u train

Linux Commands Part 2

utrains.org

*“ Before starting this lesson, make sure you are done with **Linux Commands part 1** lesson and understood the various commands taught*

*Also make sure you went through the lesson on **Linux top level directory structure**”*

Table of content

1. Review on the root directory
2. Navigating through the root directory: /var directory
3. Filter login activity on a server
4. Software management
5. Some advanced concepts

Important note!

Before starting this lesson,

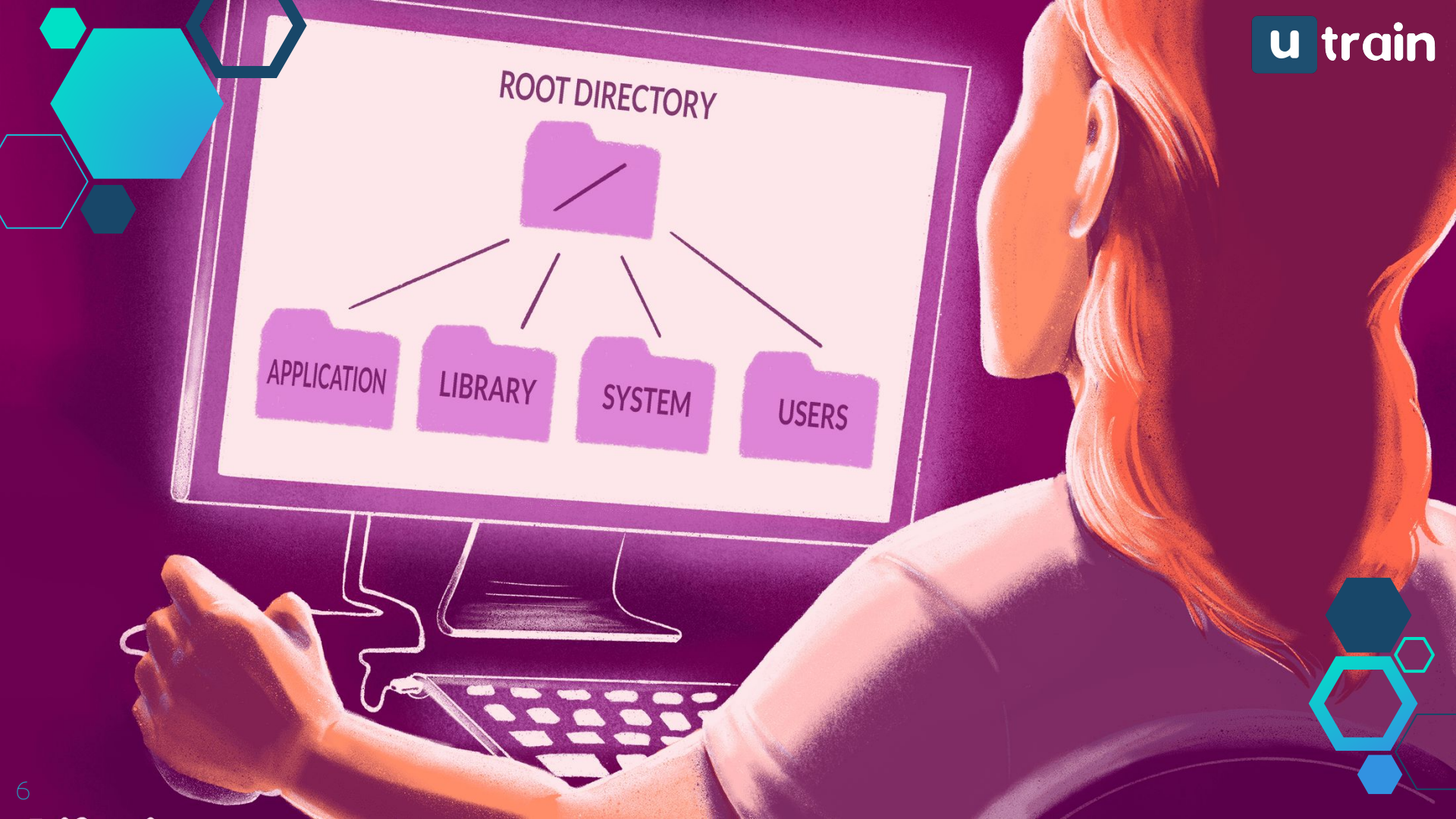
- Launch your **Visual studio code**,
- Open a **terminal** or use the one that is opened
- Check the VMs on your computer: **vagrant global-status**
- Copy the **ID** of a **Centos 7 server**. If you don't have one, **please install it now**
- Resume or start a **Centos 7 server**: **vagrant resume ID** or **vagrant up** (this works just fine)
- Connect remotely to a **Centos 7 server: vagrant ssh ID**

