Red Hat
Training and Certification

Instructor Guide

Red Hat Enterprise Linux 8.2 RH134

Red Hat System Administration II

Edition 1

RH134-RHEL8.2-en-1-20200928

Red Hat System Administration II

RH134-RHEL8.2-en-1-20200928

Red Hat Enterprise Linux 8.2 RH134
Red Hat System Administration II
Edition 1 20200928
Publication date 20200928

Authors:	Fiona Allen, Adrian Andrade, Hervé Quatremain, Victor Costea, Snehangshu Karmakar, Marc Kesler, Ed Parenti, Saumik Paul, Dallas Spohn
Editor:	Steven Bonneville, Philip Sweany, Ralph Rodriguez, David Sacco, Nicole Muller, Seth Kenlon, Heather Charles

Contributors: Artur Glogowski, Fernando Lozano, Latha Murthy, Samik Sanyal, Chetan Tiwary, Achyut Madhusudan, Rob Locke, Rudolf Kastl, Prashant Rastogi, Heider Souza, Michael Phillips

Document Conventions

References

"References" describe where to find external documentation relevant to a subject.

Note

"Notes" are tips, shortcuts or alternative approaches to the task at hand. Ignoring a note should have no negative consequences, but you might miss out on a trick that makes your life easier.

Important

"Important" boxes detail things that are easily missed: configuration changes that only apply to the current session, or services that need restarting before an update will apply. Ignoring a box labeled "Important" will not cause data loss, but may cause irritation and frustration.

Warning

"Warnings" should not be ignored. Ignoring warnings will most likely cause data loss.

Course Schedule

ILT/VT

Day 1

Activity	Time
Introduction	20 minutes (40 minutes with i18n)
Chapter 1	95 minutes
Chapter 2	130 minutes
Chapter 3	80 minutes
Chapter 4	85 minutes
Total for day	410 minutes (430 minutes)

Day 2

Activity	Time
Chapter 5	120 minutes
Chapter 6	105 minutes
Chapter 7	130 minutes
Total for day	355 minutes

Day 3

Activity	Time
Chapter 8	105 minutes
Chapter 9	105 minutes
Chapter 10	150 minutes
Total for day	360 minutes

Day 4

Activity	Time
Chapter 11	75 minutes
Chapter 12	170 minutes
Chapter 13 (started)	85 minutes
Total for day	330 minutes

Day 5

Activity	Time
Chapter 13 (continued)	115 minutes
Chapter 14	155 minutes
Total for day	270 minutes

Course Introduction

Welcome students and provide an orientation to the class, classroom hardware, and facility or VT environment.

Student Benefits

This course is specifically designed for students who have completed Red Hat System Administration I (RH124). Red Hat System Administration II (RH134) focuses on the key tasks needed to become a full time Linux Administrator and to validate those skills via the Red Hat Certified System Administrator exam. This course goes deeper into Enterprise Linux administration including filesystems and partitioning, logical volumes, SELinux, firewalling, troubleshooting, and containers.

Presentation Notes

Introduce yourself and welcome students to the class. Before starting make sure any operational requirements, including taking attendance and providing students with materials, have been met. For an in-person training event, orient students to the facility. Make sure students know the classroom hours and plans for any rest breaks and lunch.

Discuss the basic structure of the course and course timing with the students.

Objectives

Introduce your students to the main objectives of this course.

Audience and Prerequisites

Discuss the intended audience and prerequisites for this course.

Chapter 1

Improving Command-line Productivity

Overview

Run commands more efficiently by using advanced features of the Bash shell, shell scripts, and various utilities provided by Red Hat Enterprise Linux.

Schedule

ILT/VT Schedule

Section	Title	Presentation & Engagement Methods	Time (minutes)
Introduction			3
1	Writing Simple Bash Scripts	P: Lecture	15
		A: Guided Exercise	10
2	Running Commands More Efficiently Using Loops	P: Lecture	15
		A: Guided Exercise	10
3	Matching Text in Command Output with Regular Expressions	P: Lecture	15
		A: Guided Exercise	10
	Lab	Review Lab	15
Conclusion			2

Total Time: 95 minutes

Objectives

- Automate sequences of commands by writing a simple shell script.
- Efficiently run commands over lists of items in a script or from the command-line using for loops and conditionals.
- Find text matching a pattern in log files and command output using the **`grep`** command and regular expressions.

Key Takeaways

- How to create and execute simple Bash scripts.
- How to use loops to iterate through a list of items from the command-line and in a shell script.
- How to search for text in log files and configuration files using regular expressions and **`grep`**.

Instructor Tips and Suggestions

Writing Simple Bash Scripts

- This chapter is a quick introduction to Bash. Do not spend too much time on it. Quoting special characters is important, it is essential that students understand how to handle special characters.

Running Commands More Efficiently Using Loops

- The **`exit`** command can be executed with an integer between 0 and 255 representing an exit code. The exit code is passed to the parent process. It is then stored in **`?`** variable and can be accessed using **`$?`**.

Matching Text in Command Output with Regular Expressions

- Regular expressions is a pattern matching language that can be used with the **`vim`**, **`less`**, and **`grep`** commands.

