions are welcome in the Q&A at the end of this course

Course Agenda

- Day 1
  - Understanding and Installing Linux and Distributions
  - Connecting to Linux and other OS essentials
  - Essential commands part 1
- Day 2
  - Essential commands part 2
  - Common administration tasks overview

Linux Fundamentals BootCamp

1. Installation

How to get access to Linux?

Getting access to Linux for the best learning experience in this course

- For the best learning experience: install it in a virtual or physical machine
- Alternatively, use a cloud instance: free tier access is offered by major cloud providers
- Not optimal but acceptable: use a generic Linux container
- Not recommended: use WSL. WSL does offer a full Linux layer, but heavily relies on Windows tools and for that reason is not supported in this course

Understanding Distributions

- Being Open Source, all Linux components are free
- A distribution is a collection of components, gathered by a company or a community
- Many distributions exist, the most significant are:
  - Red Hat, CentOS, Fedora
  - Ubuntu, Debian, Linux Mint
  - SUSE
  - Oracle
- In this course CentOS 8 is used
- You may use another Linux distribution also

Installing CentOS 8

- Download the latest version of CentOS from centos.org
- Install in a Virtual Machine (recommended) or on hardware
- Minimal RAM: 2 GB
- Minimal disk: 20 GB
  - A second disk will be needed later in class
- Install using the "Server with GUI" pattern

Quiz Question 1

1. Which of the following Linux distributions is NOT in the same distribution family?

a. Linux Mint
b. Debian
c. Ubuntu
d. SUSE

2. Using Essential Tools

2.1 Logging in

Logging in to Linux

- Graphical or Console?
- Which User Account?
- What is sudo?
- Logging in Remotely

DEMO

- Graphical or Console?
- Which User Account?
- What is sudo?
- Logging in Remotely

2.2 Starting the Command Line

Using Commands

- Commands to get started with
  - **whoami**
  - **hostname**
  - **date**
  - **uname**
  - **passwd**
  - **touch**
  - **last**

2.3 Getting Help

Man pages common elements

- Command summary: shows how to use a command
  - Items between [brackets] are optional
  - If you see { a | b } you must choose
  - And ... means you may have more of the preceding item
- Man pages have sections
  - 1 is for end-user commands
  - 8 is for administrator (root) commands
  - 5 is for configuration files
- Many man pages have examples near the end
- Otherwise, they'll have related items near the end
- Use /sometext to search for sometext
- And use q to get out of the man page

Finding the Right **man** Page

- Man pages are summarized in the **mandb**
- Use **man –k** (or **apropos**) to search the **mandb**
- Type **mandb** if you get "nothing appropriate"

Help Systems Overview

- **man** is the primary source
- **pinfo** shows usage information for some commands
- **command --help** provides short usage overview
- /usr/share/doc contains additional help for some commands

5 Minute Minilab 2

- Find the command that you can use to change the hostname of your computer
- Are there any related commands?
- Use this command to change your computers host name
- Read the help output for **lvcreate** and find which options *must* be used

Quiz Question

3. The command **man -k user** gives "nothing appropriate" as a result. Which of the following commands is likely to fix this problem?

a. updatedb
b. mandb
c. yum install manpages
d. createdb

Quiz Question

2.Which command should you use if you're looking for a command, but you don't know the exact name of the command?

a. man
b. pinfo
c. man -k
d. help

3. Using File Tools

3.1 File Management Basics

Using ls

- Use **pwd** to find the name of the current directory
- **ls** is used to list files and directories
- **ls –a** will show hidden files also
- **ls –l** provides long listing (many details)
- Use **ls –ld /directory** to see properties and not contents of a directory
- **ls –lrt** shows a time-sorted list of files

Using Wildcards

- The **ls** command supports using wildcards:
  - ls a*
  - ls a?*
  - ls a[nm]*
  - ls a[a-e]*

Managing Directories

- **cd** is used to change directory
- **mkdir** creates a directory
- **rmdir** removes an empty directory

3.2 Understanding the Filesystem Hierarchy

Understanding the Linux Filesystem

- Directories are highly standardized on Linux
  - /usr
  - /var
  - /etc
- Complete documentation is in the Filesystem Hierarchy Standard (https://refspecs.linuxfoundation.org/fhs.shtml)
- Regular users have write-access to two directories only:
  - /home
  - /tmp
- Tip! Use **touch filename** to verify write access to a location

3.3 Understanding Absolute and Relative Paths

Understanding Path Names

- An *absolute path* contains the full name from the root directory to a file
  - /var/log/messages
- A *relative path* is related to the current directory and contains the rest that is needed to get to a file
  - log/messages if current directory is set to /var
- In relative paths, .. can be used for "one directory up"
  - If the current directory is /var/cache, ../log/messages points to the /var/log/messages file
- Tip! Use absolute paths to avoid confusion

Quiz Question

Which command do you need to use to see hidden files as well as regular files in the current directory?

