In-depth Network Security for Docker Containers

A project by Asem Majed, Wajeh Tuffaha, Mohammad Abdulhaq and Moath Qadry

Supervisor: Dr. Othman M. Othman

Table Of Contents

Background

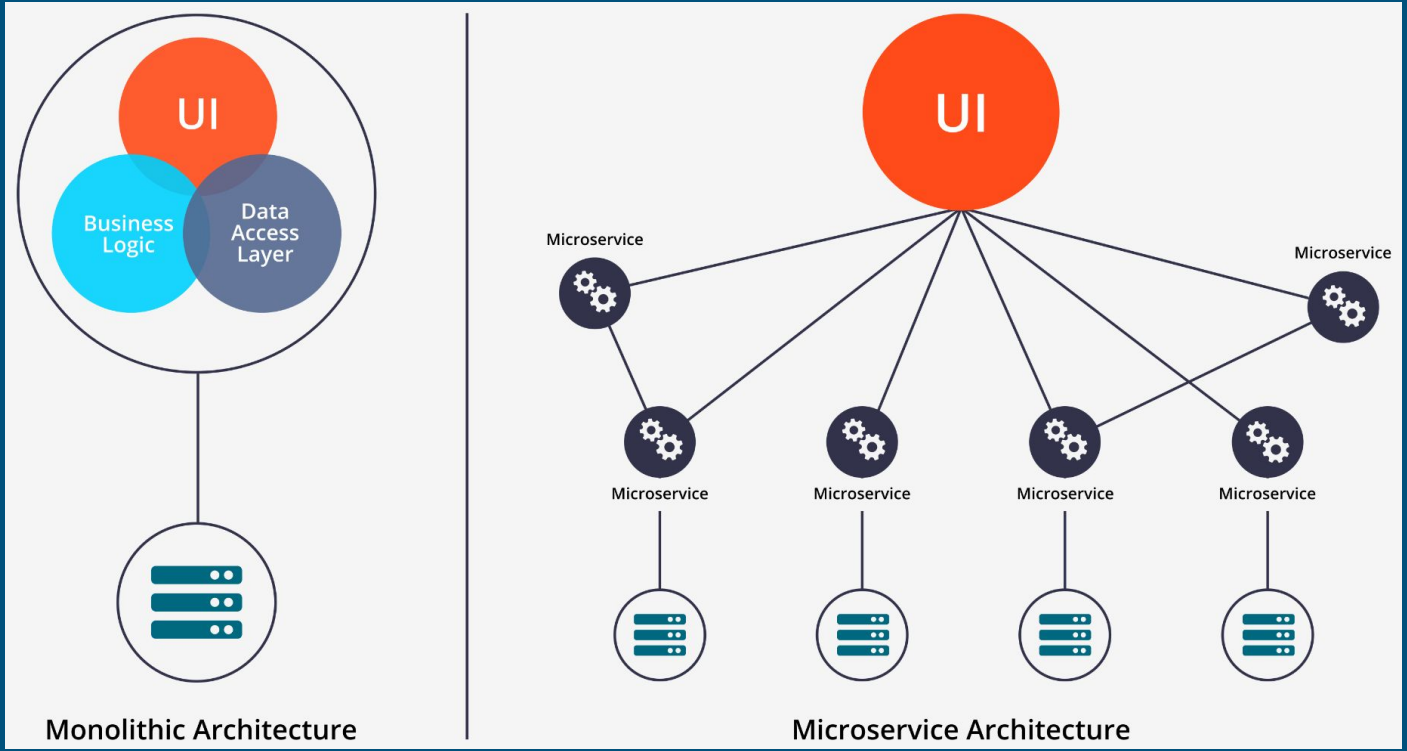
Microservices

Containers

Containers vs VMs

Microservices

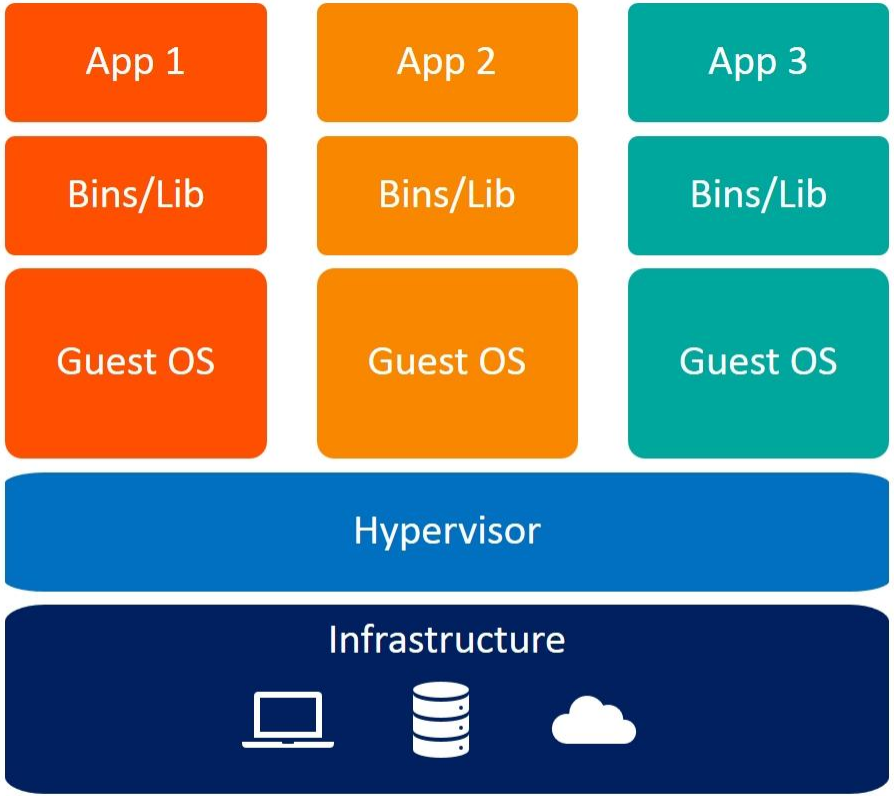
- Independence
- Resilience
- Scalability
- Lifecycle automation
- Relied on VMs