Chapter 2

Scheduling Future Tasks

Overview

Schedule tasks to automatically execute in the future.

Schedule

ILT/VT Schedule

Section	Title	Presentation & Engagement Methods	Time (minutes)
Introduction			3
1	Scheduling a Deferred User Job	P: Lecture	10
		A: Guided Exercise	10
2	Scheduling Recurring User Jobs	P: Lecture	15
		A: Guided Exercise	15
3	Scheduling Recurring System Jobs	P: Lecture	20
		A: Guided Exercise	15
4	Managing Temporary Files	P: Lecture	20
		A: Guided Exercise	15
	Quiz	Review Quiz	5
Conclusion			2

Total Time: 130 minutes

Objectives

- Set up a command that runs once at some point in the future.
- Schedule commands to run on a repeating schedule using a user's crontab file.
- Schedule commands to run on a repeating schedule using the system crontab file and directories.
- Enable and disable systemd timers, and configure a timer that manages temporary files.

Key Takeaways

- Jobs that are scheduled to run once in the future are called deferred jobs or tasks.
- Recurring user jobs execute the user's tasks on a repeating schedule.
- Recurring system jobs accomplish administrative tasks on a repeating schedule that have system-wide impact.
- The **`systemd`** timer units can execute both the deferred or recurring jobs.

Instructor Tips and Suggestions

Scheduling a Deferred User Job

- Start the discussion of this section by a asking a question such as "What would you do, if you want to accomplish a task, for example, send an email while you are out?". This question helps students to get the right impression of **`at`** and understand its use cases.
- Show students how to schedule a deferred job with **`at`** using two or three different time format. Also, show them how to inspect and manage deferred jobs.

Scheduling Recurring User Jobs

- Start explaining this section by mentioning the difference between *deferred task* and *recurring task*. Mention to the students that the package *cronie* provides different programs that help in scheduling and managing jobs intended to run on a repeating schedule.
- Show students the use of the **`crontab`** command with its various options such as **`-l`**, **`-e`**, and **`-r`**. Take an example of a user cronjob and explain the job format of the user cronjob. Ensure that the students understand the meaning of each field in the user job format.

- Mention the challenge that comes with scheduling a recurring job using **`crontab`** and that the **`crond`** daemon expects the system to be fully up and running always to execute the scheduled job. There may be situations where a system may not be fully up which would cause the scheduled job to fail. This is why the critical system administrative jobs should be scheduled as system jobs and not user jobs. Mention in the next section that they will learn how to schedule a system job.

Scheduling Recurring System Jobs

- Explain the job format of the system jobs using the syntax diagram from **`/etc/crontab`**. Mention the locations that contain the jobs intended to run hourly, daily, weekly, monthly respectively. Explain the difference between **`/etc/crontab`** and **`/etc/anacrontab`**.

- Introduce the **`systemd`** timer unit. Use **`/usr/lib/systemd/system/sysstat-collect.timer`** as an example to explain how granular time intervals can be mentioned in the **`systemd`** timer unit. Also, emphasize the recommended practice of modifying unit configuration files under **`/etc/systemd/system`** rather than modifying them under **`/usr/lib/systemd/system`**.

Managing Temporary Files

- Refer to the same section in the student guide and present the topic as given.

Chapter 3

Tuning System Performance

Overview

Improve system performance by setting tuning parameters and adjusting scheduling priority of processes.

Schedule

ILT/VT Schedule

Section	Title	Presentation & Engagement Methods	Time (minutes)
Introduction			3
1	Adjusting Tuning Profiles	P: Lecture	20
		A: Guided Exercise	10
2	Influencing Process Scheduling	P: Lecture	15
		A: Guided Exercise	15
	Lab	Review	15
Conclusion			2

Total Time: 80 minutes

Objectives

- Optimize system performance by selecting a tuning profile managed by the tuned daemon.
- Prioritize or de-prioritize specific processes with the nice and renice commands.

Key Takeaways

- The **`tuned`** service automatically modifies device settings to meet specific system needs based on a pre-defined selected tuning profile.
- To revert all changes made to system settings by a selected profile, either switch to another profile or deactivate the **`tuned`** service.
- The system assigns a relative priority to a process to determine its CPU access. This priority is called the **`nice`** value of a process.
- The **`nice`** command assigns a priority to a process when it starts. The **`renice`** command modifies the priority of a running process.

Instructor Tips and Suggestions

Adjusting Tuning Profiles

- Tuning the kernel is a popular topic and therefore this lecture has the potential to generate many questions and could impact lecture timing. Therefore, proceed with caution and remember that this section is designed to provide an overview of tuning a system, the details are covered in the RH442 Red Hat Performance Tuning course.
- If students become curious to how static tuning works you could explain that static tuning uses the predefined **`sysctl`** and **`sysfs`** settings to adjust the systems behavior based on various tuning profiles. The output of the **`sudo sysctl -a | less`** command will display the numerous kernel parameters that are available and should be enough to convince students why this subject matter requires a course of its own.

