2021

Practica 5 – Docker Networking

MARKEL ORALLO NOGUEIRA

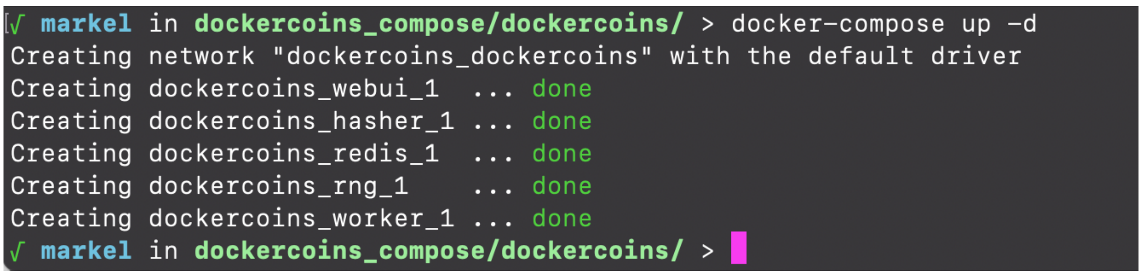
Docker networking

Siguiendo con la práctica de Dockercoins:
¿Cuál es la red que utilizan los diferentes contenedores?

```
version: "3.1"

services:
  rng:
    image: markel149/rng:latest
    networks:
    - dockercoins
    ports:
    - "8001:80"

  hasher:
    image: markel149/hasher:latest
    networks:
    - dockercoins
    ports:
    - "8002:80"

  webui:
    image: markel149/webui:latest
    networks:
    - dockercoins
    ports:
    - "8000:80"

  redis:
    image: redis
    networks:
    - dockercoins

  worker:
    image: markel149/worker:latest
    networks:
    - dockercoins

networks:
    dockercoins:
```

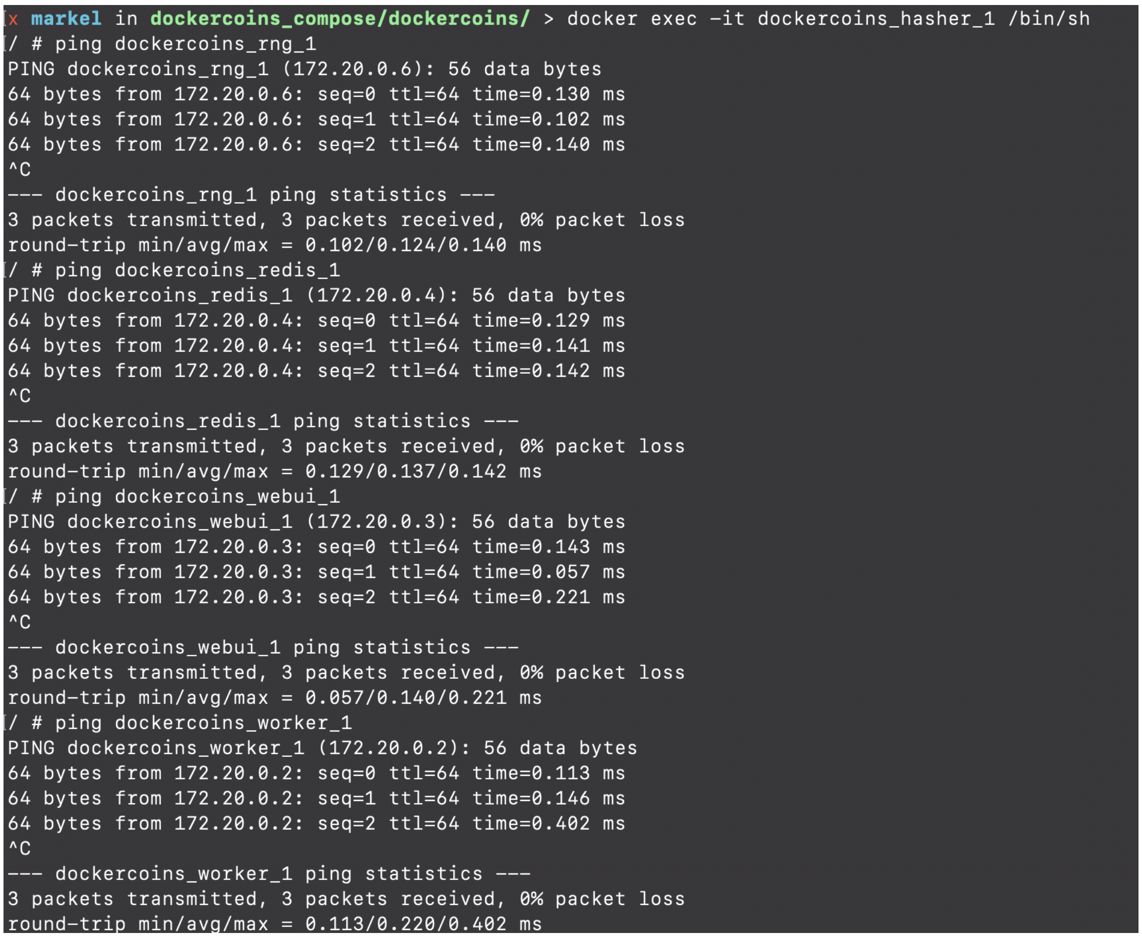
Analizando el docker-compose la red que utilizan los diferentes contenedores es la red dockercoins.