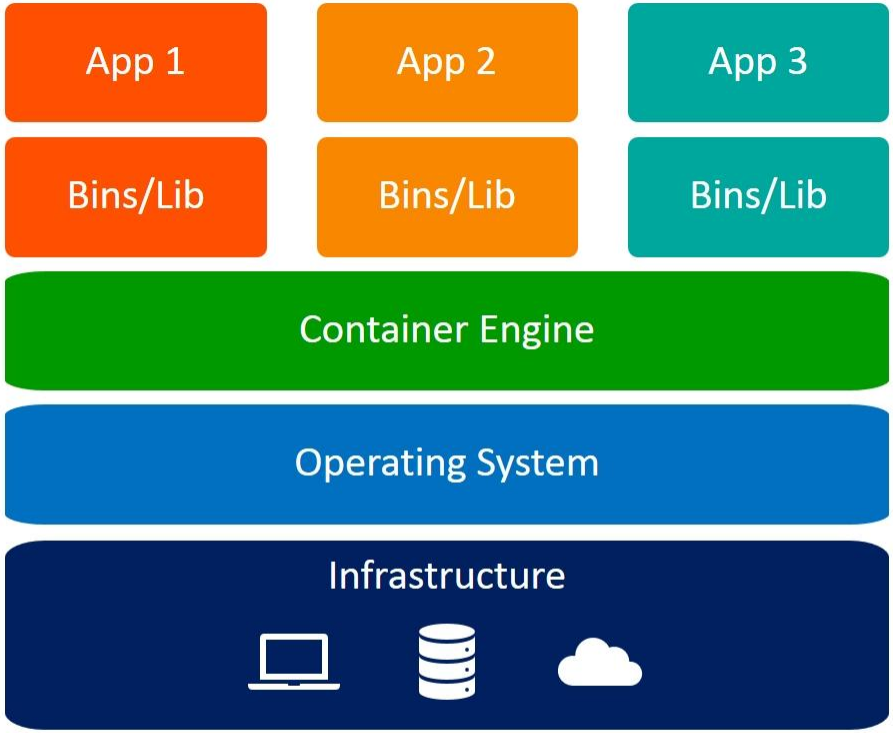
Containers

- ➔ Lightweight
- ➔ Efficient
- ➔ Move between environments
- ➔ Run independently
- ➔ Everything needed is packaged inside the container

