u train

Linux Commands Part 1

utrains.org

Table of content

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**

If you find any difficulty following these steps, please go back to the lesson **Centos-Ubuntu installation with vagrant**

Go through the steps and make sure you can **connect remotely to a vagrant virtual machine.**

That said and done, let's get started!

1

System inventory commands

System inventory

System inventory

- A **linux server** is just a computer!!!
- A computer is basically made up of the following:
  - **CPU**
  - **Memory**
  - **Hard drive**
  - **Operating system**
  - **Kernel (core or foundation of the OS)**
- Let's look at each component

The Central Processing Unit (CPU)

System inventory: CPU

- During the system inventory of a linux server, we will often be called upon to check its CPU characteristics
- To do that, we run the command: $ **lscpu**
- This command will display every useful information about the CPU (**architecture, number of cpu, Model, frequency** etc.)

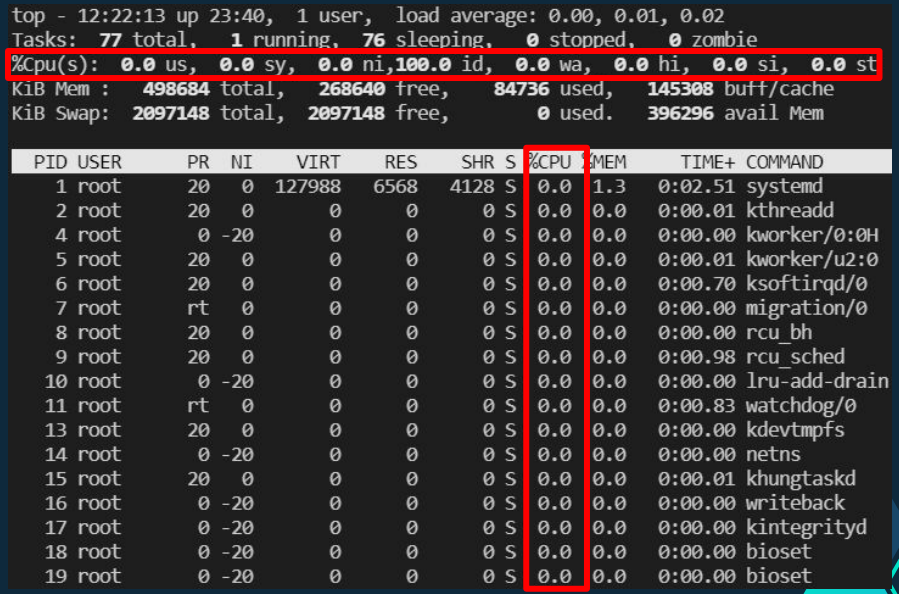
```
[vagrant@localhost ~]$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                1
On-line CPU(s) list:   0
Thread(s) per core:    1
Core(s) per socket:    1
Socket(s):             1
NUMA node(s):          1
Vendor ID:             AuthenticAMD
CPU family:            23
Model:                 104
Model name:            AMD Ryzen 7 5700U with Radeon Graphics
Stepping:              1
CPU MHz:               1796.628
BogoMIPS:              3593.25
Hypervisor vendor:     KVM
Virtualization type:   full
L1d cache:             32K
L1i cache:             32K
L2 cache:              512K
L3 cache:              8192K
NUMA node0 CPU(s):     0
```

System inventory: CPU

- To quickly check the **number of CPUs** on a server, we can run the command: $ **nproc**

```
[vagrant@localhost ~]$ nproc
1
```

- To verify the **usage percentage of the CPU** by the various programs running on the server we can use the command: $ **top**



The memory

Temporary storage

System inventory: Memory

- **To check the memory,** we can use this the **free -m** command
- This command print the
  - **total size,**
  - **the used size,**
  - **the free size**
  - and other important characteristics of the memory.
- Here is the output of the **free -m** command in the terminal

```
[vagrant@localhost ~]$ free -m
               total        used        free      shared  buff/cache   available
Mem:             486          82         263           4         141         387
Swap:           2047           0        2047
```

System inventory: Memory

- **We can also check the memory usage with the command top**
- The **top** command prints more informations than the **free -m** command

```
top - 21:26:24 up  8:44,  1 user,  load average: 0.00, 0.01, 0.04
Tasks:  78 total,   2 running,  76 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.3 us,  0.0 sy,  0.0 ni, 99.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :   498684 total,   268988 free,    84728 used,   144968 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.   396304 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    1 root      20   0  127988   6568   4128 S   0.0   1.3   0:02.28 systemd
    2 root      20   0       0      0      0 S   0.0   0.0   0:00.01 kthreadd
    4 root       0 -20       0      0      0 S   0.0   0.0   0:00.00 kworker/0:0H
    5 root      20   0       0      0      0 S   0.0   0.0   0:00.01 kworker/u2:0
    6 root      20   0       0      0      0 S   0.0   0.0   0:00.35 ksoftirqd/0
    7 root      rt   0       0      0      0 S   0.0   0.0   0:00.00 migration/0
    8 root      20   0       0      0      0 S   0.0   0.0   0:00.00 rcu_bh
    9 root      20   0       0      0      0 S   0.0   0.0   0:00.83 rcu_sched
```

