BORN2BE_ROOT

Virtual Machine

> 💡 Virtual machines are made possible through **virtualization** technology.
> Virtualization uses software to simulate **virtual hardware** (know as a hypervisor) that allows multiple VMs to run on a single machine. The physical machine is known as the host while the VMs running on it are called guests.

▼ AppArmor check status → `aa-status` / log messages → `sudo journalctl -fx`

> 💡 AppArmor is an effective and easy-to-use Linux application security system. AppArmor proactively protects the operating system and applications from external or internal threats, even zero-day attacks, by enforcing good behavior and preventing both known and unknown application flaws from being exploited.
> It supplements the traditional Unix discretionary access control (DAC) model by providing mandatory access control (MAC). It has been included in the mainline Linux kernel since version 2.6.36 and its development has been supported by Canonical since 2009.

▼ Turn off the VM

- Turn off the virtual machine `sudo poweroff` || `init 0` || `shutdown -h now`
- Shutdown and then reboot the machine `sudo shutdown -r now`
- Reboot the system `sudo reboot`

▼ Set/change system time zone → `sudo timedatectl set-timezone [time_zone]`

- To list available time zones → `timedatectl list-timezones`

▼ LVM - **Logical Volume Manager** → `lsblk`

In Linux, *Logical Volume Manager* (*LVM*) is a device mapper framework that provides *logical volume management* for the Linux kernel.

`lsblk` lists information about all available or the specified block devices.

Apt-get vs Aptitude

> 💡 Apt-get being a lower level package manager is restricted only to command line, while Aptitude being a **higher-level** tool has a default text-only interactive interface along with option of command-line operation by entering required commands.

Sudo

> 💡 The `sudo` command grants a one-time or limited-time access to root functionality. Typically, the `sudo` command is used to quickly run an administrative command, then return to the user account's regular permissions.

▼ Add a user with sudo access → (log as root) `adduser [username] sudo`

- Alternatively → `usermod -aG sudo [username]`

- Display / access sudo config file → `visudo`

▼ Sudo install and config

- Log as root and in its environment → `su -`
- Install sudo

```
apt-get update -y
apt-get upgrade -y
apt install sudo
```

- Verify sudo installation → `dpkg -l | grep sudo`
- Configure sudo → `sudo nano /var/log/sudo/sudo.log`

> 💡 `su` is an older but more fully-featured command. It can duplicate the functionality of `sudo` by use of the `-c` option to pass a single command to the shell.
>
> By default, the `su` command maintains the same shell environment.

Hostname

Display hostname → `hostnamectl | grep hostname`

Change hostname → `hostnamectl set-hostname [new_name]` || `sudo nano /etc/hostname`

Check hostname → `cat /etc/hosts`

User `./home/[user]`

- Add user → `sudo adduser [username]`

▼ Give super user privilege to an existing user

1. Open sudoers file `sudo visudo`
2. Add this line in user privilege : `[username] ALL=(ALL) ALL`

▼ Add user to a group → `sudo usermod -aG [group_name] [user_name]` or `sudo adduser [username] [group]`

> 💡 -G stands for the secondary group while -g stands for primary group

- To check if the user has been added to the group → `getent group [groupname]`

▼ List of all users → `compgen -u`

- `getent passwd | cut -d: -f1`
- `cut -d: -f1 /etc/passwd`

▼ Display user info

- Display the current user → `sudo sudo whoami`
- Displays users who are logged on the system, including the terminals they are connecting from → `who -u`
- Displays user and group ID → `id [username]`

▼ Account details → `sudo getent passwd [username]`

> 💡 **getent** is a command line utility for fetching entries from **Name Service Switch** (**NSS**) libraries from a specific system database.

▼ Information about a user → `sudo finger [username]`

> 💡 **finger** doesn’t come per-installed on many Linux systems.

    - Install finger → `sudo apt install finger`
  - ▼ User's infos from the system accounts file → `grep -i [username] /etc/passwd`

> 💡 grep command is a powerful pattern searching tool available on most if not all Linus systems. You can use it to find information about a specific user from
> the system accounts file: ***/etc/passwd*** as shown below.

