Red Hat Certified System Administrator

Filesystem ACL

Permissions

We've covered octal permissions in another lesson. A normal file looks like:

```
-rw-r--r--. 1 root   root
0 Mar 31 23:40 file
```

That file has permissions of 644 or Read/Write for the user, Read for the group, and Read for other.

Permissions

What happens when we have a file like this:

```
-rw-rw----. 1 dev   devteam
1247 Mar 31 23:40 dev-budget
```

And we need to make sure that the finance team can audit that budget?

That's where ACLs can be useful.

Access Control Lists

Access Control Lists (ACLs) allow a much more granular permission structure as opposed to the “normal” octal User/Group/Other permissions.

Looking at ACLs

The `getfacl` command is used to read filesystem ACLs.

```
getfacl: Removing leading '/'
from absolute path names
# file: srv/devteam/budget
# owner: dev
# group: dev
user::rw-
group::rw-
other::r--
```

Changing ACLs

The `setfacl` command is used to modify or set ACLs on a file or directory.

```
setfacl -m u:username:rwx file
```

```
setfacl -x u:username file
```

```
setfacl -b file
```

Default permissions

```
setfacl -d -m u:user:rwx directory
```

```
setfacl -k directory
```

```
setfacl --remove-default dir
```

Quick set ACL

If you already have ACLs set up on a specific file but can't remember exactly what was set, you can run

```
getfacl file1 | setfacl --
set-file=- file2
```

And it will copy the ACLs from file1 to file2.

Red Hat Certified System Administrator

Managing the Boot Process

BIOS/UEFI

BIOS stands for Basic Input/Output System. This is the part of the computer that handles everything from powering on to handing off the boot process to the operating system. It also manages power to the CPU and handling the configuration of RAM timings. The BIOS does a lot.

UEFI stands for Unified Extensible Firmware Interface. This new boot handler can do everything the older BIOS can do and more. For example, a computer managed by a BIOS can't boot from large (3+TB) disks. A BIOS is also limited by how much memory it can use and how it addresses hardware. UEFI, a standard since 2007, removes these limitations.

GRUB2

Once the UEFI/BIOS completes a power on self test and finishes it's part of the boot process, it hands off the next step to the boot device. This is typically a hard disk.

GRUB2 (Grand Unified Bootloader version 2) is installed on the hard disk and knows where on the disk the beginning of the Operating System is. GRUB will present a list of kernels that are available for you to boot into. You can also interact with GRUB to specify a different, non-configured, kernel to boot into. Or you could change the boot options on one of the configured kernels.

GRUB2 Commands

```
grub> ls

grub> cat

grub> search.file /grub2/grub.cfg

grub> insmod lvm
```

systemd Boot Targets

Inevitably you'll have a server that you need to interrupt the boot process for. Either you forget the root password and need to reset it, or you're having a malfunctioning piece of hardware or software that is preventing the system from fully booting.

The three most useful targets are:

`rd.break`: Breaks to an interactive shell while still in the initrd, allowing interaction before the system disk is mounted

`emergency`: Similar to `rd.break`, but mounts the system disk

`rescue`: Equivalent to the old single user mode, where some services are started and disks mounted

Red Hat Certified System Administrator

Compressing and Decompressing Files

Compression and Archives

There are two topics we need to cover:

Compression

- Encoding information in fewer bits than the original information

Archive

- A file that is a collection of a number of files/directories that can be sorted easier than its components

Archives

The most common archive command is `tar`.

`tar` stands for, and was originally used for "Tape Archive".

Creating an archive:

```
tar cvf home.tar /home
```

- `create an archive`
- `verbose`
- `file name`

Extracting an archive:

```
tar xvf home.tar
```

- `extract an archive`
- `verbose`
- `file name`

Compression

There are two main compression commands: `gzip` and `bzip2`. They both do essentially the same thing, but they use different compression algorithms.

gzip example:

```
gzip bigfile.stuff
 gzip -d bigfile.stuff.gz
```

bzip2 example:

```
bzip2 bigfile.stuff
bzip2 -d bigfile.stuff.bz2
```

Compress an Archive

While you can compress any file, you can also just compress an archive when you create it:

```
tar czvf home.tar.gz /home
```

or:

```
tar czjf home.tar.bz2 /home
```

Extracting is the same as you'd expect:

```
tar xzvf home.tar.gz /home
```

or:

```
tar xzjf home.tar.bz2 /home
```

zip Files

The other common format for archival/compression is `.zip`.