containers vs VMs



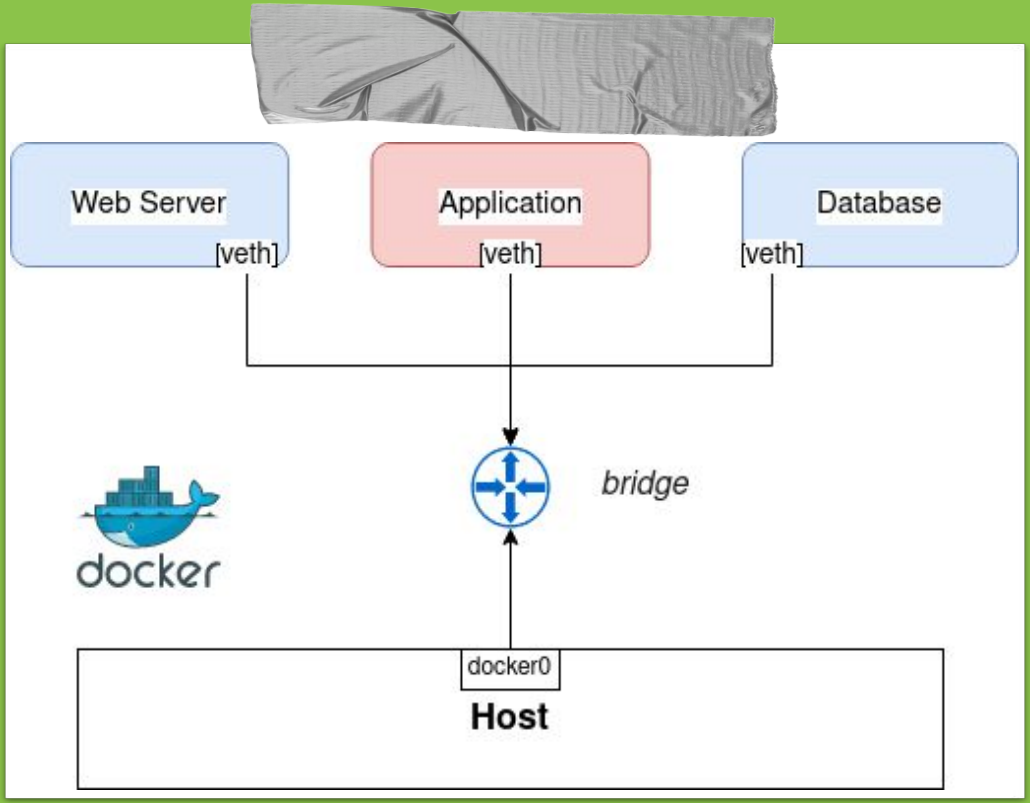
Virtual Machines



Containers

Current posture of Docker networking

Current model of Docker containers



Current posture of Docker networking

Existing vulnerabilities

- lack of isolation
  - compared to VMs
- especially at network level [1]

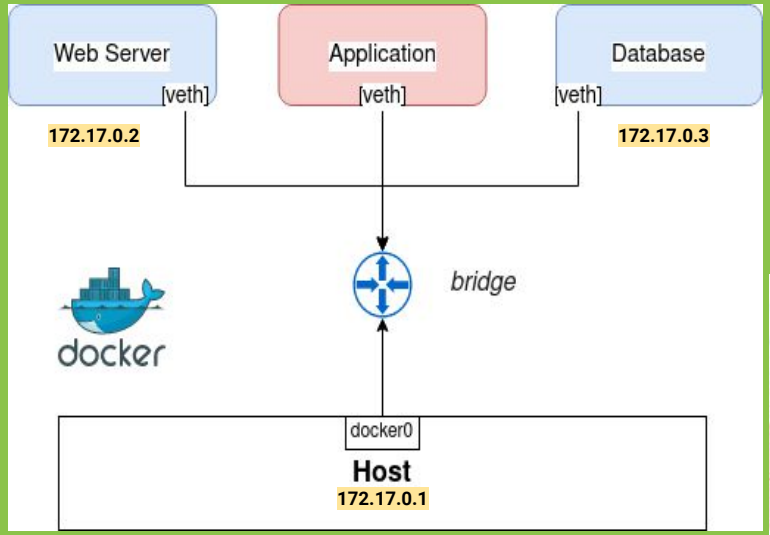
[1]

Current posture of Docker networking

Existing vulnerabilities

- Non-restricted access to containers
- Network-based attacks
  - Man In The Middle attack.
  - ARP spoofing

[1]

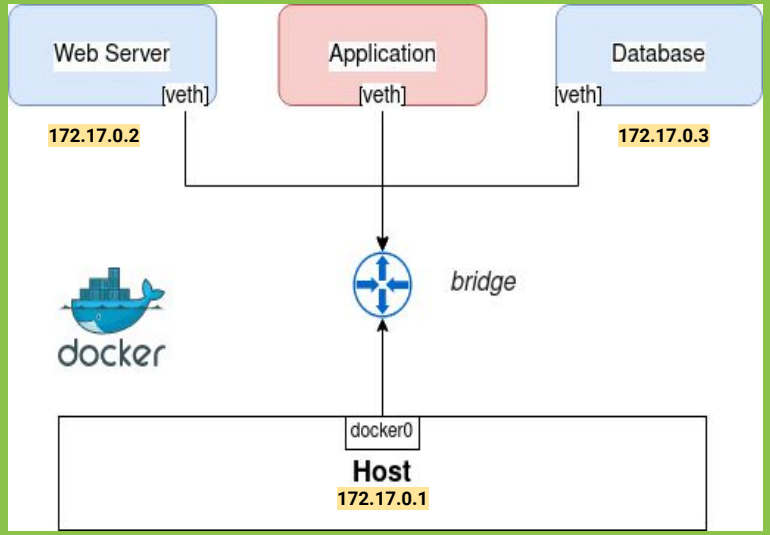


Scanning from **App** container

```
root@a61785732dff:/# nmap -p- 172.17.0.0/24
Starting Nmap 7.80 ( https://nmap.org ) at 2022-05-19 19:25 UTC
Nmap scan report for 172.17.0.1
Host is up (0.000016s latency).
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
MAC Address: 02:42:08:CB:F9:E8 (Unknown)

Nmap scan report for 172.17.0.2
Host is up (0.000018s latency).
PORT   STATE SERVICE
80/tcp open  http
MAC Address: 02:42:AC:11:00:02 (Unknown)

Nmap scan report for 172.17.0.3
Host is up (0.000017s latency).
PORT      STATE SERVICE
27017/tcp open  mongod
MAC Address: 02:42:AC:11:00:03 (Unknown)
```