  - ▼ Known users in the system → `sudo lslogins -u`

> 💡 The `u` flag only displays user accounts.

- ▼ List of last logged users → `sudo last`
  - `last -a` to show hostname on the last column
- ▼ Details of a recent login of user → `sudo lastlog -u [username]`
  - For all user → `lastlog`
- ▼ Switch user / Modify user infos
  - Switch to root account → `su root`
  - Switch (as root) to any user without knowing the password → `su - [username]`
    - Switch to new user → `sudo su - [username]`
  - Run a command as a different user `su -c [command] [other_user]`
- ▼ Remove user
  - Delete user → `sudo deluser [username]` or `grep 'username' /etc/passwd`
  - Delete user + home directory + mail spool → `sudo deluser --remove-home [username]`

Group `./etc/group/`

- **Create** a new group → `sudo groupadd [groupname]`
- **Check** if a user is in sudo group `getent group sudo`
- ▼ **Display**
  - Number of groups in Linux server → `wc -l /etc/group`
  - The groups a user belongs to → `groups [username]`
- Change user's primary group → `usermod -g [group] [username]`
- ▼ Remove group → `sudo groupdel [groupname]`
  - Check if the group has been removed → `getent group | grep [groupname]`
  - Force removal of the primary group → `sudo groupdel -f [groupname]`

Password

- Change password → `passwd [username]`
- ▼ Password strength configuration
  - Install the *libpam-pwquality* package → `sudo apt install libpam-pwquality`

  - Configure → `nano /etc/pam.d/common-password`
- Display password list parameters → `chage -l`
- Modify password expiration → `sudo nano /etc/login.defs`
  - Change minimum password age → `chage -m [number of days] [username]`

  → Reboot to apply changes

UFW `/etc/default/ufw`

> 💡 UFW (Uncomplicated Firewall) is a user-friendly front-end for managing iptables firewall rules. Its main goal is to make managing iptables easier or, as the name says, uncomplicated.
>
> To list available apps linked to UFW → `sudo ufw utf --help`

▼ **Install** → `sudo apt-get install ufw`

> 🚧 By default, the UFW firewall denies every incoming connections and only allow all outbound connections to server. This means, no one can access your server, unless you specifically open the port, while all the running services or applications on your server can be able to access the outside network.

▼ **Access** / setup UFW firewall default policy → `/etc/default/ufw`

→ `sudo ufw default deny incoming`

→ `sudo ufw default allow outgoing`

- **Enable** / Disable firewall to start on boot → `ufw enable` / `ufw disable`
- **Reset** UFW → `sudo ufw reset`
- **Add** a rule to open a port → `sudo ufw allow [port]`

▼ **Check**

- Verify which rules were added → `sudo ufw show added`
- UFW status → `sudo ufw status verbose`
- Display port `sudo ufw status`

▼ **Delete**

- Get the rule number → `sudo uwf status numbered`
  - Delete the rule → `sudo delete [rule number]`
- Delete multiple rules in once from same port : `sudo ufw delete allow [port]`
  - With specific protocol : `sudo ufw delete allow [port]/[protocol]`

> ✏️ Other rules may be added in the same way by simply specifying a name of the program

SSH `/etc/ssh/`

> 💡 SSH stands for Secure shell, it is a cryptographic network protocol for operating network services securely over an unsecured network. Typical applications include remote command-line, login, and remote command execution, but any network service can be secured with SSH.