Mostly used for files that need to be managed on both Windows and Linux servers, the `zip` command is similar to tar in that it will archive files, but different because it automatically attempts some compression.

The archive is unpacked using the `unzip` command. Neither `zip` nor `unzip` are normally installed, so you'll have to install them to use them.

Examples:

```
zip -r archivename /home
unzip archivename
```

Red Hat Certified System Administrator

Creating/Modifying User Accounts

Getting User Information

Getting relevant user information can be done a few different ways.

`id` command

`getent` command

Manually looking at `/etc/passwd`, `/etc/shadow`, `/etc/group`

Changing User Information

The easiest way to change user information is by using the `usermod` command.

`c` – Modify the user's password file comment field

`d` – Change the user's home directory. Often used with the `m` flag which moves files from the current home dir to the new one

`G` – Change the user's supplemental groups. Often used with the `a` flag which appends, rather than replaces, the supplemental groups.

`L`, `U` – Lock or Unlock the account (respectively)

Creating a User

Before we go into creating a user we should be aware of where user information is kept.

```
/etc/skel
/etc/passwd
/etc/group
/etc/default/useradd
/etc/login.defs
```

Creating a User

On Red Hat systems you can use the `useradd` command to create new users.

`c` – Text string that is entered into the comment field in `/etc/passwd`

`d` – Set the home directory

`g` – set the GID

`G` - set supplemental groups

`k` – Set the skeleton directory

`p` – Set the user's password to this encrypted password.

`r` – create a system account

`s` – Set the user's login shell

`u` – set the UID

Red Hat Certified System Administrator

Creating And Manipulating Files

What is a file?

In Linux, a file name is just a pointer to an inode that lives on a disk. That inode contains information about the data such as the file name, the physical location on the disk, and other metadata. It's essentially a file serial number, with the file name being a "human readable" format.

Creating Files

There are a few different ways to create files:

- `touch`
  - Creates a zero byte file
- `cp`
  - Copies one file to another
- `mv`
  - Moves a file from one location to another, even across file systems. This is also how you change the name of a file
- `rm`
  - Removes the inode-filename link. The data still exists on the disk drive, however.

Manipulating Files

You can also create files using your favorite text editor. The popular ones are Vim or Nano, but others like ed or Emacs exist as well.

Finding files

Since a filesystem can be organized however you see fit, it's possible that you'll forget where a specific file is.

The `find` command will help. You specify a directory to start in and the command will search the directory structure to find files that match your criteria. For example:

```
find . -name *.txt
```

Will find all files with names that end with `.txt` and output their full path to stdout.

Common ‘find’ Flags

-type

- f = regular file
  ```
  find . -type f -name httpd.conf
  ```
- d = directory
  ```
  find . -type d -name html
  ```
- l = symbolic link
  ```
  find . -type l -name redhat-release
  ```

-user

- File is owned by user (username or UID)

The ‘locate’ command

The `locate` command is a much faster version of `find`, but it’s less precise.

`locate` relies on a database that gets updated by default once a day (`/etc/cron.daily/mlocate`) so changes after that aren’t live. You can force an update by running `updatedb`.

Once that update is done, `locate` is significantly faster than `find` because it’s just searching a database rather than crawling an entire filesystem.

Reading Text Files

- `cat`
  - This reads the file and outputs all of it to stdout.
- `less` and `more`
  - Also known as “pagers” these commands will output the contents of a file to the screen, but allow you to navigate through the file as well
- `head` and `tail`
  - These commands let you look at the lines starting at the top or bottom of the file, respectively.

Text manipulation

- `sort`
  - This can be used to sort input in many ways, alphabetically numerically, or with different fields as the sort key. Sorting is also available in reverse.
- `wc`
  - Short for "word count," this utility can count the number of lines, words, characters, bytes, and the length of the longest line in the file.
- `diff`
  - Usually used to generate patch files, `diff` can display the differences between two or more files.

Red Hat Certified System Administrator

Using System Documentation

Built-in Help

For many commands, simply running `commandname --help` will output enough information to toggle your memory. Then you can run the command you need.

Other options would be `-help`, `-?`, `-h`, or passing a flag you know doesn't exist, like `-linuxacademy`.

This should throw an error and have the program print out its help screen.

man Pages

The vast majority of commands that get run on a server were installed with a manual. The easy way to interface with this manual is the `man` command. For example:

```
man ls
```

If you're not sure what the name of the program you need help with is, but you have some idea, you can run `whatis nfs` to see a list of all man pages with "nfs" in the title.