That said and done, Let's get started!

1

The root directory

Review


ROOT DIRECTORY
/
APPLICATION
LIBRARY
SYSTEM
USERS

The root directory: Review

- The **root directory** is called **/**
- This is the first or top level directory in the **file system hierarchy**.
- The file system is like an upside down tree that starts with the root directory
- Simply put, the root is the base that contains all other directories and files.
- For this lesson, connect as the root to the server (remember the **$ su** command)
  - $ **su root**
  - Enter the password: **vagrant** and hit enter

The root directory: review

- To get in the root directory, type the command: **# cd /**
- You can type **# pwd** to check the current directory

```
[root@localhost vagrant]# cd /
[root@localhost /]# pwd
/
[root@localhost /]#
```

- Let's list the content of this directory and explain the role of some subdirectories in there!

```
[root@localhost /]# ls
bin   dev  home  lib64  mnt  proc  run   srv      sys  usr     var
boot  etc  lib   media  opt  root  sbin  swapfile tmp  vagrant
```

Review on the root subdirectories

The root directory: review

Name of the file or directory	Content or function
bin	Directory that contains all the command a regular user can run in a terminal
sbin	Directory that contains all the command the root user can run in the terminal

NB: If a regular user tries to run a command that is found only in the file **sbin**, the system will display a **PERMISSION DENIED**. But if the content of **sbin** is moved to the **bin** file, then a regular user will be able to access and run them.

You can always list the content of these directories to check the commands we have learned so far. # cat bin or # cat sbin

The root directory: review

Name of the file or directory	Content or function
dev	Directory for devices like the hard drive, USB flash etc.
home	Directory that contains all the users created on the system
proc	Contains some files which keep informations concerning the hardware
root	Home directory for the root user

The root directory: review

Name of the file or directory	Content or function
opt	Folder used when installing third parties applications
var	Contains the logs (messages generated while the system is running)
etc	contains informations about configuration files and informations about users accounts

Let's open some of these directories to display their content!

2

Navigating through the Root directory: The /var directory

cd command

The /var directory

- From where we are, let's get into the **/var** directory,
- Run the commands:
  - **# pwd**
  - **# cd var**
  - **# pwd**

```
[root@localhost /]# pwd
/
[root@localhost /]# cd var
[root@localhost var]# pwd
/var
```

- Now, run ls to list the content of the current directory

```
[root@localhost var]# ls
adm    db     games   kerberos  local  log   nis  preserve  spool  yp
cache  empty  gopher  lib       lock   mail  opt  run       tmp
```

The /var directory

Let's concentrate on the **log** directory here

- Now, open and list the content of the **log** directory here: **# cd log**
- Take a look at the new path (output of the **# pwd** command)

```
[root@localhost var]# cd log
[root@localhost log]# pwd
/var/log
```

- It contains many files and directories. Let's explain the function of two important directories in here.

```
[root@localhost log]# ls
anaconda        dmesg.old          qemu-ga            tallylog
audit           grubby_prune_debug rhsm               tuned
btmp            lastlog            samba              wtmp
chrony          maillog            secure             yum.log
cron            maillog-20211220   secure-20211220
cron-20211220   messages           spooler
dmesg           messages-20211220  spooler-20211220
```

The /var directory

Name of the file or directory	Content or function
messages	It contains informations generated by the system while it is running. It is like the Kernel Buffer
secure	It generates informations about users log in activities (how users log in to the system)