- ls -l
- ls -a
- ls -h
- ls -s

3.4 Copying, Moving and Removing Files

Using **cp**

- Use **cp /dir/somefile /somedir** to copy a file
- To have the command fail if /somedir doesn't exist, change to **cp /dir/somefile /somedir/**
- Use **cp –a** to copy all files (including hidden), but use the .* wildcard to include hidden files as well
  - **cp –a ~/.* /tmp/**
- **cp –R** will copy recursively: including subdirectories
- Notice that Path names are important: consider **cp –R /tmp/ .** versus **cp –R /tmp .**

Understanding **mv**

- The **mv** command will move a file
- It is also use to remove a file

5 Minute Lab

- Create a directory /tmp/trash
- Copy all files from your home directory to this directory, including hidden files and subdirectories
- Copy the entire /tmp directory into your home directory, in such a way that after the copying you'll see a directory ~/tmp

3.5 Managing Links

Understanding Links

- A link is a file system entry that refers to another file or directory
- Hard links are pointing to the same inode on the same file system
- Symbolic links are shortcuts and add additional flexibility
  - Symbolic links can exist on a directory
  - Cross-device symbolic links can be used

Quiz Question

Which of the following are advantages of symbolic links over hard links (choose 2)?

a. Symbolic links don't consume additional inode space
b. Symbolic links can be created to a directory
c. Symbolic links can link to a file on a different device
d. Symbolic links cannot get broken

3.6 Finding Files with **find**

Finding Files

- find / -name ”hosts”
- find / -user linda
- find / -size +2G
- find / -user linda -exec cp {} /root/linda \;

Quiz Question

Which of the following find commands allows you to find files bigger than 1 GiB?

- find / -size +1G
- find -size +1G
- find / --size +1G
- find --size +1G

3.7 Archiving Files with **tar**

Understanding **tar**

- **tar** is the Tape ARchiver and was created a long time ago
- Basic use is to compress, extract, or list
  - **tar -cvf my_archive.tar /home**
  - **tar -xvf my_archive**
    - Notice this extracts to the current directory
    - Use **-C** to switch the output path
  - **tar -tvf** will show the contents of an archive
- Add compression using **-z** or **-j**

Managing File Compression

- **gzip** is the most common compression utility (in **tar** use **z**)
- **bzip2** is an alternative utility (in tar use **j**)
- **xz** is a newer very efficient utility (in tar use **J**)
- **zip** can be used as well, and has a Windows-compatible syntax
- Some other compression utilities exist as well

Quiz Question

Which of the following commands shows accurate syntax to extract the contents of file.tgz to the directory /data?

- tar zvf file.tgz /data
- tar xvf file.tgx /data
- tar xvzf file.tgz -C /data
- tar zvf file.tgx -C /data

10 Minute Lab

- Create a directory structure /tmp/files/pictures, /tmp/files/photos and /tmp/files/videos
- Copy all files that have a name starting with an a, b, or c from /etc to /tmp/files
- From /tmp/files, move all files that have a name starting with an a or b to /tmp/files/photos, and files with a name starting with a c to /tmp/files/videos
- Find all files in /etc that have a size smaller than 1000 bytes and copy those to /tmp/files/pictures
- In /tmp/files, create a symbolic link to /var
- Create a compressed archive file of the /home directory
- Extract this compressed archive file with relative file names in /tmp/archive

5. Working with Text Files

5.1 Working with **vim**

How to get started?

- **vimtutor**

5.2 Reading Text Files

Reading Text Files Command Overview

- **less**
- **more**
- **head**
- **tail**
- **cat**
- **tac**

Quiz Question

Which of the following is NOT a command that would show the contents of a file?

