

## Docker Security App:

## 1. Image Manifest Scanner - [Detection]

Wrote python script `scanImageManifest.py` to automatically read and highlight sensitive information in image manifests from docker registry.

## 2. Checking Capabilities - [Detection and Prevention]

Wrote the python script to checked if any abusable capability is present inside running containers and deleted the same.

```
alice@ubuntu:~/DockerSecurityApps$ sudo docker run --restart=always -it -d --cap-add=DAC_OVERRIDE localhost:5000/app-backend
eedeed9f22a50edfd0ecc60af9e4aa9a07e0265a865705449f9745bce5ddbea
alice@ubuntu:~/DockerSecurityApps$ sudo docker run --restart=always -it -d --cap-add=SYS_ADMIN --cap-add CAP_DAC_OVERRIDE localhost:5000/app-frontend
55047b58c3987992ec40cd5d8ed54939a172b6020b4c2bb3c308a989/b0dde0a
alice@ubuntu:~/DockerSecurityApps$ 
alice@ubuntu:~/DockerSecurityApps$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
55047b58c398        localhost:5000/app-frontend   "/bin/sh -c 'service..."   7 seconds ago      Up 7 seconds          22/tcp
eedeed9f22a5        localhost:5000/app-backend    "/bin/bash"          19 seconds ago     Up 18 seconds         0.0.0.0:5000->5000/tcp   bold_stonebreaker
7f46953d516b        registry:latest           "/entrypoint.sh /etc..." 4 weeks ago       Up 2 hours           0.0.0.0:5000->5000/tcp   modest_goldstine
alice@ubuntu:~/DockerSecurityApps$ 
alice@ubuntu:~/DockerSecurityApps$
```

```

Activities Terminal ▾ April 12 23:46
alice@ubuntu:~/DockerSecurityApp$ sudo docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
55047b586398 localhost:5000/app-frontend "/bin/sh -c 'service..." 7 seconds ago Up 7 seconds 22/tcp bold_stonebraker
eedded9f2a5 localhost:5000/app-backend "/bin/bash" 19 seconds ago Up 18 seconds 0.0.0.0:5000->5000/tcp modest_goldstine
7f46953d516b registry:latest "/entrypoint.sh /etc..." 4 weeks ago Up 2 hours 0.0.0.0:5000->5000/tcp registry

alice@ubuntu:~/DockerSecurityApp$ sudo python3 detectCapabilties.py

-----
DockerSec : Capability Finder
-----

* [Finding capabilities in container: 55047b5863 (localhost:5000/app-frontend)]
Current: = cap_chown, cap_dac_override, cap_fowner, cap_fsetid, cap_kill, cap_setgid, cap_setuid, cap_setpcap, cap_net_bind_service, cap_net_raw, cap_sys_chroot, cap_sys_admin, cap_mknod, cap_audit_write, cap_setfcap+ip
Bounding set =cap_chown, cap_dac_override, cap_fowner, cap_fsetid, cap_kill, cap_setgid, cap_setuid, cap_setpcap, cap_net_bind_service, cap_net_raw, cap_sys_chroot, cap_sys_admin, cap_mknod, cap_audit_write, cap_setfcap
Securebits: 00/0x0/1'b0
secure-noroot: no (locked)
secure-no-suid-fixup: no (locked)
secure-keep-caps: no (locked)
uid=0(root)
gid=0(root)
groups=

- [Container 55047b5863 (localhost:5000/app-frontend) has abusable capability 'cap_sys_admin']
It is advised to drop the capability unless required. Would you like to remove this capability? [y/n]: y
[Container 55047b5863 (localhost:5000/app-frontend) has abusable capability 'cap_dac_override']
It is advised to drop the capability unless required. Would you like to remove this capability? [y/n]: n
here=cap_dac_override

Skipping...
+ [Restarting the container with removed/retained capabilities]...
sudo docker stop 55047b5863
55047b5863
sudo docker run -itd --restart=always --cap-drop=cap_sys_admin --cap-add=cap_dac_override localhost:5000/app-frontend

