

NASA HW4

B09902011 陳可邦

NA

Short answers

1. "Block" just drops the packet and doesn't give the sender any response, while "Reject" tells the sender that the packet is dropped.

Source: Lab 4 Slides

2. "Interface net" matches anything in the subnet, while "interface address" only matches the 1 address assigned to said net.

https://www.reddit.com/r/PFSense/comments/6vyqw3/what_is_the_difference_between_the_interface_net/

3. pfSense is a stateful firewall, which means it stores information about connections flowing through, so the reply traffic will be allowed automatically.
















<https://docs.netgate.com/pfsense/en/latest/firewall/fundamentals.html#firewall-stateful>

pfSense

1. "Interfaces > VLAN": Add VLAN 5 & 99, assign subnets, "Interfaces > Interface assignment" > Add the VLANs, "Service > DHCP": Enable and add DNS for two vlans.
2. "Firewall > Alias", Add the values.
3. "System > Advanced", enable SSH.
4. Add rules:

Floating WAN LAN VLAN5 VLAN99

Rules (Drag to Change Order)

<input type="checkbox"/>	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
<input type="checkbox"/>	✓ 0 / 0 B	IPv4 *	VLAN99 net	*	VLAN99 net	*	*	none			  
<input type="checkbox"/>	✓ 0 / 336 B	IPv4 *	*	*	CSIE_ WORKSTATIONS	*	*	none			  
<input type="checkbox"/>	✓ 0 / 336 B	IPv4 *	*	*	VLAN5 net	*	*	none			  
<input type="checkbox"/>	✓ 0 / 0 B	IPv4 TCP	*	*	This Firewall	ADMIN_ PORTS	*	none			  
<input type="checkbox"/>	✓ 0 / 5 KiB	IPv4 *	*	*	GOOGLE_DNS	*	*	none			  

- Add rules: WAN Block source !vlan99, dest vlan99. LAN Block source !vlan99, dest vlan99. VLAN5 Block source !vlan99, dest vlan99.
- Add Schedule at 2021/5/11, then add rules:

Floating WAN LAN VLAN5 VLAN99

Rules (Drag to Change Order)

<input type="checkbox"/>	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
<input type="checkbox"/>	<div><div>✖</div><div>0 / 756 B</div></div>	IPv4 *	<div>!</div> <div>VLAN99 net</div>	*	VLAN99 net	*	*	none			<div><div></div><div></div><div></div><div></div></div>
<input type="checkbox"/>	<div><div>✖</div><div>0 / 0 B</div></div>	IPv4 *	<div></div> <div>VLAN5 net</div>	*	*	*	*	none	<div><div></div><div>may11</div></div>		<div><div></div><div></div><div></div><div></div></div>
<input type="checkbox"/>	<div><div>✔</div><div>0 / 655 KiB</div></div>	IPv4 *	<div></div> <div>VLAN5 net</div>	*	*	*	*	none			<div><div></div><div></div><div></div><div></div></div>

- See rules in 6.

SA

About Container

1

When to use container:

1. When you want to have a fixed environment so you can eliminate environment factors when collaborating.
2. When you want to quickly test your app in different environments, but don't want to set up a lot of VMs and waste time.
3. When you want to test a new tool, but don't want to pollute your system, you can use images built by the developer.
4. When you want a server run multiple application at the same time, but don't want them to interfere with one another, to prevent problems.

When to use VM:

1. When you really need security, because docker containers are not the most secure.
2. When your application is complicated, and using docker just isn't enough.
3. When you need multiple OSes, since docker use your host computer's OS, you'll need VM if you want to test your app on other OSes.

<https://www.linode.com/docs/guides/when-and-why-to-use-docker/>

2

- OCI (Open Container Initiative) is a organization whose purpose is to create an industry standard for containers runtime, currently it specificities runtime and image format. CRI (Container Runtime Interface) is an API for Kubernetes to communicate with various container runtimes.
- Docker's runtime & image supports the OCI Specs. CRI can be used to manipulate containers made by Docker.

<https://www.tutorialworks.com/difference-docker-containerd-runc-crio-oci/>

Docker Basics

1. `docker rm $(docker ps -aq)`
2. `docker rmi -f $(docker images -a -q)`
3. `docker system prune -a`
4. `docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' 5b0f1ed0dcb8`
5. `docker stats -a --format "table {{.Container}}\t{{.CPUPerc}}\t{{.MemUsage}}"`
6. `docker pull nginx:1.19.2`

```
docker run -d -p 5678:80 --name nginx-1 nginx:1.19.2
```

localhost:5678

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

7. `docker exec -it nginx-1 /bin/sh`

exec: executes a command in the container, in this case "/bin/sh".