Scanning from **Host**

```
~/docker$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED             STATUS              PORTS                                   NAMES
a61785732dff   ubuntu    "bash"                   About an hour ago   Up About an hour                                            App
609a8cf2504b   mongo     "docker-entrypoint.s…"   2 hours ago         Up 2 hours          27017/tcp                               db
19c80fc9de62   httpd     "httpd-foreground"       2 hours ago         Up 2 hours          0.0.0.0:80->80/tcp, :::80->80/tcp       web
```

Current posture of Docker networking

Mitigation techniques

- Kernel isolation:
  - namespace, Cgroups, & capabilities (host-based)
- Other solutions:
  - BASTION (container-based)
  - Cilium (host-based)

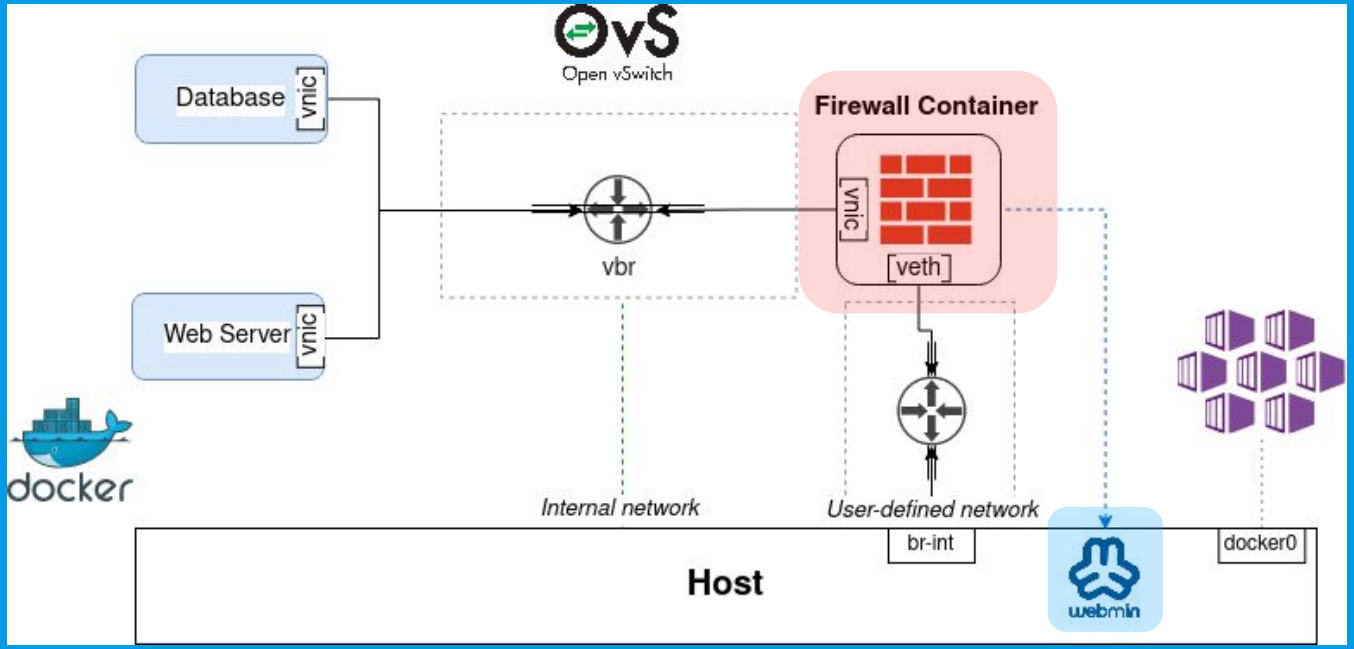
Proposed solution:

- ➔ **Main objectives**
  - Achieve network-isolation
  - Comply with Microservice model
  - Support portability & deployment features
- ➔ **Design components**
- ➔ **Practical example (scenario)**

Design Components:

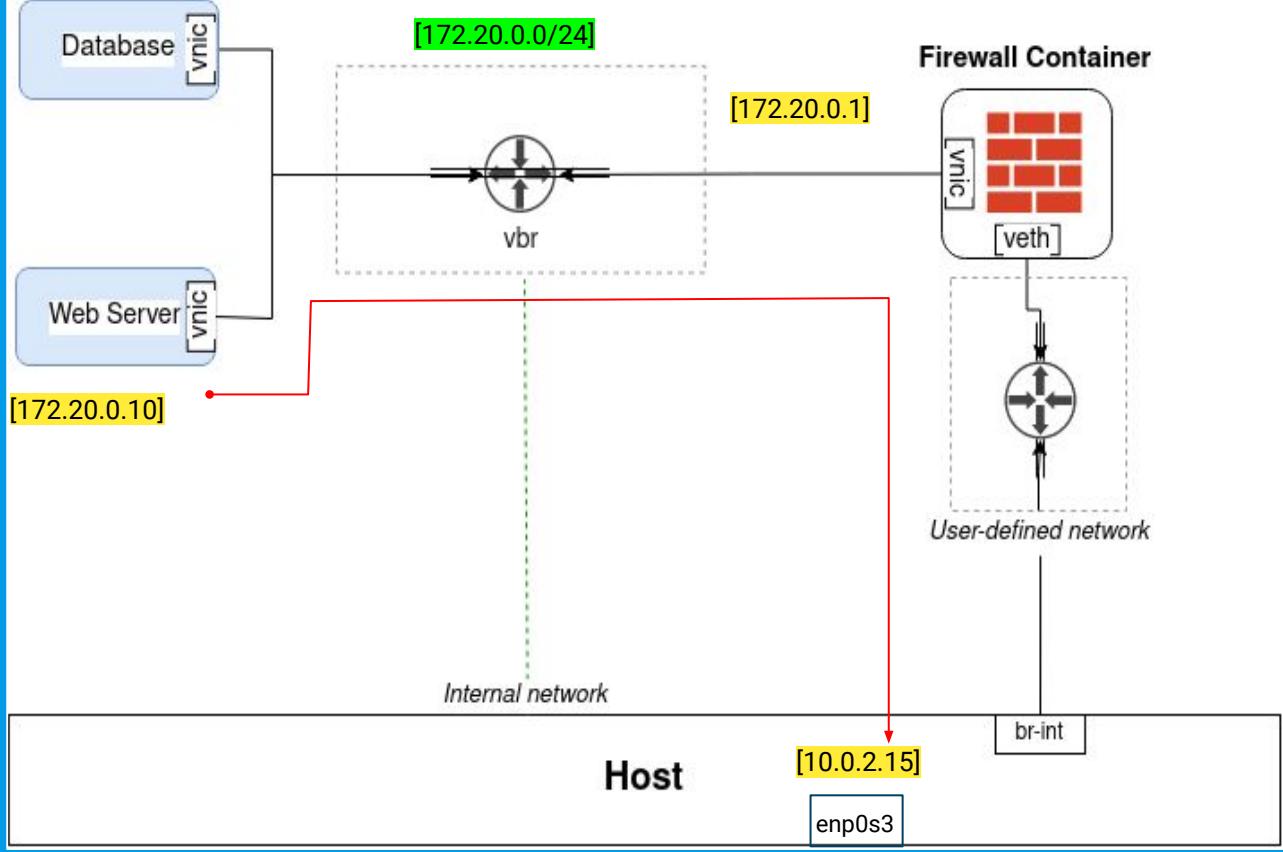
- Firewall
  - As container
  - Gateway for internal containers
  - Easy managed (Webmin)

- Internal containers
  - Connected through virtual bridge
  - Hidden



Tracepath test from 'Web Server' container to 'Host' container

Internal container uses 'Firewall container' as its default gateway




```
root@7946502961bf:/usr/local/apache2# tracepath 10.0.2.15
 1?: [LOCALHOST]                      pmtu 1500
 1:  172.20.0.1                                            1.276ms
 1:  172.20.0.1                                            0.138ms
 2:  10.0.2.15                                             0.093ms reached
     Resume: pmtu 1500 hops 2 back 2
```

Route pass through firewall

Host attached interfaces

'172.20.0.0/24' network is not visible

```
:~/docker$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP>
    ...
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP>
    ...
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP>
    ...
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
    ...
4: br-d7408f4c16d2: <BROADCAST,MULTICAST,UP,LOWER_UP>
    ...
    inet 172.18.0.1/24 brd 172.18.0.255 scope global br-d7408f4c16d2
    ...
6: vetha18dabb@if5: <BROADCAST,MULTICAST,UP,LOWER_UP>
    ...
7: ovs-system: <BROADCAST,MULTICAST>
    ...
8: vbr: <BROADCAST,MULTICAST>
    ...
```

Host routing table

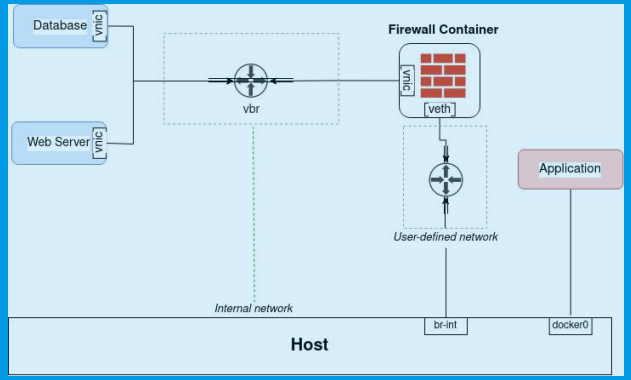
'172.20.0.0/24' network is not accessible

```
:~/docker$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         _gateway        0.0.0.0         UG    100    0        0 enp0s3
10.0.2.0        0.0.0.0         255.255.255.0   U     100    0        0 enp0s3
link-local      0.0.0.0         255.255.0.0     U     1000   0        0 enp0s3
172.17.0.0      0.0.0.0         255.255.0.0     U     0      0        0 docker0
172.18.0.0      0.0.0.0         255.255.255.0   U     0      0        0 br-d7408f4c16d2
```

Before: implementing our design

'Web' & 'DB' containers expose their ports

```
~/docker$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED              STATUS              PORTS                                  NAMES
a61785732dff   ubuntu    "bash"                   About an hour ago    Up About an hour                                           App
609a8cf2504b   mongo     "docker-entrypoint.s…"   2 hours ago          Up 2 hours          27017/tcp                              db
19c80fc9de62   httpd     "httpd-foreground"       2 hours ago          Up 2 hours          0.0.0.0:80->80/tcp, :::80->80/tcp      web
```