The Hard drive

Permanent storage

System inventory: Hard drive

- The **# lsblk** command prints the **size of the Hard drive with all the partitions included**. The output on my server is:

```
[vagrant@localhost ~]$ lsblk
NAME   MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda      8:0    0  40G  0 disk
└─sda1   8:1    0  40G  0 part /
```

- On this server, we can see only one partition of the **sda disk**
- The partitions of the sda drive will generally be named **sda1, sda2, sda3** etc

The OS

Operating system

System inventory: os version

- To check informations on the OS version of your CentOS 7 server, run the command:
  - **$ cat /etc/*release**
- The output looks like the following

```
[vagrant@localhost ~]$ cat /etc/*release
CentOS Linux release 7.8.2003 (Core)
NAME="CentOS Linux"
VERSION="7 (Core)"
ID="centos"
ID_LIKE="rhel fedora"
VERSION_ID="7"
PRETTY_NAME="CentOS Linux 7 (Core)"
ANSI_COLOR="0;31"
CPE_NAME="cpe:/o:centos:centos:7"
HOME_URL="https://www.centos.org/"
BUG_REPORT_URL="https://bugs.centos.org/"

CENTOS_MANTISBT_PROJECT="CentOS-7"
CENTOS_MANTISBT_PROJECT_VERSION="7"
REDHAT_SUPPORT_PRODUCT="centos"
REDHAT_SUPPORT_PRODUCT_VERSION="7"

CentOS Linux release 7.8.2003 (Core)
CentOS Linux release 7.8.2003 (Core)
```

The kernel

Name, release, version ...

System inventory: Kernel

- To print some system information like the kernel release, the kernel version, we use the command **uname** with a specific option
  - **$ uname -s** to print the kernel name
  - **$ uname -r** to print the kernel release

```
[vagrant@localhost ~]$ uname -s
Linux
[vagrant@localhost ~]$ uname -r
3.10.0-1127.el7.x86_64
```

  - **$ uname -a** to print all system information
- To check the manual on this command, you can run $ **man uname**
  - Press the **key Q** on your keyboard to **quit the manual page**

2

Navigation & files/directories commands

Files and directories

Creating directories

- To create a directory/folder, use the command: **mkdir directoryName**
- **Example**: let's create a directory named gym in our current directory
  - **$ mkdir gym**
- To make many directory with the same command:

  $ mkdir dirName1 dirName2 dirName3 ...
- **Example**: Let's make three directories named **work, home** and **serge**
  - **$ mkdir work home serge**

Creating files

- To create a new file, use the command: **touch directoryName**
- **Example**: let's create a file named **bottle** in our current directory
  - **$ touch bottle**
- To create many files with the same command:

  $ touch file1Name file2Name file3Name …
- **Example**: Let's make four files named line, linux, cup and class. Use $ **ls** to verify
  - **$ touch line linux cup class**

```
[vagrant@localhost ~]$ touch bottle
[vagrant@localhost ~]$ touch line linux cup class
[vagrant@localhost ~]$ ls
bottle  class  cup  gym  home  line  linux  serge  work
```

List directory content

- To list the content of a directory, we use the command **ls directoryPath**
- To list the content of the current directory, simply type: $ **ls**

```
[vagrant@localhost ~]$ ls
bottle  class  cup  gym  home  line  linux  serge  work
```