Or you can search descriptions of man pages using `apropos nfs` to get a clue of what man page to look at.

/usr/share/doc

In addition to man pages and help flags, many packages install useful information into the `/usr/share/doc/<packagename>` folder.

The contents of the folder will vary by package but if the man pages and built in help don't get you what you need, it may exist in the doc folders.

Red Hat Certified System Administrator

Linking Files

Using File Links

There are two kinds of links:

Hard link: This is link between files that point to the same inode. Hard links cannot cross filesystems. Editing one of the linked files will edit the other (because they're pointing to the same space on the disk), but removing a hard link will not remove the data on the disk. There's still one entry pointing at the inode.

Soft link: This is more like a redirect. There's no actual file or anything at the link location, just something that points back to the original file. Removing the original file doesn't remove the soft link, but it does mean that it essentially points at nothing now.

Red Hat Certified System Administrator

File Permissions

Displayed Syntax

When you do an `ls -l` on a file, the first 10 characters tell you about the permissions on that file. For example:

```
-rw-r--r-- 1 cloud_user cloud_user 0 Mon Dy Ti:me filename.txt
```

We can break that permission list down into sections:

- Type of file (`-` = regular file, d= directory, b=device, l=symbolic (soft) link)

`rw-` Owner has Read and Write permissions

`r--` Group Owner has Read permissions

`r--` Everyone else has Read permissions

Octal Permissions

Sometimes you'll need to refer to those permissions in an octal format. Using the same example, we ignore the file type.

6 = User can Read (4) and Write (2)

4 = Group can Read (4)

4 = Everyone can Read (4)

The last bit that you can add is the Execute (1) bit. So the highest number you can have would be 7 (4+2+1).

SUID and SGID

If the permissions line looks like this:

```
-rws------
```

That means that the SUID bit has been set. The SUID bit means that the effective UID of the process is that of the file owner.

If it looks like this:

```
--x--s--x
```

Then the SGID bit is set, and it is similar to the SUID bit. The effective GID of the process is that of the group owner.

Sticky Bit

If you come across a directory that has permissions set like this:

```
drwxrwxrwt
```

That means that the Sticky Bit is set. Files can be written in this directory by anyone, but only the owner can remove their files (because they're sticky... get it?)

Changing Permissions and Ownership

`chmod`

Changes the permissions on files or directores, and can use either octal or letter notation

`chown`

Changes the owner/group owner of a file or directory, and can use either UID/GID or name.

Red Hat Certified System Administrator

Configuring the Firewall

Netfilter

The piece of software in the kernel that handles firewall interactions is called `Netfilter`. There are a few different ways to interact with `Netfilter` from user space, but we're primarily going to cover `firewall-cmd` and `firewalld` in this lesson.

The “old” way

`iptables` (or `ip6tables`) is the older way of configuring `Netfilter`.

It’s still functional, but the interface isn’t as user friendly as `firewalld` can be.

If you’re familiar with `iptables` already, some of that knowledge can apply to advanced `firewalld` commands, but it is not required.

The “new” way

`firewalld` (the service) and `firewall-cmd` (the configuration tool) are the newer ways to configure the `Netfilter` portion of the kernel.

Firewall Persistence

There are two configuration areas inside `firewalld`. Runtime or Permanent. Making a change in one does not require making a change in the other.

Making a runtime change takes effect immediately, but doesn't persist through reboots or `firewalld` restart.

Making a permanent change only takes effect on reboot or `firewalld` restart.

Firewall zones

`firewalld` comes with multiple zones already built in. A zone is just a convenient grouping of rules and things to apply those rules to.