The /var directory

- Let's display the content of the file **secure:** **# cat secure** Go through by scrolling up and down.
- **NB:** This file can help us investigate if someone is trying to break into our system.

```
[root@localhost log]# cat secure
Dec 21 06:03:43 localhost su: pam_unix(su:session): session opened for
user root by vagrant(uid=1000)
Dec 21 10:08:49 localhost su: pam_unix(su:session): session closed for
user root
Dec 21 10:11:21 localhost su: pam_unix(su:session): session opened for
user root by vagrant(uid=1000)
```

The /var directory

- On the company servers, this is generally a very big file because many users login everyday on those servers
- So, instead of using # **cat** to display the content, we will use the command **more**: **# more secure**
- The **# more** command is used to display the contain of a file **one page at the time**
- We hit the **spacebar** on the keyboard to go to the next page or press **Enter** to read the file line by line
- **# more** helps us to navigate a big file page by page while **# cat** dumps everything at once.

The /var directory

- There is another useful command that can be used at the place of **# cat**. That is the **# less** command
- This command is similar to the **# more** command, it also displays one page of the file at the time but the **# less** command is faster
- In fact, **it does not load the entire file at once** like **# more**
- The result is loaded progressively and displayed one page at the time
- Hit the **q** key to quit
- **Don't hesitate to google some of this commands to better understand how they work!**

Always remember this!

While learning, when you get stuck on something, don't panic.

- Your first reaction must be to use google or youtube to find some answers to your problem.
- The best way is to make some research on your own either on **Google** or on **Youtube**. you can find tutorials and videos on youtube giving more explanations that will allow you to better understand these notions!

The /var directory

Question: **Where do all these files and directories come from?**

Answer: **These files are created during the installation of the system**

- Some informations in files such as **messages** can help the engineer to troubleshoot the computer when some system problems are encountered.
- You can open and navigate this file with the same commands we saw previously:
  - **# cat messages**, **# more messages** or **# less messages**

The /var directory

- At this stage, our current directory is **/var/log** (you can use **# pwd** to display it).
- Let's say we want to go back to the var directory. How do we proceed?
- We use the command **# cd ..**

```
[root@localhost log]# pwd
/var/log
[root@localhost log]# cd ..
[root@localhost var]# pwd
/var
```

- To go many steps behind, just separate the **..** by a **/**

Example: To go three steps behind in the path, we use

- **# cd ../../..**

```
[root@localhost var]# cd ../../..
[root@localhost /]# pwd
/
```

- **NB: The end of the path is always the root directory**

The /var directory

- When you already know the **full path** to a directory or a file, you can use that directly in the **cd** command, instead of opening directories one after the other.
- The same applies to command such as **# ls**, **# ll** etc..

Example: Let's say we want to open a directory in **/var/log.**

- We already know the name of the directory (for me here i take the directory **samba**). To open this file directly, i will type:
  - **# cd /var/log/samba**

The /var directory

- So there are **two ways** to access a file or a directory:
  - Either you **cd** progressively till you reach the file/directory
  - Or, you run the command with the **full path** to the file/directory
- To go back to the previous directory in which you were, use the command:
  - **# cd -**
- Remember anytime you want **to go back to the root directory**, you can just type **# cd** and that will be done

“ Use the cd command to navigate through your system. List the root directory and open the directories you find in there. Go through their content and get used to **cd -**, **cd ..**, **cd ../..***etc. ”*

3

Filter login activity in /var/log/secure file

The grep command

The grep command

- **The grep command is used to filter a specific string from files or outputs**
- To use this command we follow the syntax:
  - # **grep text-to-filter file-Path**
- Let's go back to the file **secure** we used previously in **/var/log**:
  - **# cd /var/log/**

Filter login activity

- Now, let's use the **grep** command on this file: **# grep root secure**
- This command goes in the file **secure**, filters all the lines on which the word **root** appear and displays them.

```
[root@localhost /]# cd /var/log/
[root@localhost log]# grep root secure
Dec 21 06:03:43 localhost su: pam_unix(su:session): session opened for user root by vagrant(uid=1000)
Dec 21 10:08:49 localhost su: pam_unix(su:session): session closed for user root
Dec 21 10:11:21 localhost su: pam_unix(su:session): session opened for user root by vagrant(uid=1000)
```