- tac
- view
- more
- type

5.3 Using **grep**

Using **grep**

- **grep** is the ultimate utility to find text
- Used in files: **grep lisa ***
- Used in command output: **ps aux | grep http**
- Many useful options:
  - -i: case-insensitve
  - -v: not-containing
  - -A 5/-B 5: shows 5 lines after/before match
  - -R: recursive

5.4 Understanding Regular Expressions

Understanding Regular Expressions

- Regular Expressions are text patterns that are used by tools like grep and others
- Don't confuse regular expressions with globbing (shell wildcards)!
- They look like file globbing, but they are not the same
  - **grep 'a*' a***
- For use with specific tools only (**grep**, **vim**, **awk**, **sed**)
- See **man 7 regex** for details
- Warning: regular expressions are powerful, but complicated! Many exceptions exist to the rules
- Apart from basic regular expressions, there are expanded regular expressions as well!

Understanding Regular Expressions

- Regular expressions are built around *atoms*; an atom specified what text is to be matched
  - Atoms can be single characters, a range of characters, or dot if you don't know the character
  - Or atoms can be a class, such as [[:alpha:]], [[:upper:]] or [[:alnum:]]
- The second element is the repetition operator, which specifies how many times a character should occur
- The third element is indicating where to find the next character

Understanding Regular Expressions

- **^** beginning of the line
- **$** end of the line
- ***** zero or more times
- **+** one or more times (Perl regex!)
- **?** zero or one time
- **.** one character

Quiz Question

How do you go from vi input mode back to command mode?

- Type exit
- Type command
- Type quit
- Press the Escape key

5.5 Using Text Processing Utilities

Common Text Processing Utilities

- **cut**: filter output from a text file
- **sort**: sort files, often used in pipes
- **tr**: translates uppercase to lowercase
- **awk**: search for specific patterns
- **sed**: powerful stream editor to batch-modify text files

sed and **awk** Examples

- **sed -n 5p /etc/passwd**
- **sed -i s/old/new/g ~/myfile**
- **sed -i -e '2d' ~/myfile**
- **for i in *conf; do sed -i 's /old/new/g' $i; done**
- awk is great to filter text: **awk -F : '{ print $4 }' /etc/passwd**
- same, but look for text "user": **awk -F : '/user/ { print $4 }' /etc/passwd**
- pinfo sed has nice examples!

5 minutes Lab

- Use **head** and **tail** to display the fifth line of the file /etc/passwd
- Use **sed** to display the fifth line of the file /etc/passwd
- Use **awk** in a pipe to filter the first column out of the results of the command **ps aux**
- Use **grep** to show the names of all files in /etc that have lines starting with the text 'root'
- Use **grep** to show all lines from all files in /etc that contain exactly 3 characters
- Use **grep** to find all files that contain the string "alex", but make sure that "alexander" is not included in the result

5. Working with the Shell

5.1 Understanding the Role of the Shell

5.2 Using I/O Redirection and Piping

Understanding Redirection

- Redirection is used to manipulate input and output of commands
- Standard input (0): <
  - **sort < /etc/services**
- Standard output (1): >
  - **ls > ~/myfile**
  - **who >> ~/myfile**
- Standard error (2): 2>
  - **grep -R root /proc 2>/dev/null**
  - **grep -R root /etc &> ~/myfile**

Understanding Piping

- A pipe is used to send the output of one command to be used as input for a second command
  - **ps aux | grep http**
- The **tee** command combines redirection and piping: It allows you to write output to somewhere, and at the same time use it as input for another command
  - **ps aux | tee psfile | grep ssh**

5.3 Working with **history**

Understanding **history**

- Commands a user types are written to ~/.bash_history
- The **history** command is used to repeat commands from this file
- Use **history -c** to clear the *current* history
- Use **history -w** to write the current history
- Use Ctrl-R for reverse-i-search
- Or use !nn to repeat a specific line from history

Quiz Question

There are different methods to repeat a command from history. Which of the following is NOT one of them?

- Ctrl-s
- !r
- !!
- !23

5.4 Using Command Line Completion

Using Bash Completion

- The [Tab] Key can be used for command line completion and works on different items
  - commands
  - variables
  - file names
- Install the bash-completion package for additional completion features

5.5 Using Variables

Understanding Variables

- A variable is a label to which a dynamic value can be assigned
- Convenient for scripting: Define the variable once, and use in a flexible way in different environments
- System variables contain default settings used by Linux
- Environment variables can be set for application use
  - Use **varname=value** to define
  - Use **echo $varname** to read
- By default, variables are only known to the current shell
  - Use **export** to export it to all subshells

Quiz Question

When starting a command, Linux will search a standard list of directories that are set in a variable. What is the name of that variable?