Influencing Process Scheduling

- In the "Reporting Nice Levels" section it may help to better understand the graphic if you open an instance of **`top`** to visually demonstrate how the nice value maps to the scheduled priority.

Chapter 4

Controlling Access to Files with ACLs

Overview

Interpret and set Access Control Lists (ACLs) on files to handle situations requiring complex user and group access permissions.

Schedule

ILT/VT Schedule

Section	Title	Presentation & Engagement Methods	Time (minutes)
Introduction			3
1	Interpreting File ACLs	P: Lecture	25
		A: Matching Quiz	5
2	Securing Files with ACLs	P: Lecture	20
		A: Guided Exercise	10
	Lab	Review	20
Conclusion			2

Total Time: 85 minutes

Objectives

- Describe use cases for ACLs, identify files that have ACLs set, and interpret the effect of those ACLs.
- Set and remove ACLs on files and define default ACLs automatically set by a directory on newly created files.

Key Takeaways

- ACLs provide fine-grained access control to files and directories.
- The **`getfacl`** command displays the ACLs on a file or directory.
- The **`setfacl`** command sets, modifies, and removes default and standard ACLs on files and directories.
- Use default ACLs for controlling new files and directories permissions.
- Red Hat Enterprise Linux uses **`systemd`** and **`udev`** to apply predefined ACLs on devices, folders, and files.

Instructor Tips and Suggestions

Interpreting File ACLs

- Access Control Lists extend the standard Linux file permissions to allow multiple named users and multiple groups to have permissions allocated to them, removing the limitation of one user, one group and the rest of the world.
- ACLs allow fine grained control for file access, including default setting of permissions for new files and directories. Access permissions can be defined and managed for multiple users and multiple groups of users.
- Highlight the limitations of normal file permissions, particularly in regard to shared directories, for example only one group, all with same permissions, or everyone.
- Show the similarities in permission flags and remind them that the parent directory-hierarchy needs to allow access for the named users and named groups.
- Do NOT delve to deep into the mount requirements.
- Demo **`ls -l`** and discuss the "**+**", map the UGO values to the ACLS, For example User and ACL User are equivalent. Discuss that group values reflect the ACL mask and mention **`chmod g`** updates the mask.

- Demo **`getfacl`** on a file and on a directory, explain the output. Mention that the output can be captured and used as input to setfacl. Discuss **`-R`** as a directory ACL backup and recovery option.
- Discuss the mask behaviour and which users and groups it affects; point out it does not affect the "owner" or "other" permissions. Cover the recalculation and setting of the mask.
- ACLs provide fine grained file permissions, removing standard Linux file permission limitations. ACLs must be supported in the mounted file system. ACL masks restrict the maximum permissions for everyone other than the file owner and "other" users. ACL precedence operates in the order you would expect. Finally, you can view ACLs on files and directories with **`getfacl`**.

Securing Files with ACLs

- **`setfacl`** is a one stop tool for adding, modifying and deleting ACLs, including default ACLs, and can be used by the file owner without root privileges.
- Demo **`setfacl -m`** covering **`u`**, **`g`**, **`m`** and settings. Make sure you cover **`-R`** and the capital **`X`** permission. Also deleting an ACL with **`-x`** and **`-b`**.
- Remind your students that **`setfacl`** would not adjust ACLs that exist but are not referenced by the **`setfacl`** command and discuss the auto creation of the user (file owner), group (owning group) and other permissions if they are not explicitly set - that is, the base ACLs.
- Also, cover that the mask is auto recalculated whenever one of the affected ACLs is changed.
- Make sure your students understand the need for the execute permission to allow directory search.
- Setfacl can add, modify and delete normal and default ACLs, including recursive updates to existing files. The mask limits the maximum permissions that can be allocated to users (other than the file owner and other users). Default ACLs make it easy to inherit appropriate file and directory permissions.

Chapter 5

Managing SELinux Security

Overview

Protect and manage the security of a server by using SELinux.

Schedule

ILT/VT Schedule

Section	Title	Presentation & Engagement Methods	Time (minutes)
Introduction			3
1	Changing the SELinux Enforcement Mode	P: Lecture	15
		A: Guided Exercise	10
2	Controlling SELinux File Contexts	P: Lecture	15
		A: Guided Exercise	10
3	Adjusting SELinux Policy with Booleans	P: Lecture	15
		A: Guided Exercise	10
4	Investigating and Resolving SELinux Issues	P: Lecture	15
		A: Guided Exercise	10
	Lab	Review Lab	15
Conclusion			2

Total Time: 120 minutes

Objectives

- Describe how SELinux protects resources and how to select the enforcement mode.
- Configure a file's SELinux context to control how processes interact with that file.
- Configure SELinux booleans to allow runtime policy changes for varying access needs.
- Investigate SELinux log messages and troubleshoot SELinux AVC denials.

Key Takeaways

- The **`getenforce`** and **`setenforce`** commands are used to manage the SELinux mode of a system.
- The **`semanage`** command is used to manage SELinux policy rules. The **`restorecon`** command applies the context defined by the policy.
- Booleans are switches that change the behavior of the SELinux policy. They can be enabled or disabled and are used to tune the policy.
- The **`sealert`** displays useful information to help with SELinux troubleshooting.