Filter login activity

Now, let's try to **login as root** user **with a wrong password** and check the root login activity in **/var/log/secure**

- Run the command **# exit** to logout of the root user account
- Try login in back with the command: **$ su** root then enter a wrong password (type any word)

```
[root@localhost log]# exit
exit
[vagrant@localhost ~]$ su root
Password:
su: Authentication failure
```

Filter login activity

- Now, login back with the correct password: **$ su** then enter **vagrant** as password
- Navigate to the **/var/log/** directory and filter the root login activity
  - # **cd /var/log/**
  - # **grep root secure**
- You can see that the login activity of the root user has been modified (One line has added)

Filter login activity

- Let's try with another word:
  - **# grep failed secure**
  - **# grep Failed secure**
- You realise that the **grep** command is **case sensitive**.
- To ignore the case sensitivity while filtering, we add the option **-i** to the command.
- Thus **# grep -i Failed secure** will display all the lines that contains the word **Failed (or failed)** from the file **secure**.

4

Software management commands

Yum, apt

Software management

- Installing applications is a necessary step if you want to use the server efficiently.
- The system will just be empty when applications are not installed.
- In fact, to host a website on your server, you need to download and install a website hosting software like **Apache**
- To host a database, you need also to install a software like **MySQL**
- Thus for each service you will provide on your server, **softwares will be needed**

Download and install softwares

Download and install software

- When working on **Windows systems**, we use to download and install softwares from the Internet step by step on the computer.
- On Linux systems, we can do this through the command line with **yum** (on CentOs) or **apt** (on Ubuntu).
- **yum** is a command we use **to download and install softwares on some Linux systems**.
- We can even use it to download other commands on our system when we encounter the **Command not found** issue

Download and install software

- To download and install a package or a software on your linux server, you need to **switch to the root user account**
- To do that, run the following command: $ **su root**
- Enter the password: **vagrant**
- While typing the password, it will not be displayed on the screen but it is taking it
- After validation, you can verify that you are now working in the **root account**:

```
[vagrant@localhost home]$ su root
Password:
[root@localhost home]#
```

Download and install software

NB: To be able to download and install softwares with yum, you need to make sure that you have an internet connexion.

- You can verify that by running the command **# ping google.com**
- Press **(Ctrl) + (c)** to quit the command (kill the process)

```
[root@localhost home]# ping google.com
PING google.com (172.217.169.14) 56(84) bytes of data.
64 bytes from lhr25s26-in-f14.1e100.net (172.217.169.14): icmp_seq=1 ttl=106 time=193 ms
64 bytes from lhr25s26-in-f14.1e100.net (172.217.169.14): icmp_seq=2 ttl=106 time=203 ms
64 bytes from lhr25s26-in-f14.1e100.net (172.217.169.14): icmp_seq=3 ttl=106 time=165 ms
64 bytes from lhr25s26-in-f14.1e100.net (172.217.169.14): icmp_seq=4 ttl=106 time=168 ms
64 bytes from lhr25s26-in-f14.1e100.net (172.217.169.14): icmp_seq=5 ttl=106 time=176 ms
```

Download and install software

- Now, let's say we want to install **finger** on our system
- This is a command used to check users accounts on the system.
- To download and install it we will run: **# yum install finger**
- **This command will look for the package finger on the net, download and install it automatically!**
  - A **confirmation question** will be ask to check if you are sure of what you are about to do.
  - Just type **y** for yes or **n** for No
- To avoid this question, you could instead use **# yum install finger -y**

Download software with yum

- When the installation is done, you can use the command **finger** to check how it works

Example:

- **# finger vagrant** gives informations about user **vagrant**
- **# finger root** gives informations about the **root**

```
[root@localhost home]# finger vagrant
Login: vagrant                          Name: vagrant
Directory: /home/vagrant                Shell: /bin/bash
On since Mon Dec 20 16:44 (UTC) on pts/0 from 10.0.2.2
   4 seconds idle
No mail.
No Plan.
```