Pon en marcha los contenedores(docker-compose up)



```
√ markel in dockercoins_compose/dockercoins/ > docker-compose up -d
Creating network "dockercoins_dockercoins" with the default driver
Creating dockercoins_webui_1  ... done
Creating dockercoins_hasher_1 ... done
Creating dockercoins_redis_1  ... done
Creating dockercoins_rng_1    ... done
Creating dockercoins_worker_1 ... done
√ markel in dockercoins_compose/dockercoins/ >
```

Entra dentro de uno de los contenedores ($ docker container exec -it <Container ID> bash) y ejecuta el comando ping nombre_contenedor



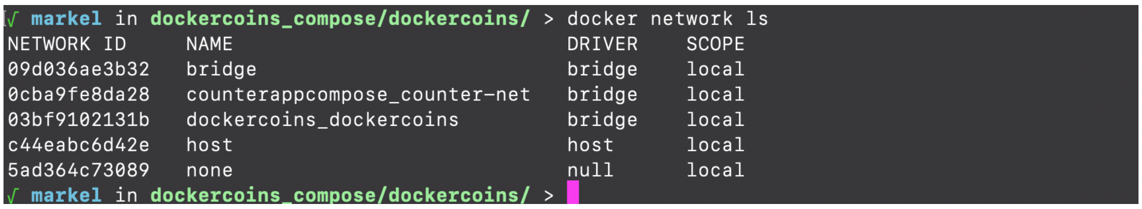
```
x markel in dockercoins_compose/dockercoins/ > docker exec -it dockercoins_hasher_1 /bin/sh
/ # ping dockercoins_rng_1
PING dockercoins_rng_1 (172.20.0.6): 56 data bytes
64 bytes from 172.20.0.6: seq=0 ttl=64 time=0.130 ms
64 bytes from 172.20.0.6: seq=1 ttl=64 time=0.102 ms
64 bytes from 172.20.0.6: seq=2 ttl=64 time=0.140 ms
^C
--- dockercoins_rng_1 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.102/0.124/0.140 ms
/ # ping dockercoins_redis_1
PING dockercoins_redis_1 (172.20.0.4): 56 data bytes
64 bytes from 172.20.0.4: seq=0 ttl=64 time=0.129 ms
64 bytes from 172.20.0.4: seq=1 ttl=64 time=0.141 ms
64 bytes from 172.20.0.4: seq=2 ttl=64 time=0.142 ms
^C
--- dockercoins_redis_1 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.129/0.137/0.142 ms
/ # ping dockercoins_webui_1
PING dockercoins_webui_1 (172.20.0.3): 56 data bytes
64 bytes from 172.20.0.3: seq=0 ttl=64 time=0.143 ms
64 bytes from 172.20.0.3: seq=1 ttl=64 time=0.057 ms
64 bytes from 172.20.0.3: seq=2 ttl=64 time=0.221 ms
^C
--- dockercoins_webui_1 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.057/0.140/0.221 ms
/ # ping dockercoins_worker_1
PING dockercoins_worker_1 (172.20.0.2): 56 data bytes
64 bytes from 172.20.0.2: seq=0 ttl=64 time=0.113 ms
64 bytes from 172.20.0.2: seq=1 ttl=64 time=0.146 ms
64 bytes from 172.20.0.2: seq=2 ttl=64 time=0.402 ms
^C
--- dockercoins_worker_1 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.113/0.220/0.402 ms
```

– Llega al destino? A todos los contenedores?

Si, llega a todos los contenedores.

– ¿Qué driver estará utilizando?

