

## Catalog

Project 1 : Running Containerized containers with Docker : .....	5
Docker Installation : .....	6
Docker client : .....	7
Pulling images and running containers : .....	8
Troubleshoot : .....	8
Running a web server in a docker container : .....	11
Listing images and containers : .....	13
Removing images and containers : .....	13
Getting shell access to a container : .....	15
Getting information about the running container : .....	16
Committing container changes to a new image : .....	16
Troubleshoot : .....	17
Tagging and pushing custom images to docker hub : .....	18
Image structure and layers : .....	19
Creating custom images using dockerfile : .....	21
Use volumes and bind mounts : .....	23
1. **Managed by Docker:** .....	23
2. **Named and Shared:** .....	23
3. **Persistence:** .....	23
4. **Performance:** .....	23
5. **Volume Drivers:** .....	23
6. **Syntax:** .....	23
1. **Host Filesystem:** .....	23
2. **No Management by Docker:** .....	23
3. **Persistence:** .....	23
4. **Performance:** .....	24
5. **Versatility:** .....	24
6. **Syntax:** .....	24
Project 2 : Securing and hardening a Linux system .....	25
Securing the openssh : .....	25
Securing the bootloader : .....	28
Enforcing password policy : .....	29
Locking or disabling user accounts : .....	30
Giving limited root privileges : .....	31
Settings user limits (Running a DOS attack without root access) Rabbit virus .....	33
Intro to cracking passwords : .....	34
Cracking Linux passwords using john the ripper .....	34
Checking files integrity with AIDE : .....	37
Scanning for rootkits (rkhunter , chkrootkit) : .....	39
Scanning for viruses with CLAMAV : .....	40
Full disk encryption using dm-crypt and LUKS : .....	42
Unlock locks encrypted drives with a key file : .....	44
Symmetric encryption using GnuPG : .....	47
Using an encrypted text editor (Encryptpad) : .....	47
Steganography : .....	48
Hide secret messages through steganography with steghide : .....	49
Scanning networks with Nmap : .....	51
Intro to netfilter and iptables : .....	53
Chain traversal : .....	53
Iptables basic usage : .....	54
Settings the default policy : .....	57
Deleting the firewall : .....	57
Filter by IP address : .....	58
Filter by ports : .....	59
Intro to stateful firewall : .....	59

Filter by MAC address : MAC filtering .....	61
Filter by date and time : .....	61
The ACCEPT and DROP target : .....	62
The LOG target : .....	63
Project 3 : Setting Up a Web and DNS Server .....	65
Getting a domain name : .....	65
Diving into the DNS protocol and installing a DNS server (Bind9) : .....	67
Setting up the authoritative bind9 dns server : .....	69
Installing a web server on the authoritative server : apache2 .....	71
Setting up virtual hosting : .....	72
Securing apache2 with openssl and digital certificates : .....	74
Access control by source ip address : (white list) .....	75
All the directives recommendations : .....	77
The <Files> directive : .....	79
The .htaccess file : .....	80
HTTP digest authentication : .....	81
The options directive and indexing : .....	83
Change the server signature : .....	85
HTTP compression : .....	86
Monitoring the server : .....	87
Install PHP : .....	89
Installing and securing the mysqlserver : .....	90
Installing phpmyadmin : .....	91
Secure phpmyadmin : .....	92
Install wordpress : .....	94
Securing wordpress : .....	95
Project 4 : Running a virtual mail server .....	97
Mail and DNS : MX records and reverse DNS .....	98
Prerequisites : .....	100
Installing software packages : .....	100
5. Configuring SMTP AUTH (SASLAUTHD and MySql) .....	103
7. Configuring Dovecot (POP3/IMAP) .....	104
Configure another port SMTP 465 in postfix in your mail server for mail submission : .....	107
My favorite : Troubleshooting .....	108
Virus scanning using Amavis and ClamAv : .....	109
Testing Amavis and clamav : .....	111
Fighting against spam : postfix access restriction .....	112
Smtpd helo restriction : .....	114
Fighting against spam using public RBL's : .....	114
Intro to spam filtering systems : .....	116
Configuring and testing rspamd : .....	118
Automating Linux Admin tasks : .....	124
Intro to Ansible : .....	124
Ansible Inventory file : .....	125
Ansible adhoc commands : the shell module .....	128
Ansible Ad-Hoc commands : the script module .....	130
Ansible Ad-hoc commands : The APT module .....	131
Ansible Ad-hoc commands : the service module .....	132
Ansible Ad-hoc commands : the user module .....	133
Scheduling tasks using Anacron : .....	137
Bonus section : Bash shell scripting .....	139
Special Variables and positional arguments : .....	142
If , elif , and else statement : .....	144
Multiple conditions and nested if statements : .....	146
Command substitution : .....	148
Comparing strings in If statement : .....	149
For loops : .....	151

While loop : .....	153
Case statement .....	154
Functions : .....	156
Variable scope in functions : .....	157
Menus in bash , the SELECT statement : .....	159
Intro to bash arrays : .....	160
Associative arrays : Dictionary .....	162
Readarray : iterate over arrays .....	163
Project : Automate user creation : .....	164



### **Connecting to a virtual private server (VPS) :**

After creating the droplet or the instance , we go to our terminal and access it through ssh  
`ssh -p 22 root@ip`  
you will be asked to provide the password to access the root user

#### **Centos :**

Vps1 centos1

#### **Full upgrade of the system :**

`apt update && apt full-upgrade -y`

#### **Securing SSH with key authentication to VPS :**

*For windows : putty key generator*

*Private key : for linux : ssh-keygen -t rsa -b 2048 -C 'key generated by taib '*

*Location : ls .ssh/*

*Then add the public ssh key to the server or droplet :*

*You can do from the settings in your droplet*

*or by using : ssh-copy-id root@ip\_of\_theserver (the public key will be appended to the key file on the server ) , you can double check by checking on the server file cat .ssh/authorized\_keys*

*Now you can log directly to the server using public key authentication without password*

`ssh root@ip_of_server`

*We can disable the passwd authentication and just rely on the RSA public key , its more secured :*

*In the server , vim /etc/ssh/sshd\_config*

*Systemctl restart ssh*

#### *Troubleshoot text editing in ubuntu vm :*

##### **1. Install `gpm`:**

```
bash                                     ⌂ Copy code
sudo apt-get install gpm
```

##### **2. Start `gpm`:**

```
bash                                     ⌂ Copy code
sudo service gpm start
```

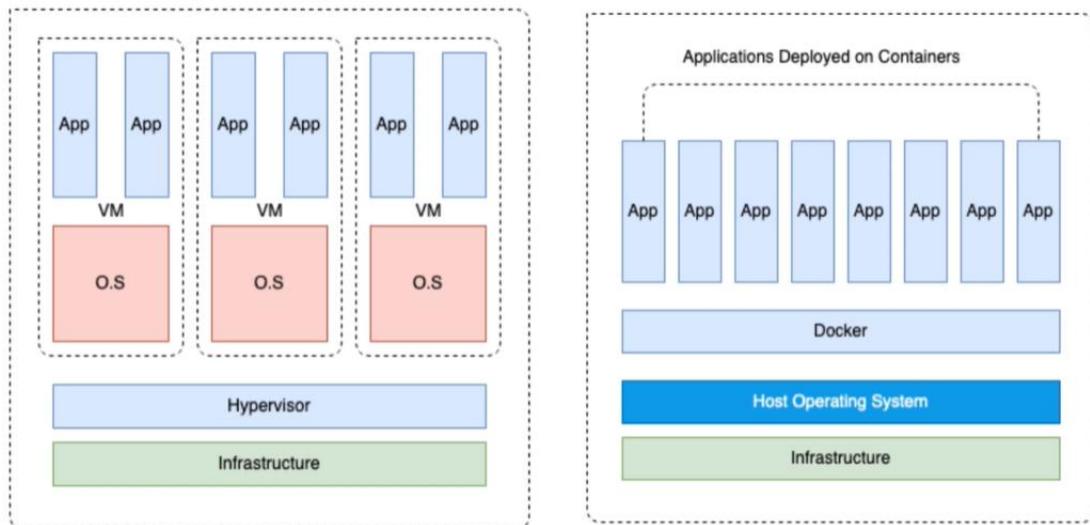
Alternatively, you can enable `gpm` to start on boot:

```
bash                                     ⌂ Copy code
sudo systemctl enable gpm
```

## **Project 1 : Running Containerized containers with Docker :**

## What is Docker?

- Docker is a platform for developers and sysadmins used to develop, deploy and run containerized applications.
- A container is a running process that encapsulates everything an application needs to run (and only those things).
- Docker makes it really easy to install and run software without worrying about setup or dependencies.
- An image is a standalone, isolated, lightweight and executable package that contains all the required components (code, runtime, system tools, libraries, settings, etc) necessary to run an application.
- The container is the runtime instance of the image. From the same image you can launch more containers.
- Images for many applications can be found on <https://hub.docker.com>



Vms are inefficient , you don't need to run a full OS for one or few apps , either way the os is wasting resources

Containers share the os and kernel , they don't need a full separate OS

For deploying different apps , compatibility program arises , some apps require python 2 and some 3 , it means more OS , more VM problems

## Docker Installation :

<https://docs.docker.com/engine/install/ubuntu/>  
<https://docs.docker.com/engine/install/centos/>

**Error : when trying to install the docker dependencies , a podman package error appears**

```
Can't be installed
- package podman-catatonit-3:4.6.0-1.module_el8+567+36ff2357.x86_64 from appstream requires podman = 3:4.6.0-1.module_el8+567
36ff2357, but none of the providers can be installed
- package podman-3:4.6.0-1.module_el8+567+36ff2357.x86_64 from appstream requires runc >= 1.0.0-57, but none of the providers
can be installed
- package podman-catatonit-3:4.6.1-1.module_el8+643+8db347f4.x86_64 from appstream requires podman = 3:4.6.1-1.module_el8+643
8db347f4, but none of the providers can be installed
- package podman-3:4.6.1-1.module_el8+643+8db347f4.x86_64 from appstream requires runc >= 1.0.0-57, but none of the providers
can be installed
- package podman-catatonit-3:4.6.1-2.module_el8+653+feef7bfe.x86_64 from appstream requires podman = 3:4.6.1-2.module_el8+653
feef7bfe, but none of the providers can be installed
```

## Solution :

The screenshot shows the IBM Edge Application Manager 4.2 documentation interface. The left sidebar includes navigation for 'IBM Edge Application Manager' (version 4.2), 'Change version', and 'Troubleshooting tips' (FAQ, Resources, Glossary, Notices). The main content area is titled 'How to uninstall Podman on RHEL?'. It contains instructions for removing Podman and installing Docker, along with several command-line examples in code blocks:

- Uninstall Packages:  
`yum remove buildah skopeo podman containers-common atomic-registries docker`
- Remove any left-over artifacts & files:  
`rm -rf /etc/containers/\* /var/lib/containers/\* /etc/docker /etc/subuid\* /etc/subgid\*`
- Delete any associated container storage:  
`cd ~ && rm -rf .local/share/containers/`

The right sidebar lists 'On this page' topics such as 'Are the currently released versions of the Horizon packages installed?' and 'Is the Horizon agent up and actively running?'.

After creating an ec2 instance in AWS , we used instant connect

To remove any existing installation of Docker : `sudo apt remove docker docker.io containerd runc`

We follow the official documentation

We check if its active

`sudo systemctl status docker`

*Check users and groups :*

`sudo cut -d: -f1 /etc/passwd  
less /etc/group`

*Add user to sudo group :*

`sudo usermod -aG sudo username`

*To check if docker client can talk to docker server :*

`Docker --version`

```
# uninstalling any old versions
sudo apt remove docker docker.io containerd runc

# adding Docker's official GPG signing key:
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

# adding the official docker repository
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"

# refreshing the apt cache
sudo apt update

# selecting the docker repository as the default one
apt-cache policy docker-ce

# installing docker
sudo apt install docker-ce docker-ce-cli containerd.io

# checking its status
sudo systemctl status docker

# adding the current user to the docker group to be able to run the docker command
sudo usermod -aG docker ${USER}

docker --version
```

## Docker client :

Is the docker image CLI that interacts with the docker daemon (sever)

`Docker --version`

`Docker version`

Docker info

Docker help

Docker container ls (after docker container help)

Docker container run hello-world ( it tries to find the mage of hello\_world locally , if not it uses the hub )

```
To generate this message, Docker took the following steps:
```

1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.  
(amd64)
3. The Docker daemon created a new container from that image which runs the executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it to your terminal.

```
To try something more ambitious, you can run an Ubuntu container with:
```

```
$ docker run -it ubuntu bash
```

## Pulling images and running containers :

You can use docker hub to pull images

Docker uses in results : repository/image\_name:tag

We can use : *docker search mysql*

Supported tags : supported version

Pull a specific tag : *docker image pull mysql:8.3.0*

Pull the latest version : *docker image pull mysql*

List docker images : *docker images*      *docker image ls*

The imageid is the sha256 hash of the image

Remove an image by tag : *docker image rm image\_name:tag*

To create and run a container from an image : *docker container run -P httpd*

To create : *docker container create -p 80:80 httpd*

To check status of containers : *docker conatiner ls -a*

To start : *docker container start "CONTAINERID"*

To run a container and get a shell with it : *docker container run -it centos*

To check if its working : we connect to the instance from another terminal to check its IP address

*ssh -l root 172-31-92-175 -p 2229*

Then we use that ip address in the browser

Check to which port httpd is connected : *docker container ls*

## Troubleshoot :

Systemctl on the host wont work , because the apache server of the container is isolated

Using the host IP didn't work so lets try the container IP address , we checked the logs and checked the container IP address

*docker logs <container\_name\_or\_id>*

*Port mapping : docker run -p 80:80 -d your\_httpd\_image*

```
Container IP : docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}'  
<container_name_or_id>
```

I found the solution : so basically you need to make sure that in the inbound rules the port is 80 allowed , but even though we added the port 80 , apparently the generated inbound port is the one needs to be accessed , so I added another rule to the inbound rules to allow all tcp connection from any host

And I run the httpd container from the instance and access it again from the windows 10 cmd to check the ips and ports generated

**cd %HOMEPATH%\Downloads**

```
ssh -i "private/public.pem" ubuntu@ec2-3-84-10-8.compute-1.amazonaws.com
```

## 2. Confirm Container IP:

- Verify the IP address of your running container. Make sure you are using the correct IP address when trying to access the Apache server.

```
bash  
Copy code  
  
docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}'  
[{"NetworkSettings": {"Networks": [{"Name": "bridge", "IPAMConfig": null, "Links": null, "Aliases": null, "NetworkID": "5a2a2a2a2a2a2a2a", "EndpointID": "5a2a2a2a2a2a2a2a", "MacAddress": "02:40:96:00:00:00", "IPv4Address": "172.31.92.175/16", "IPv6Address": null}, {"Name": "host", "IPAMConfig": null, "Links": null, "Aliases": null, "NetworkID": "5a2a2a2a2a2a2a2a", "EndpointID": "5a2a2a2a2a2a2a2a", "MacAddress": "02:40:96:00:00:00", "IPv4Address": "127.0.0.1/8", "IPv6Address": null}, {"Name": "eth0", "IPAMConfig": null, "Links": null, "Aliases": null, "NetworkID": "5a2a2a2a2a2a2a2a", "EndpointID": "5a2a2a2a2a2a2a2a", "MacAddress": "02:40:96:00:00:00", "IPv4Address": "3.84.10.8/16", "IPv6Address": null}], "Bridges": null, "ContainerID": "5a2a2a2a2a2a2a2a"}, {"NetworkSettings": {"Networks": [{"Name": "bridge", "IPAMConfig": null, "Links": null, "Aliases": null, "NetworkID": "5a2a2a2a2a2a2a2a", "EndpointID": "5a2a2a2a2a2a2a2a", "MacAddress": "02:40:96:00:00:00", "IPv4Address": "172.31.92.175/16", "IPv6Address": null}, {"Name": "host", "IPAMConfig": null, "Links": null, "Aliases": null, "NetworkID": "5a2a2a2a2a2a2a2a", "EndpointID": "5a2a2a2a2a2a2a2a", "MacAddress": "02:40:96:00:00:00", "IPv4Address": "127.0.0.1/8", "IPv6Address": null}, {"Name": "eth0", "IPAMConfig": null, "Links": null, "Aliases": null, "NetworkID": "5a2a2a2a2a2a2a2a", "EndpointID": "5a2a2a2a2a2a2a2a", "MacAddress": "02:40:96:00:00:00", "IPv4Address": "3.84.10.8/16", "IPv6Address": null}], "Bridges": null, "ContainerID": "5a2a2a2a2a2a2a2a"}]}
```

## 3. Host Machine IP:

- Confirm the IP address of your host machine. Use this IP address to access the Apache server if you are trying to connect from a browser on the host.
- If the host is running on AWS, use the public IP address or DNS name.

IT FINALLY FUCKING WORKED , so the thing is I needed to connect to either the public IP address , or the Public IPv4 DNS

Public IPv4 address	Private IPv4 addresses
3.84.10.8   <a href="#">open address</a>	172.31.92.175
Instance state	Public IPv4 DNS
Running	ec2-3-84-10-8.compute-1.amazonaws.com   <a href="#">open address</a>


**It works!**

Just disable the firewalld or add inbound rules and make the network adapter bridged

Extra info about the vpc :

**1. Virtual Private Cloud (VPC):**

- A VPC is a logically isolated section of the AWS Cloud where you can launch AWS resources. It acts as your private network in the cloud.
- Within a VPC, you can define subnets, route tables, and configure security groups to control traffic.

**2. Private IP Addresses:**

- Each instance in a VPC is assigned a private IP address. Private IP addresses are used for communication within the VPC.

**3. Public IP Addresses:**

- Instances launched in a public subnet of a VPC can be assigned a public IP address. This allows them to communicate directly over the internet.

**4. Internet Gateway:**

- An internet gateway allows communication between instances in your VPC and the internet. Instances with public IP addresses can use the internet gateway for outbound and inbound traffic.

**5. Security Groups:**

- Security groups act as virtual firewalls for your instances. You can specify inbound and outbound rules to control traffic.



---

#alias to create alias for other commands

To make alias persistent : `echo "alias vps='ssh -l root 172-31-92-175 -p 2229'" >> ~/.bashrc`

*Conclusion :*

```
## The Docker CLI ##  
## Pulling Images and Running Containers ##  
  
# testing the entire installing  
docker --version  
docker version  
docker info  
docker container run hello-world  
  
# getting help  
docker help  
docker MANAGEMENT_COMMAND help => Ex: docker container help  
  
# searching for an image on Docker Hub  
## docker search IMAGE_NAME  
docker search debian  
docker search mongo  
  
# pulling an image from docker hub  
## docker image pull IMAGE_NAME:TAG
```

```

docker image pull redis:5.0.10
docker image pull ubuntu:latest
docker image pull mysql # => by default the name of the TAG is latest

# listing local images
docker images
docker image ls

# running a container
## docker container run OPTIONS IMAGE_NAME
docker container run -P httpd

# equivalent to:
docker container create -P httpd
docker container ls -a # listing all containers
docker container start CONTAINER_ID

# getting a shell into a container
docker container run -it centos # to detach from the container without stopping it press: Ctrl + P + Q

```

### Quiz :

You've just installed Docker on Linux and you want to run the docker command as a non-privileged user. What should you do?

Add the non-privileged user to the docker group

## Running a web server in a docker container :

Public address : *curl ifconfig.me*



**ChatGPT**

To configure the firewall settings on your CentOS VM to allow inbound traffic on a specific port (e.g., port 8080), you can use the `firewall-cmd` command-line tool. Here's how you can open the port using `firewall-cmd`:

1. Open a terminal on your CentOS VM.
2. Run the following command to allow incoming traffic on port 8080:

bash

Copy code

```
sudo firewall-cmd --zone=public --add-port=8080/tcp --permanent
```

This command adds a rule to the firewall configuration to allow TCP traffic on port 8080 in the "public" zone. The `--permanent` flag makes the rule persistent across firewall reloads or system reboots.

3. After adding the rule, apply the changes to the firewall:

bash

Copy code

```
sudo firewall-cmd --reload
```

1. Open VirtualBox and ensure that your CentOS VM is powered off.
2. Select your CentOS VM from the list of virtual machines.
3. Click on "Settings" to open the VM settings.
4. In the settings window, navigate to the "Network" tab.
5. Here, you'll see the network adapters configured for your VM. Typically, you'll be using "Adapter 1" for general network connectivity.
6. Click on "Advanced" to expand the advanced network adapter settings.
7. Under the "Port Forwarding" section, click on the "+" button to add a new port forwarding rule.
8. Specify the following parameters for the new rule:
  - Name: Any descriptive name for the rule (e.g., "HTTP").
  - Protocol: TCP or UDP, depending on your requirements (usually TCP for HTTP).
  - Host IP: Leave this blank or specify the IP address of your host machine if you want to restrict forwarding to a specific interface.
  - Host Port: The port on your host machine that you want to forward (e.g., 8080).
  - Guest IP: Leave this blank or specify the IP address of your CentOS VM if you want to restrict forwarding to a specific VM interface.
  - Guest Port: The port on your CentOS VM that you want to forward (e.g., 8080).
9. Click "OK" to save the rule and close the dialog.
10. Click "OK" again to save the VM settings.

One note , when removing the outbound rules , we wont be able to connect or do anything from the internet

```
sudo docker container run -d -p 8080:80 --name mysite nginx
```

*-d : detach , to run as a daemon  
 -p 8080:80 : it will map port 8080 of the host to 80 of the container , nginx container will listen in port 80 and the host on port 8080  
 --name : name of the container  
 -P : random port*

- From the above we need to add an inbound rule for 8080
- Multiple containers can run using the same port but you have to change the published port (8080 --> 8081)
- To change the listening port of the container , we stop then

```
docker inspect <container_name_or_id>
docker run -p <new_host_port>:<container_port> --name <container_name> <image_name>
```

```
root@ip-172-31-92-175:~# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
387c87cc6318e nginx "/docker-entrypoint..." 2 minutes ago Up 2 minutes 0.0.0.0:8080->80/tcp, :::8080->80/tcp mysite
fc375e5fcf4a httpd "httpd-foreground" 48 minutes ago Up 48 minutes 0.0.0.0:80->80/tcp, :::80->80/tcp quirky_mcclintock
root@ip-172-31-92-175:~#
```

## **Listing images and containers :**

Docker images

Docker images ls -a (**for detached as well**)

Docker pull centos:latest

Docker ps

Timelap loopback interface : docker container run centos ping 127.0.0.1

#loopback interface

Exactly. The loopback interface (`**10**` ) is a special network interface that is used for communication within the same device. Unlike other network interfaces, traffic on the loopback interface doesn't go out to the physical network. Instead, it stays within the device itself. This makes it ideal for local communication without the need to involve routers, switches, or the default gateway.

When you access services on the loopback interface using addresses like `127.0.0.1` or `::1` , the communication happens entirely within the local machine. The data doesn't leave the network stack of the device; it's processed locally, and responses are sent back through the network stack to the requesting application.

*Filter result : Docker ps -a -f status=exited*

*Filter only ID : docker ps -a -q*

## **Removing images and containers :**

docker container stop , docker container rm

Docker image rm

Force removal : *docker container rm -f mysite*

Force removal all the exited containers : *docker rm \$(docker ps -a -q -f status=exited)*

The dollar sign (`\$`) is used for command substitution in a Unix-like shell (bash in this case). Here's how it works:

1. `docker ps -a -q -f status=exited`: This command lists the IDs of all Docker containers that have exited. The `-q` option is used to print only the container IDs.
2. `\$(...)`: This is the command substitution syntax. It allows the output of the command inside the parentheses to replace the entire `\$(...)` expression.
3. `docker rm \$(docker ps -a -q -f status=exited)`: This command removes all containers that have exited. The output of the `docker ps -a -q -f status=exited` command is substituted into the `docker rm` command.

- Dangling images : have no relationship to any tagged images (useless ) : remove all stopped containers

*Docker system prune -a*

*Conclusion :*

```
## Running a Web Server in a Docker Container ##  
.  
docker container run -d --p 80:80 --name mysite1 nginx  
.  
docker container run -d --p 8080:80 --name mysite2 nginx  
.  
docker container run -d --p 8081:80 --name mysite3 nginx  
  
.  
# -d => detach and run in the background  
# --name => container name, must be unique, randomly chosen from a list if it's not given  
.  
# -p X:Y => publish a container's port to the host  
# process listens on port Y in the container but is accessed on port X from the outside (LAN/Internet)  
# -P => publish all exposed ports to random ports  
  
.  
# listing local images  
docker images # old command  
docker image ls # new command  
  
.  
# listing all running containers  
docker ps # old command  
docker container ls # new command  
# -q => printing only the containers' ids  
  
.  
# listing all containers (created, running, stopped)  
docker ps -a  
docker container ls -a  
  
.  
# filtering by status  
docker container ls -a -f status=exited  
  
.  
# stopping a container  
docker container stop CONTAINER_ID|CONTAINER_NAME  
Example: docker container stop mysite1  
  
.  
# removing a container (must be stopped)  
docker container rm CONTAINER_ID|CONTAINER_NAME  
# -f => force remove the container (can be running)  
Example: docker container rm mysite1  
  
.  
# removing all stopped containers
```

```

    docker container rm $(docker container ls -f status=exited -q)

    # removing an image
    docker rmi IMAGE_NAME # => old command
    docker image rm IMAGE_NAME # => new command
    Example: docker image rm nginx

    # removing dangling images, stopped containers, dangling build cache and networks not used
    docker system prune
    # -a => remove unused images, as well

```

## Getting shell access to a container :

By default when you start a command , the bash shell is the default command that gets executed in the centos , so it get exited the moment it runs , but you can see it using `docker ps -a`

You can overwrite the standard command by adding a random command while runinng it :  
`docker run -d centos tail -f /dev/null`

If you run `docker run centos` and the container exits immediately, it might be because the default behavior of the `centos` image is to run a minimal command and then exit. This is common for base images like CentOS that are designed to be used as a base for building more specific images.

To keep the container running, you can specify a command that keeps the container alive. For example, you can use the `-d` flag to run the container in the background, and then execute a command that doesn't exit immediately:

```

bash
Copy code
docker run -d centos tail -f /dev/null

```

This command starts a CentOS container in the background (`-d`) and runs the `tail -f /dev/null` command, which essentially does nothing but keeps the container running.

- to get a shell access  
`docker container run --name centos02 -it centos`

- to get a shell access to an existing container  
`docker container exec -it "container_id" bash`

- exit the container without stopping it :  
`CTRL+P + Q`

Conclusion :

```

# getting shell access to the container
docker run -it --name=container1 centos => press Ctrl+P +Q to exit the container without stopping it

```

```

.
.
.
# getting a shell in a running container
docker container exec -it CONTAINER_ID|CONTAINER_NAME bash
Example: docker container exec -it container1 bash

.
.
.
# executing shell commands in running container
docker container exec CONTAINER_ID|CONTAINER_NAME COMMAND
Examples: docker container exec container1 cat /etc/shadow
docker container exec container1 yum -y install nmap

.
.
.
# stopping a running container
docker container stop CONTAINER_ID|CONTAINER_NAME

.
.
.
# starting a stopped container
docker container start CONTAINER_ID|CONTAINER_NAME

.
.
.
# removing a stopped container
docker container rm CONTAINER_ID|CONTAINER_NAME

.
.
.
# removing a running container
docker container rm -f CONTAINER_ID|CONTAINER_NAME

```

## **Getting information about the running container :**

Docker container ls -a  
Create nginx server : docker container run -d -p 8080:80 --name=mysite nginx  
IP and port of container : docker container port "ID"  
Container logs : docker container logs "ID"  
Container logs in real time : docker container logs -f "ID"  
Process running on the container : docker container top "ID"  
Process running on the host : ps -ef  
Real time stats about containers : docker container stats  
Full info : docker container inspect "ID" [grep -l ipaddress]  
Check If the container is reacheble : ping "CONTAINER IP"

## **Commiting container changes to a new image :**

First lets create a centos container with a shell"  
docker container run -it --name=container1 centos

Then we create a text file in the root directory and install nmap :  
Touch /root/text.txt  
Yum install nmap

## Troubleshoot :

Same problems persists , when I pull the centos image from the hub in the aws cloudshell , all the dependencies are installed and the yum install command don't produce errors , but when I want to use the yum install command in my physical machine through ssh login , it gives errors

Error : Error: Failed to download metadata for repo 'appstream': Cannot prepare internal mirrorlist:  
No URLs in mirrorlist



You

explain generate a mirrorlist for the repository.



ChatGPT

A mirrorlist is a mechanism used by package managers, like `yum` or `dnf` in CentOS and other Red Hat-based Linux distributions, to dynamically discover and choose an optimal mirror (server) for downloading software packages and updates. Instead of relying on a single server, a mirrorlist provides a list of mirrors from which the package manager can select based on factors like proximity and availability.

### 1. Repository Configuration File:

- In the `/etc/yum.repos.d/` directory, you'll find repository configuration files. These files contain information about the repository, including the URL or mirrorlist from which to download packages.

### 2. Mirrorlist vs. BaseURL:

- In a repository configuration file, you may find either a `mirrorlist` or a `baseurl` entry. The choice between them determines how the package manager fetches packages.
  - **Mirrorlist:** This is a dynamic list of mirrors maintained by the distribution. The package manager uses the mirrorlist to select a mirror based on factors like proximity and server health.
  - **BaseURL:** This is a specific URL pointing to a single mirror. The package manager uses this mirror directly for downloading packages.

Since we cant use nano or vim to edit the config file , we gonna try out another methods :

```
sudo sed 's/#baseurl/baseurl/g' CentOS-Linux-AppStream.repo (didn't work)
```

*Found the solution : its similar of the above but to different files*

```
cd /etc/yum.repos.d/
```

```

sudo sed -i 's/mirrorlist/#mirrorlist/g' /etc/yum.repos.d/CentOS-*
sudo sed -i 's|#baseurl=http://mirror.centos.org|baseurl=http://vault.centos.org|g'
/etc/yum.repos.d/CentOS-*
sudo yum update -y

```

Remember You can run multiple containers with different names from one image

Lets run another centos container

```
docker container run -d -it --name=container2 centos
```

- How do we transfer or commit container1 changes to the new container or the image :

First we start the container , we make our changes , then we exit (container1)

Then we commit the changes to a new image

```
docker commit -m "nmap installed" -a "autho" "container1 ID" "username_on_dockerhub/newname"
```

```
docker commit -m "nmap installed" -a "Taib" 5543baa467f9 taybkho/my_centos
```

S

```

root@ip-172-31-92-175:~# docker commit -m "nmap installed" -a "Taib" 5543baa467f9 taybkho/my_centos
sha256:e821172c424234a9f6077ea6e919724b6711a80f60147d57981387ba012a4b88
root@ip-172-31-92-175:~#
root@ip-172-31-92-175:~# docker images
REPOSITORY          TAG      IMAGE ID      CREATED        SIZE
taybkho/my_centos  latest   e821172c4242  3 minutes ago  558MB
dokken/centos-8    latest   fc71369c6313  42 hours ago  589MB
mysql               latest   ba6b635d3b6b  3 weeks ago   632MB
httpd               latest   59bcd61b45fd  3 weeks ago   167MB
redis               latest   3ceffe93e5ee  4 weeks ago   138MB
nginx               latest   b690f5f0fa2d5  3 months ago  187MB
hello-world         latest   d2c94e258dc9  9 months ago  13.3kB
centos              latest   5d0da3dc9764  2 years ago   231MB

```

New image was created

We can run a container from the new image

```
docker container run --name=nmap -it taybkho/my_centos
```

## Tagging and pushing custom images to docker hub :

All images have repositories except the official ones , they reside on the root level of the hub

One image can have multiple tags/version

Adding a custom tag to an existing image :

```
docker image tag_name repo
```

```
docker image tag nginx taybkho/nginx:custom
```

```

student@ubuntu-s-1vcpu-2gb-lon1-01:~$ docker image ls
REPOSITORY          TAG      IMAGE ID      CREATED        SIZE
ddandrei/my_centos  latest   182788159e60  9 minutes ago  267MB
nginx               latest   ae2feff98a0c  2 weeks ago   133MB
httpd               latest   dd85cd9bb9987  2 weeks ago   138MB
debian              latest   6d6b00c22231  2 weeks ago   114MB
centos              latest   300e315adb2f  3 weeks ago   209MB
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ docker image tag nginx ddandrei/nginx:custom
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ docker image ls
REPOSITORY          TAG      IMAGE ID      CREATED        SIZE
ddandrei/my_centos  latest   182788159e60  12 minutes ago  267MB
ddandrei/nginx      custom   ae2feff98a0c  2 weeks ago   133MB
nginx               latest   ae2feff98a0c  2 weeks ago   133MB
httpd               latest   dd85cd9bb9987  2 weeks ago   138MB
debian              latest   6d6b00c22231  2 weeks ago   114MB
centos              latest   300e315adb2f  3 weeks ago   209MB

```

Different repo and tag , same ID

Push an image to the hub :

*Docker login*  
docker image push taybkho/my\_centos

*Conclusion :*

## **Commands - Committing Changes, Tagging and Pushing Images**

## Commiting changes in a container to a new image

# STEP 1

# start the container, get shell access, make any changes and exit  
docker container run -it --name=container1 centos

# STEP 2

# create a new image from the modified container

docker commit -m "What did you do to the image" -a "Author Name" CONTAINER\_ID|CONTAINER\_NAME  
repository/new\_image\_name:tag

Example: docker commit -m "nmap installed" -a "Andrei D." container1 ddandrei/my\_centos

# STEP 3

# start containers from the image

docker image ls

docker container run -it ddandrei/my\_centos

# adding a new tag to an existing image

docker image tag nginx ddandrei/nginx:custom

docker image tag ddandrei/my\_centos:latest ddandrei/my\_centos:1.0

docker image ls

# pushing custom images to docker Hub

# STEP 1

# create an account on hub.docker.com

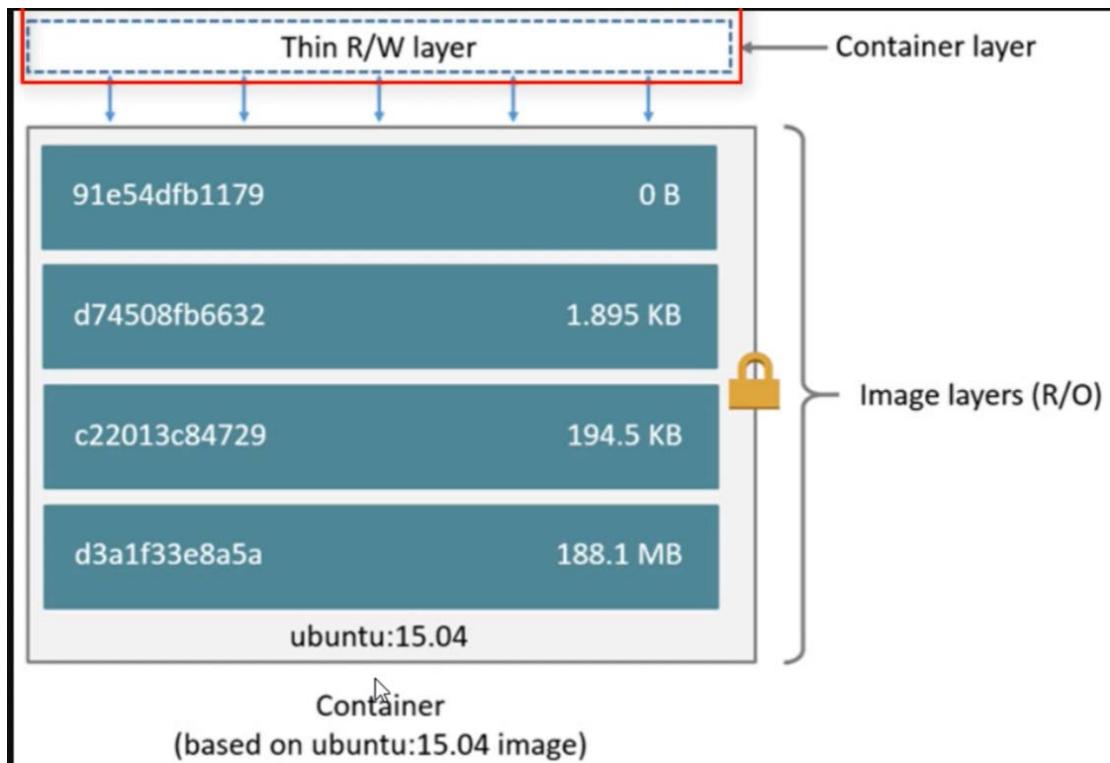
# STEP 2

docker login => enter username and password

# STEP 3

docker image push ddandrei/nginx:custom # => the username on docker hub must match the image's repository  
(ddandrei)

## **Image structure and layers :**



Docker image inspect nginx

```
    "RootFS": {
        "Type": "layers",
        "Layers": [
            "sha256:fb1bd2fc52827db4ce719cc1aafd4a035d68bc71183b3bc39014f23e9e5fa256",
            "sha256:83bdf27d9eaa4f0e8f11b95dc952e9ab92619792dbd3d72f775d54c67eae1248",
            "sha256:84e0c9ef07d70aff145d02f297c94d7d5cc0ff7d87449c20eac02f5408a48fd0",
            "sha256:06536efc503aa0d192e8a5232eb57cad57fce55a8a92dce48d45eaf284f0cc26",
            "sha256:9f21a390e3f6d9e5af2d02fdb6a622eeded91c58c9fe82af76204ad4cd854da39",
            "sha256:2b28485849eadbf9f3066df698b1579f169a7a306ce754a9809349099a58d11d",
            "sha256:f205d290cd763a4d8ddd06bee8fdb813cea79b179515fde9dec1b9fd9525856"
        ]
    }
```

Layer of the image that are save locally to be used :

Sudo ls /var/lib/docker/overlay2

Containers size :

Docker container ps -s

NAMES	SIZE
nmap	857B (virtual)
container2	0B (virtual)
wizardly_mccarthy	32.8MB (virtual)
mysite	1.09kB (virtual)
quirky_mcclintock	2B (virtual)

857 the amount of data in the disk used for the writable layer for each container

Virtual : the size of the read only and writable layers (image + container )

Multiple containers may share some or full ready only data

Two containers from different images can share some layers to avoid redundancy and extra disk space

## Creating custom images using dockerfile :

Build images automatically by reading instruction from the dockerfile (like a script)

We can check the dockerfile in the dockerhub image info

```
[ FROM debian:bookworm-slim

LABEL maintainer="NGINX Docker Maintainers <docker-maint@nginx.com>"

ENV NGINX_VERSION 1.25.3
ENV NJS_VERSION 0.8.2
ENV PKG_RELEASE 1~bookworm ]
```

NGINX VERSION : variable

Value : 1.25.3

The run command will run any command on top of the image layer and commit the changes

&& : is used to chain commands to fit in one layer , a command will run only if the previous returned successfully

Expose : the container will listen on the specific port in real time

- After checking the dockerfile from the hub , we copy its raw code and create a file in our VM , and add our own commands

```

WORKDIR /usr/share/nginx/html
EXPOSE 80
STOPSIGNAL SIGQUIT
CMD ["nginx", "-g", "daemon off;"]
~  

~  

-- INSERT --

```

This will change the pwd to the document root where the web server resides

```

root@387c87c6318e:/usr/share/nginx# cd html
root@387c87c6318e:/usr/share/nginx/html# ls
50x.html  index.html
root@387c87c6318e:/usr/share/nginx/html#

```

The bellow copy will copy page.html from docker host to html directory in nginx container

```

WORKDIR /usr/share/nginx/html
COPY page.html index.html

```

locate --ignore-case your\_search\_term

We create a the page.html text file , and then we build our image , and build the docker file in this directory

```

student@ubuntu-s-1vcpu-2gb-lon1-01:~$ vim Dockerfile
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ echo "this is a custom nginx image" > page.html
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ ls
Dockerfile  page.html
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ cat page.html
this is a custom nginx image
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ docker image build -t mynginx:1.0 .