```

```

Activities Terminal ▾ April 12 23:46
alice@ubuntu:~/DockerSecurityApp$ sudo docker run -itd --restart=always --cap-drop=cap_sys_admin --cap-add=cap_dac_override localhost:5000/app-frontend
bb2ee37421f8682f842af725ce5892df59b72942cb1f58ae40b953614524765

* [Finding capabilities in container: eeeded9f22 (localhost:5000/app-backend)]
Current: = cap_chown, cap_dac_override, cap_fowner, cap_fsetid, cap_kill, cap_setgid, cap_setuid, cap_setpcap, cap_net_bind_service, cap_net_raw, cap_sys_chroot, cap_mknod, cap_audit_write, cap_setfcap+ip
Bounding set =cap_chown, cap_dac_override, cap_fowner, cap_fsetid, cap_kill, cap_setgid, cap_setuid, cap_setpcap, cap_net_bind_service, cap_net_raw, cap_sys_chroot, cap_mknod, cap_audit_write, cap_setfcap
Securebits: 00/0x0/1'b0
secure-noroot: no (locked)
secure-no-suid-fixup: no (locked)
secure-keep-caps: no (locked)
uid=0(root)
gid=0(root)
groups=

- [Container eeeded9f22 (localhost:5000/app-backend) has abusable capability 'cap_dac_override']
It is advised to drop the capability unless required. Would you like to remove this capability? [y/n]: y
+ [Restarting the container with removed/retained capabilities]...
sudo docker stop eeeded9f22
eedded9f22
sudo docker run -itd --restart=always --cap-drop=cap_dac_override localhost:5000/app-backend
09b2e1a490c7f78d7125f2fe5d7f7adaa0ec4eb52fd4663c426fa6203818486e

* [Finding capabilities in container: 7f46953d51 (registry:latest)]
Not applicable to this container
[Container 7f46953d51 (registry:latest) does not possess any abusable capability]

alice@ubuntu:~/DockerSecurityApp$ sudo docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
09b2e1a490c7 localhost:5000/app-backend "/bin/bash" 8 seconds ago Up 7 seconds 22/tcp fervent_haslett
bb2ee37421f8 localhost:5000/app-frontend "/bin/sh -c 'service..." 18 seconds ago Up 17 seconds 0.0.0.0:5000->5000/tcp nice_varahamihira
7f46953d516b registry:latest "/entrypoint.sh /etc..." 4 weeks ago Up 2 hours 0.0.0.0:5000->5000/tcp registry

```

### 3. Docker Socket Detector - [Detection]

Python script to check if 'docker.sock' file is present inside any running container.

```
alice@ubuntu:~/DockerSecurityApp$ sudo docker run -it -d -p 22:22 -v /var/run/docker.sock:/var/run/docker.sock localhost:5000/app-frontend
028205e548380f855e8315d3e8efcfe1ba52e7e2f2613b3dcbb37c33bed
alice@ubuntu:~/DockerSecurityApp$ sudo docker run -it -d -v /var/run/docker.sock:/temp/docker.sock localhost:5000/app-backend
5f5d710b51dd661d4af352ea38a0d72977c5380982c178765333d74b8555d
alice@ubuntu:~/DockerSecurityApp$ alice@ubuntu:~/DockerSecurityApp$ sudo docker run -it -d localhost:5000/app-backend
1884557c6c3d4604b7ace0edf78541f790e5ee7eed8892f155c5edaafade4
alice@ubuntu:~/DockerSecurityApp$ alice@ubuntu:~/DockerSecurityApp$ sudo docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
1884557c6c3d localhost:5000/app-backend "/bin/bash" 5 seconds ago Up 4 seconds brave_cray
5f5d710b51dd localhost:5000/app-backend "/bin/bash" 15 seconds ago Up 14 seconds blissful_bhahba
028205e54838 localhost:5000/app-frontend "/bin/sh -c 'service..." 56 seconds ago Up 55 seconds 0.0.0.0:22->22/tcp, ::1:22->22/tcp thirsty_borg
7f46953d516b registry:latest "/entrypoint.sh /etc..." 4 weeks ago Up 2 hours 0.0.0.0:5000->5000/tcp, ::1:5000->5000/tcp registry
alice@ubuntu:~/DockerSecurityApp$ alice@ubuntu:~/DockerSecurityApp$ sudo python3 detectDockerSocket.py
```

```
alice@ubuntu:~/DockerSecurityApp$ alice@ubuntu:~/DockerSecurityApp$ sudo python3 detectDockerSocket.py
-----
[DockerSec : Docker Socket Detector]
-----
*[Finding docker socket in container: 1884557c6c]...
*[Finding docker socket in container: 5f5d710b51]...
*[Finding docker socket in container: 028205e548]...
*[Finding docker socket in container: 7f46953d51]...
[Result] :: Docker socket is found mounted inside the following running containers:
-----
Container ID : Image Name
-----
1235f5d710b51 : localhost:5000/app-backend
123028205e548 : localhost:5000/app-frontend
```