-it: keeps stdin open for input, and makes the output behave as expected.

```
> docker exec -it nginx-1 /bin/sh
# ls
bin  docker-entrypoint.d  home  media  proc  sbin  tmp
boot docker-entrypoint.sh  lib   mnt    root  srv   usr
dev  etc                  lib64 opt    run   sys   var
#
```

8. `docker exec nginx-1 cat /etc/nginx/nginx.conf`

exec: executes a command in the container, in this case "cat /etc/nginx/nginx.conf"

```
~ ..... 00:56:59
> docker exec nginx-1 cat /etc/nginx/nginx.conf

user  nginx;
worker_processes  1;

error_log  /var/log/nginx/error.log warn;
pid        /var/run/nginx.pid;

events {
    worker_connections  1024;
}

http {
    include      /etc/nginx/mime.types;
    default_type  application/octet-stream;

    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';

    access_log  /var/log/nginx/access.log  main;

    sendfile      on;
    #tcp_nopush   on;

    keepalive_timeout  65;

    #gzip  on;

    include /etc/nginx/conf.d/*.conf;
}
```

上課簡報 & stackoverflow

Docker Network

1

1. Bridge mode: The container gets a IP assigned in a subnet by the host, similar to NAT. This is the default mode for Docker.
2. Host mode: Use the same network settings as the host, can change the network settings of the host from the container. Used when you don't want any networking difference between the host and the container.
3. None: No network. Used when your container don't need internet connection.
4. Container mode: Use the same network settings as some other container of your choice. Used when you want multiple containers behave the same in networking.

<https://docs.docker.com/network/>

Creating the network: `docker network create nasa-net`

Run nginx-2: `docker run -d -p 6969:80 --name nginx-2 nginx:1.19.2`

Connect nginx-1 to the net: `docker network connect nasa-net nginx-1`

Connect nginx-2 to the net: `docker network connect nasa-net nginx-2`

```
docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}'
nginx1 nginx2 :
```

```
> docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' nginx-1 nginx-2
172.17.0.2172.18.0.3
172.17.0.3172.18.0.2
```

Ping from nginx-1 to 2:

```
root@acdd850d7f13:/# ping 172.18.0.2
PING 172.18.0.2 (172.18.0.2): 56 data bytes
64 bytes from 172.18.0.2: icmp_seq=0 ttl=64 time=0.160 ms
64 bytes from 172.18.0.2: icmp_seq=1 ttl=64 time=0.251 ms
64 bytes from 172.18.0.2: icmp_seq=2 ttl=64 time=0.250 ms
^C--- 172.18.0.2 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.160/0.220/0.251/0.043 ms
root@acdd850d7f13:/#
```

Ping from nginx-2 to 1:

```
root@4c52cbbb58ef:/# ping 172.18.0.3
PING 172.18.0.3 (172.18.0.3): 56 data bytes
64 bytes from 172.18.0.3: icmp_seq=0 ttl=64 time=0.424 ms
64 bytes from 172.18.0.3: icmp_seq=1 ttl=64 time=0.250 ms
64 bytes from 172.18.0.3: icmp_seq=2 ttl=64 time=0.920 ms
64 bytes from 172.18.0.3: icmp_seq=3 ttl=64 time=0.519 ms
64 bytes from 172.18.0.3: icmp_seq=4 ttl=64 time=0.297 ms
^C--- 172.18.0.3 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.250/0.482/0.920/0.239 ms
root@4c52cbbb58ef:/#
```

3

To find the host IP from within the container, `ping docker.host.internal` and look for the IP. The question didn't specify if we want to find the IP from within the container or the outside so I don't know what we're trying to achieve here.

<https://stackoverflow.com/questions/22944631/how-to-get-the-ip-address-of-the-docker-host-from-inside-a-docker-container>

Build Application

- When appending some commands after a `docker run` command, the commands using `CMD` will be overrides, `ENTRYPOINT` will not.
 - `ENTRYPOINT` will only be overridden by using `--override` option.
 - Their name is different.

```
FROM ubuntu
ENTRYPOINT ["echo"]
CMD ["Hello"]
```

If we run `docker run test`, it will echo `Hello`. If we run `docker run test meow`, it will echo `meow`.

<https://ithelp.ithome.com.tw/articles/10250988>

- Docker-compose is a tool for running a multi-container Docker applications. Docker-compose uses a YAML file to configure a application, while Docker use Dockerfile to run a single container.