Instructor Tips and Suggestions

Changing the SELinux Enforcement Mode

- Make sure the students understand that switching SELinux off is not an option. It should be used uniquely in troubleshooting to confirm that SELinux is the problem. They should then configure SELinux to solve the problem.

Controlling SELinux File Contexts

- The **`restorecon`** command is preferred to the **chcon**.

Adjusting SELinux Policy with Booleans

- Booleans can be difficult to understand, take time over this section. The error messages from Booleans can be confusing for administrators. Walk the students through the log files concerned and explain what messages in the log files mean.

Investigating and Resolving SELinux Issues

- The output from the **sealert** is not as intuitive as it was in Red Hat Enterprise Linux 7. For example, in the last guided exercise the **sealert** states:

```
You can generate a local policy module to allow this access.
Do
allow this access for now by executing:
# ausearch -c 'httpd' --raw | audit2allow -M my-httpd
# semodule -X 300 -i my-httpd.pp
```

  However, this will not give access to the file. To gain access to the file the students must use the **semanage** command. This might be confusing for some students.

Chapter 6

Managing Basic Storage

Overview

Create and manage storage devices, partitions, file systems, and swap spaces from the command line.

Schedule

ILT/VT Schedule

Section	Title	Presentation & Engagement Methods	Time (minutes)
Introduction			3
1	Adding Partitions, File Systems, and Persistent Mounts	P: Lecture	30
		A: Guided Exercise	15
2	Managing Swap Space	P: Lecture	20
		A: Guided Exercise	10
	Lab	Review	25
Conclusion			2

Total Time: 105 minutes

Objectives

- Create storage partitions, format them with file systems, and mount them for use.
- Create and manage swap spaces to supplement physical memory.

Key Takeaways

- You use the **`parted`** command to add, modify, and remove partitions on disks with the MBR or the GPT partitioning scheme.
- You use the **`mkfs.xfs`** command to create XFS file systems on disk partitions.
- You need to add file-system mount commands to **`/etc/fstab`** to make those mounts persistent.
- You use the **`mkswap`** command to initialize swap spaces.

Instructor Tips and Suggestions

Adding Partitions, File Systems, and Persistent Mounts

- The **`fdisk`** and **`gdisk`** commands are still available to partition disks. However, the *Configuring and managing file systems* guide at https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8-beta/html-single/configuring_and_managing_file_systems/ only uses **`parted`** and does not refer to these two commands anymore. Anaconda uses Parted through the **`libparted`** library to partition the disks during the installation process.
- **`parted`** informs the kernel when you create a partition. The kernel then triggers udev for the creation of the device file under **`/dev/`** (**`/dev/vdb1`** for example). The **`udevadm settle`** command blocks until udev has finished its work. Usually, this is a fast process, and students should not notice a pause when running the **`udevadm settle`** command. This may, however, be useful in scripts when you chain the creation of the partition and its formatting.
- When the system boots, systemd reads **`/etc/fstab`** and creates **`mount`** units from it, in **`/run`**. You can list those units with **`systemctl -t mount --all`**. Systemd uses these units to control and supervise your mounts. Every time you add or remove an entry from **`/etc/fstab`**, it is a good practice to run **`systemctl daemon-reload`** to create or delete the corresponding unit without having to reboot the system. Skipping this step should not prevent you from mounting and using the file system. See the **`systemd.mount`**(5) man page for more details.
- Do not use **`mount -a`** to check **`/etc/fstab`**. The **`mount`**(8) man page states that it is a bad practice.

- Students may ask why they should create a partition on a LUN, and not directly use the LUN block device. See the Knowledgebase: What are the advantages and disadvantages to using partitioning on LUNs, either directly or with LVM in between? [https://access.redhat.com/solutions/163853]

Managing Swap Space

- When the system boots, systemd reads **`/etc/fstab`** and creates **`swap`** units from it, in **`/run`**. You can list those units with **`systemctl -t swap --all`**. Systemd uses these units to control and supervise the swap spaces. See the **`systemd.swap`**(5) man page for more details.

- **`swapon -s`** is deprecated. Use **`swapon --show`** instead. See the **`swapon`**(8) man page.

Chapter 7

Managing Logical Volumes

Overview

Create and manage logical volumes containing file systems and swap spaces from the command line.

Schedule

ILT/VT Schedule

Section	Title	Presentation & Engagement Methods	Time (minutes)
Introduction			3
1	Creating Logical VolumesObjectivesCreating a Logical VolumePhysical Volumes	P: Lecture + Graphic	45
		A: Guided Exercise	15
2	Implementing LVM storageLogical VolumesExtending Logical Volumes	P: Lecture	30
		A: Guided Exercise	15
	Lab:	Review	20
Conclusion			2

Total Time: 130 minutes

Objectives

- Create and manage logical volumes from storage devices, and format them with file systems or prepare them with swap spaces.
- Add and remove storage assigned to volume groups, and nondestructively extend the size of a logical volume formatted with a file system.