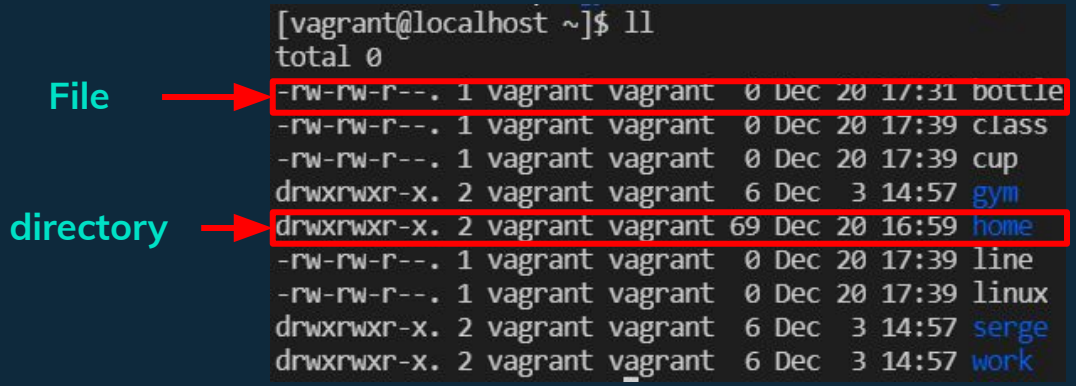
- You can also use the **# ll** or **# ls -l** commands to list the content of a directory/folder (It displays some other useful features of the files)

```
[vagrant@localhost ~]$ ll
total 0
-rw-rw-r--. 1 vagrant vagrant  0 Dec 20 17:31 bottle
-rw-rw-r--. 1 vagrant vagrant  0 Dec 20 17:39 class
-rw-rw-r--. 1 vagrant vagrant  0 Dec 20 17:39 cup
drwxrwxr-x. 2 vagrant vagrant  6 Dec  3 14:57 gym
drwxrwxr-x. 2 vagrant vagrant 69 Dec 20 16:59 home
-rw-rw-r--. 1 vagrant vagrant  0 Dec 20 17:39 line
-rw-rw-r--. 1 vagrant vagrant  0 Dec 20 17:39 linux
drwxrwxr-x. 2 vagrant vagrant  6 Dec  3 14:57 serge
drwxrwxr-x. 2 vagrant vagrant  6 Dec  3 14:57 work
```

How can we differentiate between files and directories

Files and directories

- To easily make the difference between a files and directories we can use the command **ll** or **ls -l**
- The **first character** of the line on which the file or the directory appears defines if it is a file or a directory
- A **file** starts with the character **-** while a **directory** starts with the character **d**



Change and print current directory

- To check the current working directory, we run the command: $ **pwd**

```
[vagrant@localhost ~]$ pwd
/home/vagrant
```

- To change the current directory, we use the command:
  - **cd directoryName/**

Example: Let's change the current directory to the directory **home** we previously created and create some directories and files in there:

$ cd home/ then **$ mkdir work home serge** and **$ touch cup line gym** **(Use ll to verify)**

- Check the current working directory, run $ **pwd**

Copy a file/directory

- To copy a file, we use the command: **cp fileName copyName**

Example: Let's make a copy of our previously created file **bottle**. the name of the copy is **review**: $ **cp gym review**

```
[vagrant@localhost home]$ cp gym review
[vagrant@localhost home]$ ls
cup  gym  home  line  review  serge  work
```

- To copy a directory or a folder, we add the **-r** option to our **cp** command. That is: **cp -r dirName CopyName**
- To make a copy of the **work** directory for example, we run:
- **$ cp -r work workload**

```
[vagrant@localhost home]$ cp -r work workload
[vagrant@localhost home]$ ls
cup  gym  home  line  review  serge  work  workload
```

Rename a file/directory

- To rename a file or a directory, we use the move command:
  - **mv fileName newName**
- **Example**: To change the name of the file **line** to **house** we type:
  - **mv line house**

```
[vagrant@localhost home]$ mv line house
[vagrant@localhost home]$ ls
cup  gym  home  house  review  serge  work  workload
```

- We can also use this command to **change a file or a directory location**

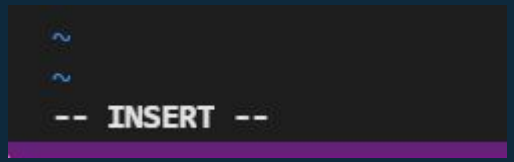
Example: To move the file **house into the directory home we created,** we run:

- **$ mv house home/**
- **$ cd /home then ls**
- **$ cd ..** to go back to the parent folder then **$ pwd** to verify

```
[vagrant@localhost home]$ mv house home/
[vagrant@localhost home]$ ls
cup  gym  home  review  serge  work  workload
[vagrant@localhost home]$ cd home/
[vagrant@localhost home]$ ls
house
```

Edit a file content

- To put or modify the content of a file, we use the command: **vi fileName**
  Example: **vi gym** helps us to open our file named **gym**
- There are two modes in **vi**:
  - The **command mode** (gym OL, OC check on the last line of the vi page)
  - The **Insert mode** (--INSERT--)
- To **change the mode**, press the **"i"** key on the keyboard. Verify the last line of the vi page



Edit a file content

- Now, you can type a text in the file:

```
I am learning how to navigate and utilize a linux server
Very soon I will be typing these same commands in a nice office and making 6
 figures income!!!

I can't wait !!!!!!!
```

- To **save the text** you have edited, press the key "**ESC**" (Escape)
- To **save and quit**, press **ESC** followed by **:wq** (this appears on the last line of the page)
- If you type the **$ ll** command in your terminal, you will realise that the size of the file has changed!