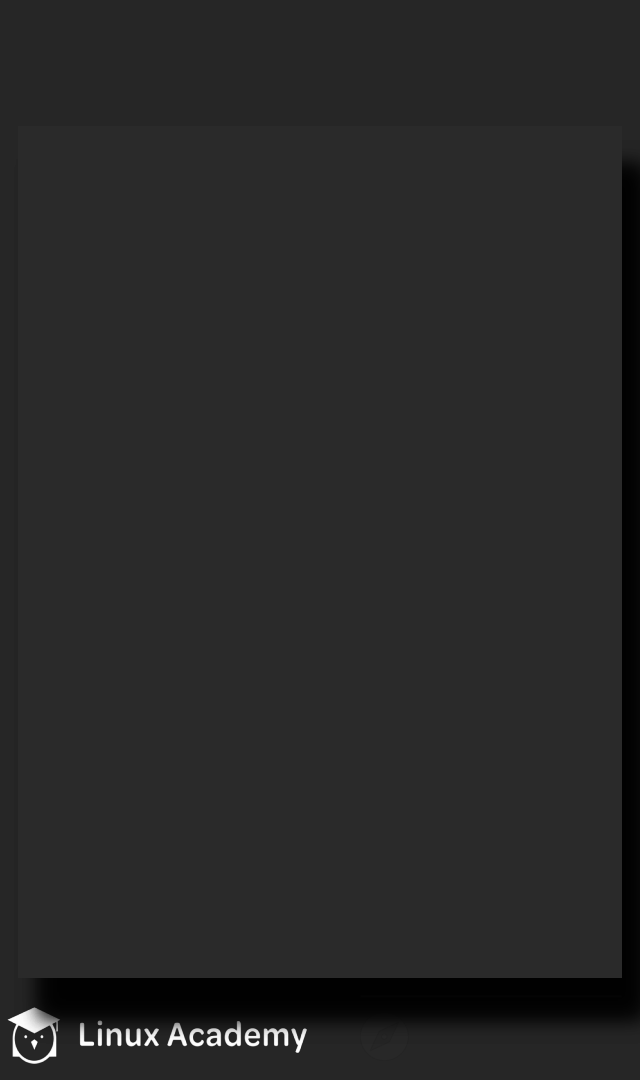
The "home" zone for example, could have a source of 192.168.0.0/24 and allow things like `samba`, `nfs`, `ntp`, and so on.

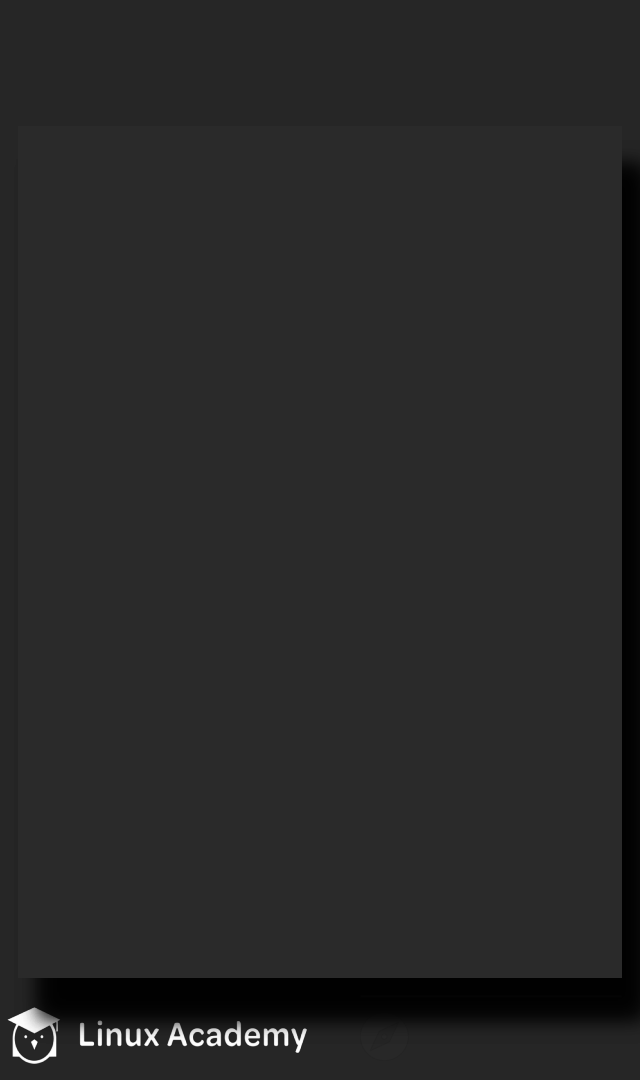
```
firewall-cmd –get-zones
firewall-cmd –get-default-zone
```