```

Rename the file : mv old\_filename new\_filename

dockerfile Copy code

```
COPY <src> <dest>
```

• `<src>` is the source path on the host machine.  
• `<dest>` is the destination path in the Docker image.

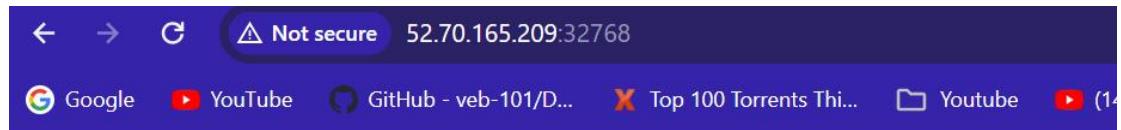
If you want to copy a file named `page.html` from the current directory on the host machine to the root directory of the Docker image and rename it to `index.html`, you can use the following line in your Dockerfile:

dockerfile Copy code

```
COPY page.html /index.html
```

```
docker container run -d -P mynginx:5.2
```

If you use a random port as the listening port , remember to edit the inbound rules



this is an nginx customer image xdd

## Use volumes and bind mounts :

In Docker, both volumes and bind mounts are mechanisms for persisting and sharing data between a Docker container and the host machine or other containers. However, they have some key differences in terms of usage, functionality, and performance.

### Volumes:

1. **Managed by Docker:**

- Volumes are managed by Docker and are stored outside of the container filesystem.
- Docker handles the creation, management, and cleanup of volumes.

2. **Named and Shared:**

- Volumes can be named, making it easier to manage and share data between containers.
- Multiple containers can share the same volume.

3. **Persistence:**

- Volumes persist data even if the container is removed.
- Data in volumes is separate from the container lifecycle.

4. **Performance:**

- Volumes generally have good performance characteristics and are suitable for various use cases.

5. **Volume Drivers:**

- Docker supports different volume drivers, allowing you to use external storage systems for volumes.

6. **Syntax:**

- To use a volume in a Dockerfile or with the `docker run` command, you specify the volume name or let Docker create a named volume.

```
```dockerfile
# In a Dockerfile
VOLUME ["/data"]

# With docker run
docker run -v mydata:/data my_image
```
```

### Bind Mounts:

1. **Host Filesystem:**

- Bind mounts link a directory or file from the host filesystem directly into the container.
- They provide direct access to files on the host.

2. **No Management by Docker:**

- Docker does not manage bind mounts; they are managed externally.
- No automatic cleanup or removal by Docker.

3. **Persistence:**

- Data in bind mounts is directly tied to the host filesystem.
- If the host file or directory is deleted, it affects the bind mount.

4. \*\*Performance:\*\*

- Bind mounts have minimal performance overhead as they directly access the host filesystem.

5. \*\*Versatility:\*\*

- Bind mounts are versatile and can be used for a wide range of scenarios, including development, debugging, and quick data sharing.

6. \*\*Syntax:\*\*

- To use a bind mount in a Dockerfile or with the `docker run` command, you specify the host path and the container path.

```
```dockerfile
# In a Dockerfile
COPY ./data /data

# With docker run
docker run -v $(pwd)/data:/data my_image
```

```

#### Choosing Between Volumes and Bind Mounts:

- Use \*\*volumes\*\* when you want Docker to manage the storage, need named and shareable storage, and want data persistence across container lifecycles.
- Use \*\*bind mounts\*\* when you need direct access to files on the host, want to share data for development or debugging, or prefer simplicity without Docker managing the storage.

In many cases, the choice between volumes and bind mounts depends on the specific requirements of your application and whether you need Docker to actively manage the storage.

```
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ docker volume create mysite
mysite
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ docker volume ls
DRIVER      VOLUME NAME
local        c63e5f8239a4a91076f8dbf00e739a851689698ae16694d706213e48ad747c1d
local        mysite
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ docker volume inspect mysite
[
  {
    "CreatedAt": "2020-12-30T09:54:44Z",
    "Driver": "local",
    "Labels": {},
    "Mountpoint": "/var/lib/docker/volumes/mysite/_data",
    "Name": "mysite",
    "Options": {},
    "Scope": "local"
  }
]
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ docker cont
container context
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ docker container run -d --name mywebapp -p 80:80 -v mysite:/usr/share/nginx/html nginx
abc697acbdad2c8370a8ad027c97473c91c755d98c2c067277944f3044d2780
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ docker container ls
CONTAINER ID  IMAGE      COMMAND             CREATED          STATUS           PORTS          NAMES
abc697acbdad2c8370a8ad027c97473c91c755d98c2c067277944f3044d2780   "/docker-entrypoint..."   6 seconds ago   Up 5 seconds   0.0.0.0:80->80/tcp   mywebapp
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ 
```

Attach the create volume to a container , and make sure that nginx has access to the assigned volume , we copy the passwd file as index.html (it will replace the orginal one , the output would be shown) :

```
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ docker volume inspect mysite
[
  {
    "CreatedAt": "2020-12-30T09:54:44Z",
    "Driver": "local",
    "Labels": {},
    "Mountpoint": "/var/lib/docker/volumes/mysite/_data",
    "Name": "mysite",
    "Options": {},
    "Scope": "local"
  }
]
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ docker cont
container context
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ docker container run -d --name mywebapp -p 80:80 -v mysite:/usr/share/nginx/html nginx
abc697acbdad2c8370a8ad027c97473c91c755d98c2c067277944f3044d2780
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ docker container ls
CONTAINER ID  IMAGE      COMMAND             CREATED          STATUS           PORTS          NAMES
abc697acbdad2c8370a8ad027c97473c91c755d98c2c067277944f3044d2780   "/docker-entrypoint..."   6 seconds ago   Up 5 seconds   0.0.0.0:80->80/tcp   mywebapp
student@ubuntu-s-1vcpu-2gb-lon1-01:~$ sudo cp /etc/passwd /var/lib/docker/volumes/mysite/_data/index.html
```

```
•          # creating a new volume
• docker volume create mysite

•          # listing all volumes
• docker volume ls

•          # inspecting a volume
• docker volume inspect mysite

•          # removing a volume
• docker volume rm mysite
• docker volume prune      # => remove all unused volumes

•          # starting a container with the volume
• docker container run -d --name mywebapp -p 80:80 -v mysite:/usr/share/nginx/html nginx
```

## Project 2 : Securing and hardening a Linux system

### Securing the openssh :

Ssh : client config  
sshd : the daemon or the server

We check the ssh config file :

*ls -l /etc/ssh*

*sshd\_config* : server configuration file

**Before making any changes the server config , always make a backup file**

To get full info use : man *sshd\_config*

*sudo cp /etc/ssh/sshd\_config /etc/ssh/sshd\_config\_backup*

After making the changes and encountering a problem , you can revert to the backup by applying

*sudo cp /etc/ssh/sshd\_config\_backup /etc/ssh/sshd\_config*

- Necessary changes :

- \*Use the latest version
- \*Change the default to a random for port that are targeted automatically and not manual attacks like ssh (22 -> 2278)
- \*Permit rootlogin no
- \*Disable password authentication (if not possible , make a strong password)
- \*Publiekeyauthentication yes
- \* Set an CLIENT ALIVE interval (a message will be sent to the client if after the time set no data was received)
- \*set a MAXAuthenttries
- \*Limit users ssh access (not everyone needs remote access)

### # Authentication:

```
#LoginGraceTime 2m
PermitRootLogin no
AllowUsers student toor u1
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

Permit ssh access only from limited IPs (admins) : by config the kernel firewall :

```
Sudo iptables -L
sudo iptables -t nat -L
```

```
Iptables -A INPUT -p tcp --dport 2278 -s 2.2.4.5 ACCEPT
```

```
root@ubuntu20-04:/home/student# iptables -A INPUT -p tcp --dport 2278 -s 2.2.4.5 -j ACCEPT
root@ubuntu20-04:/home/student# iptables -A INPUT -p tcp --dport 2278 -s 2.45.6.100 -j ACCEPT
root@ubuntu20-04:/home/student# iptables -A INPUT -p tcp --dport 2278 -j DROP
```

#### • **iptables:**

- Focuses on controlling network traffic based on IP addresses, ports, and protocols.
- Primarily used for setting up rules to allow or deny packets at the network layer (Layer 3) of the OSI model.
- Commonly used for tasks like configuring a firewall, NAT (Network Address Translation), and packet forwarding.

## Commands - SSH

- . #####
- . ## OpenSSH
- . #####
- . # 1. Installing OpenSSH (client and server)

```

# Ubuntu
sudo apt update && sudo apt install openssh-server openssh-client

# CentOS
sudo dnf install openssh-server openssh-clients

# connecting to the server
ssh -p 22 username@server_ip          # => Ex: ssh -p 2267
john@192.168.0.100
    ssh -p 22 -l username server_ip
    ssh -v -p 22 username@server_ip      # => verbose

# 2. Controlling the SSHd daemon
# checking its status
sudo systemctl status ssh            # => Ubuntu
sudo systemctl status sshd           # => CentOS

# stopping the daemon
sudo systemctl stop ssh             # => Ubuntu
sudo systemctl stop sshd            # => CentOS

# restarting the daemon
sudo systemctl restart ssh          # => Ubuntu
sudo systemctl restart sshd         # => CentOS

# enabling at boot time
sudo systemctl enable ssh           # => Ubuntu
sudo systemctl enable sshd          # => CentOS

sudo systemctl is-enabled ssh        # => Ubuntu
sudo systemctl is-enabled sshd       # => CentOS

# 3. Securing the SSHd daemon
# change the configuration file (/etc/ssh/sshd_config) and then
# restart the server
man sshd_config

a) Change the port
Port 2278

b) Disable direct root login
PermitRootLogin no

c) Limit Users' SSH access
AllowUsers stud u1 u2 john

d) Filter SSH access at the firewall level (iptables)

e) Activate Public Key Authentication and Disable Password
Authentication

```

- f) Use only SSH Protocol version 2
- g) Other configurations:  

```
ClientAliveInterval 300
ClientAliveCountMax 0
MaxAuthTries 2
MaxStartUps 3
LoginGraceTime 20
```

## Securing the bootloader :

• BIOS is a firmware interface that initializes and tests hardware components during the computer's boot process.

- The bootloader is a small program that resides in the Master Boot Record (MBR) or the EFI system partition of a storage device.
- It initiates the bootstrapping process, loading the operating system into memory.

### **Locating OS Kernel:**

- The bootloader locates the kernel of the operating system and transfers control to it, allowing the OS to take over the system.

### **GRUB (GRand Unified Bootloader):**

- GRUB is a common bootloader used in Linux systems. It can handle multiple operating systems and allows users to choose which one to boot during startup.

## Boot Sequence Order

1. BIOS executes MBR (Master Boot Record).
2. MBR executes GRUB2.
3. GRUB2 loads the kernel.
4. The kernel executes systemd which initialize the system.

### Security Checklist:

1. Physical Security (Kensington lock etc).
2. Set up a BIOS password.
3. Configure the system to boot automatically from the Linux partition.
4. Set up a password for GRUB.

- Set up a password for both the bios and grub

For bios : navigate the the security section of the BIOS

For grub :

Generate pass : *grub-mkpasswd-pbkdf2*

Add the hashed password to the grub config file :

*ls /boot/grub/grub.cfg*

*ls /etc/grub.d*

```
root@ip-172-31-92-175:/# ls /etc/grub.d
00_header 01_track_initrdless_boot_fallback 05_debian_theme 10_linux 10_linux_zfs 20_linux_xen 30_os-prober 30_uefi-firmware 40_custom 41_custom README
root@ip-172-31-92-175:/#
```

Change the custom file , because it will not be overwritten even the grub package will be updated

```
#!/bin/sh
exec tail -n +3 $0
# This file provides an easy way to add custom menu entries. Simply type the
# menu entries you want to add after this comment. Be careful not to change
# the 'exec tail' line above.
set superusers="root"
password_pbkdf2 root grub.pbkdf2.sha512.10000.79F80442D0CB864E58971BFCE423D71DF4207E688D58E4F563D94E437368B3DE1E6CC5
2D7E44CE178523E182EB0B574A333577B5455203E48273FA8568EF4A52.64C5C01EB8E7B0BEBC5F84A22AFADAFBC2C1EA151158BCA9DF3424FE
C3B39245B3108BC356E0BF55782A412784B76DB9833A56202DBD234D887C549AA554881
```

*update-grub2*

You can see the changes in *cat /boot/grub/grub.cfg*

Pass : grubmustbesecured

## Enforcing password policy :

- The password policy is located in :

*nano /etc/login.defs*

*man Login.defs*

Change the password again controls

Add : *PASS\_MIN\_LEN*

- **Add user tax :**

*sudo useradd -m -d /home/tux -s /bin/bash tux*

- Setting user password :

*sudo passwd tux*

- Change the maximum days of use using “chage” to user tax :

*Sudo chage -M 90 tux*

*-Change the minimum characters required*

*PASS\_MIN\_CLASS 4*

- Password settings :

*Sudo chage -l tux*

- Password complexity :

```
Sudo apt install libpam-pwquality  
nano /etc/pam.d/common-password
```

```
# here are the per-package modules (the "Primary" block)  
password      requisite          pam_pwquality.so retry=3 minlen=8 difok=3 ucredit=-1 lcredit=-1 dcredit=-1 ocredit=-1  
password      [success=1 default=ignore]      pam_unix.so obscure use_authok try_first_pass yescrypt
```

retry=3 minlen=8 difok=3 ucredit=-1 lcredit=-1 dcredit=-1 ocredit=-1

- `retry=3` : This specifies the number of times a user is allowed to retry entering a valid password. In this case, the limit is set to 3 retries.
- `minlen=8` : This sets the minimum length for a valid password to 8 characters.
- `difok=3` : This sets the number of characters that must differ between the old and new passwords when changing a password. In this case, at least 3 characters must be different.
- `ucredit=-1` : This enforces a requirement for at least one uppercase letter in the password ('`ucredit`' stands for uppercase credit). The value of '`-1`' means there is no specific credit requirement for uppercase letters.
- `lcredit=-1` : This enforces a requirement for at least one lowercase letter in the password ('`lcredit`' stands for lowercase credit). The value of '`-1`' means there is no specific credit requirement for lowercase letters.
- `dcredit=-1` : This enforces a requirement for at least one digit (numeric character) in the password ('`dcredit`' stands for digit credit). The value of '`-1`' means there is no specific credit requirement for digits.
- `ocredit=-1` : This enforces a requirement for at least one special character (non-alphanumeric character) in the password ('`ocredit`' stands for other credit). The value of '`-1`' means there is no specific or '\*' requirement for special characters.

## Locking or disabling user accounts :

Su - tux

- Disable authentication by passwd :

```
passwd -l tux
```

```
tux:!:6$6./8FVHdr7RNnsYC$q@Q/bybwv5vxxGXDyKBI1VTKIUIw.keZPbmXRiCJCcYQwbTxziEnbqSfJWFq9gcAkhPIFtdq4288vl:  
04:0:99999:7:::
```

```
root@ip-172-31-92-175:~# passwd --status tux  
tux L 02/14/2024 0 90 7 -1
```

- Unlock passwd :

```
Sudo passwd -u tux
```

- Completely disable a user-account :

```
sudo usermod --expiredate 1 username
```

- Reenable an account :

```
sudo usermod --expiredate -1 username  
sudo usermod --expiredate "" username
```

```
# locking password authentication  
sudo passwd -l USERNAME
```

```
•      sudo password --lock USERNAME  
•  
•          # checking the account status  
•          sudo passwd --status USERNAME  
•          sudo chage -l USERNAME  
•  
•          # unlocking password authentication  
•          sudo passwd -u USERNAME  
•  
•          # disable an account completely  
•          sudo usermod --expiredate 1 tux  
•          sudo usermod --expiredate 1970-01-02 tux  
•  
•          # checking the account expiration date  
•          sudo chage -l tux
```

## Giving limited root privileges :

-Sudo group :

*grep sudo /etc/group*

- Add an account to the sudo group :

*sudo ls -l /root/*

- Give limited access to sudoers

*cat /etc/sudoers*

*sudo update-alternatives --config editor*

*Sudo visudo*

```

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Members of the admin group may gain root privileges
%admin  ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

```

root@VM-Local:~\$ sudo visudo - Part 1

| <b>1</b> | <b>2</b> | <b>3</b>  | <b>4</b>  |
|----------|----------|-----------|---|
| root     | ALL=     | (ALL:ALL) | ALL   |
| tux      | ALL=     | (root)    | /bin/cp,/bin/ls,/usr/bin/vim,!/usr/bin/vim /etc/shadow                  |
| dan      | ALL=     |           | NOPASSWD:ALL  |
| john     | ALL=     |           | NOPASSWD:/bin/cat,/usr/bin/updatedb,PASSWD:/bin/rm,<br>NOEXEC:/bin/less |
| %sudo    | ALL=     | (ALL:ALL) | ALL   |
| %team    | ALL=     | (root)    | /usr/bin/apt update   |

*Group : %team*

*ALL = : to all hosts*

*(ALL:ALL) : ROOT CAN run commands as all users , and tux as root*

*NOPASSWD:ALL : no passwd when using commands*

*Practice :*

*First we list all the users :*

```
[ec2-user@ip-172-31-89-180 ~]$ sudo cat /etc/passwd | grep home
ec2-user:x:1000:1000:Cloud User:/home/ec2-user:/bin/bash
taib:x:1001:1001::/home/taib:/bin/bash
john:x:1002:1002::/home/john:/bin/bash
camilla:x:1003:1003::/home/camilla:/bin/bash
```

User privilege specifications :

```
nano /etc/sudoers
sudo visudo
```

*which ls , cat (absolute path)*

```
##
## Allow root to run any commands anywhere
root    ALL=(ALL)      ALL
john   ALL=(root)      /usr/bin/ls, /usr/bin/cat
```

```
[ec2-user@ip-172-31-89-180 ~]$ su - john
Password:
Last login: Sat Feb 17 17:54:09 UTC 2024 on pts/0
[john@ip-172-31-89-180 ~]$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
```

```
[john@ip-172-31-89-180 ~]$ sudo rpm update
Sorry, user john is not allowed to execute '/bin/rpm update' as root on ip-172-31-89-180.ec2.internal.
[john@ip-172-31-89-180 ~]$
```

```
# User privilege specification
root    ALL=(ALL:ALL) ALL
dan     ALL=(root)      NOPASSWD:/usr/bin/ls,/usr/bin/cat,NOPASSWD:/usr/bin/apt
```

Creating user-alias and command alias :

*Sudo visudo*

"Its good to use alias to avoid repetition , its like a STRING VARIABLE "

```
# User alias specification
User_Alias MYADMIN=dan,john

# Cmnd alias specification
Cmnd_Alias FILE=/usr/bin/cp,/usr/bin/ls,/usr/bin/touch,/usr/bin/rm

# User privilege specification
root    ALL=(ALL:ALL) ALL
dan     ALL=(root)      PASSWD:/usr/bin/ls,/usr/bin/cat,NOPASSWD:/usr/bin/apt
MYADMIN ALL=(root)      /usr/bin/netstat,FILE
```

Now camilla can run ls root

```
[camilla@ip-172-31-89-180 ~]$ sudo ls /root
[camilla@ip-172-31-89-180 ~]$
```

## Settings user limits (Running a DOS attack without root access)

### Rabbit virus

Fork bomb is a script

Vim bom

\$0 && \$0 &

Chmod +x bomb

./bomb

The system will become unusable , the vm needs to be rebooted , if its production system , the whole system will crash , it uses the fork system call

• **Explanation:**

- The script executes ` `\$0` , which represents the name of the script or shell.
- If the execution of ` `\$0` is successful (returns a zero exit status), then the second occurrence of ` `\$0` is executed.
- The second occurrence of ` `\$0` is run in the background (` &`), allowing the script to continue with the next command without waiting for the completion of the background process.

**Prevent fork bomb (replication of processes) :**

`ulimit -u`

`sudo vim /etc/security/limits.conf`

```
##          soft    core      0
##          hard    rss     10000
#@student   hard    nproc     20
#@faculty   soft    nproc     20
#@faculty   hard    nproc     50
#ftp        hard    nproc      0
#@student   -      maxlogins  4
john       hard    nproc     2000
camilla    hard    nproc     1400
```

Nproc : number of processes  
fsize , nofile , maxlogins

## Intro to cracking passwords :

Using brute force (all possible combinations), or rainbow tables (precomputed reversed hashes )

Listofrainbowtables  
Haveibeenpawned

## Cracking Linux passwords using john the ripper

John the ripper [single , dictionary , incremental(all possible char)]

[Troubleshoot :](#)

You cant install john the ripper until you log in into your Redhat account

```
sudo subscription-manager register
subscription-manager repos --list
```

Error : Repositories disabled by configuration.

Solu :

ls /etc/yum.repos.d/  
switch enabled to 1 for everything

```
[root@ip-172-31-89-180 yum.repos.d]# cat redhat-rhui.repo
```

```
[rhel-9-appstream-rhui-debug-rpms]
name=Red Hat Enterprise Linux 9 for $basearch - AppStream from RHUI (Debug RPMs)
mirrorlist=https://rhui.REGION.awesome.redhat.com/pulp/mirror/content/dist/rhel9/rhui/$releasever/$basearch/appstream/debug
enabled=1
gpgcheck=1
gpgkey:///etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release
sslverify=1
sslclientkey=/etc/pki/rhui/content-rhel9.key
sslclientcert=/etc/pki/rhui/product/content-rhel9.crt
sslcacert=/etc/pki/rhui/cdn.redhat.com-chain.crt
```

Rhui : redhat update infrastructure

```
sudo yum clean all
sudo yum makecache
```

EPEL rep : extra Package for entreprise Linux

```
sudo dnf install https://dl.fedoraproject.org/pub/epel/epel-release-latest-9.noarch.rpm
sudo dnf upgrade
```

ERROR :

```
[ec2-user@ip-172-31-89-180 yum.repos.d]$ sudo subscription-manager status
+-----+
| System Status Details
+-----+
Overall Status: Disabled
Content Access Mode is set to Simple Content Access. This host has access to content, regardless of subscription status.
System Purpose Status: Disabled
```

The currentsubscription don't allow to upgrade or add packages

<https://snapcraft.io/install/john-the-ripper/rhel>

```
sudo subscription-manager register
```

Continue :

### Single mode

```
Sudo yum install john
unshadow /etc/passwd /etc/shadow > unshadowed.txt
john -single unshadowed.txt --format=crypt
John --show unshadowed.txt
Cat ~/john/john.pot
```

### Dictionary Mode :

```
Ls /usr/share/dict
Ls /usr/share/john
Wc -l /usr/share/dict/american-english
```

```
John --wordlist=/usr/share/john/password.lst --rules unshadowed.txt  
John --show unshadowed.txt
```

*Restore a session : john -restore*

```
#####
## CRACKING PASSWORD HASHES USING JOHN THE RIPPER
#####

# Installing JTR
apt install john

# combining /etc/passwd and /etc/shadow in a single file
unshadow /etc/passwd /etc/shadow > unshadowed.txt

# cracking in single mode
john -single --format=crypt unshadowed.txt --format=crypt

# brute-force and dictionary attack
john --wordlist=/usr/share/john/password.lst --rules unshadowed.txt --format=crypt

# dictionary files:
# /usr/share/dict
# /usr/share/metasploit-framework/data/wordlists # -> on Kali Linux

# showing cracked hashes (~/.john/john.pot)
john --show unshadowed.txt

# to continue an interrupted (ctrl+c) session, run in the same directory:
john -restore

# cracking only accounts with specific shells (valid shells)
john --wordlist=mydict.txt --rules --shell=bash,sh unshadowed.txt --format=crypt

# cracking only some accounts
john --wordlist=mydict.txt --rules --users=admin,mark unshadowed.txt --format=crypt

# cracking in incremental mode (/etc/john/john.conf)
john --incremental unshadowed.txt --format=crypt

# eliminate duplicates from a wordlist (if any)
john --wordlist=mydict.txt --stdout | unique mydictionary.txt
```

## Checking files integrity with AIDE :

Advanced intrusion detection environment ; file and directory integrity checker

### AIDE - Advanced Intrusion Detection Environment.

- AIDE is a file integrity monitoring or file integrity verification tool.
- AIDE works by taking a snapshot of the state of the system. This snapshot will be used to build a database.
- When you run an integrity test, AIDE will compare the database against the real status of the system. If there are any changes between the existing snapshot and the test, AIDE will detect it and report it to you.
- AIDE is a host-based IDS (HIDS) .

Apt install aide

Vim /etc/aide/aide.conf

Aide --init

```
Ls -l /var/lib/aide/
root@ubuntu-s-1vcpu-2gb-lon1-01:~# ls -l /var/lib/aide/
total 35868
-rw-r--r-- 1 root root 57566 Dec  8 10:58 aide.conf.autogenerated
-rw----- 1 root root 18333613 Dec  8 11:02 aide.db
-rw----- 1 root root 18333613 Dec  8 11:02 aide.db.new
```

Mv /var/lib/aide/aide.db.new /var/lib/aide/aide.db

**Runtime conf file :**

update-aide.conf

Ls /var/lib/aide/

**Testing aide :**

Touch /root/abc.txt

Useradd user1

Aide -c /var/lib/aide/aide.conf.autogenerated --check > report.txt

Less report.txt

```
-----  
Added entries:  
-----  
f+++++*: /etc/subgid-  
f+++++*: /etc/subuid-  
f+++++*: /root/abc.txt  
f+++++*: /root/report.txt  
f+++++*: /var/lib/aide/aide.db  
f+++++*: /var/lib/postfix/prng_exch  
f+++++*: /var/lib/postfix/smtp_scache.db  
f+++++*: /var/spool/postfix/pid/inet.smtp
```

If the changes are okay :

Aide -c /var/lib/aide/aide.conf.autogenerated --update

Cp /var/lib/aide/aide.db.new /var/lib/aide/aide.db

**Create your own aide config file :**

Vim /root/aide.conf

---

```
database=file:/var/lib/aide/aide.db
database_out=file:/var/lib/aide/aide.db.new
database_new=file:/var/lib/aide/aide.db.new

MYRULE=u+g+p+n+s+m+sha256

/etc MYRULE
/usr MYRULE
/root MYRULE
!/usr/lib/.*
!/usr/bin/zip$
```

**Initializing the new aide database :**

Aide -c /root/aide.conf --init

**Make the produced database as the base database :**

mv /var/lib/aide/aid.db.new /var/lib/aide/aid.db

**Checking the new changes against the baseline database :**

Aide -c /root/aide/aide.conf --check

**Limit the check to only one repo like /etc :**

Aide -c /root/aide/aide.conf --limit /etc --check

```
#
# installing AIDE
apt update && apt install aide
aide -v

#
# getting help
aide --help

#
/etc/aide/aide.conf # => config file

#
### SEARCHING FOR CHANGES ###

#
# initializing the AIDE database => /var/lib/aide/aide.db.new
aideinit

#
# moving the db to the one that will be checked by AIDE
mv /var/lib/aide/aide.db.new /var/lib/aide/aide.db

#
# creating a runtime config file => /var/lib/aide/aide.conf.autogenerated
update-aide.conf # this is a command to run
```

```

# detecting changes
aide -c /var/lib/aide/aide.conf.autogenerated --check > report.txt

# updating the db
aide -c /var/lib/aide/aide.conf.autogenerated --update
# copying the newly created database as the baseline database
cp /var/lib/aide/aide.db.new /var/lib/aide/aide.db

## CREATING A CUSTOM aide.conf FILE (Example: /root/aide.conf) ##
database_in=file:/var/lib/aide/aide.db
database_out=file:/var/lib/aide/aide.db.new
MYRULE=u+g+p+n+s+m+sha256
/etc MYRULE
/usr MYRULE
/root MYRULE
!/usr/*
!/usr/share/file$


# initializing the new AIDE db
aide -c /root/aide.conf --init

# moving the new db to the baseline db
mv /var/lib/aide/aide.db.new /var/lib/aide/aide.db

# checking for any changes
aide -c /root/aide.conf --check

```

## Scanning for rootkits (rkhunter , chkrootkit) :

### Rootkits

- A **rootkit** is a collection of malicious computer software designed to enable access to a computer that is not otherwise allowed.
- After a successful intrusion into a system, usually the intruder will install a so-called "rootkit" to secure further access.
- Rootkit detection is difficult because a rootkit may be able to subvert the software that is intended to find it (rootkit scanners, antivirus).
- **NEVER TRUST A COMPROMISED MACHINE. PERIOD**

#### Rootkit Scanners:

##### 1. Rootkit Hunter (rkhunter)

rkhunter --check

##### 2. chkrootkit

chkrootkit -q

It is recommended to run them from a live disk or alternate directory , so the root scanner may trust the commands

```
Apt update && apt install rkhunter
```

**Update rkhunter datafile with the current values of the system :**  
Rkhunter --propupd

**Full system check :**

```
Rkhunter --check
```

**The system output :**

```
Tail -f /var/log/rkhunter.loh
```

**Run to show only warnings :**

```
Rkhunter --check --report-warnings-on
```

**Minimize false positives by whitelisting :**

```
Vim /etc/rkhunter.conf
```

```
/usr/bin/mawk          [ OK ]
/usr/bin/lwp-request    [ Warning ]
/usr/bin/bsd-mailx      [ OK ]
```

---

```
# SCRIPTWHITELIST=/usr/bin/lwp-request
# This is the main configuration file for Rootkit Hunter.
#
```

**Locally check for rootkits : chkrootkit**

```
Apt install chkrootkit
```

```
run : chkrootkit -q
```

## Scanning for viruses with CLAMAV :

In linux , an antivirus aint necessary

If ur running a linux based file server such an ftp or mail server , you should use an antivirus

The antivirus will protect your server from rogue windows machines uploading malicious files to your server

<https://clamav.net/>

**Install clamav :**

```
Apt install && apt install clamav clamav-daemon
```

**Checking the manual :**

```
man clamd (daemon)
Man clamdscan (clamd has to be runing)
Man clamscan
```

Man freshclam

**Freshclam status :**

Systemctl status clamav-freshclam

Ls /var/lib/clamav/  
Freshclam --version

**Daemon start :**

Systemctl start clamav-daemon (loads to virusdata in its memory but you need to wait for a while , check the log file instead to see if it can be used)

Systemctl enable clamav-daemon (to start on boot)

Tail /var/log/clamav/clamav.log

**Test virus :**

Wget [www.eicar.org/download/eicar.com](http://www.eicar.org/download/eicar.com)

clamdscan --fdpass /root

*Commands - ClamAV*

```
·      # installing clamav
·      sudo apt install && sudo apt install clamav clamav-daemon
·
·      # checking the status
·      systemctl status clamav-freshclam
·      systemctl status clamav-daemon
·
·      # starting the clamav daemon
·      systemctl start clamav-daemon
·
·      # enabling the daemon to start and boot
·      systemctl enable clamav-daemon
·
·      # getting a test virus
·      wget www.eicar.org/download/eicar.com
·
·      # scanning a directory using clamdscan
·      clamdscan --fdpass /root
·
·      # moving found viruses to a quarantine directory
·      clamdscan --move=/quarantine --fdpass /root
·
·      # scanning a directory using clamscan
·      clamscan --recursive /etc
·      clamscan --recursive --infected /quarantine
·      clamscan --recursive --infected --remove /quarantine/
```

**Before using clamscan you need to stop clamav-daemon service :**

Systemctl stop clamav-daemon

Then :

Clamscan --recursive /etc

**Clamav GUI :**

Sudo apt install clamtk  
Sudo clamtk

## **Full disk encryption using dm-crypt and LUKS :**

Full disk encryption , when applying it all the data would be lost

*Apt install cryptsetup (manage plain dm-crypt and lucks encrypted volumes )*

**Disk partition table :**

Fdisk -l

Check the device column

Remember to backup before the next step

**Feeding the partition with random data :**

This step adds an additional layer of security, making it harder for any residual data to be recovered

In the vm , you could see the usb partition in the setiings of the vm in the USB section

*dd if=/dev/urandom of=/dev/sdb status=progress*

“Mounting a partition is the process of integrating it into the file system, making its contents available to the operating system ”

**Initialize LUCKS :**

The partition should be unmounted

*Cryptsetup -y -v lucksFormat /dev/sdb*

**Open the encrypted devise :**

-Secretdata is the mapperfile , and the command file created a mapping between the disk and the secretdata in dev mapper

*cryptsetup luksOpen /dev/sdb secretdata*

```
Ls -l /dev/mapper/secretdata
```

It's a symlink to another file

```
root@kali:~$ls -l /dev/mapper/
total 0
crw----- 1 root root 10, 236 Mar  5 04:21 control
lrwxrwxrwx 1 root root      7 Mar  5 04:21 secretdata -> ../dm-0
root@kali:~$ls -l /dev/dm-0
brw-rw---- 1 root disk 254, 0 Mar  5 04:21 /dev/dm-0
root@kali:~$
```

**The status of the mapping :**

```
Cryptsetup status secretdata
```

**Format the file system :**

```
Mkfs.ext4 /dev/mapper/secretdata
```

**Mount the encrypted filesystem to the main file tree :**

```
Mount /dev/mapper/secretdata /mnt
```

**Test the disk :**

```
Ls /mnt/ : content of the disk
```

```
Cp /etc/passwd /mnt
```

```
Ls /mnt
```

```
Cat /mnt/passwd
```

**Extra info : mapping**

In the context of disk encryption, a "mapped device" refers to a device that has been created and made available after unlocking and decrypting an encrypted disk or partition. This process is commonly used with the LUKS (Linux Unified Key Setup) encryption standard in Linux.

Here's a step-by-step explanation of how a mapped device is created during the process of unlocking and decrypting an encrypted disk:

**1. \*\*Unlocking the Encrypted Disk:\*\***

- When you want to access an encrypted disk, you use the `cryptsetup` command to unlock it. This command establishes a mapping between the encrypted device and a decrypted, virtual device.