Download software with yum

- You can check the number of softwares you are able to access and install with **yum** by running the command: **# yum repolist**
- **We will learn how to increase this number in future lessons**

```
[root@localhost home]# yum repolist
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirrors.pt
 * extras: mirror.isec.pt
 * updates: mirror.ufs.ac.za
repo id                                    repo name                                  status
base/7/x86_64                              CentOS-7 - Base                            10,072
extras/7/x86_64                            CentOS-7 - Extras                             500
updates/7/x86_64                           CentOS-7 - Updates                          3,190
repolist: 13,762
```

Uninstall softwares

Uninstall software

- You can also uninstall softwares with **yum.** To do that, you must run the command: **#yum remove appName**

Example: To uninstall **finger**, we run: **# yum remove finger -y**

```
Removed:
  finger.x86_64 0:0.17-52.el7

Complete!
```

- Try to run the **finger** command again

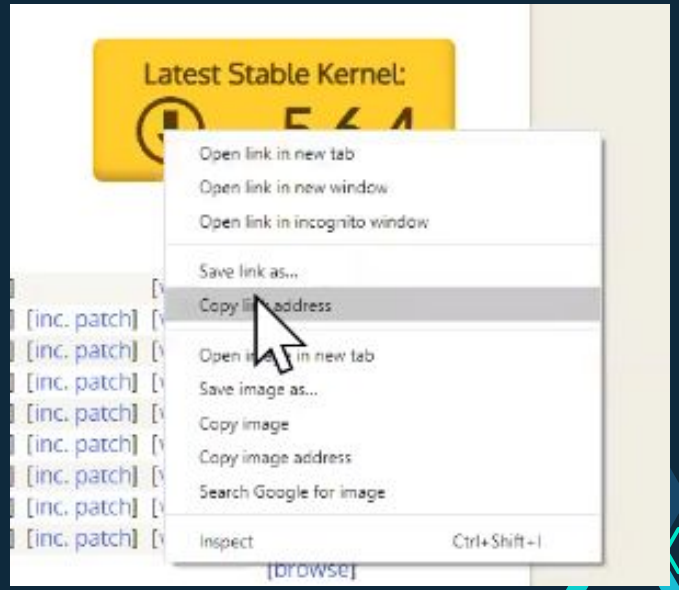
```
[root@localhost home]# finger vagrant
bash: /usr/bin/finger: No such file or directory
```

The wget command

The wget command

Let's see another useful tool in this section.

- **wget** (**Web get**) is a command we use to download a specific document from the internet. To run it, type:
- **# wget documentURL**
- **Example:** Let's go to kernel.org with our browser
- Right click on the **software download link**
- Click on **copy link address**



The wget command

- Now, in the terminal, type **# wget** (do a right click and paste the link here and hit Enter)

```
[root@localhost home]# wget https://cdn.kernel.org/pub/linux/kernel/v5.x/linux-5.15.10.tar.xz
```

- Remember to press **y** to confirm the download

Note: If you get a **Command not found** issue, just install **wget** with yum:

- **# yum install wget -y**

The wget command

- At the end of the installation process, use **ls** to verify that the document was successfully downloaded

```
[root@localhost home]# ls
cup  gym  home  linux-5.15.10.tar.xz  review  serge  work  workload
```

- If you want to remove it from your computer, you can use **# rm** as for any other file
  - **# rm linux-5.15.10.tar.xz**

The wget command

A useful tip!

You can start typing the name of the file and then use the **tabulation key** to autocomplete it.

5

Some advanced concepts

pipe, redirect, append

The pipe

The pipe

- This is a tool we use in the command line to **combine two or more commands**
- The **pipe** (**|**) takes the output of a command and passes it to the other command
- The syntax is **# command1 | command2 | command3 | etc…**
- **Example:** Let's combine the **# ll** and the **#grep** command

On most computers, you need to press and hold the **Shift** key before pressing it