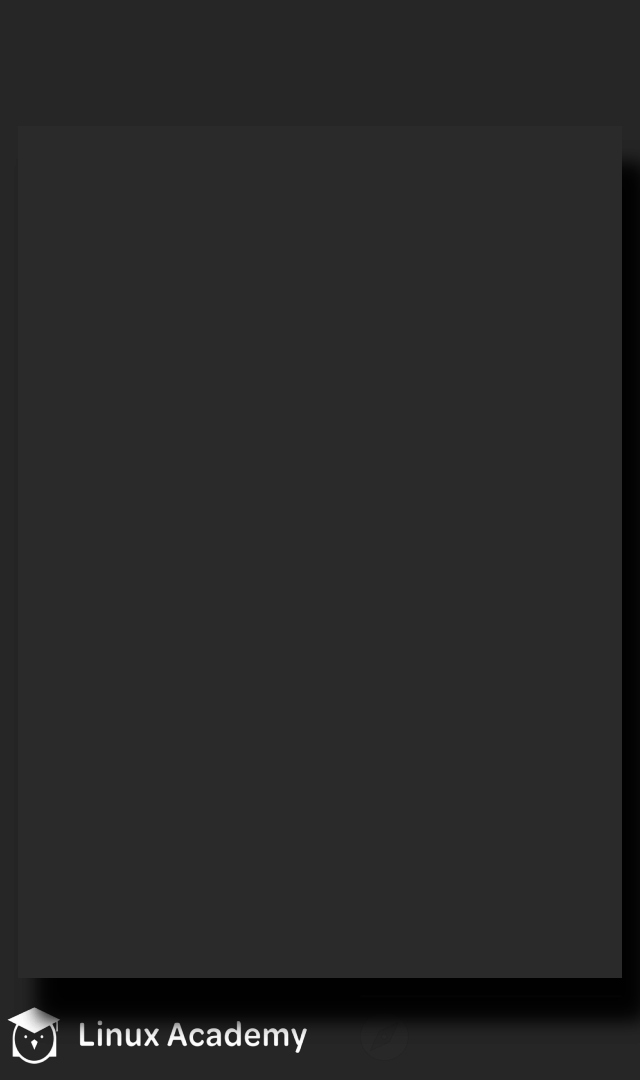
Firewall rules

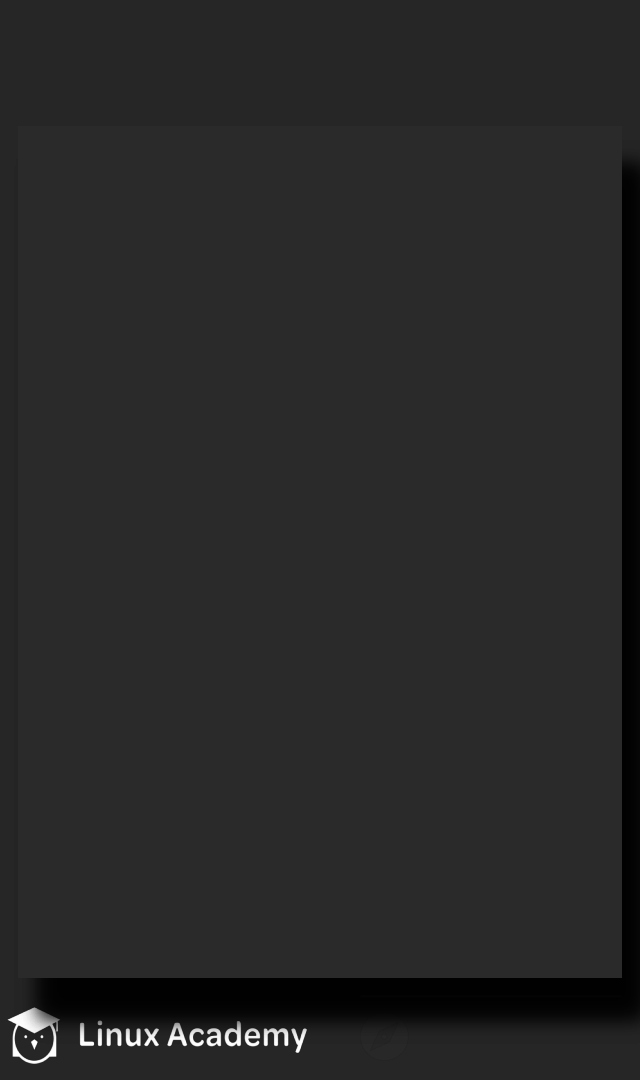
Once you know what zone you're working with, you can look at the rules that are being applied.