```
```bash
sudo cryptsetup luksOpen /dev/sdXn mymappeddevice
````
```

- In this command:

- `/dev/sdXn` is the path to the encrypted device (e.g., a partition on a hard drive).
- `mymappeddevice` is the name you give to the mapped device. You can choose any name for this.

**2. \*\*Mapped Device Creation:\*\***

- The `cryptsetup luksOpen` command creates a mapped device at `/dev/mapper/mymappeddevice`. This mapped device is now a virtual device that represents the decrypted content of the encrypted disk.

### 3. \*\*Mounting the Decrypted File System:\*\*

- After the mapped device is created, you can mount the decrypted file system to a directory using the `mount` command.

```
```bash
sudo mount /dev/mapper/mymappeddevice /mnt/mymountpoint
````
```

- Now, the content of the decrypted disk is accessible in the specified mount point.

### 4. \*\*Performing Operations:\*\*

- You can perform read and write operations on the decrypted data through the mounted directory.

### 5. \*\*Unmounting and Closing the Mapped Device (Optional):\*\*

- When you are done with the decrypted data, you can unmount the directory:

```
```bash
sudo umount /mnt/mymountpoint
````
```

- Optionally, you can close the mapped device:

```
```bash
sudo cryptsetup luksClose mymappeddevice
````
```

- Closing the mapped device releases the mapping between the encrypted and decrypted devices. If you plan to remount, you may skip this step.

The mapped device acts as an interface that allows you to work with the decrypted content of the encrypted disk as if it were a regular, unencrypted device. It is a key component in the process of accessing encrypted data securely in a Linux environment.

```
root@kali:~$umount /mnt/
root@kali:~$umount /root/secretdata/
root@kali:~$cryptsetup luksClose secretdata
root@kali:~$cryptsetup luksOpen /dev/sdb secret
Enter passphrase for /dev/sdb:
root@kali:~$ls /dev/mapper/
control secret
root@kali:~$mount /dev/mapper/ /root/secretdata/█
```

## Unlock luks encrypted drives with a key file :

LUCKS enabled devices may hold to 10 keys

**Generate a random keyfile :**

```
dd if=/dev/urandom of=/root/keyfile bs=1024 count=4
```

**Encrypt the keyfile :**

```
Chmod 400 /root/keyfile
```

```
Cryptsetup LuksAddKey /dev/sdb /root/keyfile
```

```
/dev/sda1  *      2048 163577855 163575808 78G 83 Linux
/dev/sda2      163579902 167770111 4190210 2G 5 Extended
/dev/sda5      163579904 167770111 4190208 2G 82 Linux swap / Solaris
```

```
Disk /dev/sdb: 29.78 GiB, 31954305024 bytes, 62410752 sectors
Disk model: Flash Disk
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

**Test to unlock the drive : (secret is the mapper)**

```
Cryptsetup LucksOpen /dev/sdb secret --key-file /root/keyfile
```

**Then unmount the mapper to use the data normally :**

```
Mount /dev/mapper/secret /mnt
```

```
Ls /mnt
```

Summary :

*Commands - Full Disk Encryption*

```
FULL DISK ENCRYPTION

# 1. installing cryptsetup & dm-crypt
apt install cryptsetup

# 2. Idenfity the name of the disk or partition to encrypt:
fdisk -l # for this example it will be /dev/sdc
or
dmesg # for usb sticks

# 3. Filling the disk or partition to encrypt with random data
(optional)
dd if=/dev/urandom of=/dev/sdc status=progress

# 4. Initialize the LUKS partition and set the initial passphrase
cryptsetup -y -v luksFormat /dev/sdc
```

```
WARNING!
=====
This will overwrite data on /dev/sdc irrevocably.

Are you sure? (Type uppercase yes): YES
Enter passphrase for /dev/sdc:
Verify passphrase:
Command successful.

#5. Open the encrypted device and set up a mapping name
cryptsetup luksOpen /dev/sdc secretdata
ls -l /dev/mapper

# display the status of the mapping file
cryptsetup status secretdata

# 6. Format the filesystem
mkfs.ext4 /dev/mapper/secretdata

# 7. Mount the encrypted file system into the main file tree.
mount /dev/mapper/secretdata /mnt # -> you can mount it to any
existing directory like /mnt

# 8. Unmount the encrypted disk
umount /mnt
cryptsetup luksClose secretdata

# 9. Accesing the encrypted disk after a restart or unmounting:
cryptsetup luksOpen /dev/hdc secretdata
mount /dev/mapper/secretdata /root/secret

#10. Unlocking LUKS Encrypted Drives With A Keyfile

# generating a random keyfile
dd if=/dev/urandom of=/root/keyfile bs=1024 count=4

# set the permissions (only root can read it)
chmod 400 /root/keyfile

# add the keyfile as an additional authorization method
cryptsetup luksAddKey /dev/sdc /root/keyfile

# unlock the drive using the keyfile
cryptsetup luksOpen /dev/hdc secret --key-file /root/keyfile
```

## Symmetric encryption using GnuPG :

**Encrypting a text file :**

```
gpg -c secret.txt
```

*By default it uses AES256*

**Encrypting using a custom cipher and output :**

```
gpg -c --cipher- algo blowfish -o secret_blowfish.txt.gpg secret.txt
```

**Remove the original file :**

```
Shred -vu -n 100 secret.txt
```

**Decrypt the file :**

```
Gpg -o original.txt -d secret.txt.gpg
```

**Gpg-agent :**

Why gpg didn't ask for a passphrase : it uses gpg-agent , gpg daemon keeps copy of the passphrase securely in the RAM

Make it forget the passphrase :

```
echo RELOADAGENT | gpg-connect-agent
```

Or

```
gpgconf --kill gpg-agent
```

**Change the output format to ASCII instead of binary :**

"its useful to use in text based applications "

```
Gpg -ca secret.txt
```

## Using an encrypted text editor (Encryptpad) :

It automates the gpg function

## Installation

Use the commands below to install the packages.

```
sudo add-apt-repository ppa:nilarimogard/webupd8
sudo apt update
sudo apt install encryptpad encryptcli
```

Its like notepad but the text file will be encrypted

It allows to use a passphrase and a keyfile

## Steganography :

- Steganography is the art of hiding secret information in plain-text or in clear-sight.
- Steganographic tools can easily embed secret files into images, movies, audio files or other file formats.
- The word steganography comes from the Greek word "steganos" which means "hidden" and "graph" or "graphia" which means writing.

The purpose of steganography is to hide even the mere existence of the message that is being sent.

### Steps:

1. The secret file is encrypted.
2. The encrypted secret file is embedded into a **cover file** according to a steganographic algorithm. The cover file that contains the secret message or the embedded file is called **stego file**.
3. The stego file is sent normally (in clear-text or encrypted) to the destination or is made public to be easily reached.

- Sending encrypted messages without raising suspicion, such as in countries where free speech is suppressed.
- Digital watermark of the copyright holder.
- Hiding or transporting secret information (secret documents, Bitcoin private key etc).
- Transporting sensitive data from point A to point B such that the transfer of the data is unknown.

### Extract the hidden message :

Steghide extract -sf img.jpg

### Construct a sego file :

First we need to get the color specs :

Color orange = RGP(255,153,0) or in binary :  
(11111111 , 10011001 , 00000000 )

LSB (least significant bit)= 1

Stenography will happen by hiding data in the LSB , the more bits altered , the more obvious the change

Like this

(11111111 , 10011001 , 00000000 ) --> embedding 1,0,1 -> (11111111 , 10011000 , 00000001 )

New color RGB = (255 , 152 , 1)

In an image of 1 megabite , you could embed 125 kilobytes

## Hide secret messages through steganography with steghide :

Sudo apt install steghide

Check to see if we have the same image:

sha256sum img\*

```
(kali㉿kali)-[~/Downloads]
$ sha256sum gojo*
c8faf2b024f80096ddaf9965c0e8ff19770b822f3ae7f01d9c8729a8781de72b  gojo-original.jpg
c8faf2b024f80096ddaf9965c0e8ff19770b822f3ae7f01d9c8729a8781de72b  gojo-steg.jpg
```

Check the capacity of the cover file :

steghide info gogjo-steg.jpg

Embed the secret file into the cover file :

Steghide embed -cf gojo-steg.jpg -ef secret.txt

Spot the difference :

Hash and the size

Steganalysis

```
(kali㉿kali)-[~/Downloads]
$ sha256sum gojo*
c8faf2b024f80096ddaf9965c0e8ff19770b822f3ae7f01d9c8729a8781de72b  gojo-original.
14eda0faef98cfa409c4a333df961eebb58ecb7ac1bfec7e1b4d1e7782230b5a  gojo-steg.jpg
```

#### Extract the embeded file :

Steghide extract -sf gojo.steg.jpg

```
[kali㉿kali)-[~/Downloads/stego]
└─$ mv secret.txt secret-backup.txt

[kali㉿kali)-[~/Downloads/stego]
└─$ ls
gojo-steg.jpg  secret-backup.txt

[kali㉿kali)-[~/Downloads/stego]
└─$ steghide extract -sf gojo-steg.jpg
Enter passphrase:
wrote extracted data to "secret.txt".
```

### *Commands - Steganography*

#### STEGANOGRAPHY

```
# installing steghide
apt update && apt install steghide

# embedding a secret file into a cover file
steghide embed -ef secret.txt -cf cat.jpg
Enter passphrase:
Re-Enter passphrase:
embedding "secret.txt" in "cat.jpg"... done

# getting info about a cover/stego file
steghide info cat.jpg
"cat.jpg":
format: jpeg
capacity: 47.5 KB
Try to get information about embedded data ? (y/n) n

# extracting the secret file from the stego file
steghide extract -sf cat.jpg
Enter passphrase:
wrote extracted data to "secret.txt".
```

## Scanning networks with Nmap :

**NMAP** is a network discovery and security auditing tool.

- TCP Scans:
  - SYN Scan: **-sS** (root only)
  - Connect Scan: **-sT**
- UDP Scan: **-sU**
- ICMP Scan: **-sn** or **-sP**

**Example:** `nmap -sS -p 22,100 -sV 192.168.0.1`

`nmap -sS scanme.nmap.org`

Try to scan all the ports , the admin may have changed the ssh to another port in `sshd_config`

**ICMP ping packet** is auto and if it doesn't get any response , it will give up (host is down) , we should disable the standard behaviour of sending initial ping

`Nmap -Pn target`

**Pass IDS :**

`-D` : decoy scan (to hide IP address )

```
root@linux1:/home/student# nmap -p 22 -sV 192.168.0.20 -D 192.168.0.1,192.168.0.21,192.168.0.100
```

The decoy hosts should be all running , it would be easy to find out which one to block if it's the only one up

**Pass list of targets :**

`Nmap -p 22 -iL target.txt`

**Output to a file :**

`Nmap -p 22 -iL target.txt -n -oN recon.txt`

**Timetemplate :**

`-T4`

**-A :**

Enables operating system , version detection , traceroute

*Commands - Nmap*

```
#####
## NMAP
#####
```

```

.
.
.

    ##** SCAN ONLY YOUR OWN HOSTS AND SERVERS !!! **##
    ## Scanning Networks is your own responsibility ##

.
.

    # Syn Scan - Half Open Scanning (root only)
nmap -sS 192.168.0.1

.
.

    # Connect Scan
nmap -sT 192.168.0.1

.
.

    # Scanning all ports (0-65535)
nmap -p- 192.168.0.1

.
.

    # Specifying the ports to scan
nmap -p 20,22-100,443,1000-2000 192.168.0.1

.
.

    # Scan Version
nmap -p 22,80 -sV 192.168.0.1

.
.

    # Ping scanning (entire Network)
nmap -sP 192.168.0.0/24

.
.

    # Treat all hosts as online -- skip host discovery
nmap -Pn 192.168.0.101

.
.

    # Excluding an IP
nmap -sS 192.168.0.0/24 --exclude 192.168.0.10

.
.

    # Saving the scanning report to a file
nmap -oN output.txt 192.168.0.1

.
.

    # OS Detection
nmap -O 192.168.0.1

.
.

    # Enable OS detection, version detection, script scanning, and traceroute
nmap -A 192.168.0.1

.
.

https://nmap.org/book/performance-timing-templates.html

.
.

    -T paranoid|sneaky|polite|normal|aggressive|insane (Set a timing template)
These templates allow the user to specify how aggressive they wish to be, while leaving Nmap to pick the
exact
do
not currently exist.

.
.

    # -A OS and service detection with faster execution
nmap -A -T aggressive cloudflare.com

.
.

    # Using decoys to evade scan detection
nmap -p 22 -sV 192.168.0.101 -D 192.168.0.1,192.168.0.21,192.168.0.100

.
.

    # reading the targets from a file (ip/name/network separated by a new line or a whitespace)

```

```
nmap -p 80 -iL hosts.txt  
.  
.  
# exporting to out output file and disabling reverse DNS  
nmap -n -iL hosts.txt -p 80 -oN output.txt
```

## Intro to netfilter and iptables :

A firewall controls network traffic and establishes a barrier between intranet and external network

Netfilter is a linux firewall

- **Netfilter** is a **software firewall**, a **packet filtering framework inside the Linux Kernel**.
- It enables **packet filtering, NAT, PAT, Port Forwarding** and **packet mangling**.
- Netfilter framework is controlled by the **iptables** command.
- Iptables is a tool that belongs to the **user-space** used to configure netfilter.
- Netfilter and Iptables are often combined into a single expression netfilter/iptables.
- Every Linux distribution uses netfilter/iptables, there is nothing extra that should be installed.
- **Only root user can use or configure the netfilter framework**

## Chain traversal :

### Chain Traversal in a Nutshell

- **Incoming traffic** is filtered on the **INPUT CHAIN** of the **filter table**.
- **Outgoing traffic** is filtered on the **OUTPUT CHAIN** of the **filter table**.
- **Routed traffic** is filtered on the **FORWARD CHAIN** of the **filter table**.
- **SNAT/MASQUERADE** is performed on the **POSTROUTING CHAIN** of the **nat table**.
- **DNAT/Port Forwarding** is performed on the **PREROUTING CHAIN** of the **nat table**.
- To modify values from the packet's headers add rules to the **mangle table**.
- To skip the connection tracking add rules with **NOTRACK target** to the **raw table**

## Iptables basic usage :

### iptables command

#### Basic usage:

**iptables [-t table\_name] -COMMAND CHAIN\_NAME matches -j TARGET**

| Table            | Command      | CHAIN        | matches          | Target/Jump |
|------------------|--------------|--------------|------------------|-------------|
| filter (default) | -A (append)  | INPUT        | -s source_ip     | ACCEPT      |
| nat              | -I (insert)  | OUTPUT       | -d dest_ip       | DROP        |
| mangle           | -D (delete)  | FORWARD      | -p protocol      | REJECT      |
| raw              | -R (replace) | PREROUTING   | --sport source_p | LOG         |
|                  | -F (flush)   | POSTROUTING  | --dport dest_p   | SNAT        |
|                  | -Z (zero)    | USER_DEFINED | -i incoming_int  | DNAT        |
|                  | -L (list)    |              | -o outgoing_int  | MASQUERADE  |
|                  | -S (show)    |              | -m mac           | LIMIT       |
|                  | -N           |              | -m time          | RETURN      |
|                  | -X           |              | -m quota         | TEE         |
|                  |              |              | -m limit         | TOS         |
|                  |              |              | -m recent        | TTL         |

#### Filter incoming ping request to DENY :

Iptables -t filter -A INPUT -p icmp --icmp-type echo-request -j DROP

#### List the content of the firewall :

Iptables -vnL

```
(root㉿kali)-[~]
└─# iptables -t filter -A INPUT -p icmp --icmp-type echo-request -j DROP

(root㉿kali)-[~]
└─# iptables -vnL
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target  prot opt in     out    source               destination
   0    0  DROP      1  --  *      *       0.0.0.0/0            0.0.0.0/0          icmp-type 8

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target  prot opt in     out    source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target  prot opt in     out    source               destination
```

#### Deny outgoing traffic to a malicious website :

iptables -t filter -A OUTPUT -p tcp --dport 443 -d [www.ubuntu.com](http://www.ubuntu.com) -j DROP

If you want to block outgoing traffic to port 443 for the domain "[www.ubuntu.com](http://www.ubuntu.com)," you would need to find the IP address associated with that domain and use it in your rule. For example:

```
bash
Copy code

# Find the IP address of www.ubuntu.com
$ nslookup www.ubuntu.com

# Assume the IP address is 203.0.113.1, replace it in the rule
iptables -t filter -A OUTPUT -p tcp --dport 443 -d 203.0.113.1 -j DROP
```

```
(root@kali)-[~]
# nslookup google.com
Server:      10.128.128.128
Address:     10.128.128.128#53

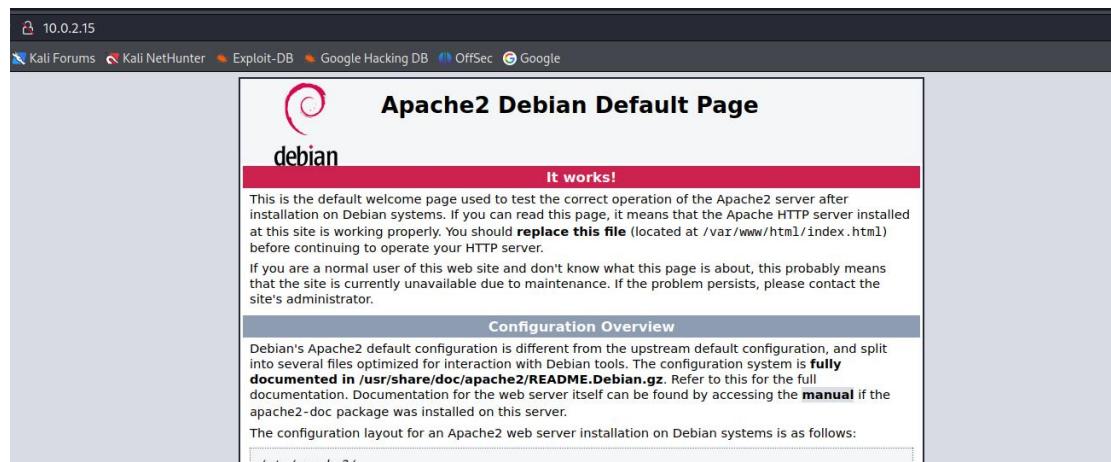
Non-authoritative answer:
Name:   google.com
Address: 142.251.39.14
Name:   google.com
Address: 2a00:1450:400d:808::200e
```

Iptables -L -vn

This wont work on websites who have multiple ip address like Google

#### **Drop incoming traffic from a local webserver :**

Apt install apache2  
 systemctl start apache2  
<http://10.0.2.15:80>



Filter table is the standard for iptables

iptables -A INPUT -p tcp --dport 80 -j DROP

#### **Drop incoming traffic to TFTP :**

Iptables -I INPUT -p udp --dport 69 -j DROP

If we use **-I** , the new rule would be the first rule in the chain

**Insert on position 3 :**

```
root@linux1:~/scripts# iptables -I INPUT 3 -p udp --dport 69 -j DROP
```

Insert = on top

Append = in the end

Chains : input , output , forward

The default policy : set to drop as a good security practice :

Iptables -P FORWARD DROP

-D :

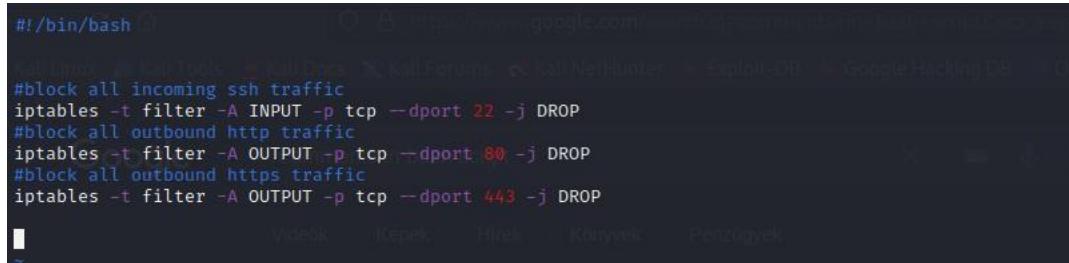
```
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target  prot opt in     out      source          destination
  0    0 ACCEPT   tcp  --  *       *       0.0.0.0/0        0.0.0.0/0          tcp dpt:22
  0    0 ACCEPT   tcp  --  *       *       0.0.0.0/0        0.0.0.0/0          tcp dpt:80
root@linux1:~/scripts# iptables -D OUTPUT 2
```

Note : we use iptables if we want to add a single rule , when configuring a firewall we should use a script , because iptable rule added from the terminal will be lost after the machine gets turned off or reboot

Script : shell file that contains shell commands

Iptables bash script :

Vim firewall.sh



```
#!/bin/bash

#block all incoming ssh traffic
iptables -t filter -A INPUT -p tcp --dport 22 -j DROP
#block all outbound http traffic
iptables -t filter -A OUTPUT -p tcp --dport 80 -j DROP
#block all outbound https traffic
iptables -t filter -A OUTPUT -p tcp --dport 443 -j DROP
```

Chmod 700 firewall.sh

./firewall.sh (take the script from the current working directory and run it )

Each time you execute the script , the rules are appended to the iptables

Solution : at the beginning add `iptables -F` , `iptables -t nat -F`



```
#!/bin/bash

#FLASH
iptables -F
iptables -t nat -F
iptables -t mangle -F
iptables -t raw -F # comments in bash script

#INBOUND
iptables -t filter -A INPUT -p tcp --dport 22 -j DROP
#OUTBOUND
iptables -t filter -A OUTPUT -p tcp --dport 80 -j DROP
iptables -t filter -A OUTPUT -p tcp --dport 443 -j DROP
```

## Settings the default policy :

### Default Policy

- Policy specifies what happens to packets that are not matched against any rule.
- By default Policy is set to accept all traffic.
- Policy can be changed only for INPUT, OUTPUT and FORWARD chains.
- Policy can be changed using -P option.

Policy can change the filter table

Check the policy with : *iptables -vnL*

Iptables -P FORWARD DROP

## Deleting the firewall :

Flush all the the chains

```
1 #!/bin/bash
2
3 #1. Set the ACCEPT policy
4 iptables -P INPUT ACCEPT
5 iptables -P OUTPUT ACCEPT
6 iptables -P FORWARD ACCEPT
7
8 #2. flush all tables
9 iptables -t filter -F
10 iptables -t nat -F
11 iptables -t mangle -F
12 iptables -t raw -F
13
14 #3. delete user-defined chains
15 iptables -X
```

## Filter by IP address :

## Filter by IP or Network Address

### 1. Match by Source IP or Network Address

Match: **-s** IP, **--source** IP

**Example:** `iptables -A INPUT -s 100.0.0.0/16 -j DROP`

### 2. Match by Destination IP or Network Address

Match: **-d** IP, **--destination** IP

**Example:** `iptables -A FORWARD -d 80.0.0.1 -j DROP`  
`iptables -A OUTPUT -d www.ubuntu.com -j DROP`

Using the domain name as destination , where the dns resolution will be done automatically

```
root@linux1:~/scripts# iptables -A OUTPUT -d www.ubuntu.com -j DROP
root@linux1:~/scripts# iptables -vnL
Chain INPUT (policy ACCEPT 11 packets, 916 bytes)
 pkts bytes target     prot opt in     out     source               destination
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
Chain OUTPUT (policy ACCEPT 11 packets, 916 bytes)
 pkts bytes target     prot opt in     out     source               destination
      2   168  DROP      all   --  *       *      0.0.0.0/0          91.189.90.58
      0     0  DROP      all   --  *       *      0.0.0.0/0          91.189.90.58
      0     0  DROP      all   --  *       *      0.0.0.0/0          91.189.89.110
      0     0  DROP      all   --  *       *      0.0.0.0/0          91.189.89.103
      0     0  DROP      all   --  *       *      0.0.0.0/0          91.189.89.118
      0     0  DROP      all   --  *       *      0.0.0.0/0          91.189.90.59
      0     0  DROP      all   --  *       *      0.0.0.0/0          91.189.89.115
```

For big companies with multiple IP addresses and domains , this wont work , you should use SQUID

**Any ip address with any mask :**

`iptables -A OUTPUT -p tcp --dport 443 -d 0/0 -j ACCEPT`

## Filter by ports :

### Filter by TCP or UDP port

---

#### 1. Match by a single port

Match option: `-p tcp --dport port`, `-p udp --sport port`

**Example:**

```
iptables -A INPUT -p tcp --dport 22 -j DROP
```

#### 2. Match by multiple ports

Match option: `-m multiport --sports | --dports port1,port2,...`

**Example:**

```
iptables -A OUTPUT -p tcp -m multiport --dports 80,443 -j ACCEPT
```

```
iptables -A OUTPUT -p tcp -m multiport --dports 80,443 -j DROP
```

```
iptables -A INPUT -p icmp --icmp-type echo-request -s 10.73.11.154 -j ACCEPT
```

**Check if a port is open :**

Scan it with nmap from another machine

## Intro to stateful firewall :

## Connection Tracking - 1

---

**Connection Tracking = Stateful Firewall**

- Connection tracking = ability to maintain **state information** about connections.
- Stateful firewalls are **more secure** than stateless firewalls.
- Stateful firewalls decide to accept or to drop packets **based on the relations** these packets are with other packets.
- Netfilter is a stateful firewall.

Connection tracking = stateful firewall

**Packet states:**

1. **NEW** - the first packet from a connection.
2. **ESTABLISHED** - packets that are part of an existing connection.
3. **RELATED** - packets that are requesting a new connection and are already part of an existing connection (Ex: FTP).
4. **INVALID** - packets that are not part of any existing connection.
5. **UNTRACKED** - packets marked within the raw table with the NOTRACK target.

Connection tracking can be used even if the protocol itself is stateless (Ex: UDP, ICMP).

You can use Iptables connection tracking with stateless protocols like UDP

**-m state --state state**, where state is a comma separated values of packet states written in UPPERCASE letters

**Example:**

```
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
```

**Implementing :**

Stateful\_firewall.sh

```
#!/bin/bash
#Allow loopback traffic
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -i lo -j ACCEPT

#STATEFUL
iptables -A INPUT -m state --state INVALID -j DROP
iptables -A OUTPUT -m state --state INVALID -j DROP

iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT

iptables -P INPUT DROP
IPTABLES -P OUTPUT DROP
```

List the firewall

```
iptables -vnL
```

**Allowing incoming manager ssh connection**

```
Iptables -A INPUT -p tcp --dport 22 -m state --state NEW -s 192.168.0.1 -j ACCEPT
```

## Filter by MAC address : MAC filtering

Ethernet and wireless can only be filtered from source and not destination

### Filter by MAC Address - 1

---

- It is possible to filter traffic only by source mac address.

Match: `-m mac --mac-source source_mac_address`

**Example:**

```
iptables -A INPUT -i wlan0 -m mac --mac-source 08:00:27:55:6f:20 -j DROP
```

Permit a list of mac addresses :

```
1#!/bin/bash
2
3 iptables -F INPUT
4
5 PERMITTED_MACS="08:00:27:04:61:21 07:00:27:04:61:21 09:00:27:04:61:21 08:00:27:04:61:56"
6
7
8 for MAC in $PERMITTED_MACS
9 do
0     iptables -A FORWARD -m mac --mac-source $MAC -j ACCEPT
1     echo "$MAC permitted"
2 done
3
4
5 iptables -P FORWARD DROP
```

I

FORWARD because we act like a router , and it should be -F FORWARD

Chmod 700 nat\_filter.sh

Use : if you want a LAN server to be accessed only by trusted hosts

### Filter by date and time :

Use : company intranet server should be available during weekdays or working hours

## Filter by Date and Time - 1

-m time option

Time match options:

```
--datestart time    Start and stop time, to be given in ISO 8601  
--datestop time    (YYYY[-MM[-DD[Thh[:mm[:ss]]]]])  
--timestart time   Start and stop daytime (hh:mm[:ss])  
--timestamp time   (between 00:00:00 and 23:59:59)  
--monthdays value  List of days on which to match, separated by comma  
[!] --weekdays value List of weekdays on which to match, sep. by comma  
--kerneltz         Work with the kernel timezone instead of UTC
```

Iptables -m time --help

### Task 1 : Permit ssh between 10am and 4pm or 16

```
Iptables -A INPUT -p tcp --dport 22 -m time --timestart 10:00 --timestamp 16:00 -j ACCEPT
```

```
#!/bin/bash  
iptables -F  
iptables -A INPUT -p tcp --dport 22 -m time --timestart 10:00 --timestamp 16:00 -j ACCEPT  
iptables -A INPUT -p tcp --dport 22 -j DROP
```

### Task 2 : https traffic to ubuntu after working hours

```
9 iptables -A FORWARD -p tcp --dport 443 -d www.ubuntu.com -m time --timestart 18:00 --timestamp 8:00 -j ACCEPT  
10 iptables -A FORWARD -p tcp --dport 443 -d www.ubuntu.com -j DROP
```

## The ACCEPT and DROP target :

### ACCEPT

- ACCEPT is a terminating target.

**Example:** iptables -A INPUT -p tcp --dport 25 -j ACCEPT

## DROP

- **DROP** is a terminating target
- DROP denies the packet and doesn't send any packet back to the source

**Example:** `iptables -I OUTPUT -p udp --dport 53 ! -d 8.8.8.8 -j DROP`

Being a terminating target, the packet will not be evaluated by the second rule

## The LOG target :

### LOG TARGET

- **LOG** is a non-terminating target.
- It logs detailed information about packet headers.
- Logs can be read with dmesg or from syslogd daemon.
- LOG is used instead of DROP in the debugging phase.
- ULOG has MySql support (extensive logging)

#### Options:

`--log-prefix`  
`--log-level`

**Example:** `iptables -A INPUT -p tcp --dport 22 --syn -j LOG --log-prefix="incoming ssh:" \`  
`--log-level info`

Log prefix tells the iptables to prefix all log messages with a specific prefix that can be used with grep

Log level , for importance

Continues to traverse the chain : non terminating

```
root@linux1:~/scripts# iptables -A INPUT -p tcp --dport 22 --syn -j LOG --log-prefix="incoming ssh traffic:" --log-level info
root@linux1:~/scripts# iptables -A INPUT -p tcp --dport 22 -j DROP
```

Dmessg

```
[ 409.068290] Bluetooth: L2CAP socket layer initialized
[ 409.068293] Bluetooth: SCO socket layer initialized
[ 409.072539] Netfilter messages via NETLINK v0.30.
[ 414.523161] device enp0s3 entered promiscuous mode
[ 433.005119] device enp0s3 left promiscuous mode
[ 440.842130] device enp0s3 entered promiscuous mode
[ 450.965033] device enp0s3 left promiscuous mode
[ 468.985942] ip_tables: (C) 2000-2006 Netfilter Core Team
[ 495.093231] nf_conntrack version 0.5.0 (16384 buckets, 65536 max)
[ 4448.842887] device enp0s3 entered promiscuous mode
[ 4485.603228] device enp0s3 left promiscuous mode
[ 4771.731019] device enp0s3 entered promiscuous mode
[ 4800.648072] device enp0s3 left promiscuous mode
[ 4907.328185] device enp0s3 entered promiscuous mode
[ 4952.125421] device enp0s3 left promiscuous mode
[ 5327.991505] incoming ssh traffic:IN=enp0s3 OUT= MAC=08:00:27:12:ae:64:08:00:27:04:61:21:08:00 SRC
=192.168.0.20 DST=192.168.0.10 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=55780 DF PROTO=TCP SPT=41370 DPT=
22 WINDOW=29200 RES=0x00 SYN URGP=0
[ 5329.023144] incoming ssh traffic:IN=enp0s3 OUT= MAC=08:00:27:12:ae:64:08:00:27:04:61:21:08:00 SRC
=192.168.0.20 DST=192.168.0.10 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=55781 DF PROTO=TCP SPT=41370 DPT=
22 WINDOW=29200 RES=0x00 SYN URGP=0
[ 5331.038632] incoming ssh traffic:IN=enp0s3 OUT= MAC=08:00:27:12:ae:64:08:00:27:04:61:21:08:00 SRC
=192.168.0.20 DST=192.168.0.10 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=55782 DF PROTO=TCP SPT=41370 DPT=
22 WINDOW=29200 RES=0x00 SYN URGP=0
[ 5335.134330] incoming ssh traffic:IN=enp0s3 OUT= MAC=08:00:27:12:ae:64:08:00:27:04:61:21:08:00 SRC
=192.168.0.20 DST=192.168.0.10 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=55783 DF PROTO=TCP SPT=41370 DPT=
22 WINDOW=29200 RES=0x00 SYN URGP=0
```

Dmesg | grep: "incoming .."