#### 4. Signature Implementation - [Prevention]

Implemented image signature verification in docker environment and wrote python script to detect if image signature does not match.

The screenshot shows a terminal window on a Kali Linux desktop environment. The terminal session starts with a command to inspect a Docker image:

```
(mkaur@Kali)-[~/notary]$ docker trust inspect --pretty localhost:5000/ubuntu:new
```

It then lists the signatures for the image:

```
Signatures for localhost:5000/ubuntu:new
```

SIGNED TAG	DIGEST	SIGNERS
new	3b96df4e7c9b10cd1176525893a6030f832868dd9f31a84fb30fc7d7018f8f24	sample_signer

It shows the list of signers and their keys:

```
List of signers and their keys for localhost:5000/ubuntu:new
```

SIGNER	KEYS
sample_signer	f5e4db567f21

Administrative keys for localhost:5000/ubuntu:new are also listed:

```
Administrative keys for localhost:5000/ubuntu:new
```

Repository Key:	Root Key:
34b7cb4af8dc4be9108cb6dc516a322582ce71798a1ea0fa295a910077d63	bb58a4307802290ea50ee60e13ee7e62d639a571dcde2f9d8a4cac5f05fb8012

The user then runs a Docker pull command with the `DOCKER_CONTENT_TRUST=1` environment variable set:

```
$ DOCKER_CONTENT_TRUST=1 docker pull localhost:5000/ubuntu:new
```

The output shows the image is up to date and being pulled from the local registry:

```
Pull (1 of 1): localhost:5000/ubuntu:new@sha256:3b96df4e7c9b10cd1176525893a6030f832868dd9f31a84fb30fc7d7018f8f24
localhost:5000/ubuntu@sha256:3b96df4e7c9b10cd1176525893a6030f832868dd9f31a84fb30fc7d7018f8f24: Pulling from ubuntu
Digest: sha256:3b96df4e7c9b10cd1176525893a6030f832868dd9f31a84fb30fc7d7018f8f24
Status: Image is up to date for localhost:5000/ubuntu@sha256:3b96df4e7c9b10cd1176525893a6030f832868dd9f31a84fb30fc7d7018f8f24
Tagging localhost:5000/ubuntu@sha256:3b96df4e7c9b10cd1176525893a6030f832868dd9f31a84fb30fc7d7018f8f24 as localhost:5000/ubuntu:new
localhost:5000/ubuntu:new
```

#### Steps:

- sudo git clone <https://github.com/theupdateframework/notary.git>
- cd notary
- sudo docker-compose up -d
- sudo docker ps
- sudo docker pull ubuntu:new
- sudo docker run -d -p 5000:5000 --restart-always --name registry registry:latest
- export DOCKER\_CONTENT\_TRUST\_SERVER=https://localhost:4443
- sudo docker trust key generate sample\_signer
- sudo docker trust signer add --key sample\_signer.pub sample\_signer localhost:5000/ubuntu:new
- sudo docker trust inspect localhost:5000/ubuntu:new
- sudo docker trust sign localhost:5000/ubuntu:new
- sudo docker trust inspect --pretty localhost:5000/ubuntu:new
- DOCKER\_CONTENT\_TRUST=1 docker pull localhost:5000/ubuntu:new