- Install → `sudo apt install openssh-server`
- ▼ Configure → `sudo nano /etc/ssh/sshd_config`

  > 💡 to set up SSH using a specific port, changes port number on line 15

  - Disable SSH login as root → `PermitRootLogin` : `no`
- Check SSH status → `systemclt status ssh`
- ▼ Allow SSH connections to server
  - Allow SSH by service name → `sudo ufw allow ssh`
  - Allow SSH by Port number → `sudo ufw [port]`
    - With specific protocol : `sudo ufw [port]/[protocol]`
  - Allow an IP address → `sudo ufw allow from [IP address]`
    - On a specific port → `sudo ufw allow from [IP address] to any port [port]`
  - Open port range, from port A to port B → `sudo ufw allow [port A]:[port B]/tcp`
  - To block all SSH connections → `sudo ufw deny ssh/tcp`

    → if using custom port : `sudo ufw deny [port]/tcp`
- ▼ Connect remotely with SSH
  - Port Forwarding (NAT mode) → `ssh [username]@[server] -p [port]`
  - Swap NAT to Bridged Adapter → `ssh [username]@[ip_add] -p [port]`
    - Get the IP addrs → `hostname -I`

- Run / execute command SSH → `ssh [user]@[server] -p [port] [command]`

CRON `/var/spool/cron/crontabs/`

> 💡 Cron is a tool in Unix that allows tasks to be automatically run in the background at regular intervals. Crontab is a file which contains the schedule of cron entries to be run and at specified times.

- Edit/create crontab file → `crontab -e`

```
*min  *h   *day   *m    *dw   /path of script to execute | wall
 -     -     -     -     -
 |     |     |     |     |
 |     |     |     |     +----- day of week (0 - 6) (Sunday=0)
 |     |     |     +------- month (1 - 12)
 |     |     +--------- day of month (1 - 31)
 |     +----------- hour (0 - 23)
 +------------- min (0 - 59)
```

> 💡 `wall` stands for write all. It is a command-line utility that displays a message on the terminals of all logged-in users.

- Display users crontabs → `cat /var/spool/cron/crontabs/[user]`
- Check active crontab from user → (log as user to check) `crontab -l`

- ▼ Manage cron services → `sudo systemctl [...] cron.service`
    - Enable crontab → `sudo systemctl enable cron.service`
    - Start crontab → `sudo systemctl start cron.service`
    - Stop crontab → `sudo systemctl stop cron.service`
    - Restart crontab → `sudo systemctl restart cron.service`
- Check cron status → `sudo systemctl status cron.service`
- Delete user crontab → `sudo crontab -u [user]`

SCRIPT

▼ This is my personal script, use it as a guide for your own script, but do not `cmd C` `cmd V` :)

```
printf "\n\n"
printf "   +--------- tmercier@student.codam.nl ---------+\n"
printf "\n"
printf "   +  Architecture :\n";
echo -ne "         |  " ;uname -s -n -r
echo -ne "         |  "; uname -v
echo -ne "         |  "; uname -m -o
echo -ne "   +  CPU physical : "; nproc --all   #print total number of processors available on the system
echo -ne "   +  vCPU : "; cat /proc/cpuinfo | grep processor | wc -l
echo -ne "   +  Memory Usage : "; free -m | awk 'NR == 2 {printf "%sMB/%sMB (%.2f%%)\n",($2-$4),$2,($2-$4)/$2*100}'
echo -ne "   +  Disk Usage : "; df -h | awk 'NR == 4 {printf "%s/%s (%s)\n",$3,$2,$5}'
echo -ne "   +  CPU Load : "; mpstat | awk 'NR==4 {printf "user %s%% | system %s%%\n",$4, $6}'
echo -ne "   +  Last Boot : "; uptime -s
echo -ne "   +  LVM Use : "; if cat /etc/fstab | grep -q "/dev/mapper/"; then echo "yes"; else echo "no"; fi
echo -ne "   +  Connexions TCP : "; netstat -t | grep ESTABLISHED | wc -l       # -t flag will ony print TCP connections
echo -ne "   +  User Log : "; w -h | wc -l      # -h flag will get the infos without the header
echo -ne "   +  Network : IP "; hostname -I | tr '\n' '(' && /sbin/ifconfig | grep ether | awk '{print $2")"}'
echo -ne "   +  Sudo : "; cat /var/log/sudo/sudo.log | wc -l | tr '\n' ' ' && printf "cmd\n"
printf "   +------------ Press ENTER to exit -----------+\n"
printf "\n" && exit
```