- SEARCH
- PATH
- search
- path

5.6 Using Other Bash Features

Understanding **alias**

- **alias** allows you to define your own commands
- By default set through /etc/profile

Using Bash Editing Commands

- Bash also includes convenient keyboard shortcuts
  - **Ctrl-l** clear screen
  - **Ctrl-u** wipe current command line
  - **Ctrl-a** move to the beginning of a line
  - **Ctrl-e** move to the end of a line
  - **Ctrl-c** interrupt the current process (break)
  - **Ctrl-d** Exit

5.7 Working with Bash Startup Files

Understanding Bash Startup Files

- /etc/environment contains a list of variables and is the first file that is processed while starting bash (empty by default on Red Hat)
- /etc/profile is executed while users login
  - /etc/profile.d is used as a snapin directory that contains additional configuration
  - ~/.bash_profile can be used as a user-specific version
  - ~/.bash_logout is processed when a user logs out
- /etc/bashrc is processed every time a subshell is started
  - A user-specific ~/.bashrc file may be used

Quiz Question

What is the name of the file that is used by Bash to set the initial environment?

- /etc/bashrc
- /etc/login
- /etc/profile
- /etc/defaults

5 Minute Lab

- Modify your environment such that after login, all users have access to the following
  - An alias with the name **ipconfig** that runs the **ip addr show** command
  - A variable with the name COLOR that is set to the value red
  - Ensure that the alias is available in subshells also

6. Managing Processes

6.1 Understanding Linux Processes and Jobs

Understanding Processes and Jobs

- Jobs are started by running commands in a shell and have their own management options
- Processes are tasks running on Linux
  - Jobs are also a process
  - Daemons are tasks that are started at startup by the systemd service manager

6.2 Managing Interactive Shell Jobs

Managing Jobs

- Jobs can be started as a foreground or as a background job
- Use **command &** to start a command as a background job
- Use **jobs** for an overview of jobs in this shell
- Use **fg** to run a background job on the foreground
- Use **Ctrl-Z** to stop a foreground job, which allows you to move it to the background using **bg**
- Use **Ctrl-C** to terminate a job
- Use **Ctrl-D** to stop the job

6.3 Monitoring Processes with **top**

top highlights

- Load average: average amount of runnable processes in the last 1, 5 and 15 minutes
- CPU statistics
  - Focus on us, sy and wa
  - Press 1 to see one line for each CPU core
- Memory statistics
  - Linux will use available memory as cache
  - Using swap doesn't have to be bad
- Press **f** to select additional display columns
- Press **w** to write new settings to ~/.toprc
- Use **top –u username** to find out specific user activity

6.4 Monitoring Processes with **ps**

Using **ps**

- The **ps** command is used in 2 dialects
  - **ps aux** is BSD dialect and shows running processes
  - **ps –ef** is System V dialect and show similar information
- Many options are available to specify what exactly must be displayed
  - **ps –e –o pid,args --forest**
  - **ps –C java –L –o pid,tid,pcpu,state,nlwp,args** will show process threads
  - **ps aux --sort pmem** will sort on memory usage

Quiz Question

Which key do you use in the top utility to see one line with statistics for each CPU in the system?

a. c
b. 1
c. w
d. r

6.5 Changing Process Priority

Understanding Process Renicing

- On Linux, there are real-time processes and normal processes
- Real-time processes are handled by a dedicated scheduler and are always handled first
- Priority of normal processes can be changed using **nice** and **renice**
  - nice values go from -20 to 19
  - Alternatively, use **r** from **top**

6.6 Managing Processes with **kill**

Understanding signals and **kill**

- Different signals can be sent to processes using **kill**
- Use **man 7 signals** for an overview
- Signal 15 (sigterm) and 9 (sigkill) can always be used
- Different variations to the **kill** command exist
  - **pkill**
  - **killall**
  - Or use **k** from **top** interface

5 Minute Lab

- From a root shell, start three processes **dd if=/dev/zero of=/dev/null** as a background job
- Use the appropriate command to show that they are running as expected
- Change process priority so that one of these three jobs gets double the amount of CPU resources
- Monitor the processes are running as expected
- Terminate all three processes