## 5. Registry Authentication - Prevention

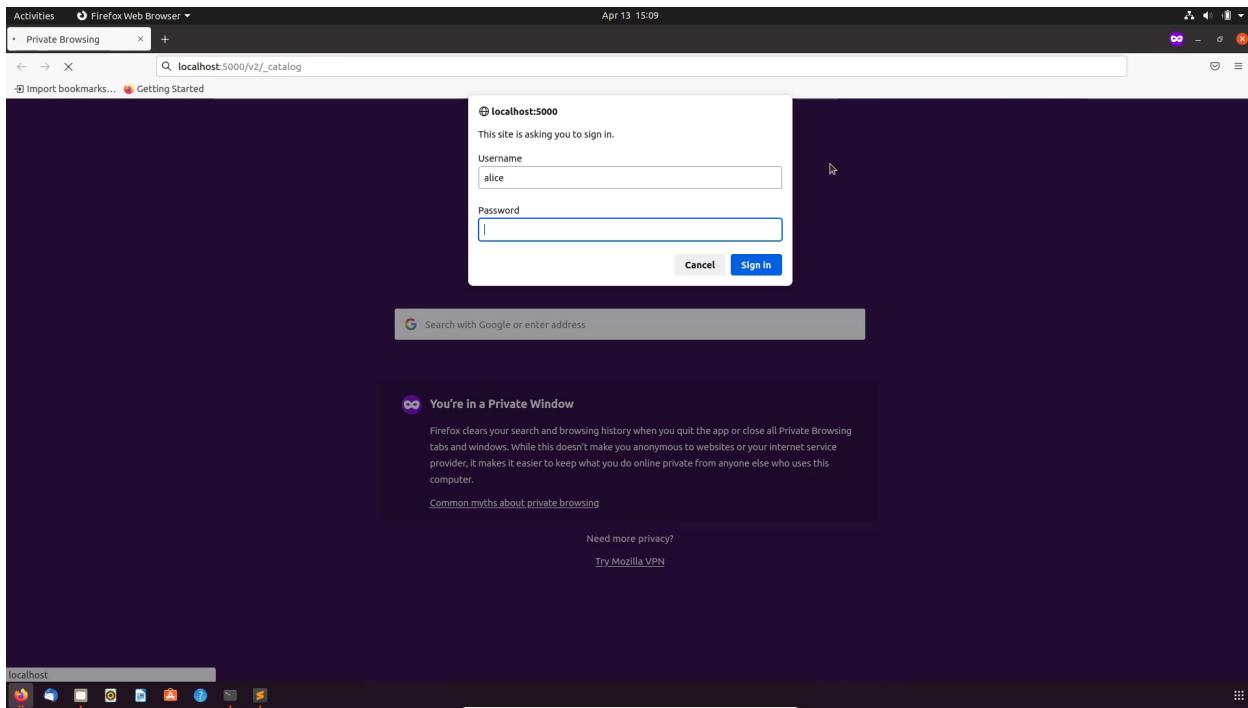
A screenshot of a Linux desktop environment showing a terminal window titled "Terminal". The terminal window has two tabs: "alice@ubuntu: ~/DockerSecurityApp" and "alice@ubuntu: ~/DockerSecurityApp". The first tab shows the command "sudo docker push localhost:5000/app-frontend:latest" being run, followed by a list of layer IDs for the image. The second tab shows the command "sudo docker pull localhost:5000/app-frontend:latest" being run, resulting in an error message: "Error response from daemon: Head \"http://localhost:5000/v2/app-frontend/manifests/latest\": no basic auth credentials". The terminal window is part of a larger desktop interface with a taskbar at the bottom.

```
alice@ubuntu:~/DockerSecurityApp$ sudo docker push localhost:5000/app-frontend:latest
The push refers to repository [localhost:5000/app-frontend]
0bc4c4d9e94b: Preparing
08417a685091: Preparing
c3d550a8a40f: Preparing
536c9219f662: Preparing
f5b9a5562dd8: Preparing
a4f8045bdcde2: Preparing
c2c989efcfcf: Preparing
bf5a7532fb1b: Preparing
1251204ef8fc: Preparing
47ef83fafe74: Preparing
df54c846128d: Preparing
be90a3f634dc: Preparing
no basic auth credentials
alice@ubuntu:~/DockerSecurityApp$ sudo docker pull localhost:5000/app-frontend:latest
Error response from daemon: Head "http://localhost:5000/v2/app-frontend/manifests/latest": no basic auth credentials
alice@ubuntu:~/DockerSecurityApp$
```

Implemented basic authentication on docker registry.