Dmesg only in the buffer memory , after reboot you should use the kern.log

```
root@linux1:~/scripts# dmesg | grep "ssh traffic" > ssh.txt
```

```
root@linux1:~/scripts# tail -f /var/log/kern.log
Dec 13 14:50:45 linux1 kernel: [ 4448.842887] device enp0s3 e
```

## Project 3 : Setting Up a Web and DNS Server

LAMP : linux , apache , mysql , php

### Getting a domain name :

Computers communicate using IP addresses ,not domain names , that's where dns resolution comes in

Log in to ubuntu.com

| No.  | Time           | Source    | Destination    | Protocol | Length | Time to Live  | Info |
|------|----------------|-----------|----------------|----------|--------|---|------|
| 3415 | 187.167637752  | 10.6.2.15 | 185.125.190.20 | TLSv1.3  | 134    | 64 Change Cipher Spec, Application Data                                       |      |
| 3416 | 187.1676391259 | 10.6.2.15 | 185.125.190.20 | TLSv1.3  | 224    | 64 Application Data   |      |
| 3419 | 187.205032455  | 10.6.2.15 | 185.125.190.20 | TCP      | 54     | 64 0x1400 -> 443 [ACK] Seq=914 Ack=568 Win=63990 Len=0                        |      |
| 3420 | 187.226382647  | 10.6.2.15 | 185.125.190.20 | TLSv1.3  | 85     | 64 Application Data   |      |
| 3468 | 198.161938940  | 10.6.2.15 | 185.125.190.20 | TCP      | 74     | 64 0x1400 -> 443 [SYN] Seq=0 Win=64240 Len=8 MSS=1468 SACK_PERM TSecr=0 Ws=32 |      |
| 3473 | 188.194829827  | 10.6.2.15 | 185.125.190.20 | TCP      | 54     | 64 0x1400 -> 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0                            |      |
| 3477 | 188.210360212  | 10.6.2.15 | 185.125.190.20 | TLSv1.3  | 571    | 64 Client Hello   |      |
| 3485 | 188.247187379  | 10.6.2.15 | 185.125.190.20 | TCP      | 54     | 64 0x1400 -> 443 [ACK] Seq=511 Ack=2921 Win=62780 Len=0                       |      |
| 3488 | 188.249191800  | 10.6.2.15 | 185.125.190.20 | TCP      | 54     | 64 0x1400 -> 443 [ACK] Seq=511 Ack=4381 Win=61320 Len=0                       |      |
| 3489 | 188.250191800  | 10.6.2.15 | 185.125.190.20 | TCP      | 54     | 64 0x1400 -> 443 [ACK] Seq=518 Ack=4532 Win=61320 Len=0                       |      |
| 3506 | 188.3108074240 | 10.6.2.15 | 185.125.190.20 | TLSv1.3  | 134    | 64 Change Cipher Spec, Application Data                                       |      |
| 3506 | 188.311254247  | 10.6.2.15 | 185.125.190.20 | TLSv1.3  | 224    | 64 Application Data   |      |
| 3511 | 188.35151533   | 10.6.2.15 | 185.125.190.20 | TCP      | 54     | 64 0x1400 -> 443 [ACK] Seq=768 Ack=5120 Win=62780 Len=0                       |      |
| 3512 | 188.354910259  | 10.6.2.15 | 185.125.190.20 | TLSv1.3  | 85     | 64 Application Data   |      |
| 3951 | 192.834763884  | 10.6.2.15 | 185.125.190.20 | TLSv1.3  | 988    | 64 Application Data   |      |

```
(root@kali)-[~]# nslookup ubuntu.com
Server:          10.128.128.128
Address:        10.128.128.128#53

Non-authoritative answer:
Name:  ubuntu.com
Address: 185.125.190.20
Name:  ubuntu.com
Address: 185.125.190.29
Name:  ubuntu.com
Address: 185.125.190.21
Name:  ubuntu.com
Address: 2620:2d:4000:1::28
Name:  ubuntu.com
Address: 2620:2d:4000:1::27
Name:  ubuntu.com
Address: 2620:2d:4000:1::26
```

### Buy a domain name :

Buy it from go daddy and namecheap

My domain : taibkho.live

Freenom.com

### Next step : run a web server on that domain :

the domain will be host on a dns server (authoritative dns server) , all the details about the domain such as the subdomains will be configured on that server

When I user tries to search for a domain name , the dns server of the user or the IPS will be served and if it doesn't have it , it will look for the authoritative dns server

-We have to indicate to the register where did we get the domain from , the name and the ip of the authoritative that will host our domain

- We will run the authoritative dns (ns1) server on a ubuntu vps (AWS) , we will use its IP for the register

Freenom.com

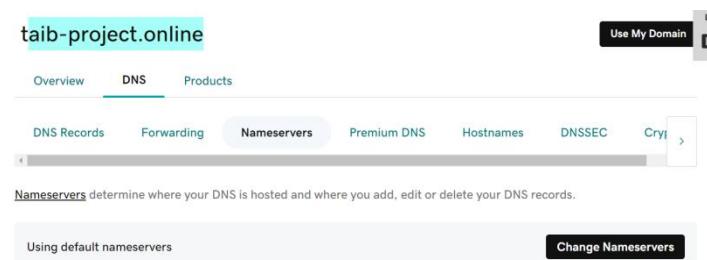
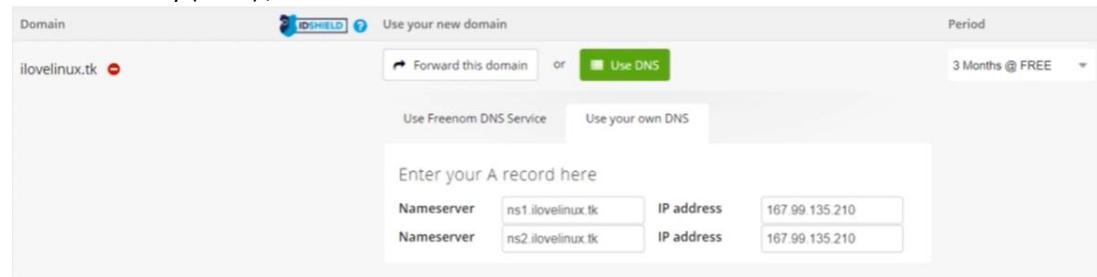
Recommended to use two nameservers for one domain

- Recursive DNS servers resolve DNS queries for clients by recursively querying authoritative DNS servers until they obtain the final answer.
- Authoritative DNS servers provide the authoritative information for specific domains. They hold the DNS records and respond to queries with the authoritative information.

taib-project.online

Ns1 : primary master , ns07.domaincontrol.com

Ns2 : secondary (slave) , ns08.domaincontrol.com



The ip is from the ubuntu host

```

└─[root@kali]~]# dig -t ns taib-project.online
; <>> DiG 9.18.16-1-Debian <>> -t ns taib-project.online
;; global options: +cmd
;; Got answer:
;; →HEADER← opcode: QUERY, status: NOERROR, id: 63537
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;taib-project.online. IN NS
;; ANSWER SECTION:
taib-project.online. 3600 IN NS ns07.domaincontrol.com.
taib-project.online. 3600 IN NS ns08.domaincontrol.com.

;; Query time: 71 msec
;; SERVER: 10.128.128.128#53(10.128.128.128) (UDP)
;; WHEN: Sun Feb 25 13:04:53 EST 2024
;; MSG SIZE rcvd: 103

```

The website on a web server

## Diving into the DNS protocol and installing a DNS server

**(Bind9) :**

**Install bind9 dns server :**

```
apt install bind9 bind9utils bind9-doc
```

```
systemctl status bind9
systemctl status named
```

**Setting bind9 to ipv4 :**

```
Vim /etc/default/named
```

```
"# run resolvconf?
RESOLVCONF=no

# startup options for the server
OPTIONS="-u bind -4"
~
```

```
systemctl reload-or-restart bind9
```

**Test the local dns server :**

```
dig -t a @localhost google.com
(sending the dns query to the dns server running at the local host )
```

You can use @1.1.1.1 for cloudflare

### Configure bind server :

The daemon running

```
root@ubuntu-s-1vcpu-2gb-fra1-01:~# ps -ef | grep named
bind      1696      1  0 08:32 ?    00:00:00 /usr/sbin/named -f -u bind
root     2059  1051  0 08:37 pts/0    00:00:00 grep --color=auto named
```

You have to configure three files :

```
cat /etc/bind/named.conf
```

```
include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
```

### Dns queries :

Recursive : do all the work

Iterative : it will refer you to other dns servers , and you have to query them

## DNS Queries

### 1. Recursive

- A recursive query is a kind of query, in which the DNS server, that received your query, will do all the job, fetching the answer, and giving it back to you. In the end, you'll get the final answer.

### 2. Iterative

- The DNS name server will not go and fetch the complete answer for your query but will give back a referral to other DNS servers, which might have the answer. Now it's your job to query those servers and find the answer.

## DNS Forwarder

A **forwarder** is another DNS server that will be queried recursively by our server.

A DNS server, configured to use a forwarder, behaves as follows:

- When the DNS server receives a query, it attempts to resolve this query.
- If the query cannot be resolved using local data, the DNS server forwards the query recursively to the DNS server that is designated as a forwarder.
- If the forwarder is not unavailable, the DNS server attempts to resolve the query by itself, using iterative queries.

### Setting a forwarder :

Setting public google dns server as forwarder , if the server doesn't know , it will send a recursive query to the forwarder and wait for final answer and send it back to the client

```
vim /etc/bind/named.conf.options
```

```
listen-on-v6 { any; };
forwarders {
    8.8.8.8;
    8.8.4.4;
};

};
```

## Setting up the authoritative bind9 dns server :

**Check the autho of a domain :**

```
dig -t ns google.com
```

**The ip of the nameserver:**

```
dig -t a ns1.google.com
```

**Check the status of the local dns server :**

```
Systemctl status bind9 or nameid
```

Check if its running

This master dns server holds the master copy of the zone file that contains dns records

**Add another domain for which this server is authoritative :**

```
vim /etc/bind/named.conf.local
```

```
; organization
//include "/etc/bind/zones.rfc1918";
```

```
zone "crystalmind.academy" {
    type master;
    file "/etc/bind/db.crystalmind.academy";
};
```

Create a new zone

type master : master zone --> master autho for the domain

File : where we add dns records for the domain

**Create a zone file :**

Using an existing template

```
cp /etc/bind/db.empty /etc/bind/db.crystalmind.academy
```

```
vim /etc/bind/db.crystalmind.academy
```

```
$ BIND reverse data file for empty rfc1918 zone
;
; DO NOT EDIT THIS FILE - it is used for multiple zones.
; Instead, copy it, edit named.conf, and use that copy.
;
$TTL 86400
@ IN SOA localhost. root.localhost. (
        1           ; Serial
        604800       ; Refresh
        86400        ; Retry
        2419200      ; Expire
        86400 )       ; Negative Cache TTL
;
@ IN NS localhost.
```

\$ : directive

@ : dns records

\$ttl : time a dns record can be cashed in another dns server , usually for non -autho

@soa : start of authority

@ns : name server record , it indicates the master and the slaves for this domain

IN : dns class , internet

Alteration :

@soa : we add the master ending with . , that indicates its fully qualified domain name

@ns : nameserver

@ = crystalmind.academy

A : ip address of the server

MX 10 : 10 is the priority

www, mail , external : subdomains

```
; BIND reverse data file for empty rfc1918 zone
;
; DO NOT EDIT THIS FILE - it is used for multiple zones.
; Instead, copy it, edit named.conf, and use that copy.
;
$TTL    86400
@       IN      SOA     ns1.crystalmind.academy. root.localhost. (
                      1           ; Serial
                      604800      ; Refresh
                      86400       ; Retry
                     2419200     ; Expire
                      86400 )     ; Negative Cache TTL
;
@       IN      NS      ns1.crystalmind.academy.
;@      IN      NS      ns2.crystalmind.academy.
ns1      IN      A       167.99.135.210  I
mail     IN      MX 10   mail.crystalmind.academy.
crystalmind.academy.          IN      A       167.99.135.210
www      IN      A       167.99.135.210
mail     IN      A       167.99.135.210
external  IN      A       91.189.88.181
```

Check the conf files :

named-checkconf

named-checkzone crystalmind.academy /etc/bind/db.crystalmind.academy

If they're errors the zone wont be loaded

systemctl restart bind9

Systemctl status bind9 will show us the loaded zone

```
Active: active (running) since Fri 2020-10-30 09:13:54 UTC; 9s ago
  Docs: man:named(8)
 Main PID: 2481 (named)
   Tasks: 5 (limit: 2345)
  Memory: 11.8M
    CGroup: /system.slice/named.service
             └─2481 /usr/sbin/named -f -u bind -4

Oct 30 09:13:54 ubuntu-s-1vcpu-2gb-fra1-01 named[2481]: managed-keys-zone: loaded serial 4
Oct 30 09:13:54 ubuntu-s-1vcpu-2gb-fra1-01 named[2481]: zone crystalmind.academy/IN: loaded serial 1
```

The **loaded serial number** inform the seconday servers that the conf file have been changes and the SOA , if the loaded serial stays the same it tells the secondary servers to ignore the conf file

```
$TTL      86400
@       IN      SOA      ns1.crystalmind.academy. root.localhost. (
                           2 ; Serial
```

#### Test the dns server after configurations :

```
dig @localhost -t ns crystalmind.academy
```

```
dig @localhost -t ns www.crystalmind.academy
```

```
;; ANSWER SECTION:
crystalmind.academy.    86400   IN      NS      ns1.crystalmind.academy.

;; ADDITIONAL SECTION:
ns1.crystalmind.academy. 86400   IN      A       167.99.135.210

;; <>> DiG 9.16.1-Ubuntu <>> @localhost -t a www.crystalmind.academy
;; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 4222
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 4096
;; COOKIE: e93071be5f36c60c010000005f9bd9d46c0bdc66eb3ab358 (good)
;; QUESTION SECTION:
;www.crystalmind.academy.      IN      A

;; ANSWER SECTION:
www.crystalmind.academy. 86400   IN      A       167.99.135.210
```

```
student@ubuntu20-04:~$ dig -t a public-dns.crystalmind.academy
```

```
external          IN      A      91.189.88.181
public-dns        IN      A      8.8.8.8
```

#### Installing a web server on the authoritative server : apache2

In the same vm :

```
Apt update && apt install apache2
```

```
Systemctl status apache2
```

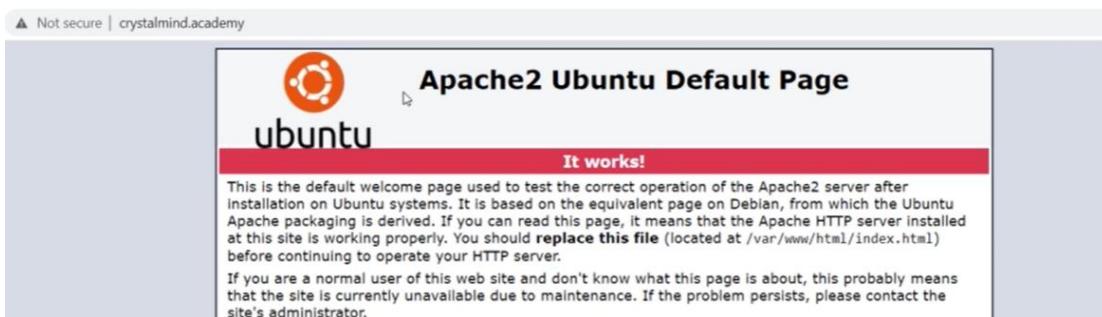
```
Sytemctl is-enabled apache2
```

```
Ufw status
```

```
Iptables -A INPUT -p tcp --d-port 80 -j ACCEPT
```

If we go to the domain

Crystalmindacademy.com



**Connect from the internet :**

Public ip : curl -4 ident.me

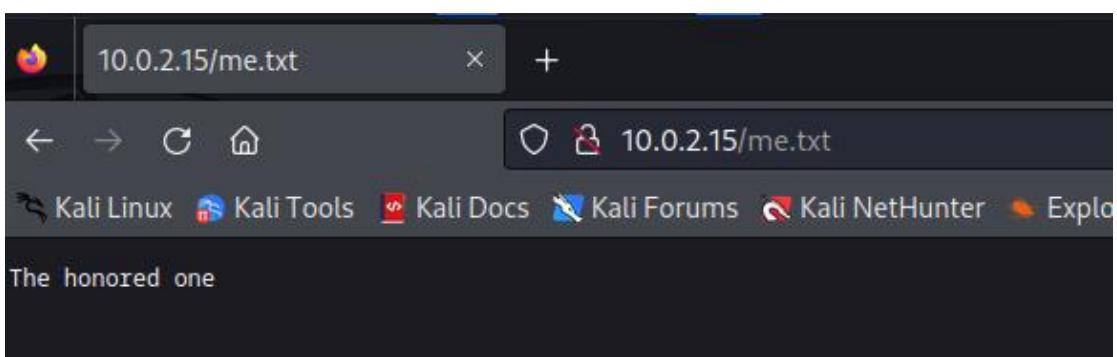
## Setting up virtual hosting :

Host more than one domain in an apache server that has a single ip address

Configure virtual hosting , apache serves files to the client form directory called document root

cd /var/www/html/

If we add a file in this directory , apache will serve It



**Create a file for each domain :**

mkdir /var/www/crystalmind.academy

**Set the ownership of the file to the user running apache :**

ps -ef | grep apache2

```
www-data    3508    3506  0  09:40 ?        00:00:00 /usr/sbin/apache2 -k start
www-data    3509    3506  0  09:40 ?        00:00:00 /usr/sbin/apache2 -k start
```

The user is www-data

chown -R www-data www-data /var/www/crystalmind.academy

This is necessary if you want the web server to have full control over the directory

```
chmod 755 /var/www/crystalmind.academy
```

**Create an index file for that particular domain :**

```
vim /var/www/crystalmind.academy/index.html
```

**Configure apache to serve the domain content when the domain is used :**

Create new virtual host :

```
cd /etc/apache2
```

```
ls sites-available
```

```
vim sites-available/crystalmind_academy.conf
```

```
{
```

If a user access the domain using Servername or alias then give them content of the document root

Optional : email address for the server admin

Custom log files

```
}
```

---

```
<VirtualHost *:80>
    ServerName crystalmind.academy
    ServerAlias www.crystalmind.academy
    DocumentRoot /var/www/crystalmind.academy

    ServerAdmin webmaster@crystalmind.academy
    ErrorLog /var/log/apache2/crystalmind_academy_error.log
    CustomLog /var/log/apache2/crystalmind_academy_access.log combined

</VirtualHost>
~
```

**Enable the created virtual host :**

```
a2ensite crystalmind_academy (without the conf)
```

```
root@ubuntu-s-1vcpu-2gb-fra1-01:/etc/apache2# ls sites-available/
000-default.conf  crystalmind_academy.conf  default-ssl.conf
root@ubuntu-s-1vcpu-2gb-fra1-01:/etc/apache2#
```

Apache will load from sites-enabled

that's why a2ensite created an asimling

```
root@ubuntu-s-1vcpu-2gb-fra1-01:/etc/apache2# ls -l sites-enabled/
total 0
lrwxrwxrwx 1 root root 35 Oct 30 09:40 000-default.conf -> ../sites-available/000-default.conf
lrwxrwxrwx 1 root root 43 Oct 30 09:59 crystalmind.academy.conf -> ../sites-available/crystalmind.academy.conf
```

Remove the asimlimg : a2dissite 000-default

```
Systemctl reload apache2
```

# **Securing apache2 with openssl and digital certificates :**

http is not encrypted , https should be enabled (over SSL/TLS)

Getting a digital certificate from CA = letsencrypt.org

Configuring the apache2 to use the certificate and encrypt the traffic using openssl

## **Installing Certbot :**

```
apt update && apt install certbot python3-certbot-apache
```

## **Request from the CA to issue a certificate to my domain :**

```
certbot -d crystalmind.academy
```

The name of the domain should be the same of the one bellow in sites-available

```
Last login: Fri Oct 30 09:09:00 2020 from 100.20.72.107
root@ubuntu-s-1vcpu-2gb-fra1-01:~# cat /etc/apache2/sites-available/crystalmind.academy.conf
<VirtualHost *:80>
    ServerName crystalmind.academy
```

If you havent set up a mail server for the domain , just use a random email

A

Y

2 redirect to https

## **If there is an error :**

```
cat /var/log/letsencrypt/letsencrypt.log
```

## **Conclusion :**

Certbot requested the certificate , create a new virtual host file for apache2 to use https , loaded the certificate to apache and reload it

Virtual host file for https domain

```
root@ubuntu-s-1vcpu-2gb-fra1-01:/etc/apache2# ls -l sites-available/
total 20
-rw-r--r-- 1 root root 1332 Apr 13 2020 000-default.conf
-rw-r--r-- 1 root root 552 Oct 30 10:12 crystalmind.academy-le-ssl.conf
```

Openssl

```
SSLCertificateFile /etc/letsencrypt/live/crystalmind.academy/fullchain.pem
SSLCertificateKeyFile /etc/letsencrypt/live/crystalmind.academy/privkey.pem
Include /etc/letsencrypt/options-ssl-apache.conf
```

## **Certbot auto renewal of certificate :**

```
systemctl status certbot.timer
```

```
certbot renew --dry-run

-----  
** DRY RUN: simulating 'certbot renew' close to cert expiry  
**          (The test certificates below have not been saved.)  
  
Congratulations, all renewals succeeded! The following certs have been renewed:  
-----
```

#### Test the domain :

Test redirecting : <http://crystalmind.academy> , should redirect to https  
https : the lock and certificate info , CA issuer and validity (connection is secure )

## Access control by source ip address : (white list)

Its like setting up a dmz , the public facing directories wont require access control , but admin directories would need identification

Lets set up the admin directory

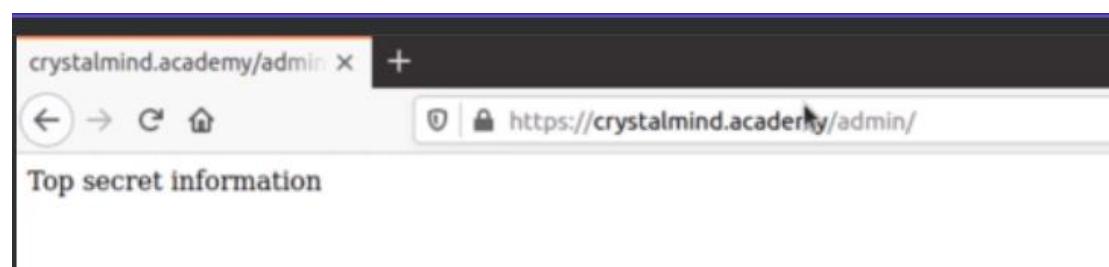
```
[root@kali]~-[var/www/html/crystal.academy]  
# cd admin  
  
[root@kali]~-[var/www/html/crystal.academy/admin]  
# ls
```

Then we create or edit the index.html file , this will be displayed when we access the server

#### Create the index.html :

```
echo "this is top secret" > index.html
```

```
crystalmind.academy/admin/
```



#### Adding the required directives :

#### Changing the virtual host file in the ssl enabled site

```
vim /etc/apache2/sites-available/crystalmind.academy-le-ssl.conf
```

The directive in closed directory section applied to that named directory and its subs , as well as the files in those directories (.htaccess file )

```
<IfModule mod_ssl.c>
<VirtualHost *:443>
    ServerName crystalmind.academy
    ServerAlias www.crystalmind.academy
    DocumentRoot /var/www/crystalmind.academy
    ErrorLog ${APACHE_LOG_DIR}/crystalmind_academy_error.log
    CustomLog ${APACHE_LOG_DIR}/crystalmind_academy_access.log combined

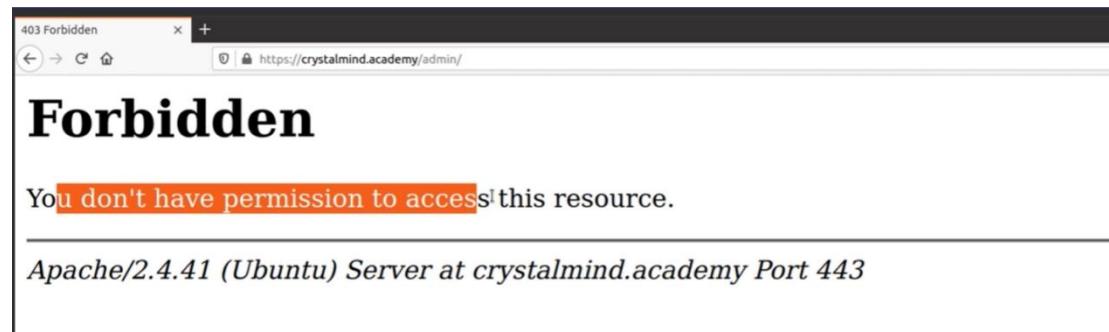
    SSLCertificateFile /etc/letsencrypt/live/crystalmind.academy/fullchain.pem
    SSLCertificateKeyFile /etc/letsencrypt/live/crystalmind.academy/privkey.pem
    Include /etc/letsencrypt/options-ssl-apache.conf

<Directory /var/www/crystalmind.academy/admin>
    Require all denied
    Require ip 6.7.73.2
</Directory>

</VirtualHost>
</IfModule>
..
```

Deny access to everyone except the one with that ip address

Systemctl restart apache2



**Adding my public ip address to the allowed ip addresses :**

Public ip : in google what is my public ip

curl ident.me

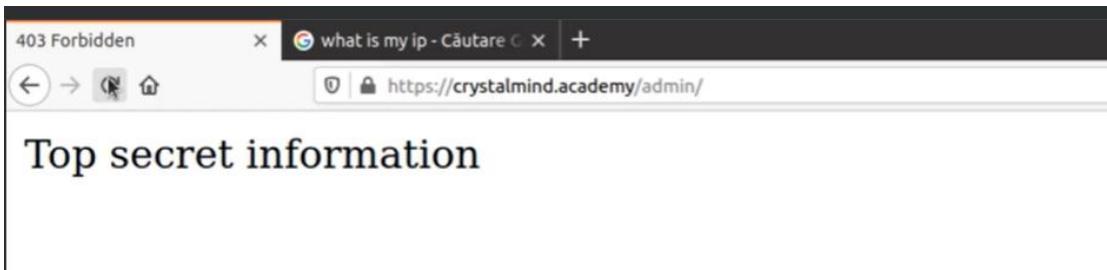
Remember you got the ip of the server , not the client

Adding the client ip to the allowed ip addresses

*Require ip "client\_ip"*

```
<Directory /var/www/crystalmind.academy/admin>
    Require all denied
    Require ip 6.7.73.2
    Require ip 86.121.14.97
</Directory>
```

Restart apache



Require local can work as well

```
<Directory /var/www/crystalmind.academy/admin>
    Require all denied
    Require ip 6.7.73.2
    Require ip 86.121.14.97
    Require ip 10.114.0.2/20
    Require local
</Directory>
```

## All the directives recommendations :

Certainly! When configuring Apache HTTP Server for security, there are several directives you can use to enhance the security of your web server. Here are some common directives along with recommendations for secure settings:

### 1. \*\*Listen:\*\*

- Specify the IP address and port to listen on. Limit to specific IP addresses if possible.

```
``apache
Listen 192.168.1.2:443
``
```

### 2. \*\*ServerAdmin:\*\*

- Set a valid email address for the server administrator.

```
``apache
ServerAdmin webmaster@example.com
``
```

### 3. \*\*ServerTokens:\*\*

- Set `ServerTokens` to `Prod` to minimize the information disclosed in the server version.

```
``apache
ServerTokens Prod
``
```

### 4. \*\*ServerSignature:\*\*

- Disable `ServerSignature` to prevent the server version information from being included in error pages.

```
```apache
ServerSignature Off
````
```

#### 5. \*\*DirectoryIndex:\*\*

- Only include necessary default files in `DirectoryIndex`.

```
```apache
DirectoryIndex index.html
````
```

#### 6. \*\*AccessFileName:\*\*

- Specify a secure name for the access control file.

```
```apache
AccessFileName .htaccess
````
```

#### 7. \*\*AllowOverride:\*\*

- Limit the directives that can be overridden in `.htaccess` files. Avoid using `All` unless necessary.

```
```apache
AllowOverride None
````
```

#### 8. \*\*Options:\*\*

- Be restrictive with `Options` to reduce the attack surface.

```
```apache
Options FollowSymLinks
````
```

#### 9. \*\*Alias:\*\*

- Use `Alias` cautiously, and ensure proper access controls.

```
```apache
Alias /images/ /var/www/images/
````
```

#### 10. \*\*Redirect:\*\*

- Use `Redirect` carefully, and avoid exposing sensitive paths.

```
```apache
Redirect /old-page http://example.com/new-page
````
```

#### 11. \*\*ProxyPass:\*\*

- Apply proper access controls and consider limiting access to backend servers.

```
```apache
ProxyPass /app http://backend-server/
````
```

#### 12. \*\*RewriteEngine:\*\*

- Use `RewriteEngine On` cautiously, and sanitize rewrite rules to prevent vulnerabilities.

```
```apache
RewriteEngine On
````
```

Remember that security is a multi-faceted aspect, and these directives are just a part of securing a web server. Additionally, regularly update your server software, apply security patches, configure firewalls, and use SSL/TLS for encrypted communication. Regularly review and audit your server configuration for security best practices. Always refer to the [official Apache HTTP Server documentation](https://httpd.apache.org/docs/2.4/) for the most accurate and up-to-date information on configuration directives.

## The <Files> directive :

Access control to a file

```
<Files "report.txt">
    Require all denied
    Require ip 3.4.56.6
</Files>
```

Wildcard :

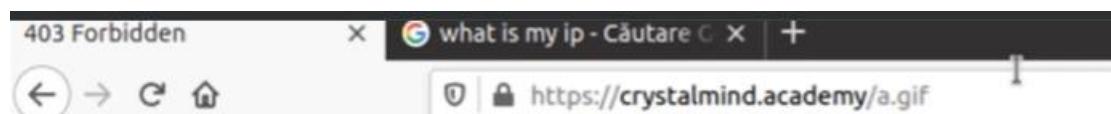
- ? - any single character
- \* - any sequence of character

Match any string , that end in .gif , .jpg , .png  
\ : cancel the special meaning of . , . can match anything  
Not : deny

```
<Files ~ "\.(gif|jp?g|png)$">
    <RequireAll>
        Require all granted
        Require not ip 86.121.14.97
    </RequireAll>
</Files>
```

Test an image :

```
root@ubuntu-s-1vcpu-1gb-fra1-01:~# cp /usr/share/apache2/icons/a.gif /var/www/crystalmind.academy/
root@ubuntu-s-1vcpu-1gb-fra1-01:~#
```



## The .htaccess file :

### The Apache Web Server

- **.htaccess** contains directives that will be applied to the directory in which it resides (and to all subdirectories and files)
- Directives can be enclosed in a **<Directory>** section or put in a **.htaccess** file
- The **AllowOverride** directive specifies, in categories, what directives will be honored if they are found in a **.htaccess** file
- You should only use **.htaccess** files when you don't have access to the main server configuration file.
- **Prefer using <Directory> ... </Directory> to .htaccess**

Only use this file when you don't have access to the conf file of the server

Allowoverride : allow the use of **.htaccess**  
the **allowoverride** only allowed in the **directory** section

```
<Directory /var/www/crystalmind.academy>
    AllowOverride All
</Directory>
```

Restart apache

### Creating the .htaccess :

Mkdir /var/www/crystalmind.academy/documents

```
root@ubuntu-s-1vcpu-1gb-fra1-01:~# mkdir /var/www/crystalmind.academy/documents
root@ubuntu-s-1vcpu-1gb-fra1-01:~# vim /var/www/crystalmind.academy/documents/.htaccess
```

Allow specific ip address to access the conf files :

```
<Files "*.*">
    Require all denied
    Require ip 86.121.14.97
</Files>
~
```

Test the access to the files :

Creating random files

```

root@ubuntu-s-1vcpu-1gb-fra1-01:~# ps -ef > /var/www/crystalmind.academy/documents/a.txt
root@ubuntu-s-1vcpu-1gb-fra1-01:~# ps -ef > /var/www/crystalmind.academy/documents/b.conf
root@ubuntu-s-1vcpu-1gb-fra1-01:~# ps -ef > /var/www/crystalmind.academy/documents/c.log
root@ubuntu-s-1vcpu-1gb-fra1-01:~# ps -ef > /var/www/crystalmind.academy/documents/d.conf
root@ubuntu-s-1vcpu-1gb-fra1-01:~# ls -l /var/www/crystalmind.academy/documents/
total 32
-rw-r--r-- 1 root root 7378 Jan 22 11:10 a.txt
-rw-r--r-- 1 root root 7378 Jan 22 11:10 b.conf
-rw-r--r-- 1 root root 7378 Jan 22 11:10 c.log
-rw-r--r-- 1 root root 7386 Jan 22 11:10 d.conf

```

Allowed from my ip address

crystalmind.academy/documents/

## Index of /documents

| <u>Name</u>                      | <u>Last modified</u> | <u>Size</u> | <u>Description</u> |
|----------------------------------|----------------------|-------------|--------------------|
| <a href="#">Parent Directory</a> |                      | -           |                    |
| <a href="#">a.txt</a>            | 2021-01-22 11:10     | 7.2K        |                    |
| <a href="#">b.conf</a>           | 2021-01-22 11:10     | 7.2K        |                    |
| <a href="#">c.log</a>            | 2021-01-22 11:10     | 7.2K        |                    |
| <a href="#">d.conf</a>           | 2021-01-22 11:10     | 7.2K        |                    |

After removing my ip address

|                                  |                       |
|----------------------------------|-----------------------|
| <a href="#">Parent Directory</a> | -                     |
| <a href="#">a.txt</a>            | 2021-01-22 11:10 7.2K |
| <a href="#">c.log</a>            | 2021-01-22 11:10 7.2K |

*Apache/2.4.41 (Ubuntu) Server at crystalmind.academy Port 443*

.htaccess update in real time , you don't need to restart the server

However .htaccess should only be used if we don't have root access , because it slows down the server

## HTTP digest authentication :

Restrict access to backend services

[https://httpd.apache.org/docs/2.4/mod/mod\\_auth\\_digest.html](https://httpd.apache.org/docs/2.4/mod/mod_auth_digest.html)

So when the user tries to access a forbidden directory , it will be prompt with a window for authentication

apt install apache2-utils

**Generating the password file :**

```
htdigest -c /etc/apache2/.htpasswd myserver johnny
```

WARNING : realm domain must match the AuthName in Apache config .

**Protect the admin directories with the password :**

Using the root conf file , or the .htaccess file

We gonna use the root conf file

```
vim /etc/apache2/sites-available/crystalmind.academy-le-ssl.conf
```

```
SSLCertificateFile /etc/letsencrypt/live/crystalmind.academy/fullchain.pem
SSLCertificateKeyFile /etc/letsencrypt/live/crystalmind.academy/privkey.pem
Include /etc/letsencrypt/options-ssl-apache.conf

<Directory /var/www/crystalmind.academy/admin>
    Require all denied
    Require ip 6.7.73.2
    Require ip 86.121.14.97
    Require ip 10.114.0.2/20
    Require local
</Directory>

<Directory /var/www/crystalmind.academy/protected>
    AuthType Digest
    AuthName "myserver"
    AuthDigestProvider file
    AuthUserFile /etc/apache2/.htpasswd
    Require valid-user
</Directory>

<Directory /var/www/crystalmind.academy>
    I
    AllowOverride All
</Directory>
```

... 10 ...

**Enable the Authdigest module that provides the authdigest to apache :**

```
a2enmod auth_digest
```

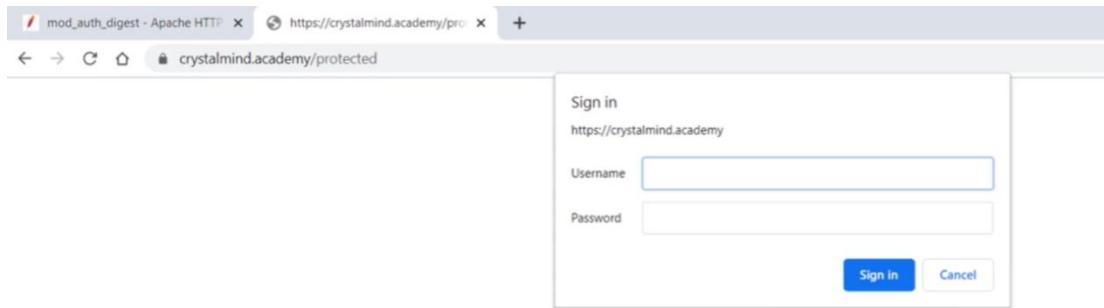
Restart apache

**Test the configuration :**

```
mkdir /var/www/crystalmind.academy/protected
```

```
Echo "protected files " > /protected/index.html
```

Try to access it through the browser



## The options directive and indexing :

### The Options Directive

The **Options** directive controls which server features are available in a particular directory.