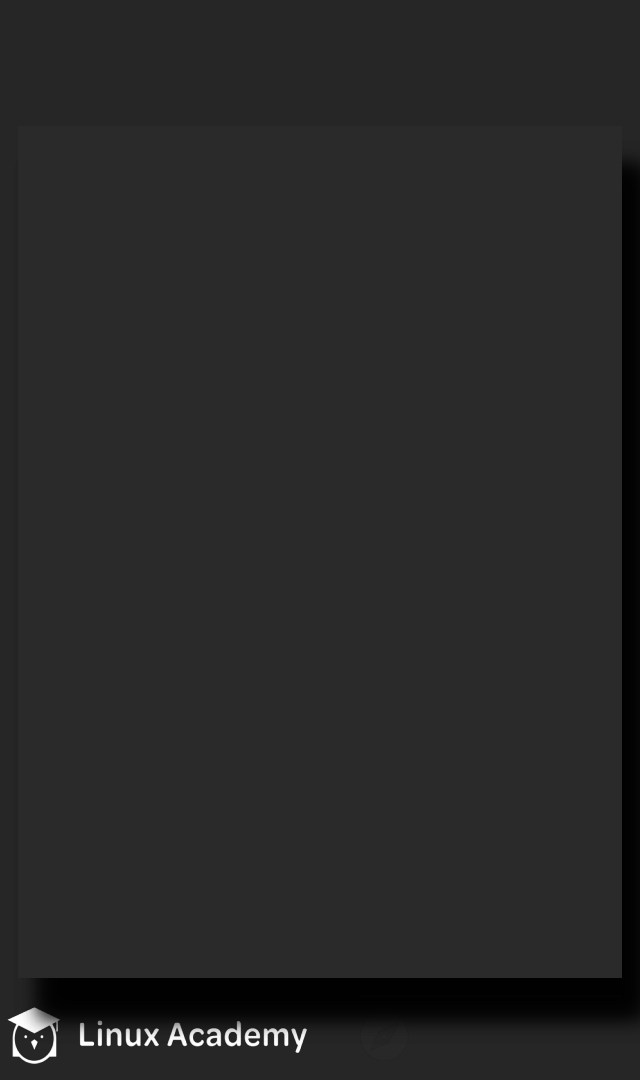
```
firewall-cmd --zone=home --list-all

firewall-cmd --zone=home –add-service=http

firewall-cmd --zone=home –add-port=80/tcp

firewall-cmd --add-source=192.168.0.0/24
```