A screenshot of a Linux desktop environment showing a terminal window titled "Terminal". The terminal window has two tabs: "alice@ubuntu: ~/DockerSecurityApp" and "alice@ubuntu: ~/DockerSecurityApp". The first tab shows the command "sudo docker login localhost:5000" being run, followed by a password prompt. The second tab shows the command "sudo docker push localhost:5000/app-frontend:latest" being run, followed by a list of layer IDs for the image. The terminal window is part of a larger desktop interface with a taskbar at the bottom.

```
alice@ubuntu:~/DockerSecurityApp$ sudo docker login localhost:5000
Username: alice
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store
Login Succeeded
alice@ubuntu:~/DockerSecurityApp$ sudo docker push localhost:5000/app-frontend:latest
The push refers to repository [localhost:5000/app-frontend]
0bc4c4d9e94b: Layer already exists
08417a685091: Layer already exists
c3d550a8a40f: Layer already exists
536c9219f662: Layer already exists
f5b9a5562dd8: Layer already exists
a4f8045bdcde2: Layer already exists
c2c989efcfcf: Layer already exists
bf5a7532fb1b: Layer already exists
1251204ef8fc: Layer already exists
47ef83fafe74: Layer already exists
df54c846128d: Layer already exists
be90a3f634dc: Layer already exists
latest: digest: sha256:de986caaa8c0471839b01c2d679e1ddec9b654c11ff8c0c80d119ccce size: 2822
alice@ubuntu:~/DockerSecurityApp$ sudo docker pull localhost:5000/app-frontend:latest
Digest: sha256:ded4d862671886caab0471839b01c2d679e1ddec9b654c11ff8c0c80d119ccce
Status: Image is up to date for localhost:5000/app-frontend:latest
localhost:5000/app-frontend:latest
alice@ubuntu:~/DockerSecurityApp$ sudo docker logout localhost:5000
Removing login credentials for localhost:5000
alice@ubuntu:~/DockerSecurityApp$ sudo docker login localhost:5000
Username: bob
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store
Login Succeeded
alice@ubuntu:~/DockerSecurityApp$
```



## Steps:

- mkdir ~/docker-registry
- cd ~/docker-registry
- mkdir data
- nano docker-compose.yml

[ version: '3'

```

services:
  registry:
    image: registry:2
    ports:
      - "5000:5000"
    environment:
      REGISTRY_AUTH: htpasswd
      REGISTRY_AUTH_HTPASSWD_REALM: Registry
      REGISTRY_AUTH_HTPASSWD_PATH: /auth/registry.password
      REGISTRY_STORAGE_FILESYSTEM_ROOTDIRECTORY: /data
    volumes:
      - ./auth:/auth
      - ./data:/data ]
  