**Values:**

- None
- All
- FollowSymLinks
- Indexes

**Example:**

<Directory>

```
Options Indexes FollowSymLinks
DirectoryIndex index.html index.php
IndexOptions FancyIndexing
```

⋮</Directory>

### Create a symlinks on the doc root directory :

In -s /etc/ /var/www/crystalmind.academy/link\_to\_etc

```
root@ubuntu-s-1vcpu-1gb-fra1-01:~# ln -s /etc/ /var/www/crystalmind.academy/link_to_etc
root@ubuntu-s-1vcpu-1gb-fra1-01:~# ls -l /var/www/crystalmind.academy/
total 24
-rw-r--r-- 1 root root 246 Jan 21 11:10 a.gif
drwxr-xr-x 2 root root 4096 Jan 21 10:43 admin
drwxr-xr-x 2 root root 4096 Jan 22 13:33 documents
-rw-r--r-- 1 root root 63 Jan 20 14:43 index.html
lrwxrwxrwx 1 root root 5 Jan 28 09:28 link_to_etc -> /etc/
drwxr-xr-x 2 root root 4096 Jan 26 08:22 protected
-rw-r--r-- 1 root root 14 Jan 21 11:00 report.txt
```

### Access the symlink through the browser :

The server would follow the symlink

← → C ⌂ 🔒 crystalmind.academy/link\_to\_etc/

## Index of /link\_to\_etc

| Name   | Last modified    | Size | Description |
|--|------------------|------|-------------|
|  Parent Directory |                  | -    |             |
|  NetworkManager/  | 2020-10-16 21:42 | -    |             |
|  PackageKit/      | 2020-10-16 21:43 | -    |             |
|  X11/             | 2020-10-16 21:41 | -    |             |

This is a security issue because the etc files would be indexed by google and can be discovered by google dorks

<https://www.exploit-db.com/google-hacking-database>

**Disabling symlinks in the root directory by using indexes :**

```
<Directory /var/www/crystalmind.academy>
    AllowOverride All
    Options Indexes
</Directory>
```

The options of indexes will overwrite the default symlink option

Only the indexes options is available on this directory

← → C ⌂ 🔒 crystalmind.academy/link\_to\_etc/

## Forbidden

You don't have permission to access this resource.

*Apache/2.4.41 (Ubuntu) OpenSSL/1.1.1f Server at crystalmind.academy Port 443*

**More on indexes :**

When a user visit a the website , and want to access a directory , the apache server will look for the index.html file , if the directory index page don't exist , then the server will depend on the "indexes" option , if the options is set for that directory , it will display the content , if it doesn't it will display forbidden .

```
root@ubuntu-s-1vcpu-1gb-fra1-01:~# mkdir /var/www/crystalmind.academy/dir1
root@ubuntu-s-1vcpu-1gb-fra1-01:~# who > /var/www/crystalmind.academy/dir1/a.txt
root@ubuntu-s-1vcpu-1gb-fra1-01:~# ls /etc/ > /var/www/crystalmind.academy/dir1/b.txt
root@ubuntu-s-1vcpu-1gb-fra1-01:~# ping -c 1 8.8.8.8 > /var/www/crystalmind.academy/dir1/c.txt
root@ubuntu-s-1vcpu-1gb-fra1-01:~# ls -l /var/www/crystalmind.academy/dir1/
total 12
-rw-r--r-- 1 root root 54 Jan 28 09:39 a.txt
-rw-r--r-- 1 root root 1882 Jan 28 09:39 b.txt
-rw-r--r-- 1 root root 243 Jan 28 09:39 c.txt
```

Dir1 don't have an index.html but the it will still be displayed by the server because the "Index" options is set on the document root directory

To disable the index option :

```
<Directory /var/www/crystalmind.academy>
    AllowOverride All
    Options -Indexes
</Directory>
```

Cant mix options with + and - with ones who don't have it wont work like bellow

Systemctl wont work

```
<Directory /var/www/crystalmind.academy>
    AllowOverride All
    Options -Indexes FollowSymLinks
</Directory>
```

Correct

```
<Directory /var/www/crystalmind.academy>
    AllowOverride All
    Options -Indexes +FollowSymLinks
</Directory>
```

Like we said when the client request a directory , the server would look for the index file of that directory , we can change the default in the document root directory section

```
<Directory /var/www/crystalmind.academy>
    AllowOverride All
    Options -Indexes +FollowSymLinks
    DirectoryIndex a.txt index.html
</Directory>
```

Now the server would look first for a.txt and if it doesn't exist it will look for the index.html  
If both don't exist , it will in options , then if nothing , then display an error

## Change the server signature :

Big red flag : the error displays info about the system

# Not Found

The requested URL was not found on this server.

Apache/2.4.41 (Ubuntu) OpenSSL/1.1.1f Server at crystalmind.academy Port 443

Vim /etc/apache2/conf-available/security.conf

```
-----  
ServerTokens Prod  
  
#  
# Optionally add a line containing the server  
# name to server-generated pages (internal er  
# listings, mod_status and mod_info output et  
# documents or custom error documents).  
# Set to "EMail" to also include a mailto: li  
# Set to one of: On | Off | EMail  
ServerSignature Off  
#ServerSignature On
```

Always restart the web server

## HTTP compression :

Apache Module mod\_deflate

That reduces bandwidth needs and cost , we use HTTP module , the output of the server will be compressed , the browser will load faster , lower usage for bandwidth will elevate site performance

**Enable moddeflate :**

[https://httpd.apache.org/docs/2.4/mod/mod\\_deflate.html](https://httpd.apache.org/docs/2.4/mod/mod_deflate.html)

a2enmod deflate

**Config deflate :**

vim /etc/apache2/mods-enabled/deflate.conf

We gonna edit the compression level

```
<IfModule mod_deflate.c>  
session <IfModule mod_filter.c>  
        AddOutputFilterByType DEFLATE text/html text/plain text/xml text/css  
        AddOutputFilterByType DEFLATE application/x-javascript application/javascript application/ecmascript  
        AddOutputFilterByType DEFLATE application/rss+xml  
        AddOutputFilterByType DEFLATE application/xml  
        DeflateCompressionLevel 7  
</IfModule>  
:/IfModule>
```

**Test the new module by config the virtual host file :**

Vim /etc/apache2/sites-enabled/crystalmind-academy-le-ssl.conf

Compression ratio

```
<VirtualHost *:443>
    ServerName crystalmind.academy
    ServerAlias www.crystalmind.academy
    DocumentRoot /var/www/crystalmind.academy
    ErrorLog ${APACHE_LOG_DIR}/crystalmind_academy_error.log
    CustomLog ${APACHE_LOG_DIR}/crystalmind_academy_access.log combined

    DeflateFilterNote ratio
    LogFormat "%{ratio}n%" "%{User-agent}i" deflate
    CustomLog "/var/log/apache2/deflate_log" deflate

    SSLCertificateFile /etc/letsencrypt/live/crystalmind.academy/fullchain.pem
    SSLCertificateKeyFile /etc/letsencrypt/live/crystalmind.academy/privkey.pem
    Include /etc/letsencrypt/options-ssl-apache.conf
```

Restart the server

**Test the compression :**

Downloading jquery library

```
root@ubuntu-s-1vcpu-1gb-fra1-01:~# cd /var/www/crystalmind.academy/
root@ubuntu-s-1vcpu-1gb-fra1-01:/var/www/crystalmind.academy# wget https://code.jquery.com/jquery-3.5.1.js
```

**Check the compression ratio :**

From the log file

tail /var/log/apache2/deflate\_log

```
root@ubuntu-s-1vcpu-1gb-fra1-01:/var/www/crystalmind.academy# ls
a.gif admin dir1 documents index.html jquery-3.5.1.js link_to_etc protected report.txt
root@ubuntu-s-1vcpu-1gb-fra1-01:/var/www/crystalmind.academy# tail /var/log/apache2/deflate_log
apache2/      apport.log      apport.log.1 apt/
root@ubuntu-s-1vcpu-1gb-fra1-01:/var/www/crystalmind.academy# tail /var/log/apache2/deflate_log
"HEAD /jquery-3.2.1.js HTTP/1.1"  -- (-%)
"GET /jquery-3.2.1.js HTTP/1.1" 79438/268039 (29%)
"GET /jquery-3.2.1.js HTTP/1.1" 79438/268039 (29%)
"GET /jquery-3.2.1.js HTTP/1.1"  -- (-%)
"GET /jquery-3.2.1.js HTTP/1.1"  -- (-%)
"GET /jquery-3.2.1.js HTTP/1.1"  -- (-%)
"GET /jquery-3.2.1.js HTTP/1.1" 79438/268039 (29%)
"GET /wp-login.php HTTP/1.1" 196 (-%) "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:62.0) Gecko/20100101"
"GET /jquery-3.5.1.js HTTP/1.1" 84781 (29%) "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/88.0.4324.104 Safari/537.36"
"--" -- (-%) "--"
root@ubuntu-s-1vcpu-1gb-fra1-01:/var/www/crystalmind.academy#
```

**Monitoring the server :**

Server logs are insufficient

By using SetHandler and server status module

**Enable the status module :**

a2enmod status

**Configure the module :**

vim /etc/apache2/mods-available/status.conf

The module is usually the big variable , the directive is the one inside it

```
<IfModule mod_status.c>
    # Allow server status reports generated by mod_status,
    # with the URL of http://servername/server-status
    # Uncomment and change the "192.0.2.0/24" to allow access from other hosts.

    <Location /server-status>
        SetHandler server-status
        Require local
        #Require ip 192.0.2.0/24
    </Location>
```

The location directive operates outside the file system , not need for a file with this server-status

Handler is an action activated on apache

Require (limit to only the ip addresses provided to see the output )

Restart apache

Can be accessed by me



Server status

**Simulate concurrent connections :**

Ab -n 100 -c 10 https://crystalminds.academy or local host

100 request , max 10 request running at the same time

Srv	PID	Acc	M	CPU	SS	Req	Dur	Conn	Child	Slot	Client	Protocol	VHost	Request
0-0	6697	0/12/12	_	0.00	3	0	22	0.0	0.04	0.04	127.0.0.1	http/1.1		
1-0	6698	0/13/13	_	0.01	3	0	27	0.0	0.05	0.05	127.0.0.1	http/1.1		
2-0	6699	0/13/13	_	0.01	3	1	28	0.0	0.05	0.05	127.0.0.1	http/1.1		
3-0	6700	0/14/14	_	0.02	3	2	22	0.0	0.05	0.05	127.0.0.1	http/1.1		
4-0	6701	0/12/12	_	0.01	3	0	17	0.0	0.04	0.04	127.0.0.1	http/1.1		
5-0	6702	0/12/12	R	0.01	3	0	29	0.0	0.04	0.04	86.121.14.97	http/1.1		
6-0	6703	0/12/12	_	0.00	3	0	24	0.0	0.04	0.04	127.0.0.1	http/1.1		
7-0	6704	0/13/13	W	0.01	0	0	20	0.0	0.05	0.05	86.121.14.97	http/1.1	crystalmind.academy:443	GET /server-status HTTP/1.1

**Set the browser to refresh the page ever n seconds : real time**



## Install PHP :

Apt update && apt install php php-mysql libapache2-mod-php

Systemctl restart php

Php -v

**Test if php and apache work together and handle dynamic content :**

Vim /var/www/crystalmind.academy/test.php

```
<?php
    phpinfo();
?>
```

Now lets see it on the browser

<https://crystalminds.academy/test.php>



## **Installing and securing the mysqlserver :**

To manage the data of my server

### **Install mysql server :**

Apt install mysql-server

It will start automatically

Systemctl status mysql

Sql daemon process : ps -ef | grep mysql

### **Secure mysql :**

Run the security script that comes preinstalled to remove week standard configurations and set admin password

mysql\_secure\_installation

```
root@ubuntu-s-1vcpu-2gb-fra1-01:/etc/apache2# mysql_secure_installation
Securing the MySQL server deployment.

Connecting to MySQL using a blank password.

VALIDATE PASSWORD COMPONENT can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD component?

Press y|Y for Yes, any other key for No: y

There are three levels of password validation policy:

LOW    Length >= 8
MEDIUM Length >= 8, numeric, mixed case, and special characters
STRONG Length >= 8, numeric, mixed case, special characters and dictionary      file

Please enter 0 = LOW, 1 = MEDIUM and 2 = STRONG: 1
Please set the password for root here.                                     I

New password: ■
```

The root mentioned here has no relation with the linux root user but the root of the sql server

Remove the anonymos and test database

Yes for privilege tables

### **Test the sql server :**

mysql -u root

If you run the command as system root it will not ask for a password

## Installing phpmyadmin :

It is to manage mysql and mariadb

Before installing it , if the “Validate password plugin for mysql ” is enabled , it will trigger an error when ur trying to set a password for the phpmyadmin user

**Connect to the mysql server :**

mysql (as root)

**Uninstall the plugin :**

```
mysql> UNINSTALL COMPONENT "file://component_validate_password"
-> ;■
```

It will prevent the component from being loaded to the server

**Install myphpadmin :**

```
apt update && apt install phpmyadmin php-mbstring php-zip php-gd php-json php-curl
```

Yes for db-config

We notice a new config file was added

```
root@ubuntu-s-1vcpu-1gb-fra1-01:~# cat /etc/apache2/conf-enabled/
charset.conf          other-vhosts-access-log.conf  serve-cgi-bin.conf
javascript-common.conf  phpmyadmin.conf
localized-error-pages.conf   security.conf
```

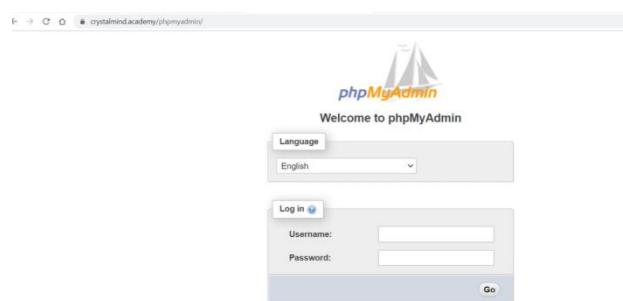
Then we enable mbstring

```
phpenmod mbstring
```

Systemctl restart apache2

**Access phpmyadmin :**

Crystalmind.academy/phpmyadmin



Create a phpmyadmin user with admin privileges :

```
mysql
```

```
CREATE USER
```

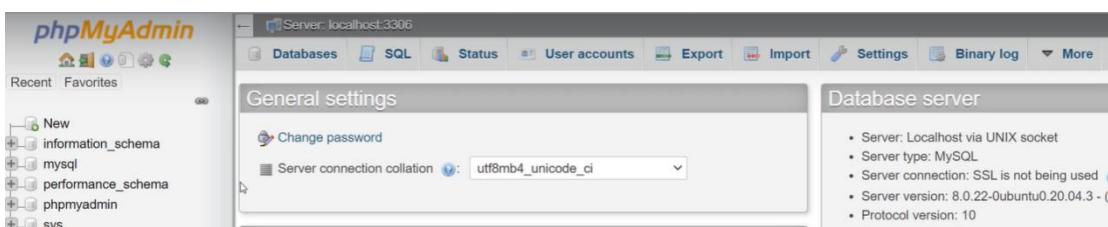
```
mysql> CREATE USER 'toor'@'localhost' IDENTIFIED WITH caching_sha2_password BY 'toorpass123=';
Query OK, 0 rows affected (0.01 sec)
```

With my sql\_native\_password can work as well , look for the most updated one

```
GRANT PRIVILIGES
```

```
mysql> GRANT ALL PRIVILEGES ON *.* TO 'toor'@'localhost' WITH GRANT OPTION;
```

SUCCESS



## Secure phpmyadmin :

### Securing phpMyAdmin

- **Update** all components of the LAMP stack to their latest versions (phpMyAdmin, Apache, PHP, MySql);
- **Limit** the access to phpMyAdmin by source IP address (Require IP);
  - Use <Directory /usr/share/phpmyadmin> ... </Directory> or .htaccess
- **Add** an extra layer of authentication using the Apache built-in features;
  - Use <Directory /usr/share/phpmyadmin> ... </Directory> or .htaccess
- **Change** the URL used to access phpMyAdmin;
  - Change the Alias directive in /etc/phpmyadmin/apache.conf
- **Use** only HTTPS;
- **User** only strong passwords;

Limit the access : /etc/phpmyadmin/apache.conf

Add extra layer

```
<Directory /var/www/crystalind.academy/admin>
    Require all denied
    Require ip 6.7.73.2
    Require ip 86.121.14.97
    Require ip 10.114.0.2/20
    Require local
</Directory>
```

/usr/share/phpmyadmin

```
#Alias /phpmyadmin /usr/share/phpmyadmin
Alias /phpmy21 /usr/share/phpmyadmin
```

### Commands - phpMyAdmin

```
## Installing phpMyAdmin ##

# Disabling MySql Validate Password Plugin
mysql>UNINSTALL COMPONENT "file://component_validate_password";
mysql>exit;

# Installing phpMyAdmin
sudo apt update && apt install phpmyadmin php-mbstring php-zip php-gd php-json php-curl

# Creating a MySql dedicated user for phpmyadmin
mysql>CREATE USER 'toor'@'localhost' IDENTIFIED WITH
caching_sha2_password BY 'toorpass123=';
    # or using mysql_native_password (backward compatible)
mysql>CREATE USER 'toor1'@'localhost' IDENTIFIED WITH
mysql_native_password BY 'toorpass123=';
    mysql>GRANT ALL PRIVILEGES ON *.* TO 'toor'@'localhost' WITH GRANT
OPTION;

phpMyAdmin is accessed at https://server_ip_or_domain/phpmyadmin

Any change made to phpMyAdmin will be done by editing
/etc/phpmyadmin/apache.conf

# Change the phpMyAdmin URL:
a) Open /etc/phpmyadmin/apache.conf
b) Change the Alias directive:
    Alias /NEW_NAME /usr/share/phpmyadmin
It will be accessed at https://server_ip_or_domain/NEW_NAME

To limit access to phpMyAdmin by source IP (host) or to add HTTP
Authentication add the
Apache directives that protect /usr/share/phpmyadmin in
a <Directory> ... </Directory> section (in /etc/phpmyadmin/apache2)
or in .htaccess (in /usr/share/phpmyadmin)
```

## **Install wordpress :**

Content management system written in php

### **Install wordpress :**

Apache is running , php , mysql as well

First we should create an sql database to store the data of the web app

Mysql -u root

CREATE DATABASE wordpress ;

#create a user that has all access

CREATE USER 'wordpressuser'@'localhost' IDENTIFIED BY 'Pass123.' ;

#grant priviliges to all tables in the database

GRANT ALL PRIVILEGES ON wordpress.\* TO 'worpressuser'@'localhost' ;

#reload privileges

FLUSH PRIVILEGES ;

Exit

### **Download the latest version of wordpress :**

Cd /var/tmp (to have write access , every user has one here )

Wget <https://wordpress.org/latest.tar.gz>

Tar -xvf latest.tar.gz

### **Move the wordpress to the virtual host directory :**

Remove everything first

```
root@ubuntu-s-1vcpu-2gb-fra1-01:/tmp# ls -l /var/www/crystalmind.academy/
total 8
-rw-r--r-- 1 root root 62 Oct 30 09:51 index.html
-rw-r--r-- 1 root root 21 Oct 30 10:28 test.php
root@ubuntu-s-1vcpu-2gb-fra1-01:/tmp# rm -rf /var/www/crystalmind.academy/*
root@ubuntu-s-1vcpu-2gb-fra1-01:/tmp# mv
```

Mv wordpress/\* /var/www/crystalmind.academy

### **Changing the ownership of the files to the same as the webserver runs :**

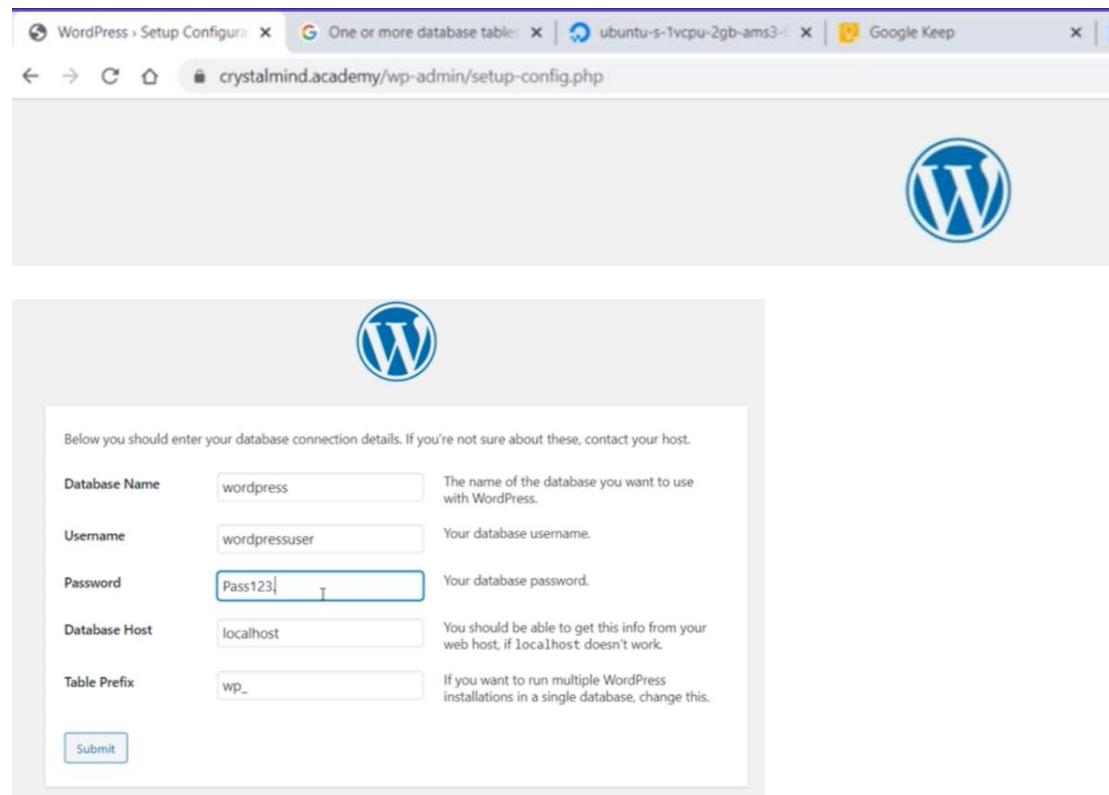
#checking the user for apache2

ps -ef | grep apache2

```
#setting up the changement
```

```
Chown -R www-data www-data /var/www/crystalmind.academy
```

### Finalize the installation on the web browser :



### Save the password somewhere

Site Title	My Personal Blog
Username	wpadmin
Usernames can have only alphanumeric characters, spaces, underscores, periods, and the @ symbol.	
Password	jfqj^SwN0SHl5w&jZw <small>Strong</small>
<b>Important:</b> You will need this password to log in. Please store it in a secure place.	
Your Email	webmaster@crystalmind.academy
Double-check your email address before continuing.	
Search engine visibility	<input type="checkbox"/> Discourage search engines from indexing this site <small>It is up to search engines to honor this request.</small>

### Posts - add new

## Securing wordpress :

<https://www.wpbeginner.com/wordpress-security/>

crystalminds.academy/wp-admin

# WordPress Security

1. Always use the **latest version** of WordPress and keep all plugins up to date.
2. Use only **strong passwords** (min. 10 random characters including special ones).
3. **Limit login attempts** using a plugin or a WAF.
3. Install a security plugin or a WAF (Web Application Firewall). Exemple: **WordFence**.
4. Add **2-step verification** (using a security plugin).
5. Protect **wp-admin** directory (source IP access or username and passwords).
6. **Make backups** regularly and test them.

-check the updates  
- add a wap



Go to login security

Enable MFA

#protect the wp-admin

```
root@ubuntu-s-1vcpu-2gb-fra1-01:/var/www/crystalmind.academy# ls
index.php      wp-admin          wp-config.php   wp-links-opml.php  wp-settings.php
license.txt    wp-blog-header.php  wp-content     wp-load.php      wp-signup.php
readme.html    wp-comments-post.php wp-cron.php   wp-login.php    wp-trackback.php
wp-activate.php wp-config-sample.php wp-includes   wp-mail.php     xmlrpc.php
```

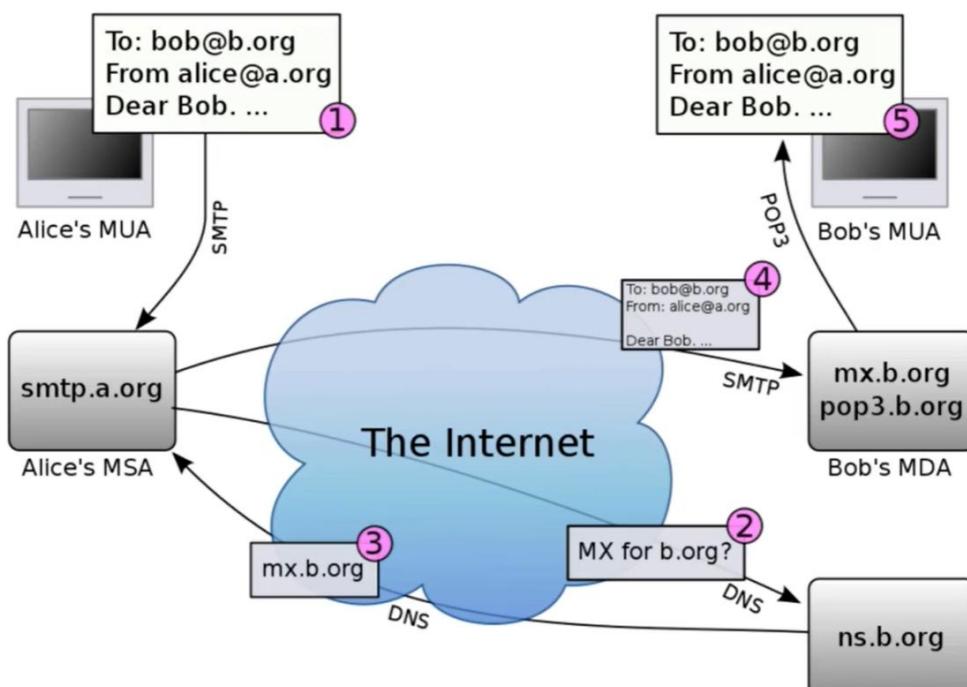
<https://www.wpbeginner.com/wordpress-security/>

#add a backup

Search for plugin backup

## Project 4 : Running a virtual mail server

Complex mail system for a large company



MUA : outlook (mail user agent )

Smtp.a.org : mail server (gonna use postfix mail server) port 25

The source SMTP server will determine from the message envelope the destination domain , which is the part after @ in this case b.org , and then queries the dns server for the MX resource of the destination domain

**MX :** mail exchanger , is the destination mail or the smtp server ,

MX records are DNS records that specify the mail servers responsible for receiving email on behalf of a domain

**OTHER DNS RECORDS : A ,AAAA , MX , NS SOA**

**Third step :** the dns server responds with the mx record of the destination domain ,

Now the source smtp knows the name and ip address of the destination smtp server

**POP VS IMAP :** pop deletes the email after retrieving it , imap saves and synchronize the message across different devices , imap is prefered

Mail config :

# Virtual Mail System Components

---

- **Postfix** (Mail/SMTP Server, MTA)
  - STARTTLS used for encryption
  - SMTP AUTH (SASL) used for authentication
- **Dovecot** (IMAP/POP3 Server, MDA)
  - (SSL/TLS) used for encryption
- **MySql**
  - stores the domains and the virtual users
- **Spam Filter** (Rspamd/SpamAssassin)

Handle multiple email from multiple domains , the users are virtual , they don't exist in /etc , we will store them in an sql database , the connection between the client and postfix mail server will be secured with STARTTLS

## Mail and DNS : MX records and reverse DNS

### Mail and DNS

---

#### Requirements:

1. An **A** record, to point the system's FQDN to the mail server IPv4 address.  
*mail.crystalmind.academy. IN A 206.81.X.X*

2. An **MX** record for the domain.

*@ IN MX 10 mail.crystalmind.academy.  
mail IN A 206.81.X.X*

3. An **SPF** record for the domain

*crystalmind.academy. IN TXT "v=spf1 mx ~all"*

4. A **PTR** Record (Reverse DNS)

#### A record :

Dig -t a mail.crystalmind.academy

#### Mx record :

Zonefile : cat /etc/bind/db.crystalmind.academy

```

,
@      IN      NS      ns1.crystalmind.academy.
ns1    IN      A       206.81.23.94
@      IN      MX     10 mail.crystalmind.academy.
crystalmind.academy.   IN      A       206.81.23.94
www    IN      A       206.81.23.94
mail   IN      A       206.81.23.94

```

10 is the priority

**Checking the mx record :**

dig -t mx crystalmind.academy

**Setting up an SPF record :**

A txt record that displays the hosts that are allowed to send mails on behalf of our domain

Its an email authentication techniques to prevent attackers from using our domain to send malicious emails or spams

Vim /etc/bind/db.crystalmind.academy

```

mail    IN      A       206.81.23.94
crystalmind.academy.   IN      TXT      "v=spf1 mx ~all"

```

don't forget the . after the domain name

Mx : we are approving the domain mail servers in the mx record

~all : if the spf checks fails the result would be a soft failure , this info will be checked by other mail servers that receives mails from our domain and use spf , if our domain doesn't use it , receiving mail servers cannot make sure that we r the ones sending the message and might considerate spam

Check the spf record

Dig -t txt crystalmind.academy

**PTR record (reverse dns ) :**

IP to domain

Most email servers will perform a reverse dns lookup on the ip of the server requesting resources , and may not accept email from the server if the ptr record aint set up

The PTR record can only be set by the host provider owning the IP address

*Dig -x ip address of our server*

For some cloud providers , the ptr record will be set automatically based on the hostname you provide

Changing host name :

Hostnamectl

```
root@ubuntu-s-2vcpu-4gb-fra1-01:~# hostnamectl
  Static hostname: ubuntu-s-2vcpu-4gb-fra1-01
    Icon name: computer-vm
      Chassis: vm
    Machine ID: 151f5f1290cb4584b9252d211004dafa
      Boot ID: ab1fcdf9626c418c816ae315fc7016be
  Virtualization: kvm
Operating System: Ubuntu 20.04.1 LTS
      Kernel: Linux 5.4.0-54-generic
    Architecture: x86-64
root@ubuntu-s-2vcpu-4gb-fra1-01:~#
```

Hostnamectl set-hostname newhostname(mail.crystalmind.academy)

Vim /etc/hosts (change the default to the one above)

```
# Your system has configured 'manage_etc_hosts' as True.
# As a result, if you wish for changes to this file to persist
# then you will need to either
# a.) make changes to the master file in /etc/cloud/templates/hosts.debian tmpl
# b.) change or remove the value of 'manage_etc_hosts' in
#      /etc/cloud/cloud.cfg or cloud-config from user-data
#
127.0.1.1 mail.crystalmind.academy
127.0.0.1 localhost
```

Then reboot the vm or instance

If we run dig we should see the mail.crystalmind.academy in the ptr record

## Prerequisites :

- Set up a droplet , with a static ip adress
  - a domain acquired, set up our own dns server as authoritative for that domain
  - the dns server is running bind9 and mx record set up
- The reverse dns was configured automatically with correlation to the hostname chosen
- sql server with root access
  - digital certificate used by apache2 , that would be used for the mail server as well

## Installing software packages :

Installation guide

### 1. Installing Software Packages

```
apt update && apt install postfix postfix-mysql postfix-doc dovecot-common dovecot-imapd dovecot-pop3d libsasl2-2 libsasl2-modules libsasl2-modules-sql sasl2-bin libpam-mysql mailutils dovecot-mysql dovecot-sieve dovecot-managesieved
```

System is-enabled postfix

## 2. Configuring MySQL and Connect it With Postfix

The mail server virtual users and passwords required for both postfix and dovecot is stored in mysql

```
mysql -u root
mysql> CREATE DATABASE mail;
mysql> USE mail;
mysql> CREATE USER 'mail_admin'@'localhost' IDENTIFIED BY 'mail_admin_password';
#grant user access to all tables in mail database
mysql> GRANT SELECT, INSERT, UPDATE, DELETE ON mail.* TO 'mail_admin'@'localhost';
#apply the changes without reloading the server
mysql> FLUSH PRIVILEGES;
mysql> CREATE TABLE domains (domain varchar(50) NOT NULL, PRIMARY KEY (domain));
#users table to store their emails and passwords
mysql> CREATE TABLE users (email varchar(80) NOT NULL, password varchar(128) NOT NULL,
PRIMARY KEY (email));
#all the emails that are going to be forwarded to other emails
mysql> CREATE TABLE forwardings (source varchar(80) NOT NULL, destination TEXT NOT NULL,
PRIMARY KEY (source));
mysql> exit
```

## 3. Configuring Postfix to communicate with MySQL

#how post fix will get the data from my sql , bellow is mysql data

a) vim /etc/postfix/mysql\_virtual\_domains.cf  
user = mail\_admin  
password = mail\_admin\_password  
dbname = mail  
query = SELECT domain FROM domains WHERE domain='%s'  
hosts = 127.0.0.1

b) vim /etc/postfix/mysql\_virtual\_forwardings.cf  
#how to get data from the forwarding table  
user = mail\_admin  
password = mail\_admin\_password  
dbname = mail  
query = SELECT destination FROM forwardings WHERE source='%s'  
hosts = 127.0.0.1

c) vim /etc/postfix/mysql\_virtual\_mailboxes.cf  
#create a virtual mailbox  
user = mail\_admin  
password = mail\_admin\_password  
dbname = mail  
query = SELECT CONCAT(SUBSTRING\_INDEX(email,'@',-1),'/',SUBSTRING\_INDEX(email,'@',1))  
FROM users WHERE email='%s'  
hosts = 127.0.0.1

d) vim /etc/postfix/mysql\_virtual\_email2email.cf  
#virtual email mapping file  
user = mail\_admin  
password = mail\_admin\_password  
dbname = mail  
query = SELECT email FROM users WHERE email='%s'  
hosts = 127.0.0.1

e) Setting the ownership and permissions

```
#the access change to only the owner
chmod o-rwx /etc/postfix/mysql_virtual_*
#the owner is postfix now
chown root.postfix /etc/postfix/mysql_virtual_*
```

4. Creating a user and group for mail handling : (because the users don't exist in /etc)

```
#create a group and user called vmail with static id 5000
groupadd -g 5000 vmail
#all the emails will be stored in /var/vmail
useradd -g vmail -u 5000 -d /var/vmail -m vmail
```

5. Configuring postfix

```
postconf -e "myhostname = mail.crystalmind.academy"
postconf -e "mydestination = mail.crystalmind.academy, localhost, localhost.localdomain"
postconf -e "mynetworks = 127.0.0.0/8"
postconf -e "message_size_limit = 31457280"
#maximum size in bytes of a message 30mb
postconf -e "virtual_alias_domains ="
postconf -e "virtual_alias_maps = proxy:mysql:/etc/postfix/mysql_virtual_forwardings.cf,
mysql:/etc/postfix/mysql_virtual_email2email.cf"
#how postfix should connect to sql server
postconf -e "virtual_mailbox_domains = proxy:mysql:/etc/postfix/mysql_virtual_domains.cf"
postconf -e "virtual_mailbox_maps = proxy:mysql:/etc/postfix/mysql_virtual_mailboxes.cf"
postconf -e "virtual_mailbox_base = /var/vmail"
#where all the mails will be saved
postconf -e "virtual_uid_maps = static:5000"
#user vmail id used to deliver mails
postconf -e "virtual_gid_maps = static:5000"
postconf -e "smtpd_sasl_auth_enable = yes"
postconf -e "broken_sasl_auth_clients = yes"
#broken enables interoperability with remote smtp clients that implements an obsolete version of sasl
postconf -e "smtpd_sasl_authenticated_header = yes"
#enables SASL authentication in postfix smtp server , if a remote smtp client is authenticated , the permit_sasl_authenticated access restriction below can be used to permit relay access or to permit the client to send emails using this server
postconf -e "smtpd_recipient_restrictions = permit_mynetworks, permit_sasl_authenticated,
reject_unauth_destination"
#recipient restriction means that only my networks which is the local host or 127.0.0.1 and sasl authenticated clients are allowed to send emails using this server , important security settings
postconf -e "smtpd_use_tls = yes"
#announces STARTTLS support to remote smtp clients without requiring tls encryption
postconf -e "smtpd_tls_cert_file = /etc/letsencrypt/live/crystalmind.academy/fullchain.pem"
#the path to the digital certificate and privatekey used by the server for tls encryption same used by apache
postconf -e "smtpd_tls_key_file = /etc/letsencrypt/live/crystalmind.academy/privkey.pem"
postconf -e "virtual_transport=dovecot"
#indicates that the local delivery to the users mailbox is done by dovecot
postconf -e 'proxy_read_maps = $local_recipient_maps $mydestination $virtual_alias_maps
$virtual_alias_domains $virtual_mailbox_maps $virtual_mailbox_domains $relay_recipient_maps
$relay_domains $canonical_maps $sender_canonical_maps $recipient_canonical_maps
$relocated_maps $transport_maps $mynetworks $virtual_mailbox_limit_maps'
```

<https://www.postfix.org/postconf.5.html>

See the parameter value from main.cf  
Postconf -n message\_size\_limit

Postconf -d (the default value )

## 5. Configuring SMTP AUTH (SASLAUTHD and MySql)

### SMTP AUTH (SASLAUTHD with PAM and MySql)

- The original SMTP did not provide any form of authentication.
- **SMTP Authentication (SMTP AUTH)** is an extension of the SMTP whereby a client may log in using an authentication mechanism supported by the server.
- SMTP AUTH is implemented by something called **SASL** (Simple Authentication and Security Layer).
- The SASL implementation for Postfix uses either a library called **Cyrus SASL** or **Dovecot SASL**. Check support: `postconf -a`
- The application that handles SASL is called SASL Authentication Daemon or **saslauthd**.
- **PAM** (Pluggable Authentication Modules) provides authentication for **saslauthd**. It practically says how to access the MySql backend.