Key Takeaways

- LVM allows you to create flexible storage by allocating space on multiple storage devices.
- Physical volumes, volume groups, and logical volumes are managed by a variety of tools such as **`pvcreate`**, **`vgreduce`**, and **`lvextend`**.
- Logical volumes can be formatted with a file system or swap space, and they can be mounted persistently.
- Additional storage can be added to volume groups and logical volumes can be extended dynamically.

Instructor Tips and Suggestions

Creating Logical VolumesObjectivesCreating a Logical VolumePhysical Volumes

This section starts with LVM concepts. Explain some of the benefits of LVM over basic storage. Some examples:

- Flexible capacity: When using logical volumes, file systems can extend across multiple disks, since you can aggregate disks and partitions into a single logical volume.
- Resizeable storage pools: You can extend logical volumes or reduce logical volumes in size with simple software commands, without reformatting and repartitioning the underlying disk devices.
- Convenient device naming: Logical storage volumes can be managed in user-defined and custom named groups.

There are other features such as mirroring, striping, and snapshots though those are not covered in this chapter. There is no need to go into these additional features. Just be aware of them should an inquisitive student ask.

Before going into the details of implementing LVM storage, perform a high level overview of the steps that are required. An illustration is provided to assist with the discussion.

Cover the details of implementing LVM. The **`parted`** command is now used for creating partitions.

Implementing LVM storageLogical VolumesExtending Logical Volumes

This section covers the details of extending logical volumes. Some items to be aware of:

- Just as with the previous section, the **`parted`** command is used to create partitions.
- Logical volumes formatted with GFS2 or XFS can not be reduced. This is a known issue and has not changed as of the time of writing. This is not covered in this section; however, instructors should be aware of this fact. Logical volumes formatted with ext4 can be reduced.
- There are times when a logical volume must be brought offline in order to perform some tasks. Performing file system checks, reducing logical volumes, and extending logical volumes formatted as swap are some examples.

Chapter 8

Implementing Advanced Storage Features

Overview

Manage storage using the Stratis local storage management system and use the VDO volumes to optimize storage space in use.

Schedule

ILT/VT Schedule

Section	Title	Presentation & Engagement Methods	Time (minutes)
Introduction			3
1	Managing Layered Storage with StratisDescribing the Architecture of StratisInstalling and Enabling Stratis	P: Lecture	20
		A: Guided Exercise	20
2	Managing Stratis File SystemsCompressing and Deduplicating Storage with VDO	P: Lecture	15
		A: Guided Exercise	20
	Lab	Review Lab	25
Conclusion			2

Total Time: 105 minutes

Objectives

- Manage multiple storage layers using Stratis local storage management.
- Optimize the use of storage space by using VDO to compress and deduplicate data on storage devices.

Key Takeaways

- The Stratis storage management solution implements flexible file systems that grow dynamically with data.
- The Stratis storage management solution supports thin provisioning, snapshotting, and monitoring.
- The Virtual Data Optimizer (VDO) aims to reduce the cost of data storage.
- The Virtual Data Optimizer applies zero-block elimination, data deduplication, and data compression to optimize disk space efficiency.

Instructor Tips and Suggestions

Managing Layered Storage with StratisDescribing the Architecture of StratisInstalling and Enabling Stratis

- Explain the basic architecture of Stratis and how it helps user to easily manage different layers of storage features using a single tool. Do not get into too much low-level details to explain how each storage layer functions.
- Show the students how to use the **`stratis`** command to manage Stratis pools and file systems.

Managing Stratis File SystemsCompressing and Deduplicating Storage with VDO

- Introduce Virtual Data Optimizer to the students.
- Explain the VDO phases such as Zero-Block elimination, Deduplication, and Compression. Mention that these phases help VDO to reduce the data footprint on the storage devices.
- Show the students how to use the **`vdo`** command to manage VDO volumes.

Chapter 9

Accessing Network-Attached Storage

Overview

Access network-attached storage using the NFS protocol.

Schedule

ILT/VT Schedule

Section	Title	Presentation & Engagement Methods	Time (minutes)
Introduction			3
1	Mounting Network-Attached Storage with NFS	P: Lecture	25
		A: Guided Exercise	15
2	Automounting Network-Attached Storage	P: Lecture	25
		A: Guided Exercise	15
	Lab	Review	20
Conclusion			2

Total Time: 105 minutes

Objectives

- Mount, use, and unmount an NFS export from the command line and at boot time.
- Configure the automounter with direct and indirect maps to automatically mount an NFS file system on demand, and unmount it when it is no longer in use.

Key Takeaways

- Mount and unmount an NFS export from the command line.
- Configure an NFS export to automatically mount at startup.
- Configure the automounter with direct and indirect maps, and describe their differences.