The pipe

- Here we use **grep** command to filter the output of the **ll** command using the pipe
  - **# ll | grep work**

```
[root@localhost home]# ll
total 119096
-rw-rw-r--. 1 vagrant vagrant         0 Dec 20 20:23 cup
-rw-rw-r--. 1 vagrant vagrant       174 Dec 21 01:31 gym
drwxrwxr-x. 2 vagrant vagrant        19 Dec 21 01:21 home
-rw-r--r--. 1 root    root    121948408 Dec 17 09:39 linux-5.15.10.tar.xz
-rw-rw-r--. 1 vagrant vagrant         0 Dec 20 20:33 review
drwxrwxr-x. 2 vagrant vagrant         6 Dec 20 20:31 serge
drwxrwxr-x. 2 vagrant vagrant         6 Dec 20 20:31 work
drwxrwxr-x. 2 vagrant vagrant         6 Dec 21 01:14 workload
[root@localhost home]# ll|grep work
drwxrwxr-x. 2 vagrant vagrant         6 Dec 20 20:31 work
drwxrwxr-x. 2 vagrant vagrant         6 Dec 21 01:14 workload
```

Redirect and append

Redirect and Append

- **Redirect** (**>**) and **append** (**>>**) are used to **capture the output of a command in a file.**
- **Example 1:** Let's start with redirect (>). Take a look at the following image:

```
[root@localhost home]# ls
cup  gym  home  linux-5.15.10.tar.xz  review  serge  work  workload
[root@localhost home]# ls > file1
```

- Here we listed the current directory with **# ls** and secondly we did the same but this time around, **we redirected the output to a file named file1 using the redirection sign >**

Redirect and Append

- **NB:** If file1 does not exist, the redirection sign will create it before dumping the output of the **ls** command in it.
- When listing the content of the current directory again, we realise that a new file has been added ie. **file1**.

```
[root@localhost home]# ls
cup  file1  gym  home  linux-5.15.10.tar.xz  review  serge  work  workload
```

Redirect and Append

- Let's display the content of **file1** to see if our command worked correctly: **# cat file1**

```
[root@localhost home]# cat file1
cup
file1
gym
home
linux-5.15.10.tar.xz
review
serge
work
workload
```

- <u>Compare the output of the # ls command with the content of file1</u>

Redirect and Append

- **Example 2:** Let's list the content of the current directory again with **# ll** and redirect it into file1 : **# ll > file1**

```
[root@localhost home]# ll
total 119100
-rw-rw-r--. 1 vagrant vagrant         0 Dec 20 20:23 cup
-rw-r--r--. 1 root    root           67 Dec 21 08:50 file1
-rw-rw-r--. 1 vagrant vagrant       174 Dec 21 01:31 gym
drwxrwxr-x. 2 vagrant vagrant        19 Dec 21 01:21 home
-rw-r--r--. 1 root    root    121948408 Dec 17 09:39 linux-5.15.10.tar.xz
-rw-rw-r--. 1 vagrant vagrant         0 Dec 20 20:33 review
drwxrwxr-x. 2 vagrant vagrant         6 Dec 20 20:31 serge
drwxrwxr-x. 2 vagrant vagrant         6 Dec 20 20:31 work
drwxrwxr-x. 2 vagrant vagrant         6 Dec 21 01:14 workload
[root@localhost home]# ll > file1
```

- Now if you check the content of file1 with **# cat file1**, you will realise that the previous content is no more found in the file! It contains only the output of the **# ll** command.

Redirect and Append

- Redirect (>) and append (>>) do almost the same thing. But when you use append, the new content is added at the end of the file.
- **That means append keeps the old content while redirect erases it.**
- **Example:** Check the content of file2 at the end of these commands

```
[root@localhost home]# echo "It is not easy but it is fun" > file2
[root@localhost home]# cat file2
It is not easy but it is fun
[root@localhost home]# echo "I can't wait to start working" >> file2
[root@localhost home]# cat file2
It is not easy but it is fun
I can't wait to start working
```

“ Play around with these commands and do not hesitate to post your questions when you encounter some problems. ”

See you guys in the next lesson!

Thanks!

Any questions?

You can find us at:

website: http://utrains.org/

Phone: +1 (302) 689 3440

Email: contact@utrains.org

Click on the link below to contact the support team for any issue!

utrains.org/support/

Create a ticket for your problem and we will get back to you soon!