```

- docker-compose up
- sudo nano /etc/nginx/sites-available/localhost:5000

[ location / {

```

# Do not allow connections from docker 1.5 and earlier
# docker pre-1.6.0 did not properly set the user agent on ping, catch "Go *" user agents
if ($http_user_agent ~ "^^(docker|v1\.(3|4|5(?:!\.[0-9]-dev))|Go ).*$") {
    return 404;
}

proxy_pass          http://localhost:5000;
proxy_set_header Host      $http_host; # required for docker client's sake
proxy_set_header X-Real-IP   $remote_addr; # pass on real client's IP
proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
proxy_set_header X-Forwarded-Proto $scheme;
proxy_read_timeout 900;
}

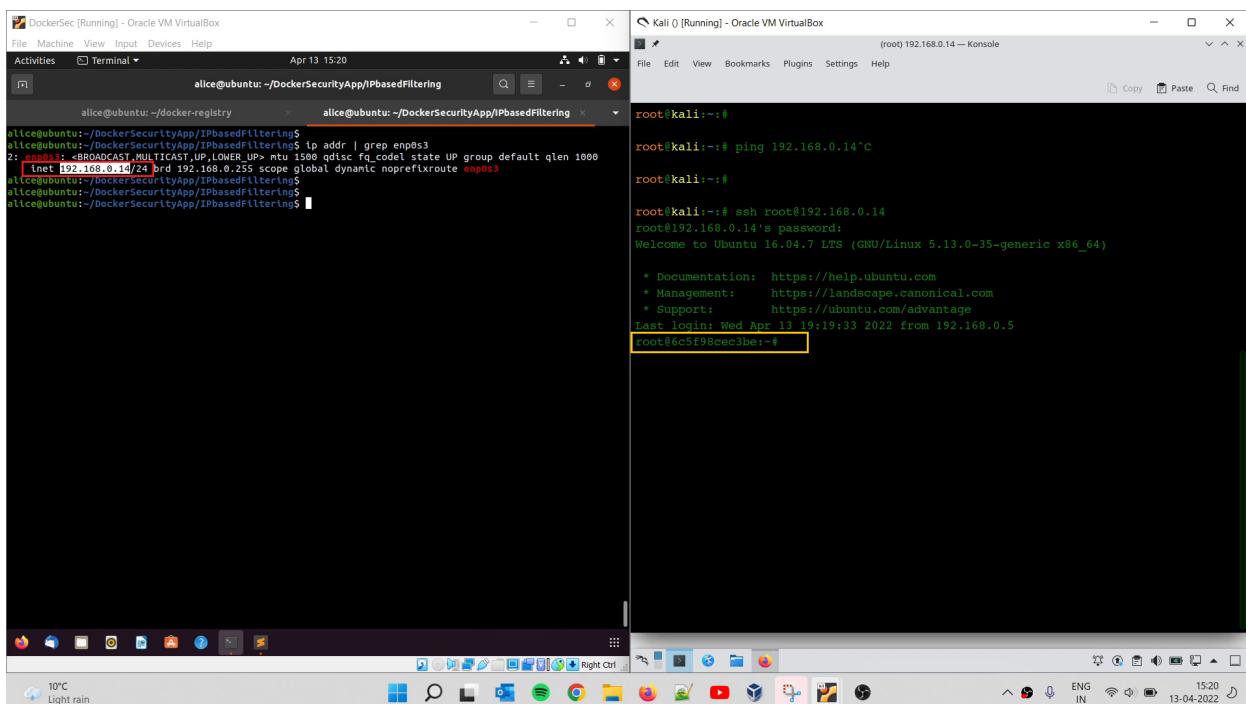
...

```

- sudo systemctl restart nginx
- docker-compose up
- sudo apt install apache2-utils -y
- mkdir ~/docker-registry/auth
- cd ~/docker-registry/auth
- htpasswd -Bc registry.password alice
- htpasswd -B registry.password bob
- nano ~/docker-registry/docker-compose.yml
- cd ~/docker-registry
- docker-compose up

## 6. IP Based Filtering - [Prevention]

Implemented IP Address whitelisting on SSH login using IPtables.



The screenshot shows two terminal windows side-by-side. The left window is on a Kali Linux host (IP 192.168.0.14) and the right window is on a Ubuntu guest machine (IP 192.168.0.13). Both windows are running within Oracle VM VirtualBox.

**Left Terminal (Kali Linux):**

```
alice@kali:~$ ssh root@192.168.0.13
root@kali:~$
```

**Right Terminal (Ubuntu Guest):**

```
alice@ubuntu:~/DockerSecurityApp/IPbasedFiltering$ sudo iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    REJECT     tcp  --  192.168.0.5          anywhere             reject-with icmp-port-unreachable
2    ACCEPT     tcp  --  192.168.0.0/24       anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED

Chain FORWARD (policy DROP)
num  target     prot opt source               destination
Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination
alice@ubuntu:~/DockerSecurityApp/IPbasedFiltering$
```

**Bottom Taskbar:**

- Icons for various applications like File Manager, Terminal, and Network.
- Weather widget showing 9°C Light rain.
- Date and time: 13-04-2022 15:34.
- Language and keyboard settings: ENG IN.

### Commands:

```
sudo iptables -L --line-numbers
sudo iptables -A INPUT -p tcp -s 192.168.0.5 -j REJECT
sudo iptables -A INPUT -p tcp -s 192.168.0.0/24 --dport 22 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
```