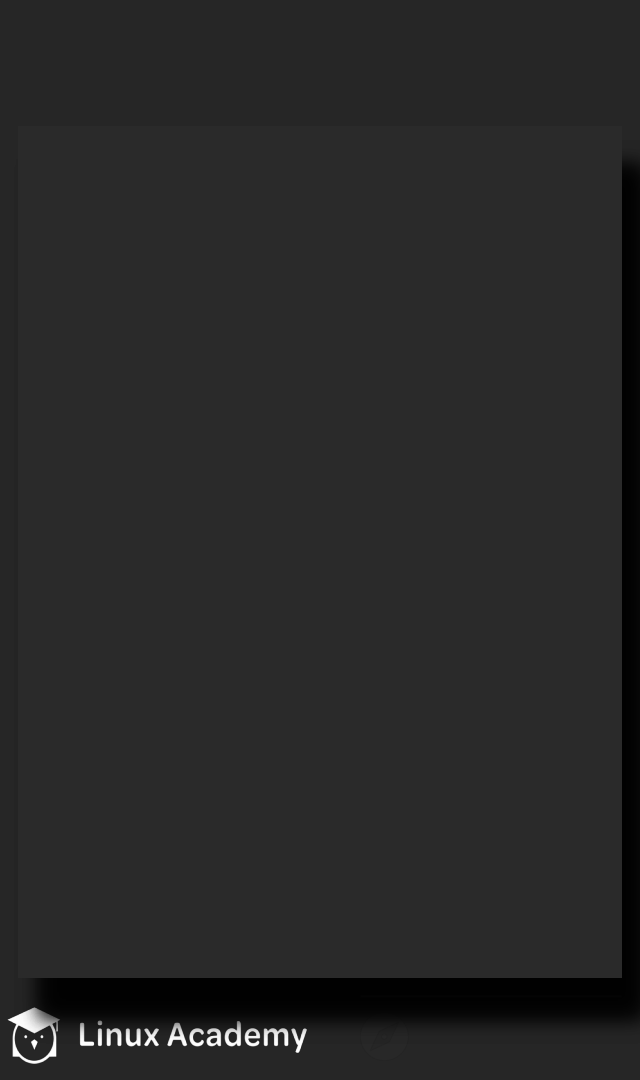

Linux Academy

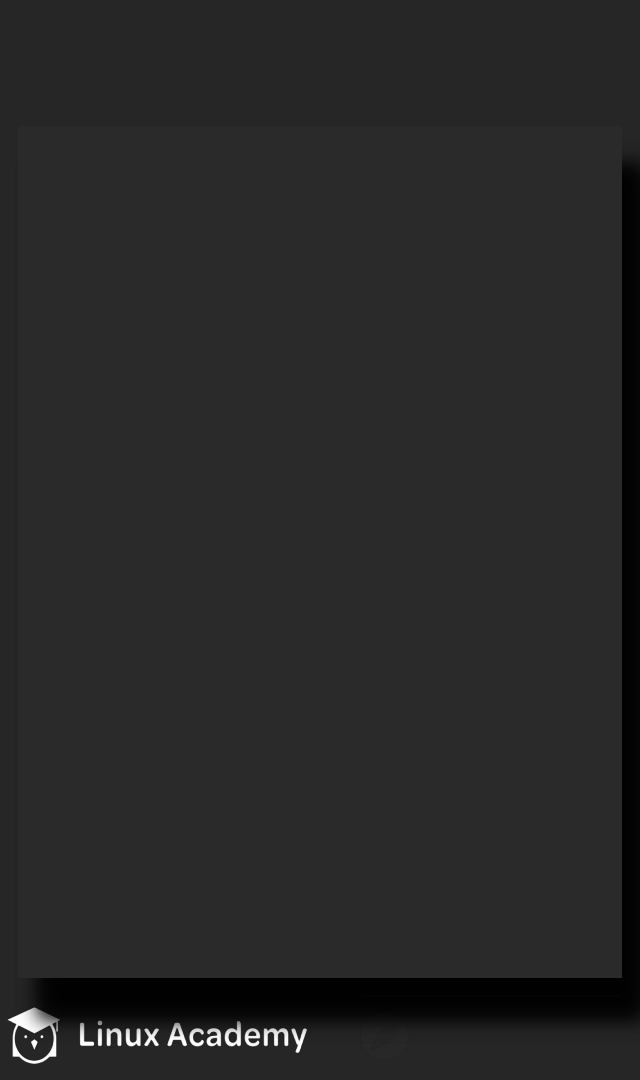

Linux Academy


Linux Academy


Linux Academy


Linux Academy


Linux Academy


Linux Academy

Red Hat Certified System Administrator

Using grep and Regular Expressions

What is grep?

grep is a program designed for finding matching patterns.

It can be used by itself:

```
grep Error /var/log/messages
```

Or as part of a pipe:

```
find . -name *.txt | grep taxes
```

grep Common Flags

`grep`

`-i:` Makes the search case insensitive

`-r:` Makes the search recursive through a directory structure

`-v:` Makes the search find all instances where there is not a pattern match

`-w:` Makes the search match on a word rather than any pattern match

grep Regular Expressions Pt. 1

`^`	Match expression at the start of a line, as in `^A`
`$`	Match expression at the end of a line, as in `A$`.
`\`	Turn off the special meaning of the next character, as in `\^`
`[ ]`	Match any one of the enclosed characters, as in `[aeiou]`, and use a hyphen for a range, as in `[0-9]`
`[^ ]`	Match any one character except those enclosed in `[ ]`, as in `[^0-9]`

grep Regular Expressions Pt. 2

`.` Match a single character of any value, except end of line

`*` Match zero or more of the preceding character or expression

`\{x,y\}` Match x to y occurrences of the preceding

`\{x\}` Match exactly x occurrences of the preceding

`\{x,\}` Match x or more occurrences of the preceding

grep Regular Expressions

Search files for lines with the word *linux*:

```
grep linux files
```

With *linux* at the start of a line:

```
grep '^linux' files
```

With *linux* at the end of a line:

```
grep 'linux$' files
```

Show lines containing only *linux*:

```
grep '^linux$' files
```

Lines starting with '^s', \ escapes the ^:

```
grep '\^s' files
```

Search for either *Linux* or *linux*:

```
grep '[Ll]inux' files
```

Search for BOB, Bob, BOb or BoB:

```
grep 'B[oO][bB]' files
```

Search for blank lines:

```
grep '^$' files
```

Search for pairs of numeric digits:

```
grep '[0-9][0-9]' file
```

Red Hat Certified System Administrator

Group Management

Getting Group Information

Groups allow us to group users together for a set of permissions. Getting relevant group information can be done a few different ways.

`id` command

`getent` command

Manually looking at `/etc/group`

Adding Groups

You may find it necessary to manually add groups to your server. It's relatively easy to do so using the `groupadd` command.