Instructor Tips and Suggestions

Mounting Network-Attached Storage with NFS

- NFS is the default method for Linux and UNIX style operating systems to share network storage resources. Knowing how to set up and connect to NFS shares is essential, and selecting the right connection method will make it easier to provide access to the NFS shares when they are needed.
- Some NFS shares are need transiently and others are needed all the time. Understanding how to connect using the **`mount`** command and the **`/etc/fstab`** file supports these use cases.
- Cover the basic features of NFSv4 as the default NFS version on RHEL8 and NFSv3 is the older supported version, TCP as the protocol (which may mean firewalld needs to be opened), neither UDP or **`RPCBIND`** are needed in RHEL8, using a NFSv4-only solution, Server/Client.
- Demo manual mounts, show how to browse the server by mounting the root export and how NFS shares are visible. We are using an NFSv4 share in the class, so you cannot demo **`showmount`** command. Point out it is not reliable for NFSv4 and should only be used to query NFSv3 servers.
- Show the students the new **`nfsconf`** tool, and the new **`/etc/nfs.conf`** configuration file.
- Show the students how to configure an NFSv4 only client withe the **`nfsconf`** tool, and the **`/etc/nfs.conf`** configuration file.
- Demo the **`/etc/fstab`**, **`mount`** method and **`umount`** commands.

Automounting Network-Attached Storage

- The auto-mounter does not require users to specifically mount and unmount shares (requiring **`root`** or **`sudo`** privileges), or permanently mount the share via the **`/etc/fstab`** which may consume system and network resources.
- The auto-mounter connects and disconnects on demand, and for all configured users on the system.
- Make sure your students understand the benefits the auto-mounter brings to the table.
- **`autofs`** is typically not installed by default, demo installing it.
- Demo creating the master map file under **`/etc/auto.master.d/`**. Make sure the mandatory extension of "**`autofs`**" is covered and that the file can be called anything. The mapping filename is absolute.
- Demo creating the mapping files. Cover that the file name is not important but by convention the file is located in /etc and is named auto.xxx with a meaningful extension. Discuss the key values as absolute for direct and just the relative path for indirect. Highlight that the last directory is managed by autofs and in the case of indirect is created and removed automatically as needed. The options are the same as the **`mount`** command options, although mention **`-fstype=`** option for other file systems. Finally the format of the share server:/path/to/share.
- Direct maps use an absolute mount point, indirect maps use either a directory name or an **`*`** and **`&`** to identify mount-points and locations.
- Demo the use of the **`*`** and **`&`** and make sure students understand what they do and how they could use them to mount from a common parent directory to a common local parent, based on the specified path. Mount options need to be shared as well as the parent directories!

Chapter 10

Controlling the Boot Process

Overview

Manage the boot process to control services offered and to troubleshoot and repair problems.

Schedule

ILT/VT Schedule

Section	Title	Presentation & Engagement Methods	Time (minutes)
Introduction			3
1	Selecting the Boot Target	P: Lecture	20
		A: Guided Exercise	20
2	Resetting the Root Password	P: Lecture	20
		A: Guided Exercise	20
3	Repairing File System Issues at Boot	P: Lecture	15
		A: Guided Exercise	20
	Lab	Review	30
Conclusion			2

Total Time: 150 minutes

Objectives

- Describe the Red Hat Enterprise Linux boot process, set the default target used when booting, and boot a system to a non-default target.
- Log in to a system and change the root password when the current root password has been lost.
- Manually repair file system configuration or corruption issues that stop the boot process.

Key Takeaways

- **`systemctl reboot`** and **`systemctl poweroff`** reboot and power down a system, respectively.
- **`systemctl isolate`** ***`target-name`*****`.target`** switches to a new target at runtime.
- **`systemctl get-default`** and **`systemctl set-default`** can be used to query and set the default target.
- Use **`rd.break`** on the kernel command line to interrupt the boot process before control is handed over from the **`initramfs`**. The root file system is mounted read-only under **`/sysroot`**.
- The emergency target can be used to diagnose and fix file-system issues.

Instructor Tips and Suggestions

Selecting the Boot Target

- For more information on the boot process, refer to *An introduction to the Linux boot and startup processes* at https://opensource.com/article/17/2/linux-boot-and-startup
- **`initramfs`** files are built from 2 **`cpio`** archives. The best way to inspect **`initramfs`** files is through the **`lsinitrd`** command. For more information on the **`initramfs`** file format, refer to the Knowledgebase: How to extract/unpack/uncompress the contents of the initramfs boot image? [https://access.redhat.com/solutions/2037313]
- During the exercise, depending on the classroom environment and the physical keyboard layout, some students may struggle to enter the **`root`** password, **`redhat`**, and to type commands in emergency and rescue modes. Once logged in in the emergency or rescue shell, students can use the **`loadkeys`** ***`lang`*** command to select a new layout temporarily. The available layout files are in the **`/lib/kbd/keymaps/xkb/`** directory. The following example set different layouts.

```
[root@servera ~]# loadkeys it
[root@servera ~]# loadkeys fr
[root@servera ~]# loadkeys de
[root@servera ~]# loadkeys ch-fr
```

Resetting the Root Password

- The process to recover from a lost **root** password has not changed since Red Hat Enterprise Linux 7.
- In the boot loader, the line to update by appending **rd.break** may start by **linux**, **linux16**, or **linuxefi** depending on the environment.