An open mail relay is a security issue because anyone can send a mail without providing authentication , like using gmail smtp server , you need to provide your username and password

a) Creating a directory where saslauthd will save its information:

`mkdir -p /var/spool/postfix/var/run/saslauthd`

b) Editing the configuration file of saslauthd: `vim /etc/default/saslauthd`

```
START=yes
DESC="SASL Authentication Daemon"
NAME="saslauthd"
MECHANISMS="pam"
MECH_OPTIONS=""
THREADS=5
OPTIONS="-c -m /var/spool/postfix/var/run/saslauthd -r"
```

c) Creating a new file: `vim /etc/pam.d/smtp`

#provides authentication for saslauthd and how to access mysql backend , crypt 3 that means using md5

```
auth required pam_mysql.so user=mail_admin passwd=mail_admin_password host=127.0.0.1
db=mail table=users usercolumn=email passwdcolumn=password crypt=3
account sufficient pam_mysql.so user=mail_admin passwd=mail_admin_password host=127.0.0.1
db=mail table=users usercolumn=email passwdcolumn=password crypt=3
```

```
d) vim /etc/postfix/sasl/smtpd.conf  
pwcheck_method: saslauthd  
mech_list: plain login  
log_level: 4
```

```
e) Setting the permissions  
chmod o-rwx /etc/pam.d/smtp  
chmod o-rwx /etc/postfix/sasl/smtpd.conf
```

```
f) Adding the postfix user to the sasl group for group access permissions:  
usermod -aG sasl postfix
```

```
g) Restarting the services:  
systemctl restart postfix  
systemctl restart saslauthd
```

## 7. Configuring Dovecot (POP3/IMAP)

Dovecot is a well known pop3 /imap server  
Master.cf controls the postfix daemon process , a process that works in the parameters defined in master.cf

```
[root@mail ~]# ll /etc/postfix/  
Dovecot (POP3/IMAP)  
total 136  
-rw-r--r-- 1 root root 99 Nov 23 08:49 dynamicmaps.cf  
drwxr-xr-x 2 root root 4096 Jun 19 17:11 dynamicmaps.cf.d  
-rw-r--r-- 1 root root 2430 Nov 23 09:40 main.cf  
-rw-r--r-- 1 root root 27120 Nov 23 08:49 main.cf.proto  
lrwxrwxrwx 1 root root 31 Nov 23 08:49 makedefs.out -> /usr/share/postfix/makedefs.out  
-rw-r--r-- 1 root root 6208 Nov 23 08:49 master.cf
```

Postfix will use dovecot as mail delivery agent , so we need to add dovecat the master.cf

a) At the end of /etc/postfix/master.cf add:

```
dovecot unix - n n - - pipe  
flags=DRhu user=vmail:vmail argv=/usr/lib/dovecot/deliver -d ${recipient}
```

```
mailman unix - n n - - pipe  
flags=FR user=list argv=/usr/lib/mailman/bin/postfix-to-mailman.py  
${nexthop} ${user}  
dovecot unix - n n - - pipe  
flags=DRhu user=vmail:vmail argv=/usr/lib/dovecot/deliver -d ${recipient}
```

b) Edit Dovecot config file: vim /etc/dovecot/dovecot.conf

Erase all the content and add the below

Mail location: is where dovecot looks for emails

passdb : password database used to authenticate users , dovecot-sql tell dovecot how to connect to sql and extract usernames and passwords , **don't forget to create the args file**

Protocols options : specifies which protocols dovecot supports

Service auth : will set up the socket , user and permissions , used to communicate with postfix

Protocol LDA : specifies LDA settings

Service stats : gather stats from imap /pop3

```
log_timestamp = "%Y-%m-%d %H:%M:%S "
mail_location = maildir:/var/vmail/%d/%n/Maildir
managesieve_notify_capability = mailto
managesieve_sieve_capability = fileinto reject envelope encoded-character vacation subaddress comparator-i;ascii-numeric relational regex imap4flags copy include variables body enotify environment mailbox date
namespace {
    inbox = yes
    location =
    prefix = INBOX.
    separator = .
    type = private
}
passdb {
    args = /etc/dovecot/dovecot-sql.conf
    driver = sql
}
protocols = imap pop3

service auth {
    unix_listener /var/spool/postfix/private/auth {
        group = postfix
        mode = 0660
        user = postfix
    }
    unix_listener auth-master {
        mode = 0600
        user = vmail
    }
    user = root
}

userdb {
    args = uid=5000 gid=5000 home=/var/vmail/%d/%n allow_all_users=yes
    driver = static
}

protocol lda {
    auth_socket_path = /var/run/dovecot/auth-master
    log_path = /var/vmail/dovecot-deliver.log
    mail_plugins = sieve
    postmaster_address = postmaster@example.com
}

protocol pop3 {
    pop3_uidl_format = %08Xu%08Xv
}
```

```

service stats {
    unix_listener stats-reader {
        user = dovecot
        group = vmail
        mode = 0660
    }
    unix_listener stats-writer {
        user = dovecot
        group = vmail
        mode = 0660
    }
}

ssl = yes
ssl_cert = </etc/letsencrypt/live/crystalmind.academy/fullchain.pem
ssl_key = </etc/letsencrypt/live/crystalmind.academy/privkey.pem

c) vim /etc/dovecot/dovecot-sql.conf
driver = mysql
connect = host=127.0.0.1 dbname=mail user=mail_admin password=mail_admin_password
default_pass_scheme = PLAIN-MD5
password_query = SELECT email as user, password FROM users WHERE email='%u';

d) Restart Dovecot
systemctl restart dovecot

```

## 8. Adding Domains and Virtual Users.

```

mysql -u root
mysql>USE mail;
mysql>INSERT INTO domains (domain) VALUES ('crystalmind.academy');
mysql>insert into users(email,password) values('u1@crystalmind.academy', md5('pass123'));
mysql>insert into users(email,password) values('u2@crystalmind.academy', md5('pass123'));
Mysql>SELECT * FROM users ;
mysql>quit;

```

### **Test the SASL authentication server :**

```

testsaslauthd -u user1@crystalmind.academy -p pass123 -f /var/spool/postfix/var/run/saslauthd/mux
-s smtp
root@mail:~# testsaslauthd -u user1@crystalmind.academy -p pass123 -f /var/spool/postfix/var/run/saslauthd/mux -s smtp
.0: OK "Success."

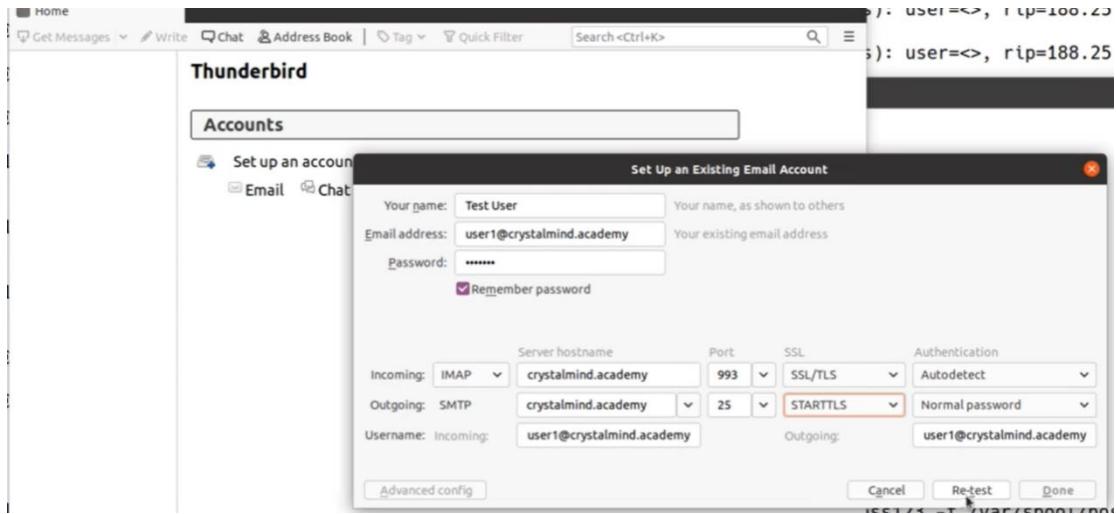
```

## 9. Testing the Mail System.

Set up a Mail client like Mozilla Thunderbird/outlook, send and receive mail to both local and external accounts.

The server hostname should be the same as the one in the certificate or it will give an error

In the username the domain is important because you can have user1 in another domain handled by postfix



Adding second account : Account actions , add mail account , USER2

Test sending email from two local accounts , send email from user1 to user2

And to external account , sent to it and reply from it

Problems can arise if port 25 is closed in our domain or ISP is blocking it , you cant connect to port 25 from your LAN

**Configure another port SMTP 465 in postfix in your mail**

**server for mail submission :**

Some mail client like outlook prefer it in the first place

## Netstat -tupan 465

Active Internet connections (servers and established)						
Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State	PID/Program name
tcp	0	0	10.114.0.2:53	0.0.0.0:*	LISTEN	701/named
tcp	0	0	10.19.0.5:53	0.0.0.0:*	LISTEN	701/named
tcp	0	0	206.81.23.94:53	0.0.0.0:*	LISTEN	701/named
tcp	0	0	127.0.0.1:53	0.0.0.0:*	LISTEN	701/named
tcp	0	0	127.0.0.53:53	0.0.0.0:*	LISTEN	554/systemd-resolve
tcp	0	0	0.0.0.0:25	0.0.0.0:*	LISTEN	3077/master

Postfix master listens on port 25

## Vim /etc/postfix/master.cf

```
# -o milter_macro_daemon_name=ORIGINATING
smtps      inet  n       -       y       -       -          smtpd
```

## Systemctl restart postfix

Now it listens on port 465

tcp 0 0.0.0.0:465 0.0.0.0:\*

# My favorite : Troubleshooting

## Ping the server from the internet :

Ping mail.crystalmind.academy

## Test if port 25 is open :

```
Nmap -p 25 mail.crystalmind.academy -Pn  
Nmap -p 465 mail.crystalmind.academy -Pn
```

## From windows :

```
Telnet mail.crystalmind.academy 25
```

## Monitor the mail server :

```
tail -f /var/log/mail.log
```

## Increase the logging level : add -v

```
Vim /etc/postfix/master.cf
```

```
# ======  
smtp      inet  n      -      y      -      -      -      smtpd -v  
#smtp      inet  n      -      y      -      -      1      postscreen
```

Systemctl restart postfix

Resend an email and check the log

## Check the authentication issues :

```
tail -f /var/log/auth.log
```

If authentication problems persist use pam debug

```
vim /etc/pam.d/smtp
```

```
[auth required pam_mysql.so user=mail_admin passwd=mail_admin_password host=127.0.0.1 db=mail table=users usercolumn=email passwdcolumn=password crypt=3 debug]  
[account sufficient pam_mysql.so user=mail_admin passwd=mail_admin_password host=127.0.0.1 db=mail table=users usercolumn=email passwdcolumn=password crypt=3 debug]
```

So the thing is when we edit the conf file , we should remember which service is connected to it and then we should restart it after we modify the log file

Systemctl restart saslauthd

We can check the debug mode in the logs

```
Tail -f /var/log/mail.log / auth.log
```

## **Virus scanning using Amavis and ClamAv :**

Scan the emails for viruses before being accepted by the mail server

I guess its protecting windows computers from themselves

Amavis is a high performance interface between a mailserver such as postfix and content filter such as antivirus or antispam filtering systems , these content filters will scan the headers and body of the email , such as clamav and spamassassin

**Install amavis :**

Apt update && apt install amavisd-new

E : subprocess error code

You should set the value of the variable “my hostname” to your domain name

Vim /etc/amavis/conf.d/05-node\_id

```
$myhostname = "mail.crystalmind.academy";
```

Systemctl status amavis

Amavisd-new -V

Systemctl enable amavis

```
Feb 15 11:59:14 mail amavis[5571]: No ext program for .deb, tried: ar
Feb 15 11:59:14 mail amavis[5571]: No ext program for .zoo, tried: zoo
Feb 15 11:59:14 mail amavis[5571]: No ext program for .doc, tried: ripole
Feb 15 11:59:14 mail amavis[5571]: No decoder for .F
Feb 15 11:59:14 mail amavis[5571]: No decoder for .doc
Feb 15 11:59:14 mail amavis[5571]: No decoder for .zoo
```

Amavis doesn't support some archive types because viruses spread in attachments (phishing)

So we need to install packages to allow amavis to extract the content of attachments

2. Installing required packages for scanning attachments

```
apt install arj bzip2 cabextract cpio rpm2cpio file gzip lhasa nomarch pax rar unrar p7zip-full unzip zip
lzip lzip liblzip4-tool lzop unrar-free
```

Amavis works as an smtp proxy , which means an email is fed to it through SMTP , checked and returned back to the mail server , through a new smtp connection so I have to tell postfix to enable content filtering by sending every incoming mail to Amavis

Amavis listens on localhost and port 10024

Configuring Postfix (/etc/postfix/main.cf)

```
postconf -e 'content_filter = smtp-amavis:[127.0.0.1]:10024'
```

```
postconf -e 'smtpd_proxy_options = speed_adjust'
```

Speed adjust will delay the postfix connection to the content filter until the entire mail has been received

-Configure postfix to use a special smtp client component called smtp-amavis to deliver email messages to amavis

Add to the end of /etc/postfix/master.cf

```
smtp-amavis unix - - n - 2 smtp
  -o syslog_name=postfix/amavis
  -o smtp_data_done_timeout=1200
  -o smtp_send_xforward_command=yes
  -o disable_dns_lookups=yes
  -o max_use=20
  -o smtp_tls_security_level=none
```

```
dovecot unix - n n - - pipe
  flags=DRhu user=vmail:vmail argv=/usr/lib/dovecot/deliver -d ${recipient}
smtp-amavis unix - - n - 2 smtp
  -o syslog_name=postfix/amavis
  -o smtp_data_done_timeout=1200
  -o smtp_send_xforward_command=yes
  -o disable_dns_lookups=yes
  -o max_use=20
  -o smtp_tls_security_level=none
```

-Add some lines that tell postfix to run an additional smtp daemon that listens on localhost and port 10025 , to receive the emails back from amavis

```
127.0.0.1:10025 inet n - n - - smtpd
  -o syslog_name=postfix/10025
  -o content_filter=
  -o mynetworks_style=host
  -o mynetworks=127.0.0.0/8
  -o local_recipient_maps=
  -o relay_recipient_maps=
  -o strict_rfc821_envelopes=yes
  -o smtp_tls_security_level=none
  -o smtpd_tls_security_level=none
  -o smtpd_restriction_classes=
  -o smtpd_delay_reject=no
  -o smtpd_client_restrictions=permit_mynetworks,reject
  -o smtpd_helo_restrictions=
  -o smtpd_sender_restrictions=
  -o smtpd_recipient_restrictions=permit_mynetworks,reject
  -o smtpd_end_of_data_restrictions=
  -o smtpd_error_sleep_time=0
  -o smtpd_soft_error_limit=1001
  -o smtpd_hard_error_limit=1000
  -o smtpd_client_connection_count_limit=0
  -o smtpd_client_connection_rate_limit=0
  -o
receive_override_options=no_header_body_checks,no_unknown_recipient_checks,no_address_map
pings
```

Restart postfix

Integrate amavis which is just an interface and not an antivirus to clamav which is an antivirus

#### 5. Installing ClamAV

```
apt install clamav clamav-daemon
```

Freshclam is a daemon that periodically checks for virus database definitions updates , installs them and notify clamav daemon to refresh its in memory virus database cache

```
Systemctl status clamav-freshclam
```

```
Systemctl status clamav-daemon
```

E : if it says condition failed that means the virus signatures are not downloaded yet

Wait and restart clamav-daemon

#### 6. Turning on virus-checking in Amavis.

```
In /etc/amavis/conf.d/15-content_filter_mode
```

```
@bypass_virus_checks_maps = (  
    \%bypass_virus_checks, \@bypass_virus_checks_acl, \$bypass_virus_checks_re);
```

Uncomment:

```
@bypass_virus_checks_maps = (  
    \%bypass_virus_checks, \@bypass_virus_checks_acl, \$bypass_virus_checks_re);
```

-The communication between Amavis and clamav will be done through socketfile , and we need to add the user to the amavis group so they have the necessary permissions on the socket file

```
Usermod -aG amavis clamav
```

#### 7. Restarting Amavis and ClamAv

```
systemctl restart amavis; systemctl restart clamav-daemon
```

8- Check the communication between amavis and clamav

```
Journalctl -eu amavis
```

```
Feb 15 12:11:48 mail amavis[8436]: Using primary internal av scanner code for ClamAV-clamd
```

## Testing Amavis and clamav :

Send an email with a malicious attachment

Eicar.org for the file

Then send it to our mail server user1.crystalmind.academy

We can see that the mail server blocked it

Tail -f /var/log/mail.log

In another access : journalctl -eu amavis

```
Feb 15 12:25:22 mail amavis[661]: amavisd-new.
Feb 15 12:25:22 mail systemd[1]: Started LSB: Starts amavisd-new mailfilter.
Feb 15 12:25:22 mail amavis[1244]: No $altermme,          not using it
Feb 15 12:25:22 mail amavis[1244]: No ext program for .deb, tried: ar
Feb 15 12:25:22 mail amavis[1244]: No ext program for .zoo, tried: zoo
Feb 15 12:25:22 mail amavis[1244]: No ext program for .doc, tried: ripole
Feb 15 12:25:22 mail amavis[1244]: No decoder for .F
Feb 15 12:25:22 mail amavis[1244]: No decoder for .doc
Feb 15 12:25:22 mail amavis[1244]: No decoder for .zoo
Feb 15 12:25:22 mail amavis[1244]: Using primary internal av scanner code for ClamAV-clamd
Feb 15 12:25:22 mail amavis[1244]: Found secondary av scanner ClamAV-clamscan at /usr/bin/clamscan
Feb 15 12:27:06 mail amavis[1268]: (01268-01) Blocked INFECTED (Win.Test.EICAR_HDB-1) {DiscardedOpen
lines 166-177/177 (END)}
rcpt=1 (queue active)
Feb 15 12:27:06 mail postfix/smtpd[2901]: disconnect from mail-41104.protonmail.ch[185.70.41.104] ehlo=2 starttls=1 mail=1 rcpt=1 data=1 quit=1 commands=7
Feb 15 12:27:06 mail amavis[1268]: (01268-01) Blocked INFECTED (Win.Test.EICAR_HDB-1) {DiscardedOpen
Relay,Quarantined}, [185.70.41.104]:43342 [51.89.119.103] <andrei.cma@protonmail.com> -> <user1@crystalmind.academy>, quarantine: P/virus-PNg8Ttv6V1Azi, Queue-ID: DDA4617AD1C, Message-ID: <AF_nhm7NcK6
egJUhudjreRNU3GiUrrUSv2pS2_fI0HuTqdGHBGC1Fnvfpk6zlkRmtzYe1bSrC6HuRFZrdm1kafr4VXVdZE43z8s5CrK3BM=@protonmail.com>, mail_id: PNg8Ttv6V1Azi, Hits: -, size: 4378, dkim_sd=protonmail:protonmail.com, 103 ms
Feb 15 12:27:06 mail postfix/amavis/smtp[2914]: DDA4617AD1C: to=<user1@crystalmind.academy>, relay=127.0.0.1[127.0.0.1]:10024, delay=0.93, delays=0.81/0.01/0/0.1, dsn=2.7.0, status=sent (250 2.7.0 Ok,
discarded, id=01268-01 - INFECTED: Win.Test.EICAR_HDB-1)
Feb 15 12:27:06 mail postfix/qmgr[2896]: DDA4617AD1C: removed
```

## Fighting against spam : postfix access restriction

# Fighting Against Spam

1. Postfix Access Restrictions
2. Spam Filtering Systems: SpamAssassin, Rpamd

Telnet and sending a text to ourselves

```

telnet 192.168.0.1 25
Trying 192.168.0.1...
Connected to 192.168.0.1 (192.168.0.1).
Escape character is '^].
220 mail.example.com ESMTP Postfix
HELO mail.example.com
250 mail.example.com
MAIL FROM:<me@example.com>
250 2.1.0 Ok
RCPT TO:<me@example.com>
250 2.1.5 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
To:<me@example.com>
From:<me@example.com>
Subject: Test Mail
This is a just test message
.
250 2.0.0 Ok: queued as 301AE20034
QUIT
221 2.0.0 Bye

```



```

# <-smtpd_client_restrictions
# <-smtpd_helo_restrictions
#
# <-smtpd_sender_restrictions
#
# <-smtpd_recipient_restrictions
#
# <-smtpd_data_restrictions
#
# <-header_checks
#
# <-body_checks
#

```

Restriction list will keep checking until a decision is made to permit or deny or defer (try again later)

Smtp hello , sender , recipient restriction are checked before the body of the mail is received or sent

Therefore the spam might be rejected early in the smtpd process without consuming additional time and resources

#### **Adding those restriction :**

We're adding it in the postfix config  
vim /etc/postfix/main.cf

The message has to pass all the restrictions

```

smtpd_helo_required = yes
smtpd_helo_restrictions =
  permit_mynetworks,
  permit_sasl_authenticated,
  reject_invalid_helo_hostname,
  reject_non_fqdn_helo_hostname,
  reject_unknown_helo_hostname,
  permit

smtpd_sender_restrictions =
  permit_mynetworks,
  permit_sasl_authenticated,
  reject_unknown_sender_domain,
  reject_non_fqdn_sender,
  reject_unknown_reverse_client_hostname,
  reject_unknown_client_hostname,
  permit

smtpd_recipient_restrictions =
  permit_mynetworks,
  permit_sasl_authenticated,
  reject_unauth_destination,
  reject_unauth_pipeline,
  reject_unknown_recipient_domain,
  reject_non_fqdn_recipient,
  check_client_access hash:/etc/postfix/rbl_override,
  reject_rhsbl_helo dbl.spamhaus.org,
  reject_rhsbl_reverse_client dbl.spamhaus.org,
  reject_rhsbl_sender dbl.spamhaus.org,
  reject_rbl_client zen.spamhaus.org,
  permit

```



## **Smtpd helo restriction :**

```
smtpd_helo_required = yes
smtpd_helo_restrictions =
    permit_mynetworks,
```

Unknown hello hostname : don't have dns or mx records

Reject\_non\_fqdn\_hostname : non fully qualified hostname required by rfc

### **Sender :**

Reject\_unknown\_sender\_domain = unknown dns record or record in general

Reject\_unknown\_reverse\_hostname = don't have PTR record , this could lead to false positives because they're legit mail servers that don't have PTR record set

Reject\_unknown\_client\_hostname = domain to ip

### **Recipient :**

-Permit\_my\_network = permit senders from our own network

-permit\_sasl\_authentication = senders that are already authenticated using sasl

-Reject\_unauth\_destination = tells postfix to reject emails if the recipient domains are not hosted locally and that means that postfix is not the final destination , without this our server would be an open relay

-reject\_unauth\_pipeline = stop mail from bulk mail software , that improperly uses pipelining in order to speed deliveries without waiting for authorization

- reject\_unknown\_recipient\_domain : reject the email if our mail server is not the final destination

## **Fighting against spam using public RBL's :**

The restrictions that we set are kinda basic because servers that respect the standards will still send spams

We can use RBS , real time blacklist , dns based list

You can check it online using debouncer

<http://debouncer.com/>

Free blacklisting services : spamhaus.org

It will reject the ip address if its blacklisted

In the recipient :

```
reject_non_fqdn_recipient,  
reject_rhsbl_helo dbl.spamhaus.org,  
reject_rhsbl_reverse_client dbl.spamhaus.org,  
reject_rhsbl_sender dbl.spamhaus.org,  
reject_rbl_client zen.spamhaus.org,  
permit
```

Create a whitelist for legitimate domains that are blacklisted online :

vim /etc/postfix/rbl\_override

Write a whitelisted domain that will be allowed to send emails to our server

```
gooddomain.com  OK  
ilikeit.com     OK  
~  
~
```

Postfix won't use this file, but postmap will

Postmap /etc/postfix/rbl\_override

We can check the file in the postfix directory

```
root@mail:~# ls /etc/postfix/  
dynamicmaps.cf      main.cf.proto      mysql_virtual_email2email.cf  postfix-files.d  
dynamicmaps.cf.d    makedefs.out      mysql_virtual_forwardings.cf  postfix-script  
main.cf             master.cf        mysql_virtual_mailboxes.cf   rbl_override  
main.cf-17.2.2021   master.cf.proto   post-install                rbl_override.db  
main.cf-with-restrictions  mysql_virtual_domains.cf  postfix-files  sasl
```

Then add this to recipient

```
check_client_access hash:/etc/postfix/rbl_override  
reject_rhsbl_helo dbl.spamhaus.org,
```

Systemctl restart postfix

It's a good idea to test the server after such modifications and check the log file meantime

### *Config - Postfix Access Restrictions*

```
.      smtpd_helo_required = yes  
.      smtpd_helo_restrictions =  
.          permit_mynetworks,  
.          permit_sasl_authenticated,  
.          reject_invalid_helo_hostname,  
.          reject_non_fqdn_helo_hostname,  
.          reject_unknown_helo_hostname,
```

```

        permit

smtpd_sender_restrictions =
    permit_mynetworks,
    permit_sasl_authenticated,
    reject_unknown_sender_domain,
    reject_non_fqdn_sender,
    reject_unknown_reverse_client_hostname,
    reject_unknown_client_hostname, #could trigger false-positives
    permit

smtpd_recipient_restrictions =
    permit_mynetworks,
    permit_sasl_authenticated,
    reject_unauth_destination,
    reject_unauth_pipelining,
    reject_unknown_recipient_domain,
    reject_non_fqdn_recipient,
    check_client_access hash:/etc/postfix/rbl_override,
    reject_rhsbl_helo dbl.spamhaus.org,
    reject_rhsbl_reverse_client dbl.spamhaus.org,
    reject_rhsbl_sender dbl.spamhaus.org,
    reject_rbl_client zen.spamhaus.org,
    permit

```

## Intro to spam filtering systems :

**Rspamd** is an external filtering system

- **Rspamd** evaluates the messages by a number of rules including keywords in the mail contents matched by regular expressions, statistical analysis, and custom services such as domain or IP blacklists.
- Rspamd is a permanent process that runs on the mail server.
- It listens to connections from Postfix using the mail filter (**milter**) protocol. Every time an email enters the system, Postfix will send it to Rspamd to have its content checked. Rspamd runs a lot of checks on the email and computes a **total score**. **The higher the score, the more likely the email would be considered spam.**
- According to this spam score and the user's settings, Rspamd recommends an action for the mail server to apply to the message, for example, to pass, reject or add a header.

**Instaaling and cong rspamd :**

We assume there is a working mail system consisting of POSTFIX MTA , dovecot IMAP server , and dns is set up correctly as well

Install redis : it is used by rspamd to store non volatile data and cache volatile data  
Apt update && apt install redis-server

Install rspamd :

```
## INSTALLING RSPAMD AND POSTFIX INTEGRATION ##
# All commands are run as root
```

1. Installing Redis as storage for non-volatile data and as a cache for volatile data  
apt update && apt install redis-server

2. Adding the repository GPG key  
wget -O- https://rspamd.com/apt-stable/gpg.key | sudo apt-key add -

3. Enabling the Rspamd repository  
echo "deb http://rspamd.com/apt-stable/ \$(lsb\_release -cs) main" | sudo tee -a /etc/apt/sources.list.d/rspamd.list

4. Installing Rspamd  
apt update && apt install Rspamd  
Pgrep -a rspamd  
Cd /etc/rspamd (do not change settings here because software update will change them)  
To not overide the changes in rspamd.conf we should create config files in :  
local.d/  
Override.d/  
Now lets create the persistent conf files bellow :

5. Configuring the Rspamd normal worker to listen only on localhost interface  
vim /etc/rspamd/local.d/worker-normal.inc  
bind\_socket = "127.0.0.1:11333";

6. Enabling the milter protocol to communicate with postfix:  
vim /etc/rspamd/local.d/worker-proxy.inc  
bind\_socket = "127.0.0.1:11332";  
milter = yes;  
timeout = 120s;  
upstream "local" {  
default = yes;  
self\_scan = yes;  
}

#### 1. Milter Protocol:

Milter operates as a protocol that allows external programs (milters) to interact with the email processing pipeline of an MTA, such as Postfix. This interaction occurs at different stages of the email delivery process.