<https://docs.docker.com/compose/>

- `-p` : maps a port of the host to a port of the container.
 - `-w` : set working directory
 - `-v ${PWD}:/app` : Put current directory to `/app` inside the container.
 - `--network nasa-net` : use nasa-net as network
 - `-e` : set environment variables

```
version: "3.8"
services:
  app:
    image: node:12-alpine
```

```

command: sh -c "echo helloworld"

ports:
  - 3000:3000

working_dir: /app

volumes:
  - ./:/app

environment:
  MYSQL_HOST: mysql
  MYSQL_USER: root
  MYSQL_PASSWORD: secret

mysql:
  image: mysql:5.7
  volumes:
    - ./mysql-data:/var/lib/mysql
  environment:
    MYSQL_ROOT_PASSWORD: secret

networks:
  default:
    external:
      name: nasa-net

```

```

Creating hw4_app_1 ... done
Creating hw4_mysql_1 ... done
Attaching to hw4_mysql_1, hw4_app_1
app_1 | helloworld
mysql_1 | 2021-04-25 20:15:02+00:00 [Note] [Entrypoint]:
mysql_1 | 2021-04-25 20:15:02+00:00 [Note] [Entrypoint]:
mysql_1 | 2021-04-25 20:15:02+00:00 [Note] [Entrypoint]:
mysql_1 | 2021-04-25 20:15:03+00:00 [Note] [Entrypoint]:
mysql_1 | 2021-04-25T20:15:03.210971Z 0 [Warning] TIMEST
defaults_for_timestamp server option (see documentation f
mysql_1 | 2021-04-25T20:15:03.234432Z 0 [Warning] Settin
case insensitive
hw4_app_1 exited with code 0
mysql_1 | 2021-04-25T20:15:10.746578Z 0 [Warning] InnoDB
mysql_1 | 2021-04-25T20:15:13.578944Z 0 [Warning] InnoDB
mysql_1 | 2021-04-25T20:15:13.678119Z 0 [Warning] No exi
that this server has been started. Generating a new UUID:
mysql_1 | 2021-04-25T20:15:13.692135Z 0 [Warning] Gtid t
pened.
mysql_1 | 2021-04-25T20:15:15.181766Z 0 [Warning] CA cer
mysql_1 | 2021-04-25T20:15:15.225005Z 1 [Warning] root@

```

4. (a) `docker-compose up -d`
- (b) `docker-compose pause $(docker-compose ps --services)`

(c) `docker-compose down -v`

Docker in Docker

1. Dockerfile:

```
FROM ubuntu:18.04
RUN apt-get update
RUN apt-get -y install apt-transport-https ca-certificates curl gnupg lsb-release
RUN curl -fsSL https://download.docker.com/linux/ubuntu/gpg | gpg --dearmor -o
/usr/share/keyrings/docker-archive-keyring.gpg
RUN echo "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-
keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
| tee /etc/apt/sources.list.d/docker.list > /dev/null
RUN apt-get update
RUN apt-get -y install docker-ce docker-ce-cli containerd.io

RUN usermod -aG docker $(whoami)
RUN newgrp docker
RUN touch dk.sh; echo "service docker start" >> dk.sh; echo "docker run hello-
world" >> dk.sh; echo "/bin/bash" >> dk.sh
RUN chmod +x dk.sh;
CMD ["/bin/sh", "dk.sh"]
```

2. `docker build -t dind-nasa-hw4:v1.0.0 .` `-t` for the image name and tag, `.` is the directory.
3. `docker run --cap-add=NET_ADMIN --privileged dind-nasa-hw4:v1.0.0`
`--privileged` needed to start docker daemon, `--cap-add=NET_ADMIN` needed to pull hello-world for some reason.

```
/Volumes/Transcend/NTU/nasa-2021-csie/hw4 / main ↑1 !1 ?4 .....
> docker run --cap-add=NET_ADMIN --privileged chenkb/dind-nasa-hw4:v1.0.0
* Starting Docker: docker
...done.
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
b8dfde127a29: Pulling fs layer
b8dfde127a29: Verifying Checksum
b8dfde127a29: Download complete
b8dfde127a29: Pull complete
Digest: sha256:f2266cbfc127c960fd30e76b7c792dc23b588c0db76233517e1891a4e357d51
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

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


Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/
```

<https://stackoverflow.com/questions/45731683/docker-pull-operation-not-permitted>


4. `docker tag dind-nasa-hw4:v1.0.0 chenkb/dind-nasa-hw4:v1.0.0`

`docker login`

`docker push chenkb/dind-nasa-hw4:v1.0.0`

 **docker hub**

[Explore](#) [chenkb/dind-nasa-hw4](#)



chenkb/dind-nasa-hw4 ☆
By [chenkb](#) • Updated 2 hours ago
Container

[Overview](#) [Tags](#)