After: implementing our design

Only *firewall container* exposes selected port

```
:~/docker$ docker ps
CONTAINER ID   IMAGE                        COMMAND                  CREATED          STATUS   PORTS                  NAMES
d05b87e4dd57   mongo                        "docker-entrypoint.s…"   9 minutes ago    Up                              db
7946502961bf   httpd                        "httpd-foreground"       13 minutes ago   Up                              web
ed82ee6b357f   abukareem/working:working3.0 "/bin/sh -c '\"/etc/w…"  16 minutes ago   Up       0.0.0.0:80->80/tcp     FW
```

Implementation:

- **Configuration methods**
- **Practical Example**
- **Evaluation**

Implementation:

- **Configuration methods:**
- **CLI**
- **Docker-compose**
- **Bash Script**
- **Practical Example**
- **Evaluation**

```yaml
version: "3.9"
services:

  FW:
    image: abukareem/docker-firewall
    container_name: FW
    #ports:
      #- "80:80"
      #- "10000:10000"
    restart: unless-stopped
    volumes:
      - FWV1:/var/log
    networks:
      FWN:
        ipv4_address: 172.18.0.254
    stdin_open: true
    tty: true
    privileged: true
  web:
    image: httpd
    container_name: web
    cap_add:
      - NET_ADMIN
    stdin_open: true # docker run -i
    tty: true        # docker run -t
  db:
    image: mongo
    container_name: db
    cap_add:
      - NET_ADMIN
    stdin_open: true # docker run -i
    tty: true        # docker run -t
networks:
  FWN:
```

Implementation:

- **Configuration methods:**
- CLI
- Docker-compose
- Bash Script
  - Firewall container configuration
  - Services containers configuration
  - Add virtual interfaces

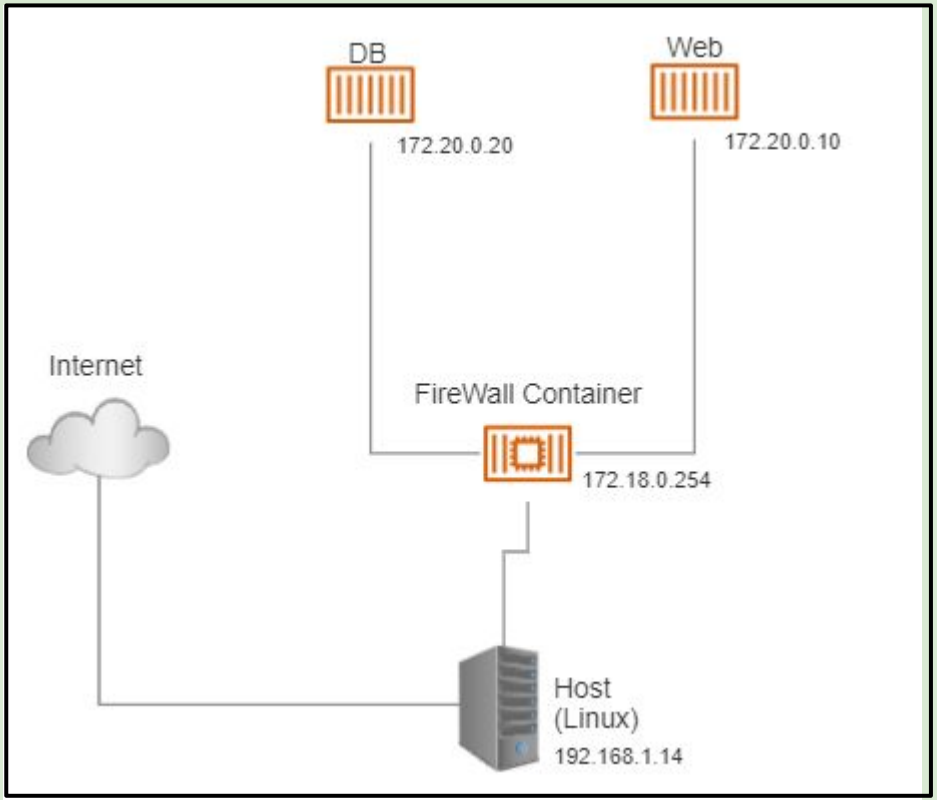
- **Practical Example**
- **Evaluation**

```
┌──(kali㉿kali)-[~/Downloads]
└─$ sudo bash all.sh

1. Configure Firewall Container
2. Add Service
3. Add Virtual Interface
4. Exit
Enter choice:
```

Implementation:

- Configuration methods
- Practical Example:
- Topology
- Procedure
- Results
- Evaluation



Implementation:

- **Configuration methods**
- **Practical Example:**
- **Topology**
- **Procedure**
- **Results**
- **Evaluation**