Repairing File System Issues at Boot

- After editing **/etc/fstab**, the exercise asks students to run the **systemctl daemon-reload** command for systemd to recreate the **mount** targets from the updated **/etc/fstab** file. Because the next step is to reboot the system, this **systemctl daemon-reload** command is not necessary in that situation. It is, however, a good practice to get used to running that command after editing **/etc/fstab**.

Chapter 11

Managing Network Security

Overview

Control network connections to services using the system firewall and SELinux rules.

Schedule

ILT/VT Schedule

Section	Title	Presentation & Engagement Methods	Time (minutes)
Introduction			3
1	Managing Server Firewalls	P: Lecture	20
		A: Guided Exercise	10
2	Controlling SELinux Port Labeling	P: Lecture	20
		A: Guided Exercise	10
	Lab	Review	10
Conclusion			2

Total Time: 75 minutes

Objectives

- Accept or reject network connections to system services using firewalld rules.
- Control whether network services can use specific networking ports by managing SELinux port labels.

Key Takeaways

- The **`netfilter`** subsystem allows kernel modules to inspect every packet traversing the system. All incoming, outgoing or forwarded network packets are inspected.
- The use of **`firewalld`** has simplified management by classifying all network traffic into zones. Each zone has its own list of ports and services. The **`public`** zone is set as the default zone.
- The **`firewalld`** service ships with a number of pre-defined services. They can be listed using the **`firewall-cmd --get-services`** command.
- Network traffic is tightly controlled by the SELinux policy. Network ports are labeled. For example, port **`22/TCP`** has the label **`ssh_port_t`** associated with it. When a process wants to listen on a port, SELinux checks to see whether the label associated with it is allowed to bind that port label.
- The **`semanage`** command is used to add, delete, and modify labels.

Chapter 12

Installing Red Hat Enterprise Linux

Overview

Install Red Hat Enterprise Linux on servers and virtual machines.

Schedule

ILT/VT Schedule

Section	Title	Presentation & Engagement Methods	Time (minutes)
Introduction			3
1	Installing Red Hat Enterprise LinuxObjectivesBuild Images with ComposerInstalling RHEL with the Graphical Interface	P: Lecture	20
		A: Guided Exercise	30
2	Installing Red Hat Enterprise Linux ManuallyAutomating Installation with Kickstart	P: Lecture	45
		A: Guided Exercise	20
3	Installing and Configuring Virtual Machines	P: Lecture	15
		A: Quiz	5
	Lab	Review	30
Conclusion			2

Total Time: 170 minutes

Objectives

- Install Red Hat Enterprise Linux on a server.
- Automate the installation process using Kickstart.
- Install a virtual machine on your Red Hat Enterprise Linux server using Cockpit.

Key Takeaways

- The RHEL 8 binary DVD includes Anaconda and all repositories required for installation.
- The RHEL 8 boot ISO includes the Anaconda installer, accessing repositories over the network during installation.
- The Kickstart system performs unattended installations.
- Kickstart files can be created using the Kickstart Generator website or by copying and editing **`/root/anaconda-ks.cfg`**.
- The *virt* Yum module provides the packages for a RHEL system to become a virtualization host.
- The *cockpit-machines* package adds the **Virtual Machines** menu to Cockpit.

Instructor Tips and Suggestions

Installing Red Hat Enterprise LinuxObjectivesBuilding Images with ComposerInstalling RHEL with the Graphical Interface

- The manual installation of Red Hat Enterprise Linux 8 is very similar to a RHEL 7 installation.
- To burn the binary DVD, you need a dual layer DVD because of the size of the ISO file.
- There is only a single build of RHEL 8 per processor architecture. The variants such as Server, Workstation, or Desktop do not require a specific ISO anymore.
- The **System Purpose** item allows administrators to define what the system will be used for and what support level it should receive. Anaconda gives this information to Subscription Manager which can more accurately choose the appropriate subscription for the system.

Installing Red Hat Enterprise Linux ManuallyAutomating Installation with Kickstart

- The **`system-config-kickstart`** command has been replaced by the Kickstart Generator website. While students are not expected to create a kickstart file this way, it would be worth reviewing the options on the site to become aware of what capabilities are available in case a student should ask.
- Students should be made aware that **`/root/anaconda-ks.cfg`** is an excellent starting point for creating kickstart files. Making a copy of this file and modifying it is a much more desirable solution than creating a kickstart file from scratch.

Installing and Configuring Virtual Machines

- The **`virt-manager`** graphical tool is still available and supported but is deprecated. Red Hat recommends using the Web Console interface instead.
- Web Console is limited in the features and configuration options it provides. When you need those advanced features, use **`virsh`** or the deprecated **`virt-manager`** tool.

Chapter 13

Running Containers

Obtain, run, and manage simple lightweight services as containers on a single Red Hat Enterprise Linux server.