7. Configure postfix to use Rspamd  
postconf -e "milter\_protocol = 6"  
postconf -e "milter\_mail\_macros = i {mail\_addr} {client\_addr} {client\_name} {auth\_authen}"  
postconf -e "milter\_default\_action = accept"  
postconf -e "smtpd\_milters = inet:127.0.0.1:11332"  
postconf -e "non\_smtpd\_milters = inet:127.0.0.1:11332"

8. Restarting Rspamd and Postfix  
systemctl restart rspamd; systemctl restart postfix

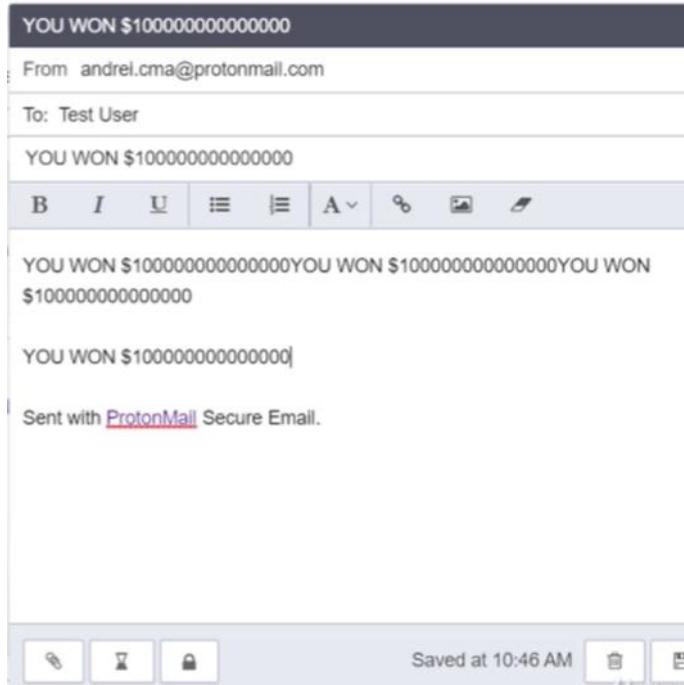
## Configuring and testing rspamd :

Real time rspamd log file :

Tail -f /var/log/rspamd/rspamd.log

Then we send an email from gmail to our mail server :

It is important to craft the email to look like spam



Then we go check the logs in our rspamd

```
iADPi-x1GjIqNDc3NtXfkNa0ppcONFZcvT0U8pxVwBmWESmfkXMRdDpmKeQ6oNYJ2Gnc2AUQ=@protonmail.com>, qid: <5CDED1  
.70.40.137, from: <andrei.cma@protonmail.com>, (default: F (no action): [2.90/15.00] [SUBJ_ALL_CAPS(1.80  
E64_TEXT_BOOGUS(1.00){},SUBJECT_HAS_CURRENCY(1.00){},DMARC_POLICY_ALLOW(-0.50){protonmail.com;quarantine;  
-0.20}{protonmail.com:s=protonmail;},R_SPF_ALLOW(-0.20){+ip4:185.70.40.0/24;},MIME_BASE64_TEXT(0.10){,M  
{multipart/alternative;text/plain;},ARC_NA(0.00){},ASN(0.00){asn:62371, ipnet:185.70.40.0/24, country:CH  
.00}{protonmail.com:+;},FREEMAIL_ENVFROM(0.00){protonmail.com;},FREEMAIL_FROM(0.00){protonmail.com;},FREE  
.00){protonmail.com;},FROM_EQ_ENVFROM(0.00){},FROM_HAS_DN(0.00){},HAS_PHPMAILER_SIG(0.00){},HAS_REPLYTO(  
a@protonmail.com;},MID_RHS_MATCH_FROM(0.00){},MIME_TRACE(0.00){0:+;1:+;2:~;},RCPT_COUNT_ONE(0.00){1;},RCV  
.00){0;},RCVD_IN_DNSWL_NONE(0.00){185.70.40.137:from;},REPLYTO_EQ_FROM(0.00){},RWL_MAILSPIKE_VERYGOOD(0.  
37:from;},TO_DN_ALL(0.00){},TO_MATCH_ENVRCPT_ALL(0.00){}}], len: 2367, time: 641.697ms, dns req: 20, dig  
cf4d5fcfd32d625243798df6>, rcpts: <user1@crystalmind.academy>, mime_rcpts: <user1@crystalmind.academy>  
2021-02-11 08:46:54 #4248(rspamd_proxy) <b3e88c>; proxy; rspamd_protocol_http_reply: regexp statistics: 0  
scanned, 4 regexps matched, 174 regexps total, 49 regexps cached, 0B scanned using pcre, 2.10KiB scanned  
2021-02-11 08:46:54 #4248(rspamd_proxy) <fa3c62>; proxy; proxy_milter_finish_handler: finished milter cor  
2021-02-11 08:47:16 #4249(controller) <imhkkk>; monitored; rspamd_monitored_dns_cb: DNS query blocked on  
m (127.0.0.1 returned), possibly due to high volume
```

The SUBJ\_ALL\_CAPS : is a symbol , the 1.80 is the score that will added to the total score to determine if the email will be accepted or rejected

If we want to change the standard scoring for a symbol

Vim /etc/rspamd/local.d/groups.conf

```

77.40.137.11:587 [user1@crystalmind.academy] (getmail) [2.50/15.00] [B0B5_ALL_CARS(1.00)
E64_TEXT_BOGUS(1.00){},SUBJECT_HAS_CURRENCY(1.00){},DMARC_POLICY_ALLOW(-0.50){protonmail.com;quarantine; -0.20){protonmail.com:s=protonmail;},R_SPF_ALLOW(-0.20){+ip4:185.70.40.0/24;},MIME_BASE64_TEXT(0.10){},M {multipart/alternative;text/plain;},ARC_NA(0.00){},ASN(0.00){asn:62371, ipnet:185.70.40.0/24, country:CH .00}{protonmail.com:+;},FREEMAIL_ENVFROM(0.00){protonmail.com;},FREEMAIL_FROM(0.00){protonmail.com;},FREI .00){protonmail.com;},FROM_EQ_ENVFROM(0.00){},FROM_HAS_DN(0.00){},HAS_PHPMAILER_SIG(0.00){},HAS_REPLYTO(( a@protonmail.com;),MID_RHS_MATCH_FROM(0.00){},MIME_TRACE(0.00){0:+;1:+;2:+;},RCPT_COUNT_ONE(0.00){1;},RC .00){0;},RCVD_IN_DNSWL_NONE(0.00){185.70.40.137:from;},REPLYTO_EQ_FROM(0.00){},RWL_MAILSPIKE_VERYGOOD(0.3 7:from;},TO_DN_ALL(0.00){},TO_MATCH_ENVRCPT_ALL(0.00){}}], len: 2367, time: 641.697ms, dns req: 20, dige cf4d5fc32d625243798df6>, rcpts: <user1@crystalmind.academy>, mime_rcpts: <user1@crystalmind.academy> 2021-02-11 08:46:54 #4248(rspamd_proxy) <>3e88c>; proxy; rspamd_protocol_http_reply: regexp statistics: ( scanned, 4 regexps matched, 174 regexps total, 49 regexps cached, 0B scanned using pcre, 2.10KiB scanned 2021-02-11 08:46:54 #4248(rspamd_proxy) <fa3c62>; proxy; proxy_milter_finish_handler: finished milter cor 2021-02-11 08:47:16 #4249(controller) <imhkkk>; monitored; rspamd_monitored_dns_cb: DNS query blocked on m (127.0.0.1 returned), possibly due to high volume
]

symbols {
    "SUBJECT_HAS_CURRENCY"{
        weight = 5.0;
    }
}

```

Systemctl restart rspamd

#### The actions recommended by rspamd :

Cat /etc/rspamd/actions.conf

Actions or final decisions based on total score

```

actions {
    reject = 15; # Reject when reaching this score
    add_header = 6; # Add header when reaching this score
    greylist = 4; # Apply greylisting when reaching this score (will emit `so

```

Greylisting is a mechanism where the email is temporarily rejected and wait for the sending server to try again , after a waiting time greylisting will accept the email , the idea is to reject email from systems that don't have sending queue , after some time the email will be accepted

#### Modify or alter the standard actions scores :

Vim /etc/rspamd/local.d/actions.conf

So we're kinda making it stricter

Typically, the `X-Spam` header includes a spam score or some indication of the likelihood that email is spam. The value may be numeric, with a higher score indicating a higher likelihood of spam. Additionally, it may include information about the tests or criteria that contributed to the spam score.

```
reject = 10
add_header = 4
greylist = 8
~
~
~
~
~
~
~
~
-- INSERT --
```

Greylist is technically soft reject

If we send the email again , we can see the X-SPAM header in thunderbird

```
MIME-Version 1.0
Content-Type multipart/alternative; boundary="b1_DDBdXdPSvMKdZkKujMzijLobZHj8F5HJrjDOMja8"
X-Spam-Status No, score=-0.3 required=10.0 tests=ALL_TRUSTED,DKIM_SIGNED,
DKIM_VALID,DKIM_VALID_AU,DKIM_VALID_EF,FREEMAIL_FROM,HTML_MESSAGE,
LOTS_OF_MONEY,MONEY_FREEMAIL_REPTO,SUBJ_ALL_CAPS shortcircuit=no
autolearn=disabled version=3.4.4
Checker-Version SpamAssassin 3.4.4 (2010-01-24) on mailout.protonmail.ch
X-Spam Yes
```

**Adding more points for the FREEMAIL FROM :**

vim /etc/rspamd/local.d/groups.conf

```
symbols {
    "SUBJECT_HAS_CURRENCY"{
        weight = 5.0;
    }
    "FREEMAIL_FROM"{
        weight = 4.0
    }
}
```

We restart rspamd and then we send the email again

And the new email will be rejected

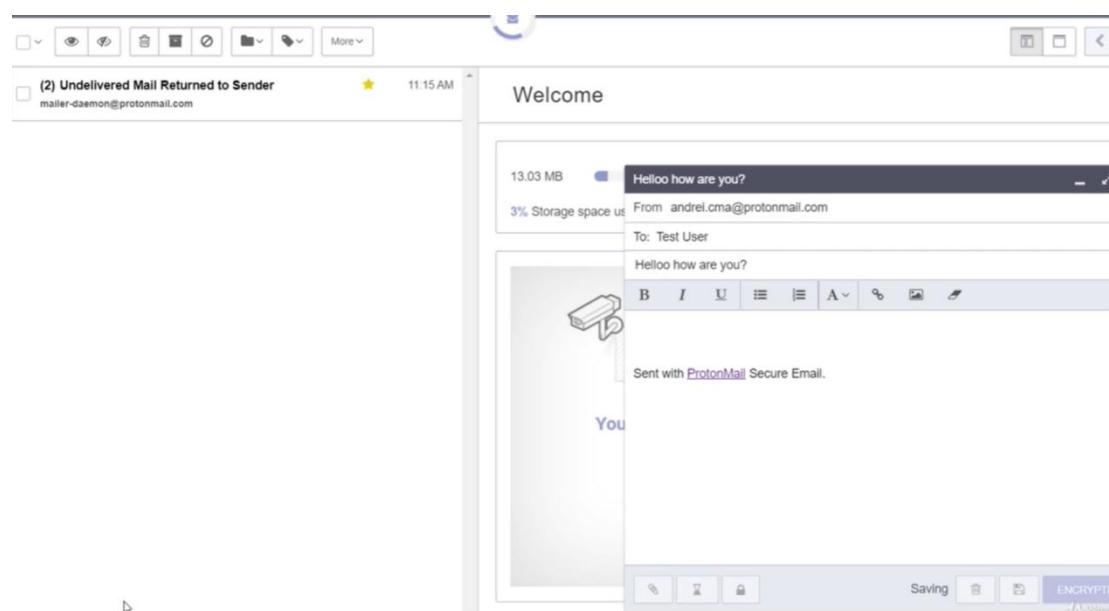
**For troubleshooting purposes , add a header will all the tests and scores to the mail as you see in the logs :**

Vim /etc/rspamd/override.d /milter\_headers.conf

```
extendend_spam_headers = true;  
~  
~  
~  
~  
~  
~  
~  
~  
-- INSERT --
```

Restart rspamd

Send new email



Mail headers in thunderburd looks like the logs for rspamd

```
X-Spamd-Result: default: False [6.10 / 10.00]; HAS_REPLYTO(0.00)[andrei.cma@protonmail.com];  
    FREEMAIL_FROM(4.00)[protonmail.com]; R_SPF_ALLOW(-0.20)[+ip4:185.70.40.0/24];  
    MIME_BASE64_TEXT_BOGUS(1.00)[]; TO_DN_ALL(0.00)[]; DKIM_TRACE(0.00)  
    [protonmail.com:+]; DMARC_POLICY_ALLOW(-0.50)[protonmail.com,quarantine];  
    MIME_BASE64_TEXT(0.10)[]; RCVD_COUNT_ZERO(0.00)[0]; FROM_EQ_ENVFROM(0.00)[];  
    MIME_TRACE(0.00)[0:+,1:+,2:~]; SUBJECT_ENDS_QUESTION(1.00)[]; ASN(0.00)[asn:62371,  
    ipnet:185.70.40.0/24, country:CH]; MID_RHS_MATCH_FROM(0.00)[]; FREEMAIL_ENVFROM(0.00)  
    [protonmail.com]; ARC_NA(0.00)[]; R_DKIM_ALLOW(-0.20)[protonmail.com:s=protonmail];  
    REPLYTO_FQ_FROM(0.00)[]; FROM_HAS_DN(0.00)[]; TO_MATCH_ENVRCPT_ALL(0.00)[].
```

### **Postfix log monitoring using pflogsumm :**

The standard log is tail -f /var/log/mail.log

Pflogsumm \; gives a summary of postfix logs

Apt install pflogsumm

### **Generate a report for today :**

Pflogsumm -d today /var/log/mail.log

For yesterday just change to yesterday

For week without -d

Pflogsumm /var/log/mail.log | less or > mail\_report.txt

### **To report problems first in the report :**

Pflogsumm /var/log/mail.log --problems-first | less

### **Full reason of rejection of the mail :**

Pflogsumm /var/log/mail.log --rej-add-from --verbose-msg-detail | pipeless

We will see message bound details and message reject details

Per-Hour Traffic Summary					
time	received	delivered	deferred	bounced	rejected
0000-0100	0	0	0	0	0
0100-0200	0	0	0	0	0
0200-0300	0	0	0	0	0
0300-0400	0	0	0	0	0

Pay special attention to message reject details , to check the reasons and if they're any false positives

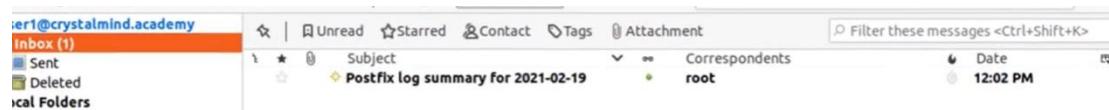
```

message reject detail
-----
RCPT
  554 5.7.1 <spameri@tiscali.it>: Relay access denied; from=<spameri@tiscali.it> to=<spameri@tiscali.it> proto=ESMTP helo=<SVRTERM> (total: 1)
    1   190.0.36.70 (spameri@tiscali.it)
  554 5.7.1 <spameri@tiscali.it>: Relay access denied; from=<spameri@tiscali.it> to=<spameri@tiscali.it> proto=ESMTP helo=<WIN-QGF0VKPDT6T> (total: 2)
    1   165.231.143.235 (spameri@tiscali.it)
    1   196.242.244.25 (spameri@tiscali.it)
  554 5.7.1 <spameri@tiscali.it>: Relay access denied; from=<spameri@tiscali.it> to=<spameri@tiscali.it> proto=ESMTP helo=<admin-PC> (total: 1)
    1   43.245.203.243 (spameri@tiscali.it)
  554 5.7.1 <spameri@tiscali.it>: Relay access denied; from=<spameri@tiscali.it> to=<spameri@tiscali.it> proto=ESMTP helo=<win2008r2RDP> (total: 1)
    1   37.49.225.252 (spameri@tiscali.it)
  554 5.7.1 <spameri@tiscali.it>: Relay access denied; from=<spameri@tiscali.it> to=<spameri@tiscali.it> proto=ESMTP helo=<win2012r2RDP> (total: 1)
    1   77.247.110.15 (spameri@tiscali.it)
-----

```

**Send the report to your email address :**

```
Pflogsumm /var/log/mail.log --rej-add-from --verbose-msg-detail | mail -s "Postfix log summary for $(date +%F)" user1@crystalmind.academy
```



**Send automaticcally the report everyday :**

```
Crontab -e
```

```
2
```

```
0 minute 8 hour (morning)
```

```
0 8 * * *
```

```
0 8 * * * pflogsumm -d yesterday /var/log/mail.log --rej-add-from --verbose-msg-detail | mail -s "Postfix log summary for $(date +%F)" user1@crystalmind.academy
```

```
0 8 * * * pflogsumm /var/log/mail.log --rej-add-from --verbose-msg-detail | mail -s "Postfix log summary for $(date +%F)" user1@crystalmind.academy
```

## Automating Linux Admin tasks :

### Intro to Ansible :

#### Ansible - Automate for Everyone

- Ansible is a configuration management system used to configure, automate, monitor, and troubleshoot devices in **large networks**;
- It's written in Python and is **OpenSource** under **GPL Licence**;
- There is a **control node** and many **managed nodes** (network devices we manage with Ansible). **Windows OS is not supported as a control node at this moment (only WLS)**;
- Ansible uses an **agentless architecture**. No application is installed on the managed nodes and no daemon, process or agent is running;
- Ansible uses **SSH** for device configuration. Any system that can be configured using SSH can also be configured using Ansible;
- When managing Windows it users native PowerShell remote support instead of SSH;

Ansible modules :

[https://docs.ansible.com/ansible/2.9/modules/list\\_of\\_all\\_modules.html](https://docs.ansible.com/ansible/2.9/modules/list_of_all_modules.html)

Install ansible on the controlling machine :

Ansible is a linux software , can be only run there , the managed nodes can be anything

This vm is called ansible (the control node)

Sudo apt update && sudo apt install ansible

Prerequisites :

One ansible control node , and two managed nodes ( 3 or 4 linux vms )

Its necessary that the control node has ssh access to the managed , and because we r gonna do configuration tests , we need root access to the managed  
We don't need root access for the control

The infrastructure :

```
ansible@ubuntu-s-1vcpu-1gb-fra1-01:~$ cat servers.txt
207.154.254.221
167.172.186.63
139.59.145.232
```

#### Testing the ssh connection to the managed :

-l : login\_name : Specifies the user to log in as on the remote machine. This also may be specified on a per-host basis in the configuration file.

```
ssh -l student -p 22 IP
```

Then we enter the password for that user

```
#do the same test for all managed
```

## Ansible Inventory file :

### Ansible Components

- **Inventory** is a **list of managed nodes**. An inventory file is also sometimes called a "host file". The inventory can specify information such as IP address, hostname or domain-name for each managed node. Default location: /etc/ansible/hosts  
An inventory file can be formatted as an INI or YAML file.
- **Modules** are the **units of code ansible executes**. Each module has a particular use, from administering users on a specific type of database to managing VLAN interfaces on a specific type of network device or to installing, configuring and starting a specific server like for example Apache2 on a Linux distribution.
- **Tasks** are **units of action** in Ansible. You can execute a single task once with an Ad-Hoc command or multiple tasks in a playbook.
- **Playbooks** are ordered lists of tasks. We can run those tasks in that order repeatedly.
- Playbooks are written in **YAML** and are easy to read, write, share and understand.

Create a hosts file in the inventory file

Vim hosts

```
[servers]
vps1 ansible_host=207.154.254.221
vps2 ansible_host=167.172.186.63
vps3 ansible_host=139.59.145.232
```

Servers : the group

Vps1 : the alias

Ansible\_host = predefined parameter

### List all the created hosts :

Test if ansible will parses the hosts from the hosts file : ./hosts is the path to the inventory file

```
ansible -i ./hosts --list-hosts-all
```

```
ansible@ubuntu-s-1vcpu-1gb-fra1-01:~$ ansible -i ./hosts --list-hosts all
hosts (3):
vps1
vps2
vps3
```

### Test how ansible communicates with the managed nodes :

We can see that ansible detected the nodes from up , but lets test the connection

```
ansible -i ./hosts vps1 -m ping -u student -k
```

*-m ping : use the ping module but this one checks SSH authentication to each managed nodes*

*-k : ask for the password*

*Student : the remote user*

E :

```
ansible@ubuntu-s-1vcpu-1gb-fra1-01:~$ ansible -i ./hosts vps1 -m ping -u student -k
SSH password:
vps1 | FAILED! => {
    "msg": "to use the 'ssh' connection type with passwords, you must install the sshpass program"
}
```

R :

```
sudo apt install sshpass
```

E :

```
vps1 | FAILED! => {
    "msg": "Using a SSH password instead of a key is not possible because Host Key checking is enabled and sshpass does not support this. Please add this host's fingerprint to your known_hosts file to manage this host."
}
```

R :

We need to disable the hostkey checking in ansible config file , check where its located first

Ansible --version

If its "none" :

```
ansible@ubuntu-s-1vcpu-1gb-fra1-01:~$ ansible --version
ansible 2.10.8
  config file = None
```

Sudo apt install ansible-core

This will help us generate a configuration file , we redirect to the standard config name and we move it to the standard ansible location

```
ansible-config init --disable > ansible.cfg
```

```
sudo mkdir /etc/ansible
```

```
sudo mv ansible.cfg /etc/ansible
```

**Check if the config file is created :**

```
ansible --version
```

```
ansible@ubuntu-s-1vcpu-1gb-fra1-01:~$ ansible --version
ansible [core 2.12.0]
  config file = /etc/ansible/ansible.cfg
```

**Configuring ansible :**

```
Sudo vim /etc/ansible/ansible.cfg
```

Remember we did all this to change the hostkey to false so ansible can check connection to the hosts

```
# uncomment this to disable SSH key host checking
host_key_checking = False
```

Then we run again

```
ansible -i ./hosts vps1 -m ping -u student -k
```

```
ansible@ubuntu-s-1vcpu-1gb-fra1-01:~$ ansible -i ./hosts vps1 -m ping -u student -k
SSH password:
vps1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

Its working we have got pong , ansible authenticated succefully

**Gather info about hosts in JSON format :**

```
ansible -i ./hosts servers -m setup -u student -k
```

**For a group test with same credentials :**

```
ansible -i ./hosts hosts -m ping -u student -k (you can use all for all hosts in inventory)
```

For automation :

```
vim hosts
```

**Remember :** never use plaintext for passwords , use ansible volt or public key encryption

The bellow are behavioral inventory variables

```
[servers]
vps1 ansible_host=207.154.254.221
vps2 ansible_host=167.172.186.63
vps3 ansible_host=139.59.145.232

[servers:vars]
ansible_user=student
ansible_ssh_pass=ansible123-
ansible_port=22
~
```

ansible -i ./hosts servers -m setup

ansible -i ./hosts servers -m ping

"no need for user or -k "

**For group test with not same credentials :**

Remember with -k , it will need to ask for password , this would bad for automation

We gonna set up the conf file to bypass that

vim hosts

Inventory variable can be set per host or group

Unique credentials : ssh key generation script

```
[servers]
vps1 ansible_host=207.154.254.221 ansible_user=student
vps2 ansible_host=167.172.186.63
vps3 ansible_host=139.59.145.232

[servers:vars]
ansible_user=student
ansible_ssh_pass=ansible123-
ansible_port=22
~
```

The vps1 variable will overwrite the group variable

**Ansible adhoc commands : the shell module**

## Ansible Ad-Hoc Commands

- **Ad-Hoc** commands can be used to do quick and simple things like checking the logs, checking if a process is running, or installing a package on a list of servers.
- **Playbooks** are used for big deployments, orchestration or system configuration
- We use the **ansible** command to run an Ad-Hoc command and the **ansible-playbook** command to run a playbook.
- Both Ad-Hoc commands and Playbooks use modules to perform different tasks. Modules are units of code that do the actual work in Ansible.

Ansible controls 200 nodes using adhoc or playbook

-**Adhoc commands** are not reusable like : reboot the server , update the system ...

```
ansible -i ./hosts servers -m shell -a "df -h" -u student -k
```

```
vps2 | CHANGED | rc=0 >>
Filesystem      Size  Used Avail Use% Mounted on
udev            474M   0    474M  0% /dev
tmpfs           99M  936K  98M  1% /run
/dev/vda1        25G  1.9G  23G  8% /
tmpfs           491M   0    491M  0% /dev/shm
tmpfs           5.0M   0    5.0M  0% /run/lock
tmpfs           491M   0    491M  0% /sys/fs/cgroup
/dev/vda15       105M  9.2M  96M  9% /boot/efi
/dev/loop0        56M   56M  0  100% /snap/core18/1885
/dev/loop1        71M   71M  0  100% /snap/lxd/16922
/dev/loop2        31M   31M  0  100% /snap/snapd/9607
/dev/loop3        32M   32M  0  100% /snap/snapd/10707
/dev/loop4        56M   56M  0  100% /snap/core18/1944
/dev/loop5        68M   68M  0  100% /snap/lxd/18150
tmpfs           99M   0    99M  0% /run/user/1000

vps1 | CHANGED | rc=0 >>
Filesystem      Size  Used Avail Use% Mounted on
udev            474M   0    474M  0% /dev
tmpfs           99M  936K  98M  1% /run
/dev/vda1        25G  1.9G  23G  8% /
tmpfs           491M   0    491M  0% /dev/shm
tmpfs           5.0M   0    5.0M  0% /run/lock
tmpfs           491M   0    491M  0% /sys/fs/cgroup
/dev/vda15       105M  9.2M  96M  9% /boot/efi
/dev/loop0        56M   56M  0  100% /snap/core18/1885
/dev/loop1        71M   71M  0  100% /snap/lxd/16922
/dev/loop2        31M   31M  0  100% /snap/snapd/9607
/dev/loop3        32M   32M  0  100% /snap/snapd/10707
/dev/loop4        56M   56M  0  100% /snap/core18/1944
/dev/loop5        68M   68M  0  100% /snap/lxd/18150
tmpfs           99M   0    99M  0% /run/user/1000
```

Processes for each node

**Ansible threads : -f**

Usually ansible uses 5 simultaneous threads , if you have more than 5 hosts , you can edit the number like bellow

```
ansible -i ./hosts servers -m shell -a "lsmem" -f 10          #ram
                                              Free -m          #freeram
                                              Tail -n 3 /etc/passwd #last 3 created users
```

#mac address of the interface with public ip of each host

```
ansible -i ./hosts servers -m shell -a "ip address show dev eth0 | grep ether"
```

```
ansible@ubuntu-s-1vcpu-1gb-fra1-01:~$ ansible -i ./hosts servers -m shell -a "ip address show dev eth0 | grep ether"
vps1 | CHANGED | rc=0 >>
  link/ether c2:62:53:0f:5b:33 brd ff:ff:ff:ff:ff:ff
vps2 | CHANGED | rc=0 >>
  link/ether 9e:1a:14:72:19:25 brd ff:ff:ff:ff:ff:ff
vps3 | CHANGED | rc=0 >>
  link/ether 1e:14:63:3e:1f:de brd ff:ff:ff:ff:ff:ff
```

Cut he mac address : cut -delimeter - field

```
ansible -i ./hosts servers -m shell -a "ip address show dev eth0 | grep ether | cut -d ' ' -f6"
```

```
ansible@ubuntu-s-1vcpu-1gb-fra1-01:~$ ansible -i ./hosts servers -m shell -a "ip address show dev eth0 | grep ether | cut -d' ' -f6"
vps2 | CHANGED | rc=0 >>
9e:1a:14:72:19:25
vps1 | CHANGED | rc=0 >>
c2:62:53:0f:5b:33
vps3 | CHANGED | rc=0 >>
1e:14:63:3e:1f:de
```

Negating the >>

```
ansible@ubuntu-s-1vcpu-1gb-fra1-01:~$ ansible -i ./hosts servers -m shell -a "ip address show dev eth0 | grep ether | cut -d' ' -f6 | grep -v '>>'"
9e:1a:14:72:19:25
c2:62:53:0f:5b:33
1e:14:63:3e:1f:de
```

ansible -i ./hosts servers -m shell -a "ip address show dev eth0 | grep ether | cut -d' ' -f6 | grep -v '>>'"

## Ansible Ad-Hoc commands : the script module

If you have a backup script and you want to run it on each server of your network infrastructure

The classic method : use cp to copy the script to the remote server and run the script there

This cant be done for a 1000 remote host

### The automated method :

supported for linux and windows

- lets first create the intended script : create a backup of the etc folder

vim backup.sh

tar -czf /root/etc-\$(date +\$F).tar.gz /etc

Save the script

- `tar`: This command is used for manipulating archives. In this case, it's creating an archive.
- `-czf`:
  - `--c`: Create a new archive.
  - `--z`: Compress the archive using gzip.
  - `--f`: Specify the name of the archive file.
- `/root/etc-\$(date +\$F).tar.gz`:
  - `/root/etc-`: This is the path where the archive file will be created. It specifies the `/root` directory, and the filename starts with `etc-`.
  - `\$(date +\$F)`: This is a command substitution. It inserts the current date in a specific format into the filename. `\${F}` is a format specifier that represents the year and month (e.g., 2022-02).
  - `.tar.gz`: This is the extension of the archive file. It indicates that the archive is compressed using gzip.
- `/etc`: This is the directory being archived. The `/etc` directory typically contains configuration files and settings for the system.

### **Run it on multiple hosts :**

```
ansible -i ./hosts servers -m script -a "./backup.sh" --become -K  
  
#remember depending on the script , and what does it contain , like in this case we need to run the  
ansible command as root to have access to the etc command , --become to become root
```

### **Check if the backup archive was created :**

We connect to a random one , and check the repos

```
ssh -l student 207.154.254.221 ( hosts ip addr )
```

```
student@ubuntu-s-1vcpu-1gb-fra1-02:~$ sudo ls /root/  
[sudo] password for student:  
etc-2021-01-15.tar.gz snap
```

### **Avoid entering password to become root :**

Add become pass

```
[servers:vars]  
ansible_user=student  
ansible_ssh_pass=ansible123-  
ansible_become_pass=ansible123-  
ansible_port=22
```

Now we can run the ansible command without pass

## **Ansible Ad-hoc commands : The APT module**

Apt module to automate software management

If you use redhat or centos use yum instead of apt

### **Install a software package to all the hosts :**

Lets install nmap

```
ansible -i hosts servers -m apt -a "name=nmap state=present update_cache=true" -u student -k --  
become -K
```

```
#updatecache will update the package cache if its out of date  
#state : desired package state
```

```

ansible@ubuntu-s-1vcpu-1gb-fra1-01:~$ ansible -i hosts -m apt -a "name=nmap state=present update_cache=true"
-u student -k --become -K

student@ubuntu-s-1vcpu-1gb-fra1-01:~$ nmap -p 80 -sV scanme.nmap.org
Command 'nmap' not found, but can be installed with:
  I
sudo snap install nmap # version 7.91, or
sudo apt install nmap # version 7.80+dfsg1-2build1
See 'snap info nmap' for additional versions.
student@ubuntu-s-1vcpu-1gb-fra1-01:~$ █

```

**Remove a package from all hosts :**

```

ansible -i hosts servers -m apt -a "name=nmap state=absent purge=yes" -u student -k --become -K

#u student -k --become -K : not necessary
#purge : force delete of conf files
#--become : is necessary because you need be root on the host to remove packages

```

**Full upgrade of the systems in the hosts :**

```
Ansible -i hosts servers -m apt -a "upgrade=full" --become
```

**Remove dependencies that are not needed anymore :**

```
ansible -i hosts servers -m apt -a "autoremove=yes autoclean=yes" --become
```

## Ansible Ad-hoc commands : the service module

Control services on remote hosts

**Install nginx :**

```
ansible -i hosts servers -m service -a "name=nginx state=present update_cache=true" --become
```

**Making sure it started :**

```
ansible -i hosts servers -m service -a "name=nginx state=started" --become
```

Systemctl status nginx (in the host)

**Reload the service :**

```
ansible -i hosts servers -m service -a "name=nginx state=restarted" --become
```

**Stop the service :**

```
ansible -i hosts servers -m service -a "name=nginx state=stopped" --become
```

**Make the service starts on boot :**

```
ansible -i hosts servers -m service -a "name=nginx enabled=yes" --become
```

## Ansible Ad-hoc commands : the user module

Used to manage groups , user accounts , user attributes

Add = present  
Remove=absent

**Create a group in all remote hosts :**

```
ansible -i ./hosts servers -m groups -a "name=developers state=present" --become
```

In host

```
student@ubuntu-s-1vcpu-1gb-lon1-02:~$ tail -n 3 /etc/group
systemd-coredump:x:999:
student:x:1000:
developers:x:1001:
```

**Remove group from all remote hosts :**

```
ansible -i ./hosts servers -m groups -a "name=developers state=absent" --become
```

**Create a user in all hosts that belongs to a group :**

```
Ansible -i ./hosts servers -m users "name=john state=present groups=developers create_home=yes
comment="new user" shell=/bin/bash generate_ssh_key=yes" --become
```

#developers would be the secondary group of john"

```
student@ubuntu-s-1vcpu-1gb-lon1-02:~$ tail -n 3 /etc/passwd
lxd:x:998:100::/var/snap/lxd/common/lxd:/bin/false
student:x:1000:1000::/home/student:/bin/bash
john:x:1001:1002:new user:/home/john:/bin/sh
```

**Add a user to more than one secondary group :**

```
Ansible -i ./hosts servers -m users "name=john state=present groups=developers,sudo,adm
create_home=yes comment="new user" shell=/bin/bash generate_ssh_key=yes" --become
```

```
student@ubuntu-s-1vcpu-1gb-lon1-02:~$ groups john
john : john adm sudo developers
```

**Task automation and scheduling using cron :**

Schedule command or tasks or as know , cron jobs to run on specific times

Cron is run as daemon and its used to run repetitive task like backup , monitoring disk space usage ...

Crontab : contains users cronjobs  
Ls /var/spool/cron/crontabs/

#manage crontab files  
*Which crontab*

**- Crontab config files for systemwide**

ls -l /etc/crontab

**-User specific cronjobs**

ls /var/spool/cron/crontabs

**- Show the crontab for the current user**

crontab -l

**- Edit the current crontab with date**

crontab -e

23 # m h dom mon dow command

Minute , hour , day of month , month , day of week , command

\* means always , like every week

Minimum time in cron is a minute , if you want for less , create a white loop bash script

#daily backup at 6

0 16 \* \* \* /root/backup.sh

23 # m h dom mon dow command  
24 0 16 \* \* \* /root/backup.sh

1. **Minute (0 - 59):**
  - Denotes the minute at which the command is scheduled to run.
  - For example, `5` means the command will run at the 5th minute of the hour.
2. **Hour (0 - 23):**
  - Specifies the hour at which the command is scheduled to run.
  - For example, `2` means the command will run at 2 AM.
3. **Day of the Month (1 - 31):**
  - Indicates the day of the month on which the command is scheduled to run.
  - For example, `15` means the command will run on the 15th day of the month.
4. **Month (1 - 12 or names):**
  - Specifies the month during which the command is scheduled to run.
  - You can use numeric values (`1` for January, `2` for February, etc.) or use the first three letters of the month's name (e.g., `Jan`, `Feb`).
  - For example, `3` or `Mar` means the command will run in March.
5. **Day of the Week (0 - 7 or names, where both 0 and 7 represent Sunday):**
  - Represents the day of the week on which the command is scheduled to run.
  - You can use numeric values (`0` or `7` for Sunday, `1` for Monday, etc.) or use the first three letters of the day's name (e.g., `Sun`, `Mon`).
  - For example, `4` or `Thu` means the command will run on Thursday.

For different hours : 4,6,8

A cronjob : on weekdays , between 9 and 17 , once per hour

0 9-17 \* \* 1-5 /root/firewall.sh

```
--> ""
23 # m h dom mon dow   command
24 0 6 * * 0 /root/backup.sh
25 0 4,6,10 * * * /root/check_space.sh
26 0 9-17 * * 1-5 /root/firewall.sh
27 10 4,21 */3 * * /root/task.sh
TACERT
```

At minute 10 , at 4 and 21 , every 3 days

@yearly /root/happy\_new\_year

@monthly

@weekly : once a week at midnight on Sunday

@reboot : at boottime

**Cron log file :**

```
tail -f /var/log/syslog in ubuntu
      /var/log/cron in centos
```

```
[root@ubuntu20-04 ~]# tail -f /var/log/syslog
Aug  7 11:38:01 ubuntu20-04 CRON[2981]: (student) CMD (date >> /tmp/date_and_time.txt)
```

**Remove crontab file :**

```
crontab -r -u student
```

```
crontab -l (to check if its removed )
```

Use absolute path when using crontab , common mistakes beginner use is use relative path

#### **Users access to crontab command :**

The crontab allow and deny decides which users are allowed to use the crontab

Man crontab

#### **Generate crontab online :**

<https://crontab-generator.org/>

#### **System cronjobs :**

Move your cron scripts to the bellow directories and they will run as system cronjobs

Ls /etc/cron.