Provide necessary information:
firewall container creation.

```
Enter choice: 1
Enter container name:
FireWall
Enter network name:
network-1
*Network does not exist, a new one will be created*
Enter network address: (including '/subnet')
172.20.0.0/24
Enter network gateway:
172.20.0.1
NETWORK network-1 CREATED!
Enter container ip address:
172.20.0.254
Enter volume name:
Container_Disk
*Volume does not exist, a new one will be created*
VOLUME Container_Disk CREATED!
```

Implementation:

- **Configuration methods**
- **Practical Example:**
- **Topology**
- **Procedure**
- **Results**
- **Evaluation**

Provide needed information:
Network parameters.

```
Enter choice: 1
Enter container name:
FireWall
Enter network name:
network-1
*Network does not exist, a new one will be created*
Enter network address: (including '/subnet')
172.20.0.0/24
Enter network gateway:
172.20.0.1
NETWORK network-1 CREATED!
Enter container ip address:
172.20.0.254
Enter volume name:
Container_Disk
*Volume does not exist, a new one will be created*
VOLUME Container_Disk CREATED!
```

Implementation:

- **Configuration methods**
- **Practical Example:**
- **Topology**
- **Procedure**
- **Results**
- **Evaluation**

Provide needed information:
Volume information.

```
Enter choice: 1
Enter container name:
FireWall
Enter network name:
network-1
*Network does not exist, a new one will be created*
Enter network address: (including '/subnet')
172.20.0.0/24
Enter network gateway:
172.20.0.1
NETWORK network-1 CREATED!
Enter container ip address:
172.20.0.254
Enter volume name:
Container_Disk
*Volume does not exist, a new one will be created*
VOLUME Container_Disk CREATED!
```

Implementation:

Provide necessary parameter:
Virtual Bridge.

- **Configuration methods**
- **Practical Example:**
- **Topology**
- **Procedure**
- **Results**
- **Evaluation**

```
==========================
CONTAINER FireWall CREATED!
==========================
(Adding a virtual interface to the container)
Enter virtaul bridge name and virtual interface name: (in-line)
vpr nic-1
VIRTUAL INTERFACE SUCCESSFULLY ADDED!
Enter ip address for nic-1 interface (including '/subnet'):
172.19.0.0/24
```

Implementation:

- **Configuration methods**
- **Practical Example:**
- **Topology**
- **Procedure**
- **Results**
- **Evaluation**

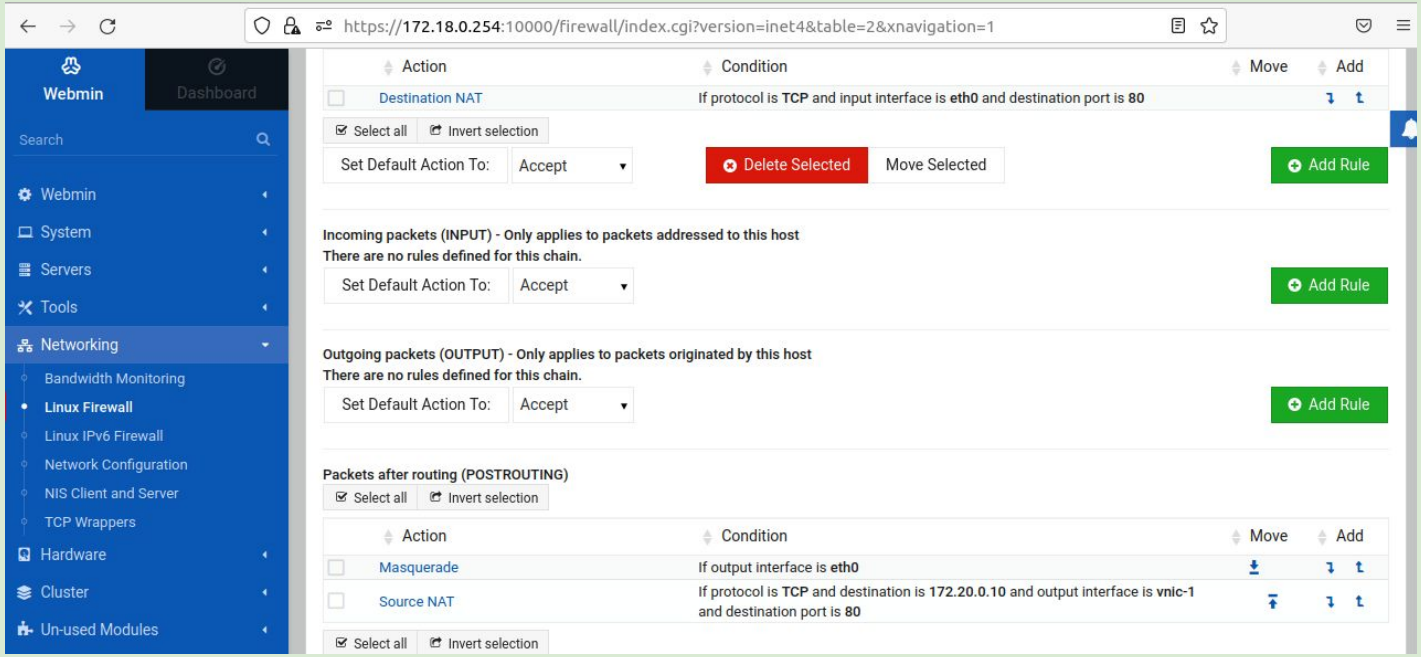
Provide necessary parameter:
Web container creation.

```
1. Configure Firewall Container
2. Add Service
3. Add Virtual Interface
4. Exit
Enter choice: 2
Enter image name:
httpd
Enter service name:
WEB
==========================
CONTAINER WEB CREATED!
==========================
(Adding a virtual interface to the container)
Enter virtaul bridge name and virtual interface name: (in-line)
vpr-1 nic-2
VIRTUAL INTERFACE SUCCESSFULLY ADDED!
(PLEASE WAIT... )
DONE!
```