Chapter Schedule

ILT/VT Schedule

Section	Title	Presentation & Engagement Methods	Time (minutes)
Introduction			3
1	Introducing Containers	P: Lecture	20
		A: Quiz	5
2	Running a Basic Container	P: Lecture	20
		A: Guided Exercise	10
3	Finding and Managing Container Images	P: Lecture	15
		A: Guided Exercise	10
4	Performing Advanced Container Management	P: Lecture	15
		A: Guided Exercise	20
5	Attaching Persistent Storage to a Container	P: Lecture	10
		A: Guided Exercise	15
6	Managing Containers as Services	P: Lecture	15
		A: Guided Exercise	20
	Lab	A: Review	20
Conclusion			2

Total Time: 200 minutes

Chapter Objectives

By the end of this chapter, you should be able to:

- Explain what a container is and how to use one to manage and deploy applications with supporting software libraries and dependencies.
- Install container management tools and run a simple rootless container.
- Find, retrieve, inspect, and manage container images obtained from a remote container registry and stored on your server.
- Run containers with advanced options, list the containers running on the system, and start, stop, and kill containers.
- Provide persistent storage for container data by mounting a directory from the container host inside a running container.
- Start, stop, and check the status of a container as a systemd service.

Key Takeaways

Key takeaways from this chapter include the following:

- Containers provide a lightweight way to distribute and run an application and its dependencies that may conflict with software installed on the host.
- Containers run from container images that you can download from a container registry or create yourself.
- Podman, provided by Red Hat Enterprise Linux, directly runs and manages containers and container images on a single host.
- Containers can be run as **`root`**, or as non-privileged rootless containers for increased security.
- You can map network ports on the container host to pass traffic to services running in its containers. You can also use environment variables to configure the software in containers.
- Container storage is temporary, but you can attach persistent storage to a container using the contents of a directory on the container host, for example.
- You can configure Systemd to automatically run containers when the system starts up.

Instructor Tips and Suggestions

Introducing Containers

- Useful reference for instructors on running containers: *Section 1.1. Running containers without Docker* at https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html-single/building_running_and_managing_containers/index#starting-with-containers_building-running-and-managing-containers

Running a Basic Container

- If a student makes a mistake when running a container and wishes to start over, then the student first needs to delete the wrong container. To do so, the student can run the two following commands to delete all the containers on their system:

```
[student@servera ~]$ podman stop --all
4e934d5e43c7638500bb0d8ab2cc74572207d67b1377ac2ccb21dbf8d62978bf
[student@servera ~]$ podman rm --all
4e934d5e43c7638500bb0d8ab2cc74572207d67b1377ac2ccb21dbf8d62978bf
```

Attaching Persistent Storage to a Container

- With rootless containers, Podman maps the user IDs inside the container with unprivileged user IDs on the host. The following example starts a MariaDB container, and then shows the user ID of the **`mysqld`** process from inside the container and from the host.

```
[user@host ~]$ podman run -d --name mydb -e MYSQL_USER=user -e
 MYSQL_PASSWORD=redhat -e MYSQL_DATABASE=inventory registry.redhat.io/rhel8/
mariadb-103:1-102
4e934d5e43c7638500bb0d8ab2cc74572207d67b1377ac2ccb21dbf8d62978bf
[user@host ~]$ podman exec -ti mydb ps naux
    USER     PID %CPU %MEM    VSZ   RSS TTY STAT START  TIME COMMAND
      27       1  4.3  3.3 1528620 62040 ?  Ssl  11:16  0:00 /usr/libexec/mysqld
[user@host ~]$ ps naux | grep mysqld
  100026    6089  1.3  3.3 1528620 62040 ?  Ssl  11:16  0:00 /usr/libexec/mysqld
```

  Inside the container, the process has the user ID 27. From the host point of view, the process has the user ID 100026. With that example, when preparing a host directory for the database, you must make sure that the user ID 100026 has read/write access to it.

- For simplicity, when using share storage in exercises, instructions ask students to set the directory mode to 777 or ensure that the mode allows access. A more realistic way would be to set the directory ownership to the user ID running the container processes (100026 in the preceding example). However, that would require to compute that user ID, and then to run the **`chown`** command as **`root`**. Dealing with user namespaces and SELinux on rootless containers [https://www.redhat.com/sysadmin/user-namespaces-selinux-rootless-containers] provides an easier way to set the ownership with the **`podman unshare`** command.

Managing Containers as Services

- To use **`systemctl --user`** commands, users must log in at the console or directly through SSH. If students use **`su`** or **`sudo`** to switch to the **`student`** account, instead of using SSH, the typical error message when using **`systemctl --user`** is as follows:

```
[student@servera ~]$ systemctl --user
Failed to connect to dbus: No such file or directory
```

  See the **`pam_systemd`**(8) man page.

Chapter 14

Comprehensive Review

Review tasks from *Red Hat System Administration II*

Chapter Schedule

ILT/VT Schedule

Section	Title	Presentation & Engagement Methods	Time (minutes)
Introduction			3
1	Fixing Boot Issues and Maintaining Servers	A: Review Lab	45
2	Configuring and Managing File Systems and Storage	A: Review Lab	45
3	Configuring and Managing Server Security	A: Review Lab	45
4	Running Containers	A: Review Lab	20
Conclusion			2

Total Time: 160 minutes

Objectives

- Review tasks from *Red Hat System Administration II*