7. Managing Software

7.1 Understanding Software Installation Options

Understanding Installation Options

- From source files
- From compressed archives
- From packages
- From a repository

7.2 Installing Software from Source

Installing from Source

- Software can be delivered in a compressed tar ball
- This may contain a setup script, or just source files
- To get this on your computer, you need to install it, which often involves compiling
- Compiling makes software ready to run on a specific platform
- You'll need different tools, like the C compiler GCC and the make utility
- The disadvantage: there is no central registration on your machine of software that was installed this way

Quiz Question

When installing software from source, which command typically takes care of all the software compilation?

a. config
b. setup
c. install
d. make

7.3 Understanding Software Packages

Understanding Software Packages

- A Package is a tar ball with something in addition
  - A script to copy files to the right location
  - A database to keep track of what is installed
- Packages typically focus on the software they want to install, and use dependencies for related software packages
- That means the dependency needs to be installed before you can install the package
- With a result that can be a real dependency hell

7.4 Understanding Software Managers

Working with Software Managers

- Software Managers were developed to fix the dependency problems
- They do so by working with repositories
- Repositories typically are on-line resources where packages are stored
- Before installing a package, the software manager analyzes the dependencies and will try to fetch the dependencies from the repositories
- Repositories are provided by the Linux distribution, or software vendors, or you may create your own
- Common software managers are yum and apt

7.5 Managing Packages with **yum**

Working with **yum**

- **yum** uses repositories that are in /etc/yum.repos.d
- The command was written to be intuitive
  - **yum install**
  - **yum search**
  - **yum remove**
- **yum** also allows working with package groups
  - **yum groups list**
  - **yum groups install**
- Finding the right package may be a challenge, but there's **yum provides** to fix that
- And **yum history** allows you to undo changes

Quiz Question

To install software on Linux, repositories are used. Which of the following is a valid name of a configuration file that stores repositories?

a. /etc/yum.repo
b. /etc/yum.repos.d/myrepo
c. /etc/yum/myrepo
d. /etc/yum.repos.d/my.repo

7.5 Managing Packages with **rpm**

Managing Packages with **rpm**

- The **rpm** command should not be used anymore for package installation
- It is still useful to work with the RPM database and allows you to do package queries
  - **rpm -qf /my/file** will tell you which package a file is from
  - **rpm -ql mypackage** queries the database to list package contents
- to query the package file instead of the package contents, add a **p** to the options
  - **rpm -qpc mypackage.rpm** lists configuration files in a downloaded package file
  - **rpm -qp --scripts mypackage.rpm** shows scripts that may be present in a package

5 Minute Lab

- Find the RPM package that contains the **sealert** binary and install it
- After installing it, generate a list of files in that package
- What package comes the **ip** command from?

8. User and Permissions Basics

Managing Users

- While creating a user, the user becomes member of a primary group with the name of that user
- Primary group is important for file permissions
- Users can be added to secondary groups as well
- Use **useradd** to create a user
  - **useradd linda**
  - **useradd –G sales linda**
- Use **groupadd** to add groups
  - **groupadd sales**
- Use **usermod** to change user properties
  - **usermod –aG sales anna**
- Use **passwd** to set passwords for users

Understanding Basic Permissions

- Basic Permissions can be set to files and directories
- **Read (4)**
  - On files: read contents
  - On directories: list contents of directories with **ls**
- **Write (2)**
  - On files: modify contents
  - On directories: add and remove files
- **Execute (1)**
  - On files: run the file if it contains executable code
  - On directories: change into the directory using **cd**

Managing File Ownership

- To manage permissions, file ownership is managed first
- By default, the user who creates a file becomes user-owner, the primary group of that used becomes group-owner, and everybody else is the others owner (ugo)
- Use **chown** to change user ownership
  - **chown anna myfile**
  - **chown anna:sales /data/sales**
- Use **chgrp** to change group ownership
  - **chgrp sales /data/sales**

Managing File Permissions

- **chmod** is used to manage file permissions
- In absolute mode, it sets permissions to ugo:
  - **chmod 750 myfile**
- Alternatively, relative mode can be used:
  - **chmod +x myscript**

Quiz Question

A user needs to be added to a new group. Which command will do this, without overwriting current group assignments?