Implementation:

- **Configuration methods**
- **Practical Example:**
- **Topology**
- **Procedure**
- **Results**
- **Evaluation**

NAT Configurations.



Implementation:

- **Configuration methods**
- **Practical Example:**
- **Topology**
- **Procedure**
- **Results**
- **Evaluation**

Filter table

Allow http connections

Forwarded packets (FORWARD) - Only applies to packets passed through this host

Select all | Invert selection

	Action	Condition	Move	Add
☐	Accept	If protocol is **TCP** and input interface is **eth0** and output interface is **vnic-1** and destination port is **80** and TCP flags **SYN** (of **SYN**) are set and state of connection is **NEW**		
☐	Accept	If input interface is **eth0** and output interface is **vnic-1** and state of connection is **ESTABLISHED,RELATED**		
☐	Accept	If input interface is **vnic-1** and output interface is **eth0** and state of connection is **ESTABLISHED,RELATED**		

Select all | Invert selection

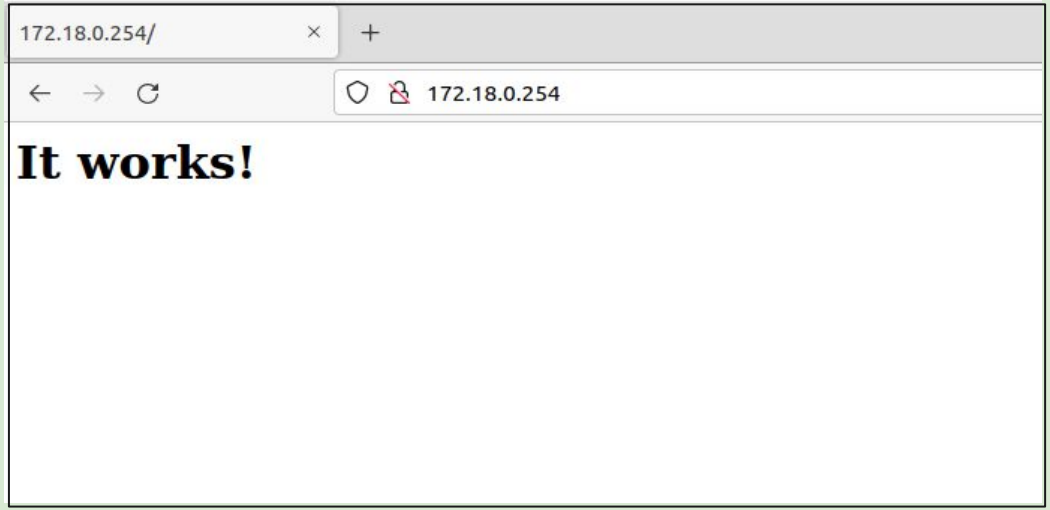
Set Default Action To: Drop

Delete Selected | Move Selected | Add Rule

Implementation:

- Configuration methods
- Practical Example:
- Topology
- Procedure
- Results
- Evaluation

Accessibility



Inaccessibility

```
:~/docker$ telnet 172.18.0.254 27017
Trying 172.18.0.254...
telnet: Unable to connect to remote host: Connection refused
```

Implementation:

- Configuration methods
- Practical Example
- Evaluation:

Port scanning

```
root@a61785732dff:/# nmap -p- 172.17.0.0/24
Starting Nmap 7.80 ( https://nmap.org ) at 2022-05-19 19:25 UTC
Nmap scan report for 172.17.0.1
Host is up (0.000016s latency).
PORT    STATE SERVICE
22/tcp open  ssh
80/tcp open  http
MAC Address: 02:42:08:CB:F9:E8 (Unknown)

Nmap scan report for 172.17.0.2
Host is up (0.000018s latency).
PORT    STATE SERVICE
80/tcp open  http
MAC Address: 02:42:AC:11:00:02 (Unknown)

Nmap scan report for 172.17.0.3
Host is up (0.000017s latency).
PORT       STATE SERVICE
27017/tcp open  mongod
MAC Address: 02:42:AC:11:00:03 (Unknown)
```

```
root@f1123a497252:/# nmap -p- 172.20.0.0/24
Starting Nmap 7.80 ( https://nmap.org ) at 2022-05-20 14:32 UTC
```

Conclusion

Thank you for listening!

Any questions?

Implementation:

- Configuration methods
- Practical Example:
  - Topology
  - Procedure
  - Results
  - Evaluation

DHCP server Configuration.