`g` – specify the Group ID

Changing a User's Groups

`usermod` can be used to change a user's primary group and a user's supplementary groups.

A user can have a single primary group, but any number of supplementary groups.

`g` – change a user's primary group

`G` – change a user's supplementary group. The a flag can be used to append a group rather than replace.

Using Supplementary Groups

Directories can be set to restrict access to members of a specific group.

Changing Group Information

The easiest way to change group information is by using the `groupmod` command.

`g` – Change the Group ID

`n` – Change the name of a group

Red Hat Certified System Administrator

Updating and Installing Packages

Red Hat Package Manager

The `rpm` command is historically how software is installed on Red Hat servers. It's also one way to manage software on the server after installation.

`i` – Install

`h` – Give progress information

`v` – Verbose

`U` – Upgrade a package

`e` – Erase/Remove a package

We'll discuss these for completions sake, but these are very rarely correct to use.

`nodeps` – Ignore dependencies

`force` – Ignore errors

Repositories

The `yum` command uses network software repositories to figure out what packages are available, and what dependencies those packages have.

We can look at what repos we have configured by using the `yum repolist` command.

We can install packages using `yum install`, remove packages using `yum remove`, and search available packages using `yum search`.

Red Hat Certified System Administrator

Logging

“Standard” Logs

Most logs are written to `/var/log` in a normal text format. Some boot information, normal messages, and most services write their logs in that directory. What services write where is partially controlled by the `rsyslog` service. Looking at `/etc/rsyslog.conf` will show what services log where, and at what log priority.

Rotating Logs

The longer a log gets, the more difficult it is to read or interpret meaningful information from.

Fortunately there is a service that will automatically rotate logs based on configurable settings, so that you can keep older logs for reference and always have fresh information available.

The `logrotate` service's configuration file is at: `/etc/logrotate.conf`.

systemd and journalctl

In addition to the text-based log files in `/var/log`, systemd keeps logs stored in a binary, searchable format. By default it is not persistent across reboots, but that can be changed.

```
mkdir /var/log/journal
chgrp systemd-journal
/var/log/journal
chmod 2775 /var/log/journal
systemctl restart \
systemd-journal.service
```

Why Use the Journal?

Since the journal is binary, it's difficult to interact with. So why use it?

Simply put, it's powerful.

Rather than searching through every log file for errors, run:

```
journalctl -p err
```

To find errors since yesterday:

```
journalctl -p err -since \
yesterday
```

To find all messages associated with UID 1000:

```
journalctl _UID=1000
```

Red Hat Certified System Administrator

Logging in and Switching Users

Using the **su** Command

The **su** command (substitute user) allows you to substitute another user for your own when logged in.

You'll be prompted for a password and then, if successful, placed in a shell belonging to the user specified in the command.

User Environments

A user environment is a collection of directories, files, and settings that set up how things look when a user logs in, what commands are run, and what that user has access to.

By default, the `su` command retains the environment of the user who ran the command. So if Lucy has a different `$PATH`, for example, just typing `su lucy` won't set your path up the way hers is.

For that to happen you need to use the `-` argument to `su`. So your command would be `su - lucy` if you wanted to inherit Lucy's `$PATH`.

Single Command with `su -c`

If all you want to do is run a single command as the substitute user, you can run `su -c fdisk` (as an example). In this instance you don't really need to pass the `-` argument because your intent is not to get a shell, just to run a single command.

Red Hat Certified System Administrator

LVM

Logical Volume Manager

Rather than addressing disks, LVM provides a method of addressing a pool of space to manage storage.

A Logical Volume sits inside a Volume Group that can span Physical Volumes. A Volume Group can contain a number of Logical Volumes.

When a Physical Disk is set up for LVM, metadata is written at the beginning of the disk for normal usage, and the end of the disk for backup usage.

Creating a Physical Volume

```
pvcreate /dev/device \
/dev/device2 /dev/device3
```

There aren't any common flags for this command – its usage is pretty straightforward.

Creating a Volume Group

```
vgcreate myVG /dev/device \
/dev/device1 /dev/device2
```

One common switch is:

`s` - Set the Physical Extent size

This changes the default physical extent from 4MiB to whatever you set it to.

Note that this has no effect on normal I/O performance – large numbers of extents can slow down LVM tools, but for actual usage it doesn't matter as of LVM2.

Creating a Logical Volume

```
lvcreate -L 100m myVG
```

Common switches include:

`n` – allows you to set the Logical Volume's name.

`l` – use extents rather than a specified size for calculating the size of the logical volume