```
student@ubuntu20-04:~$ ls /etc/cron.  
cron.d/ cron.daily/ cron.hourly/ cron.monthly/ cron.weekly/
```

One thing about cronjobs , that if the system is down on the particulare timeslot , when the server is up it will not run the missed cronjob

```
#####
## Task Scheduling using Cron
#####

# editing the current user's crontab file
crontab -e

# listing the current user's crontab file
crontab -l

# removing the current user's crontab file
crontab -r

## COMMON EXAMPLES ##
# run every minute
* * * * * /path_to_task_to_run.sh

# run every hour at minute 15
15 * * * * /path_to_task_to_run.sh

# run every day at 6:30 PM
30 18 * * * /path_to_task_to_run.sh

# run every Monday at 10:03 PM
3 22 * * 1 /path_to_task_to_run.sh
```

```

.
.
.
# run on the 1st of every Month at 6:10 AM
10 6 1 * * /path_to_task_to_run.sh

.
.
.
# run every hour at minute 1, 20 and 35
1,20,35 * * * * /path_to_task_to_run.sh

.
.
.
# run every two hour at minute 10
10 */2 * * * /path_to_task_to_run.sh

.
.
.
# run once a year on the 1st of January at midnight
@yearly /path_to_task_to_run.sh

.
.
.
# run once a month at midnight on the first day of the month
@monthly /path_to_task_to_run.sh

.
.
.
# run once a week at midnight on Sunday
@weekly /path_to_task_to_run.sh

.
.
.
# once an hour at the beginning of the hour
@hourly /path_to_task_to_run.sh

.
.
.
# run at boot time
@reboot /path_to_task_to_run.sh

.
.
.
All scripts in following directories will run as root at that interval:
/etc/cron.hourly
/etc/cron.daily
/etc/cron.hourly
/etc/cron.monthly
/etc/cron.weekly
.
.
```

## Scheduling tasks using Anacron :

Anacron is used to schedule tasks on systems that don't run all the time

### Config anacron :

Vim /etc/anacrontab

Run-parts : runs all the executables in the directory

```
# These replace cron's entries
1      5      cron.daily      run-parts --report /etc/cron.daily
7      10     cron.weekly    run-parts --report /etc/cron.weekly
@monthly      15     cron.monthly   run-parts --report /etc/cron.monthly
```

All the executables that are found in /etc/cron.weekly , will be run at frequency of weekly

10 : is the delay to not overrun the server

Cron.weekly : job identifier , name of the file that would be created under /var/spool/anacron

**Check for syntax errors :**

anacron -T

Sudo anacron -d

```
student@ubuntu20-04:~$ sudo anacron -d
Anacron 2.3 started on 2020-08-07
Will run job `backup' in 1 min.
Job `backup' started
Job `backup' terminated
Normal exit (1 job run)
student@ubuntu20-04:~$ 

student@ubuntu20-04:~$ sudo cat /var/spool/anacron/
backup      cron.daily   cron.monthly  cron.weekly
student@ubuntu20-04:~$ sudo cat /var/spool/anacron/backup
[sudo] password for student:
20200807
```

---

## Bonus section : Bash shell scripting

#question

Answer

#which shell is running  
Echo \$0

#default shell of the user  
cat /etc/passwd

```
vps1:x:1000:1000:vps1:/home/vps1:/bin/bash
```

#shell script

an executable text file that contains shell commands

the junior sysadmin just runs the script , the senior makes the script and know how to access sql databases

linux don't care about extension

#directories where the shell is looking for executables : E : command not found  
Echo \$PATH

run a script from current dir : ./first\_script.sh  
with absolute path : pwd/first\_script.sh

### The bash shebang and comments :

If we don't use shebang , it will run the script using the default shell  
Hard links are names to the same file , same inode number = same file

```
[root@vps1 scripts]# which -a bash  
/bin/bash  
/usr/bin/bash
```

```
student@ubuntu20-04:~/scripts$ ls -li /bin/bash /usr/bin/bash  
918160 -rwxr-xr-x 1 root root 1183448 iun 18 18:44 /bin/bash  
918160 -rwxr-xr-x 1 root root 1183448 iun 18 18:44 /usr/bin/bash
```

```
#!/bin/bash  
#!/bin/python3 (which python3)
```

Chmod 700

Remove permission : chmod -x first\_script.sh

Comment : like python #

In vim : vim ~/.vimrc

### Running bash scripts :

```
./first_script  
Absolute path  
Bash first_script.sh (no need for execution permission )  
  
.first_script = the script will run in a subshell  
source first_script = the script will run in the current shell = . first_script.sh
```

### Variables in bash :

A name for a memory location that contains a value that can be manipulated

Os=linux (no spaces)

Distro="mx linux"

No spaces

no float

no type declaration

Variables cannot start with a number

Getting the value from a variable : referencing vs assigning

Assigning : os=centos

Referencin : echo \$os

Using double quotes : echo "I using \$os" = im using centos

Using single quote : = im using \$os

### Cancelling the special meaning of dollar sign : using / or "

Echo "im using \\$os as \$os"

### Create a variable from different variables :

distro=ubuntu

os=linux

my\_distro = "\$os \$distro"

```
[root@vps1 scripts]# dsitro=ubuntu
[root@vps1 scripts]# os=linux
[root@vps1 scripts]# my_distro = "$os $dsitro"
bash: my_distro: command not found...
[root@vps1 scripts]# my_distro="$os $dsitro"
[root@vps1 scripts]# echo my_distro
my_distro
[root@vps1 scripts]# echo $my_distro
linux ubuntu
```

### Search for a shell variable :

```
[root@vps1 scripts]# set | grep distro
my_distro='linux ubuntu'
```

**Unset a variable :**

Unset distro

**System or environment variables :**

Are usually in CAPITALS

```
[root@vps1 scripts]# echo $USER
root
[root@vps1 scripts]# echo $HOME
/root
[root@vps1 scripts]#
```

User lowercase for user variables

**Make a variable constant and read only :**

```
declare -r logdir="/var/log"
```

#make a script

That shows the word count of the passwd file and the first 5 lines and its last 4 lines , declare the variable as pass\_stat

Pass\_stat.sh

```
1#!/bin/bash
2file="/etc/passwd"
3echo "file count is :"
4wc -l $file
5echo -----
6echo "first 5 lines"
7head -n 5 $file
8echo -----
9echo "last 4 lines"
10tail -n 4 $file
```

```
[root@vps1 scripts]# source pass_stat.sh
file count is :
45 /etc/passwd
-----
first 5 lines
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin/nologin
daemon:x:2:2:daemon:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
-----
last 4 lines
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
avahi:x:70:70:Avahi mDNS/DNS-SD Stack:/var/run/avahi-daemon:/sbin/nologin
tcpdump:x:72:72::/sbin/nologin
vps1:x:1000:1000:vps1:/home/vps1:/bin/bash
```

#### Getting user input :

use read

```
student@ubuntu20-04:~/scripts$ read -p "Enter the IP address" ip
Enter the IP address100.0.0.1
student@ubuntu20-04:~/scripts$ echo $ip
100.0.0.1
student@ubuntu20-04:~/scripts$ ping -c 1 $ip
PING 100.0.0.1 (100.0.0.1) 56(84) bytes of data.
64 bytes from 100.0.0.1: icmp_seq=1 ttl=55 time=124 ms
--- 100.0.0.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 123.884/123.884/123.884/0.000 ms
```

#### Entering a password without showing it in the shell :

read -s -p “Enter a password : “ pswd

Echo \$pswd

## Special Variables and positional arguments :

apt install nginx

Apt is the command or script  
Install first argument  
Nginx second argument

## Special Variables and Positional Arguments

```
./script.sh filename1 dir1 10.0.0.1
```

\$0 is the name of the script itself (script.sh).  
\$1 is the first positional argument (filename1)  
\$2 is the second positional argument (dir1)  
\$3 is the last argument of the script (10.0.0.1)  
\$9 would be the ninth argument and \${10} the tenth

\$# is the number of the positional arguments  
"\$\*" is a string representation of all positional arguments: \$1, \$2, \$3 ....  
\$? is the most recent foreground command exit status

```
student@ubuntu20-04:~/scripts$ ./arguments.sh linux windows mac 10
$0 is ./arguments.sh
$1 is linux
$2 is windows
$3 is mac
$* is linux windows mac 10
$# is 4
```

Applying positional argument :

### WITH POSITIONAL ARGUMENTS YOU DON'T NEED "READ" VARIABLE

Example : takes a file , display its content , then create a compressed copy of it , "this is not an easy one "

```
student@ubuntu20-04:~/scripts$ chmod +x display_and_compress.sh
student@ubuntu20-04:~/scripts$ ls
arguments.sh  dir1          file_stat.sh    myscript1
block_ip.sh   display_and_compress.sh  first_script.sh
student@ubuntu20-04:~/scripts$ ./display_and_compress.sh arguments.sh
Displaying the contents of arguments.sh ...
```

```
1 #!/bin/bash
2 echo "Displaying the contents of $1 ..."
3 sleep 2
4 cat $1
5 echo
6 echo "Compressing $1 ..."
7 sleep 2
8 tar -czvf "$1.tar.gz" $1
```

```
[root@vps1 scripts]# ./display_compress.sh pyscript
Displaying the content of pyscript :
#!/bin/python3
print("I'm the honored one")

The compressed format of the file :
pyscript
[root@vps1 scripts]# ls
display_compress.sh  drl  first_script.sh  malicious_ip.sh  pass_stat.sh  pyscript  pyscript.tar.gz
[root@vps1 scripts]#
```

## If , elif , and else statement :

---

```
if [ some_condition_is_true ]
then
    //execute this code
elif [ some_other_condition_is_true ]
then
    //execute_this_code
else
    //execute_this_code
fi
```

The commands are evaluated sequentially , if the first is true , the others are not evaluated

Exercise :

Display the content of a file only if the type of the file is regular

---

```
1 #!/bin/bash
2 if [ -f $1 ]
3 then
4     cat $1
5 fi
6
```

Man test (for testing conditions ) that's where you can find -f

```
[root@vps1 scripts]# source display_regular.sh pyscript
#!/bin/python3
print("I'm the honored one")
```

Recommendation for strings input or first positional argument to use “\$1”

Extra editing from test commands :

```

-b FILE
    FILE exists and is block special

-c FILE
    FILE exists and is character special

-d FILE
    FILE exists and is a directory

-e FILE
    FILE exists

-f FILE
    FILE exists and is a regular file

```

---

```

1 #!/bin/bash
2 if [ -f "$1" ]
3 then
4     echo "The argument is a file, displaying its contents..."
5     sleep 1
6     cat $1
7 elif [ -d "$1" ]
8 then
9     echo "The argument is directory, running ls -l ..."
10    sleep 1
11    ls -l $1
12 else
13     echo "The argument ($1) is neither a file nor a directory."
14 fi
15

```

### Single bracket and double bracket :

[ test ] vs [[ test ]]

1. Single Brackets:

```

bash                                         ⌂ Copy code

if [ expression ]; then
# commands
fi

```

2. String Comparison:

```

bash                                         ⌂ Copy code

if [ "$var1" = "$var2" ]; then
# strings are equal
fi

```

3. Numeric Comparison:

```

bash                                         ⌂ Copy code

if [ "$num1" -eq "$num2" ]; then
# numeric values are equal
fi

```

4. File Tests:

```

bash                                         ⌂ Copy code

if [ -f "filename" ]; then
# file exists and is a regular file
fi

```

## 2. Extended Test Constructs:

- Supports extended conditional expressions.
- No need for quoting variables.

```
bash
```

Copy code

```
if [[ $var1 == $var2 ]]; then
    # strings are equal
fi
```

## 3. Logical Operators:

- Supports logical operators (`&&`, `||`) for compound conditions.

```
bash
```

Copy code

```
if [[ $num -gt 10 && $num -lt 20 ]]; then
    # numeric value is between 10 and 20
fi
```

## 4. Pattern Matching:

- Supports pattern matching (globbing).

```
bash
```

Copy code

```
if [[ $string == "pattern"* ]]; then
    # string starts with "pattern"
fi
```

-eq , -ge , -gt , -le , -lt , -ne , -ef

Exercise : check if the user is a minor or not

```
1#!/bin/bash
2#lets create an input asking for age
3echo "Welcome to roller coster "
4sleep 1
5read -p "What's your age ? " age
6#if [[ $age =~ ^[0-9]+$ ]]; then
7#    # echo "Write a valid age "
8
9if [[ $age -lt 18 ]]; then
10    echo "You're a minor , fuck outta here "
11elif [[ $age -eq 18 ]]; then
12    echo "Alright , you barely made it , enter "
13elif [[ $age -gt 100 ]]; then
14    echo "You're supposed to be dead , go back home "
15elif [[ $age -ge 18 ]]; then
16    echo "Enter bruh "
17
18fi
```

## Multiple conditions and nested if statements :

```

1 #!/bin/bash
2 #lets create an input asking for age
3 echo "Welcome to roller coster "
4 sleep 1
5 read -p "What's your age ? " age
6 #if [[ $age =~ ^[0-9]+$ ]] ; then
7     # echo "Write a valid age "
8
9 if [[ $age -lt 18 ]] ; then
10    echo "You're a minor , fuck outta here "
11 elif [[ $age -eq 18 ]] ; then
12    echo "Alright , you barely mede it , enter "
13 elif [[ $age -gt 100 ]] ; then
14    echo "You're supposed to be dead , go back home "
15 elif [[ $age -ge 18 ]] ; then
16    echo "Enter bruh "
17
18 fi

```

His version

```

1 #!/bin/bash
2 read -p "Enter you age:" age
3
4 if [[ $age -lt 18 ]] && [[ $age -ge 0 ]]
5 then
6     echo "You are minor!"
7 elif [[ $age -eq 18 ]]
8 then
9     echo "Congratulations, you've just become major!"
10 elif [[ $age -gt 18 ]] && [[ $age -le 100 ]]
11 then
12     echo "You are major"
13 else
14     echo "Invalid age!"
15 fi

```

Display content script

Only display if there is at least one argument with the script

Indentation

```

1 #!/bin/bash
2 if [[ $# -eq 1 ]]
3 then
4 if [[ -f "$1" ]]
5 then
6     echo "The argument is a file, displaying its contents..."
7     sleep 1
8     cat $1
9 elif [[ -d "$1" ]]
10 then
11     echo "The argument is directory, running ls -l ..."
12     sleep 1
13     ls -l $1
14 else
15     echo "The argument ($1) is neither a file nor a directory."
16 fi
17
1. Select with Shift + V
2. Define the selection with arrow keys
3. Press > (Shift + .)

```

3,5 A

Exit

Shift + V

Arrow keys

Shift + .

```
1#!/bin/bash
2if [[ $# -eq 1 ]]
3then
4    if [[ -f "$1" ]]
5    then
6        echo "The argument is a file, displaying its contents..."
7        sleep 1
8        cat $1
9    elif [[ -d "$1" ]]
10   then
11       echo "The argument is directory, running ls -l ..."
12       sleep 1
13       ls -l $1
14    else
15        echo "The argument ($1) is neither a file nor a directory."
16    fi
17 else
18    echo "The script should be run with an argument."
19 fi
20
```

## Command substitution :

Run a shell command and store its output into a variable for later use

```
now=`date`
```

```
student@ubuntu20-04:~$ now=`date`
student@ubuntu20-04:~$ echo $now
Wed 16 Sep 2020 01:48:22 PM EEST
```

Second method :

```
user=$(who)
```

```
student@ubuntu20-04:~$ output="$(ps -ef | grep bash)"
student@ubuntu20-04:~$ echo "$output"
student      2015     2006  0 13:48 pts/0    00:00:00 /bin/bash
student      2192     2015  0 13:50 pts/0    00:00:00 /bin/bash
student      2194     2192  0 13:50 pts/0    00:00:00 grep --color=auto bash
student@ubuntu20-04:~$
```

Task : create an archive of the /etc directory with the current date and hour and minute

```
c_date=date +%F_%H%M
```

```
tar -czvf etc-$c_date.tar.gz /etc
```

So if we archive without date , the current will be overwritten , so we add the date in the naming

## Comparing strings in If statement :

Comparing two strings , quoting string variables is better to avoid word splitting and globing issues  
!= not equal

```
1#!/bin/bash
2 read -p "First string : " str1
3 sleep 1
4 read -p "Second string : " str2
5 if [[ $str1 == $str2 ]] ; then
6     echo " They're equal "
7 else
8     echo " They're not equal "
9
10 fi
```

```
[root@vps1 scripts]# source compare_strings.sh
First string : taib
Second string : taib
They're equal
[root@vps1 scripts]# source compare_strings.sh
First string : taib
Second string : Taib
They're not equal
```

### Checking for substrings :

If [[ "\$str1" == \*"Linux"\* ]]

```
1#!/bin/bash
2
3 str1="Nowadays, Linux powers the servers of the Internet."
4
5 if [[ "$str1" == *"Linux"* ]]
6 then
7     echo "The substring Linux is there."
8 else
9     echo "The substring Linux IS NOT there."
```

### Check if the string is empty or not :

-n : non zero length

```
1#!/bin/bash
2
3 my_str=""
4
5 if [[ -z "$my_str" ]]
6 then
7     echo "String is zero length."
8 else
9     echo "String IS NOT zero lenght."
0 fi
1
2 if [[ -n "$my_str" ]]
3 then
4     echo "String is not zero length."
5 else
6     echo "String is zero length."
7 fi
```

## Lab : testing network connections

Continuously checking the network connection for a server or default gateway , if its down inform the sysadmin

We can use the 100% packet lost

```
1 #!/bin/bash
2 output=$(ping -c 3 $1)
3 #echo "$output"
4
5 if [[ "$output" == *"100% packet loss"* ]]
6 then
7     echo "The network connection to $1 is not working."
8 else
9     echo "The network connection to $1 is working."
```

## Conclusion

```
# if [ some_condition_is_true ]
# then
# //execute this code
# elif [ some_other_condition_is_true ]
# then
# //execute_this_code
# else
# //execute_this_code
# fi
## Examples:

i=1
if [[ $i -lt 10 ]]
then
echo "i is less than 10."
fi
#####
i=100
if [[ $i -lt 10 ]]
then
echo "i is less than 10."
else
echo "i is greater than or equal to 10."
fi
#####
i=10
if [[ $i -lt 10 ]]
then
echo "i is less than 10."
elif [[ $i -eq 10 ]]
then
echo "i is 10"
else
echo "i is greater than or equal to 10."
fi
```

```

#####
### TESTING CONDITIONS => man test ###

### For numbers (integers) ###
# -eq equal to
# -ne not equal to
# -lt less than
# -le less than or equal to
# -gt greater than
# -ge greater than or equal to

### For files:
# -s file exists and is not empty
# -f file exists and is not a directory
# -d directory exists
# -x file is executable by the user
# -w file is writable by the user
# -r file is readable by the user

### For Strings
# = the equality operator for strings if using single square brackets [ ]
# == the equality operator for strings if using double square brackets [[ ]]
# != the inequality operator for strings
# -n $str str is nonzero length
# -z $str str is zero length

# && => the logical and operator
# || => the logical or operator

```

## For loops :

### The For Loop

---

```

for item in LIST
do
  COMMANDS
done

```

- The list can be a series of strings separated by spaces, a range of numbers, output of a command, an array, and so on.

Exercice : iterate over a string

```
1#!/bin/bash
2for os in Ubuntu CentOS Slackware Kali
3do
4    echo "os is $os"
5done

v
v
v
v
v
v

student@ubuntu20-04:~/scripts$ ./for1.sh
os is Ubuntu
os is CentOS
os is Slackware
os is Kali
```

1,1 All

If a string contains a space you should put it in double quote so it can be considered one work or substitue it

```
1 #!/bin/bash
2 for os in Ubuntu CentOS Slackware Kali MX Linux
3 do
4     echo "os is $os"
5 done
```

## Iterating over a list of integer

```
7 for num in {3..7}
8 do
9     echo "num is $num"
10 done
```

**With increment of 5**

```
7 for num in {10..100..5}
8 do
9     echo "num is $num"
10 done
```

**Displaying the content of each file in the current directory :**

/\* : any file in the current directory

```
1 #!/bin/bash
2
3 for item in ./*
4 do
5     if [[ -f "$item" ]]
6     then
7         echo "Displaying the contents of $item"
8         sleep 1
9         cat $item
10        echo "#####"
11
12    fi
13 done
```

## Rename every file in the current directory

\*.txt : it will iterate of txt files

```
1 #!/bin/bash
2
3 for file in *.txt
4 do
5     mv "$file" "renamed_by_script_$file"
6 done

.
.
.
.

:tudent@ubuntu20-04:~/scripts$ ls
age.sh          connection_testing.sh   empty_string.sh  myscript1
arguments.sh    dir1                   file_stat.sh    renamed_by_script_123.txt
arguments.sh.tar.gz  display_and_compress.sh first_script.sh renamed_by_script_abc.txt
block_ip.sh     display.sh            for1.sh        rename_files.sh
compare_strings.sh drop_ip.sh          for2.sh        substring.sh
:tudent@ubuntu20-04:~/scripts$
```

## While loop :

```
while CONDITION
do
    COMMANDS
done
```

- The set of commands are executed as long as the given condition evaluates to **true**.

Exercice : while i is smaller than 0 , print im the greatest

```
[root@vps1 scripts]# cat while1.sh
#!/bin/bash
i=0
while [[ $i -lt 10 ]]
do
    echo "I WILL WIN "
done
```

```
1 #!/bin/bash
2 i=0
3
4 while [[ $i -lt 10 ]]
5 do
6     echo "i: $i"
7     ((i++)) # let i=i+1
8
9 done
```

**Arithmatic operations :**

```
a=2  
b=1  
c=$((a+b))  
Echo $c
```

**Infinite loop :**

---

```
1 #!/bin/bash  
2 while [ 1 -eq 1 ]  
3 do  
4     echo "Infinite loop, press CTRL + C to exit."  
5     sleep 1  
6 done
```

**Check for the running process :**

**Process\_monitoring.sh**

---

```
1 #!/bin/bash  
2  
3 while :  
4 do  
5     output=$(pgrep -l $1)  
6     if [[ -n "$output" ]]  
7     then  
8         echo "The process \"$1\" is running."  
9     else  
10        echo "The process \"$1\" IS NOT running."  
11    fi  
12    sleep 3  
13 done
```

T

## Case statement

The case construct to test strings and numbers and it's a simple form of "if and else statement "

### Use `case` Statement When:

1. **Multiple Conditions:**
  - If you have multiple conditions to check and each condition corresponds to a specific pattern or value, a `case` statement can be more concise.
2. **Pattern Matching:**
  - `case` statements are particularly useful when you need to match patterns or compare against a list of possible values.
3. **Readability:**
  - When dealing with multiple options, a `case` statement can improve the readability of your script, making it easier to understand and maintain.

## The Case Statement

```
case EXPRESSION in
    PATTERN_1)
        STATEMENTS
    ;;
    PATTERN_2)
        STATEMENTS
    ;;
    PATTERN_N)
        STATEMENTS
    ;;
    *)
        STATEMENTS
    ;;
esac
```

Master Linux Administration  
By Andrei Dumitrescu

Home

./favorite\_pet.sh

```
1#!/bin/bash
2
3 echo -n "Enter your favorite pet:"
4 read PET
5
6 case "$PET" in
7     dog)
8         echo "Your favorite pet is the dog."
9         ;;
10    cat|Cat)
11        echo "You like cats."
12        ;;
13    fish|"African Turtle")
14        echo "Fish or turtles are great!"
15        ;;
16    *)
17        echo "Your favorite pet is unknown!"
18 esac
```

\$# : gives the count of arguments

\$@ : gives the arguments themselves as an array

SIGHUP : SIGHUP is a signal in Unix-like operating systems, including Linux, that stands for "Hang Up." This signal is primarily used to instruct processes to reload configuration files or restart themselves

Kill -l : send the specified signal to the process

```
[root@vps1 scripts]# kill -l
 1) SIGHUP      2) SIGINT      3) SIGQUIT      4) SIGILL      5)
SIGTRAP
 6) SIGABRT     7) SIGBUS      8) SIGFPE       9) SIGKILL     10)
SIGUSR1
11) SIGSEGV     12) SIGUSR2     13) SIGPIPE     14) SIGALRM     15)
SIGTERM
16) SIGSTKFLT   17) SIGCHLD     18) SIGCONT     19) SIGSTOP     20)
SIGTSTP
21) SIGTTIN     22) SIGTTOU     23) SIGURG      24) SIGXCPU     25)
SIGXFSZ
26) SIGVTALRM   27) SIGPROF     28) SIGWINCH    29) SIGIO       30)
```

\$1 : the signal  
\$2 : the process

```
[root@vps1 scripts]# sleep 1001 &
[1] 17932
[root@vps1 scripts]# sleep 1001
^C
[root@vps1 scripts]# pgrep sleep
17931
17932
```

```
if [ $# -ne 2 ]
then
    echo "Run the script with 2 arguments: SIGNAL and PID."
    exit
fi

case "$1" in
    1) echo "Sending the SIGHUP signal to $2"
        kill -SIGHUP $2
        ;;
    2) echo "Sending the SIGINT signal to $2"
        kill -SIGINT $2
        ;;
    15) echo "Sending the SIGTERM signal to $2"
        kill -15 $2
        ;;
    *) echo "Signal number $1 will not be delivered."
        ;;
esac
```

```
student@ubuntu20-04:~/scripts$ ./signal.sh 1 5064
Sending the SIGHUP signal to 5064
[1]+  Hangup                  sleep 1001
```

## Functions :

Create code that can be reused for designing

```
1 #!/bin/bash
2
3 function print_something () {
4     echo "I'm a simple function!"
5 }
6
7 display_something () {
8     echo "Hello functions!"
```

### Functions arguments :

In bash we don't put the arguments in () , we put them as input arguments like

Display\_something picture

Picture is argument and \$1

**For other programming languages :** they have the concept of return value , bash function don't allowed us  
return status : 0 --> everything went okay

```
12 create_files (){
13     echo "Creating $1"
14     touch $1
15     chmod 400 $1
16     echo "Creating $2"
17     touch $2
18     chmod 600 $2
19 }
```

### Function lines\_in\_files

\$1 found in \$2

```
1
2 function lines_in_file () {
3     grep -c "$1" "$2"
4 }
5
6 print_something
7 display_something
8
9 create_files aa.txt bb.txt
0 echo $?
1
2 n=$(lines_in_file "usb" "/var/log/dmesg")    I
3 echo $n
```

N = how many times it occurs

## Variable scope in functions :

Scope : global or local variables , in bash the variables are global , can be used anywhere in the script

```
1 #!/bin/bash
2 var1="AA"
3 var2="BB"
4
5 function func1() {
6     echo "Inside func1: var1=$var1, var2=$var2"
7 }
8
9 func1
^
^
```

If we change a local variable , it will affect the global as well

```

1 #!/bin/bash
2 var1="AA"
3 var2="BB"
4
5 function func1() {
6     var1="XX"
7     echo "Inside func1: var1=$var1, var2=$var2"
8 }
9
10 func1
11 echo "After calling func1: var1=$var1, var2=$var2"

```

variable\_scope.sh" 11L, 168C written 11,19 All

```

tudent@ubuntu20-04:~/scripts$ ./variable_scope.sh
nside func1: var1=AA, var2=BB
tudent@ubuntu20-04:~/scripts$ ./variable_scope.sh
nside func1: var1=XX, var2=BB
fter calling func1: var1=XX, var2=BB
tudent@ubuntu20-04:~/scripts$ █

```

### Settings a local variable :

Settings a local one is a good practice

```

1 #!/bin/bash
2 var1="AA"
3 var2="BB"
4
5 function func1() {
6     var1="XX"
7     local var2="YY"
8     echo "Inside func1: var1=$var1, var2=$var2"
9 }
10
11 func1
12 echo "After calling func1: var1=$var1, var2=$var2"

```

variable\_scope.sh" 12L, 185C written 7,15-22 All

```

nside func1: var1=XX, var2=BB
fter calling func1: var1=XX, var2=BB
tudent@ubuntu20-04:~/scripts$ ./variable_scope.sh
nside func1: var1=XX, var2=YY I
fter calling func1: var1=XX, var2=BB

```

## Menus in bash , the SELECT statement :

### Bash Menus (Select)

```
select ITEM in LIST
do
    COMMANDS
done
```

- **ITEM** is a user defined variable and the **LIST** is a series of strings, numbers or output of commands.

```
1 #!/bin/bash
2 select COUNTRY in Germany France USA "United Kingdom"
3 do
4     echo "COUNTRY is $COUNTRY"
5     echo "REPLY is $REPLY"
6 done
```

Menus.sh

```
1 #!/bin/bash
2 select COUNTRY in Germany France USA "United Kingdom"
3 do
4     echo "COUNTRY is $COUNTRY"
5     echo "REPLY is $REPLY"
6 done
~
~
~
```

1,1      All

```
student@ubuntu20-04:~/scripts$ ./menus.sh
1) Germany
2) France
3) USA
4) United Kingdom
#? 1
COUNTRY is Germany
REPLY is 1
#? █
```

The default is #? , you can change it by adding the PS3 variable or prompt

```

2 PS3="Choose your country:"
3 select COUNTRY in Germany France USA "United Kingdom" Quit
4 do
5     case $REPLY in
6         1)
7             echo "You speak German."
8             ;;
9         2)
10            echo "You speak French."
11            ;;
12        3)
13            echo "You speak American English."
14            ;;
15        4)
16            echo "You speak British English."
17            ;;
- INSERT --

```

3,59

50%

## Intro to bash arrays :

Arrays can store multiple data , arrays are simply lists that can be computed



### Create an array :

Using ( ) or array[0]=100

```

student@ubuntu:~/scripts$ ages=(20 22 40 38)

student@ubuntu:~/scripts$ numbers[0]=100
student@ubuntu:~/scripts$ numbers[1]=200
student@ubuntu:~/scripts$ 
student@ubuntu:~/scripts$ 
student@ubuntu:~/scripts$ echo ${numbers[@]}
100 200

```

**Print the whole array :**

```
student@ubuntu:~/scripts$ ages=(20 22 40 38)
student@ubuntu:~/scripts$ echo ${ages[@]}
20
student@ubuntu:~/scripts$ echo ${ages[*]}
20 22 40 38
student@ubuntu:~/scripts$ echo ${!ages[@]}
20 22 40 38
student@ubuntu:~/scripts$
```

There is a difference between \* and @

**Print the indexes of an array and how many items :**

```
student@ubuntu:~/scripts$ echo ${#ages[*]}
4
student@ubuntu:~/scripts$ echo ${!ages[*]}
4
```

**Select an element from an array :**

```
student@ubuntu:~/scripts$ echo ${ages[0]}
20
student@ubuntu:~/scripts$ echo ${ages[2]}
40
```

No out of index error

The indexes don't need to be consecutive

```
student@ubuntu:~/scripts$ names[7]="Maya"
student@ubuntu:~/scripts$ echo ${names[@]}
Dan Alina Diana Maya
student@ubuntu:~/scripts$ echo ${!names[@]}
0 1 2 7
```

**Remove an item from an array :**

```
student@ubuntu:~/scripts$ unset names[1]
student@ubuntu:~/scripts$ echo ${names[@]}
Dan Diana Maya
student@ubuntu:~/scripts$ echo ${!names[@]}
0 2 7
```

**Add an element to the array**

years+=(2023 2024)

**Slice an array :**

#Starts at a specific index

Echo \${years[\*]: 2}

```
student@ubuntu:~/scripts$ echo ${years[@]:2}
2020 2021 2022 2023 2024 2025 2026
```

```
#access based on indexes

student@ubuntu:~/scripts$ echo ${years[@]}
2018 2019 2020 2021 2022 2023 2024 2025 2026
student@ubuntu:~/scripts$ echo ${years[@]:2}
2020 2021 2022 2023 2024 2025 2026
student@ubuntu:~/scripts$ echo ${years[@]:2:4}
2020 2021 2022 2023
```

5<sup>th</sup> included

## Associative arrays : Dictionary

Key value pair , only works with declare , the only difference between this and index arrays is that we use strings instead of indexes

```
student@ubuntu:~/scripts$ declare -A userdata
student@ubuntu:~/scripts$ userdata[username]="youradmin"
student@ubuntu:~/scripts$ userdata[password]="fdsa4."
student@ubuntu:~/scripts$ userdata[uid]=1010
student@ubuntu:~/scripts$ echo ${userdata[username]}
youradmin
student@ubuntu:~/scripts$ echo ${userdata[@]}
fdsa4. 1010 youradmin
student@ubuntu:~/scripts$ echo ${!userdata[@]}
password uid username
student@ubuntu:~/scripts$ userdata[login]=$(date --utc +%s)
student@ubuntu:~/scripts$ echo ${userdata[@]}
fdsa4. 1661506614 1010 youradmin
student@ubuntu:~/scripts$ userdata[login]=$(date +%T)
student@ubuntu:~/scripts$ echo ${userdata[@]}
fdsa4. 12:37:47 1010 youradmin
student@ubuntu:~/scripts$ userdata+=([shell]="Bash" [admin]="False")
student@ubuntu:~/scripts$ echo ${userdata[@]}
False fdsa4. 12:37:47 Bash 1010 youradmin
student@ubuntu:~/scripts$
```

Declare a read only array :

```
student@ubuntu:~/scripts$ declare -r -A SUPERSTARS=(
> [Germany]="Boney M"
> [USA]="Bon Jovi"
> [England]="The Beatles"
> )
student@ubuntu:~/scripts$ echo ${SUPERSTARS[@]}
Bon Jovi The Beatles Boney M
student@ubuntu:~/scripts$ SUPERSTARS[USA]="Metallica"
bash: SUPERSTARS: readonly variable
```

Unset userdata[password]

## Readarray : iterate over arrays

Readarray converts the input into standard array

```
student@ubuntu:~/scripts$ readarray months
Jan
Feb
Mar
Apr
May
Jun
^C
student@ubuntu:~/scripts$ echo ${months[@]}
Jan Feb Mar Apr May Jun
student@ubuntu:~/scripts$ cat months.txt
January
February
March
April
May
June
July
August
September
October
November
December
student@ubuntu:~/scripts$ readarray months< <(cat months.txt)

student@ubuntu:~/scripts$ readarray months< <(cat months.txt)
student@ubuntu:~/scripts$ echo ${months[@]}
January February March April May June July August September October November December
student@ubuntu:~/scripts$ echo ${!months[@]}
0 1 2 3 4 5 6 7 8 9 10 11
```

Input redirection : the content of what follows will be redirected to readarray

There is always a trailing \n , we can remove the trailing using -t

```
student@ubuntu:~/scripts$ echo ${months[@]@Q}
$'January\n' $'February\n' $'March\n' $'April\n' $'May\n' $'June\n' $'July\n' $'August\n' $'September
\n' $'October\n' $'November\n' $'December\n'
student@ubuntu:~/scripts$ readarray -t months< <(cat months.txt)
student@ubuntu:~/scripts$ echo ${months[@]@Q}
'January' 'February' 'March' 'April' 'May' 'June' 'July' 'August' 'September' 'October' 'November' 'December'
```

Example 2 :

Users< : input redirection  
<(cut -d: -f1 /etc/passwd) : command substitution

```
student@ubuntu:~/scripts$ readarray users< <(cut -d: -f1 /etc/passwd)
student@ubuntu:~/scripts$ echo ${users[@]}
root daemon bin sys sync games man lp mail news uucp proxy www-data backup list irc gnats nobody syst
emd-network systemd-resolve systemd-timesync messagebus syslog _apt tss uuidd tcpdump avahi-autoipd u
sbmux rtkit dnsmasq cups-pk-helper speech-dispatcher avahi kernoops sanded nm-openvpn hplip whoopsie c
olord geoclue pulse gnome-initial-setup gdm student systemd-coredump vboxadd sshd sysadmin10
```

Iterating over arrays :

Create a script that add the execution access to all the files in the given directory

The image shows two terminal windows side-by-side. The left window displays a bash script with comments and code for adding execute permissions to files in a directory. The right window shows the results of running the script, including file listing and modification details.

```

root@vps1:~/scripts
File Edit View Search Terminal Help
1 #!/bin/bash
2
3 #the purpose of this script is to add execute access to the scripts in the current directory
4
5 #converting the files into array
6 readarray -t access< <(ls /root/scripts)
7 for f in ${access[@]}
8 do
9     chmod +x $f
10
11
12
13 done
14
15
16

root@vps1:~/scripts
Edit View Search Terminal Help
: exec.sh first_script.sh malicious_ip.sh pass_stat.sh pyscript
: ipt.tar.gz while1.sh
:@vps1 scripts]# source exec.sh
:@vps1 scripts]# ls -l
. 7104
-xr-x. 1 root root 489 Mar 9 22:07 age.sh
-xr-x. 1 root root 185 Mar 9 23:39 compare_strings.sh
-xr-x. 1 root root 56 Mar 10 01:19 connection_testing.sh
-xr-x. 1 root root 52 Mar 10 01:01 conn_test.sh
-xr-x. 1 root root 149 Mar 9 20:11 display_compress.sh
-xr-x. 1 root root 213 Mar 9 22:50 display_regular.sh
-xr-x. 2 root root 22 Mar 9 17:34 dr1
-xr-x. 1 root root 131 Mar 9 20:29 drop_ip.sh
-xr-x. 1 root root 7216469 Mar 9 23:21 etc-2024-03-09_2320.tar
-xr-x. 1 root root 296 Mar 10 05:10 exec.sh
wxrwx. 1 root root 56 Mar 9 17:33 first_script.sh
-xr-x. 1 root root 151 Mar 9 19:16 malicious_ip.sh
-xr-x. 1 root root 196 Mar 9 18:40 pass_stat.sh
-xr-x. 1 root root 45 Mar 9 17:51 pyscript
-xr-x. 1 root root 161 Mar 9 20:12 pyscript.tar.gz
-xr-x. 1 root root 70 Mar 10 01:53 while1.sh
:@vps1 scripts]# mv exec.sh add_x_access.sh
:@vps1 scripts]# ./age.sh

```

## Project : Automate user creation :

First we create a text file containing the users and their group

user.txt

---

```

1 john:management
2 diana:programming
3 paul:accounti

```

Vim user\_and\_groups.sh

---

```

1 #!/bin/bash
2 readarray accounts< <(cat ./users.txt)
3
4 for account in "${accounts[@]}"
5 do
6     echo $account
7     user=$(echo $account | cut -d: -f1)
8     group=$(echo $account | cut -d: -f2)
9     groupadd $group
10    echo "Group $group added successfully!"
11    useradd -G $group $user
12    echo "User $user added successfully!"
13 done

```

Check if the user or group already exists :

```

do
    #echo $account
    user=$(echo $account | cut -d: -f1)
    group=$(echo $account | cut -d: -f2)
    if [[ -z "$(grep -w $group /etc/group)" ]]
    then
        groupadd $group
        echo "Group $group added successfully!"
    else
        echo "Group $group already exists!"
    fi

    if [[ -z "$(grep -w $user /etc/passwd)" ]]
    then
        useradd -G $group $user
        echo "User $user added successfully!"
    else
        echo "User $user already exists!"

done

```

```

[root@vps1 scripts]# source project.sh
Group front has been added succefuly
User tim has been added succefuly
Group police has been added succefuly
User lucy has been added succefuly
Group accounting has been added succefuly
User adam has been added succefuly
Group reception has been added succefuly
User Alina has been added succefuly
[root@vps1 scripts]# source project.sh
Group already exists
User already exists
[root@vps1 scripts]# tail -4 /etc/passwd
tim:x:1002:1004::/home/tim:/bin/bash
lucy:x:1003:1006::/home/lucy:/bin/bash
adam:x:1004:1008::/home/adam:/bin/bash
Alina:x:1005:1010::/home/Alina:/bin/bash

```

This was successful , however what if we remove a user from our system and we want to automatically update the system by removing him or her

## **Project 6 : Linux high availability clustering on veritas cluster**

**server**



