- groupmod -a username groupname
- usermod -a groupname username
- usermod -aG groupname username
- groupmod -aU username groupname

Quiz Question

Which permission is needed for an ordinary user to delete files?

write on the file

write on the directory

write and execute on the file

read, write and execute on the directory

10 Minute Lab

- Create users anna and linda, and make sure they are member of the secondary group sales
- Create the directory /data/sales
- Ensure that all members of the group sales can add, delete and modify files in this directory
- Ensure that others have no permissions at all

9. Using Networking

Understanding Linux Networking

- Every distribution has its own solution to manage persistent networking
  - Red Hat uses **nmcli** / **nmtui**
  - Ubuntu uses netplan
  - SUSE uses SUSE Manager
- For managing runtime networking, the same tools are used throughout
  - **ip addr show** (**ip a**) to show current IP address configuration
  - **ip route show** to show the routing table
  - **cat /etc/resolv.conf** to show networking
  - Do **NOT** use **ifconfig**
- To test connectivity, use **ping**

10. Using Storage Devices

Understanding Mounts

- To use a storage device in Linux, it must be mounted
- Storage devices are available through device files in **/dev**
- Use **lsblk** to find a list of available devices
- Common devices:
  - /dev/sda1
  - /dev/vda1
  - /dev/nvme0n1p1
- By mounting a device it is made accessible by accessing a directory
  - **mount /dev/sdb1 /mnt**
- That directory cannot be used by anything else
- Persistent mounts need to be configured through /etc/fstab

11. Using Systemd

11.1 Understanding Systemd

Understanding Systemd

- Systemd is the manager of everything
- It's the first thing that is started after starting the Linux kernel
- It starts processes and can do that in parallel
- It also manages mounts, timers, paths, and much more
- It is event driven, which means that it can react to specific events
- The items that are managed by systemd are called units
- Default units are in /usr/lib/systemd/system, custom units are in /etc/systemd

11.2 Managing Services

Managing Services

- **systemctl -t help** shows unit types
- **systemctl list-unit-files** lists all installed units
- **systemctl list-units** lists active units
- **systemctl enable name.service** enables but doesn't start a service
- **systemctl start name.service** starts a service
- **systemctl disable name.service** disables but doesn't stop a service
- **systemctl stop name.service** stops a service
- **systemctl status name.service** gives information about the service

Modifying Service Configuration

- **systemctl cat name.service** reads current unit configuration
- **systemctl show** shows all available configuration parameters
- **systemctl edit name.service** allows you to edit service configuration
- After modifying service configuration, use **systemctl daemon-reload**
- Next, use **systemctl restart name.service** to restart the service

11.3 Managing Targets

Understanding Targets

- A target is a group of services
- Some targets are isolatable, which means that you can use them as a state your system should be in
  - emergency.target
  - rescue.target
  - multi-user.target
  - graphical.target
- Use **systemctl list-dependencies name.target** to see the contents and dependencies of a systemd target

Managing Targets

- **systemctl start name.target**
- **systemctl isolate name.target**
- **systemctl list-dependencies name.target**
- **systemctl get-default**
- **systemctl set-default name.target**

Quiz Question

Which of the following is the most accurate definition of a systemd target?

a. A target is a group of services
b. A target is a group of units
c. A target is used to define how a system should start up
d. A target defines the services that should be started on system boot

5 Minute Lab

- Examine the contents of the sshd service
- Ensure the sshd service will be automatically started after a reboot
- Find out what is used as the default target
- Show a list of all active unit files

12. Using Log Files

Logging Overview

- Syslog is the legacy service that takes care of logging
- Syslog on modern Linux is implemented through **rsyslogd**
- **systemd-journald** is a systemd-integrated log service

Working with **journalctl**

- **journalctl** shows the complete journal
- **journalctl -u <unit>** shows information about specific unit (use tab completion)
- **journalctl --dmesg** shows kernel messages
- combined filters can be used: **journalctl -u crond --since yesterday --until 9:00 -p info**

Understanding Rsyslog

- The **rsyslogd** service works with facility, priority, and destination
- The facility is what **rsyslogd** should be logging for
- The priority indicates the severity of a log event
- The destination defines where the message should be written to

Quiz Question

Which of the following is the most efficient command to find the last log messages that have been generated by the sshd process?

a. tail -f /var/log/messages
b. tail -f /var/log/secure
c. journalctl
d. systemctl status sshd