# Project Task1 Report

## A. Install and get familiar with Docker

### 1. Install Docker

Follow the tutorials from Docker's official website (https://docs.docker.com/desktop/linux/install/ubuntu/) to install Docker on Ubuntu.
Run command `sudo docker run hello-world` to check if it was installed:

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

Run command `service docker status` to check docker status:

```
enizumi@ubuntu:~$ service docker status
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2022-06-16 00:18:33 PDT; 5min ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 15383 (dockerd)
      Tasks: 9
     Memory: 54.5M
     CGroup: /system.slice/docker.service
             └─15383 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Jun 16 00:18:32 ubuntu dockerd[15383]: time="2022-06-16T00:18:32.630735574-07:00" level=warning msg="Your ker>
Jun 16 00:18:32 ubuntu dockerd[15383]: time="2022-06-16T00:18:32.631027691-07:00" level=info msg="Loading con>
Jun 16 00:18:32 ubuntu dockerd[15383]: time="2022-06-16T00:18:32.820617121-07:00" level=info msg="Default bri>
Jun 16 00:18:33 ubuntu dockerd[15383]: time="2022-06-16T00:18:33.008069276-07:00" level=info msg="Loading con>
Jun 16 00:18:33 ubuntu dockerd[15383]: time="2022-06-16T00:18:33.144009134-07:00" level=info msg="Docker daem>
Jun 16 00:18:33 ubuntu dockerd[15383]: time="2022-06-16T00:18:33.144362143-07:00" level=info msg="Daemon has >
Jun 16 00:18:33 ubuntu systemd[1]: Started Docker Application Container Engine.
Jun 16 00:18:33 ubuntu dockerd[15383]: time="2022-06-16T00:18:33.163697615-07:00" level=info msg="API listen >
Jun 16 00:20:13 ubuntu dockerd[15383]: time="2022-06-16T00:20:13.471952718-07:00" level=info msg="ignoring ev>
Jun 16 00:23:24 ubuntu dockerd[15383]: time="2022-06-16T00:23:24.338640026-07:00" level=info msg="ignoring ev>
lines 1-21/21 (END)
```

### 2. Pull images

use the `docker images` command to view local images:

```
enizumi@ubuntu:~$ docker image ls
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
hello-world   latest    feb5d9fea6a5   8 months ago   13.3kB
enizumi@ubuntu:~$ docker image pull alpine:latest
latest: Pulling from library/alpine
2408cc74d12b: Pull complete
Digest: sha256:686d8c9dfa6f3ccfc8230bc3178d23f84eeaf7e457f36f271ab1acc53015037c
Status: Downloaded newer image for alpine:latest
docker.io/library/alpine:latest
enizumi@ubuntu:~$ docker image ls
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
alpine        latest    e66264b98777   3 weeks ago    5.53MB
hello-world   latest    feb5d9fea6a5   8 months ago   13.3kB
```

### 3. Run and manage containers

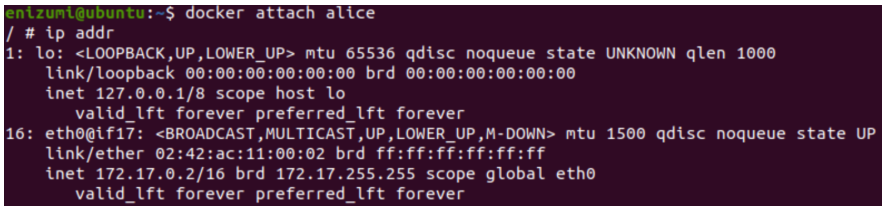
Use command `docker run` to start 2 containers on host1, and another container on host2. Use `docker ps` to view existing containers:

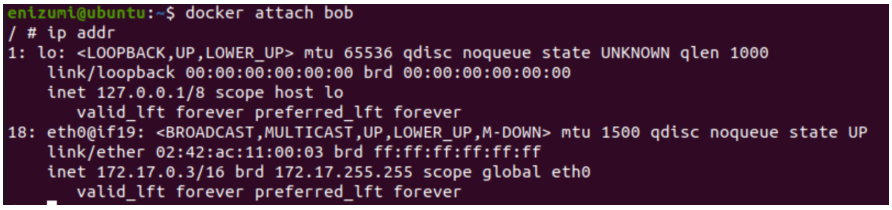
```
enizumi@ubuntu:~$ docker ps -a
CONTAINER ID   IMAGE           COMMAND     CREATED          STATUS          PORTS     NAMES
356d287643e5   alpine:latest   "/bin/sh"   6 seconds ago    Up 5 seconds              bob
6bd431194483   alpine:latest   "/bin/sh"   17 seconds ago   Up 14 seconds             alice
```

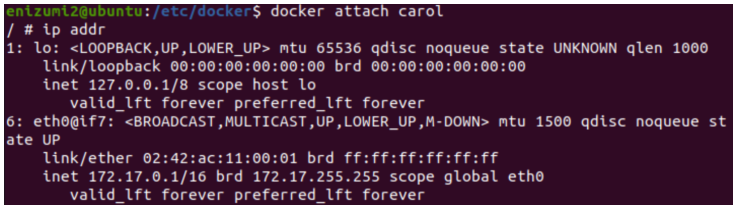
```
enizumi2@ubuntu:~$ docker ps -a
CONTAINER ID   IMAGE           COMMAND     CREATED          STATUS          POR
TS     NAMES
48aea0baae2b   alpine:latest   "/bin/sh"   34 minutes ago   Up 34 minutes
       carol
```

### 4. Enter containers and check their net status

Enter 3 containers and check ip address respectively:




IP address of alice is 172.17.0.2, bob is 172.17.0.3, carol is 172.17.0.1.

alice cannot ping carol because they do not share a network. alice and bob are on the same host, so they share the bridge network by default.