Display the content of a file

- To display the content of a file we use the command **# cat fileName**
- **Example**: Let's display the content of the previously edited file **gym:**
  - **# cat gym**

```
[vagrant@localhost home]$ cat gym
I am learning how to navigate and utilize a linux server
Very soon I will be typing these same commands in a nice office and making 6
 figures income!!!

I can't wait !!!!!!!
```

Remove or delete a file

- To remove or delete a file, we use the command: **$ rm fileName**
- Using **$ rm** on the file, a confirmation question will be asked. Type “**y**” for **YES** or “**n**” for **NO**
- To avoid this question, we use: **$ rm -f** to force the deletion of the file.
- **Example**: Let’s delete the file **cup**
  - **$ rm cup** or **$ rm -f cup**
  - **$ ls** to verify

Remove or delete a directory/folder

- To remove or delete a directory/folder, we still use the command **$ rm** but with the option **-r**
- **Example**: to remove the directory **work**, we do:
  - **$ rm -r work**
- To avoid the confirmation question, instead type:
  - **$ rm -rf work**

Recall

Do you remember the function of these commands?
$ mkdir $ touch $ cat $ rm $ cp $ vi

- **$ mkdir** is used to create a folder
- **$ touch** is used to create a file
- **$ cat** is used to display the content of a file
- **$ rm** is used to remove or delete a file
- **$ cp** is used to copy a file
- **$ vi** is use to edit the content of a file

3

Other useful system commands

Man, history, echo, grep

The man command

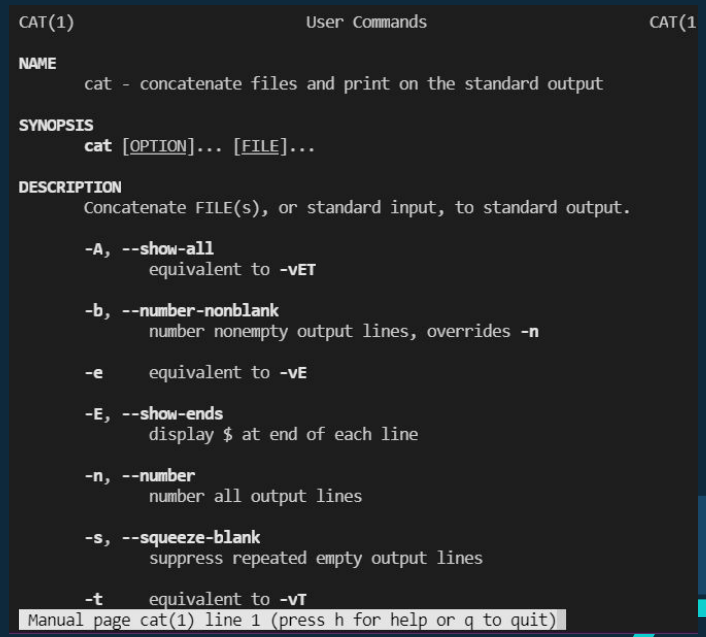
The man command

- As a beginner in Linux, you may find some difficulties using commands and knowing the syntax of some useful commands
- **The man command** in Linux is **used to display the user manual of any command that we can run on the terminal**.
- This command help you to get a better understanding of any other command in Linux

The man command

- The man command provides a detailed view of a specific command.
- Its syntax is: $ **man command**

  Example: to display the manual for the cat command, we run: **$ man cat**
- **Note :** hit “q” to exit the prompt and use the up & down arrow to navigate



The echo command

The echo command

- This command is used to display a message on the console or on the screen
- To use it, we run: **$ echo message**

Example:

$ echo Welcome to the class please get on your keyboard and type

```
[vagrant@localhost home]$ echo Welcome to the class please get on your keybo
ard and type
Welcome to the class please get on your keyboard and type
```

- Play around with this command and print some message on the console.

The history command

The history command

- The **# history** command displays all the commands you have typed in the session.
- Each command is displayed on a single line for better visibility

```
[vagrant@localhost home]$ history
    1  id
    2  uptime
    3  cat /etc/*release
    4  cal
    5  date
    6  uptime
    7  id
    8  exit
    9  date
   10  date --date="yesterday"
   11  date --date="10 days ago"
   12  uptime
   13  cal
   14  cal 12 2021
   15  cal 1
   16  cleat
   17  clear
   18  whoami
   19  id
   20  pwd
   21  cd /tmp
```

“ Don’t try to memorize all of this!

Just try to understand how it works, practise it and have fun playing around with commands ”

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!