## B. Install and configure the overlay network

### 1. Install and configure etcd

Install etcd, and edit `/usr/lib/systemd/system/etcd.service`. The following script is an example of host1.

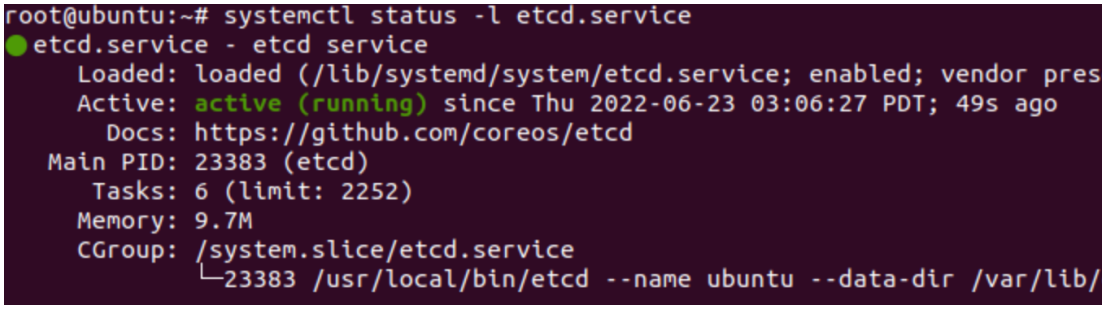
```
[Unit]
Description=etcd service
Documentation=https://github.com/coreos/etcd

[Service]
User=etcd
Type=notify
ExecStart=/usr/local/bin/etcd \
 --name ${ETCD_NAME} \
 --data-dir /var/lib/etcd \
 --initial-advertise-peer-urls http://${ETCD_HOST_IP}:2380 \
 --listen-peer-urls http://${ETCD_HOST_IP}:2380 \
 --listen-client-urls http://${ETCD_HOST_IP}:2379,http://127.0.0.1:2379 \
 --advertise-client-urls http://${ETCD_HOST_IP}:2379 \
 --initial-cluster-token etcd-cluster-1 \
 --initial-cluster ${ETCD_NAME}=http://192.168.117.131:2380,etcd-
2=http://192.168.117.129:2380 \
 --initial-cluster-state new \
 --heartbeat-interval 1000 \
 --election-timeout 5000
Restart=on-failure
```

```
RestartSec=5

[Install]
WantedBy=multi-user.target
```

Check if etcd is working,




Check if the cluster is built successfully,

```
root@ubuntu:~# etcdctl  member list
7e18438618ccbf60, started, ubuntu, http://192.168.117.129:2380, http://192
.168.117.129:2379, false
e418dc762b130398, started, ubuntu, http://192.168.117.131:2380, http://192
.168.117.131:2379, false
```

**2. Install and configure flannel**

Download flannel, and edit .json file.
use etcd to synchronize the config by

```
ECTDCTL_API=2 etcdctl set /coreos.com/network/config < ~/flannel-network-
config.json
```

notice that flannel does not support etcd3 api, so here I add 'enable api 2' in etcd config file. Check if it is synchronized by `etcdctl get /coreos.com/network/config` on host2. Edit `/usr/lib/systemd/system/flanneld.service` as follows (not exactly the same),

```
[Unit]
Description=flannel
After=etcd.service network.target

[Service]
ExecStart=/usr/local/bin/flanneld --etcd-
endpoints=http://192.168.117.131:2379,http://192.168.117.129:2379

[Install]
WantedBy=multi-user.target
```

Check if flanneld is working well, there is a new interface `flannel.100` with subnet ip `10.12.160.0`.

```
● flanneld.service - flannel
     Loaded: loaded (/lib/systemd/system/flanneld.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2022-06-23 07:00:07 PDT; 8s ago
   Main PID: 24411 (flanneld)
      Tasks: 6 (limit: 2252)
     Memory: 6.7M
     CGroup: /system.slice/flanneld.service
             └─24411 /usr/local/bin/flanneld --etcd-endpoints=http://192.168.117.131:2379,http://192.168.117.129:2379
```

Check the ip interface,

```
root@ubuntu:/usr/local/bin# ip r
default via 192.168.117.2 dev ens33 proto dhcp metric 100
10.12.160.0/20 via 10.12.160.0 dev flannel.100 onlink
169.254.0.0/16 dev ens33 scope link metric 1000
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1 linkdown
192.168.117.0/24 dev ens33 proto kernel scope link src 192.168.117.129 metric 100
```

**3. Configure Docker to use the overlay network**

Edit `/usr/lib/systemd/system/docker.service` to set the default network of docker to be the flannel overlay network.
Look for parameters in `/run/flannel/subnet.env`, add `--bip=${FLANNEL_SUBNET} --mtu=${FLANNEL_MTU}` in service config file.
Restart docker, use `docker network inspect bridge` to check its ip, now it becomes the same as flannel subnet.

## Test the overlay network

After configuring etcd and flannel, etcd assigns subnet ip addr range for each host, and those subnets are connect if in the same cluster. The ip addrs of containers changed because new subnet ip addrs are assinged for dockers in the corresponding etcd cluster, then docker will assign distinct ip addrs for its containers. Since the containers are now in the same overlay network, they can ping each other through flannel.

**Vedio Links**

.mov: https://jbox.sjtu.edu.cn/l/718YLZ
.mp4: https://jbox.sjtu.edu.cn/l/s1wTff