Esta utilizando el driver “bridge”.



```
√ markel in dockercoins_compose/dockercoins/ > docker network ls
NETWORK ID     NAME                            DRIVER    SCOPE
09d036ae3b32   bridge                          bridge    local
0cba9fe8da28   counterappcompose_counter-net   bridge    local
03bf9102131b   dockercoins_dockercoins         bridge    local
c44eabc6d42e   host                            host      local
5ad364c73089   none                            null      local
√ markel in dockercoins_compose/dockercoins/ >
```

Si hacemos docker network inspect dockercoins_dockercoins

```
[
    {
        "Name": "dockercoins_dockercoins",
        "Id":
"03bf9102131bd8e9a328fa6e5e8d61fe2a383cfc9d3bf77530c7713ef3b1da45",
        "Created": "2021-09-27T20:22:22.268898752Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "172.20.0.0/16",
                    "Gateway": "172.20.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": true,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {

"1128ae84ef8bc1ede9855d54015069943c4af90d863687f9a1331e92cb50f374": {
                "Name": "dockercoins_webui_1",
                "EndpointID":
"94933092bc605f5e109a58f7f31edc8113f9be845bef2e6bee98ca81b8d2d0dd",
                "MacAddress": "02:42:ac:14:00:03",
                "IPv4Address": "172.20.0.3/16",
                "IPv6Address": ""
            },

"52b80343223c201c4a17adae713352054c55696d17cae65a96748c8b3eff3eb3": {
                "Name": "dockercoins_redis_1",
                "EndpointID":
"076bd1db64eddd63a2b28d3d685e7672dc080d1ee3113ec8f9783f31914791f7",
                "MacAddress": "02:42:ac:14:00:04",
                "IPv4Address": "172.20.0.4/16",
                "IPv6Address": ""
            },

"59efcc7e2e24d9b8d81c85d790aa41d4b3b6ccc53baaec38ae0975687274cbc4": {
                "Name": "dockercoins_hasher_1",
                "EndpointID":
"1c9b1d0a62694b5cd296d4a8ec062641b9dddca5887fcdddd787a8b6714f5254",
                "MacAddress": "02:42:ac:14:00:05",
                "IPv4Address": "172.20.0.5/16",
                "IPv6Address": ""
            },

"c0061048816a68aca15a4cb7de704ef3662068f1e3121fa0a83e8e4d12c251c0": {
```

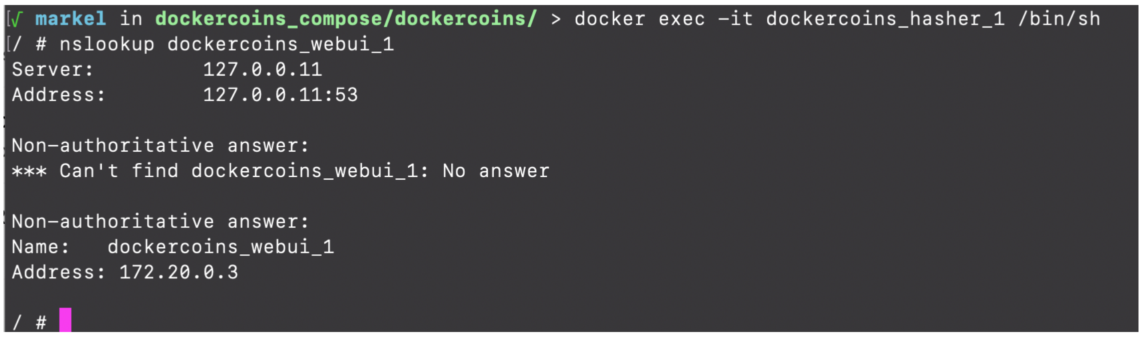
```
                "Name": "dockercoins_worker_1",
                "EndpointID":
"a3832e182afe237160c7a735ac407402c819862f30ad96a88a50834506269a13",
                "MacAddress": "02:42:ac:14:00:02",
                "IPv4Address": "172.20.0.2/16",
                "IPv6Address": ""
            },

"c995c8d3bd56e4122cd95b5fb7e123c8979a42a1cd76fa10540f26b51b386b6e": {
                "Name": "dockercoins_rng_1",
                "EndpointID":
"350fb7838ebd43183e52f8f4c83043b1fa63d8c1cd099648cae4b8ebaef9777f",
                "MacAddress": "02:42:ac:14:00:06",
                "IPv4Address": "172.20.0.6/16",
                "IPv6Address": ""
            }
        },
        "Options": {},
        "Labels": {
            "com.docker.compose.network": "dockercoins",
            "com.docker.compose.project": "dockercoins",
            "com.docker.compose.version": "1.29.2"
        }
    }
]
```

Podemos ver en el output del comando anterior que los 5 contenedores que están conectados a la red son los contenedores que hemos levantado con docker-compose:

- Dockercoins_rng_1
- Dockercoins_worker_1
- Dockercoins_redis_1
- Dockercoins_hasher_1
- Dockercoins_webui_1

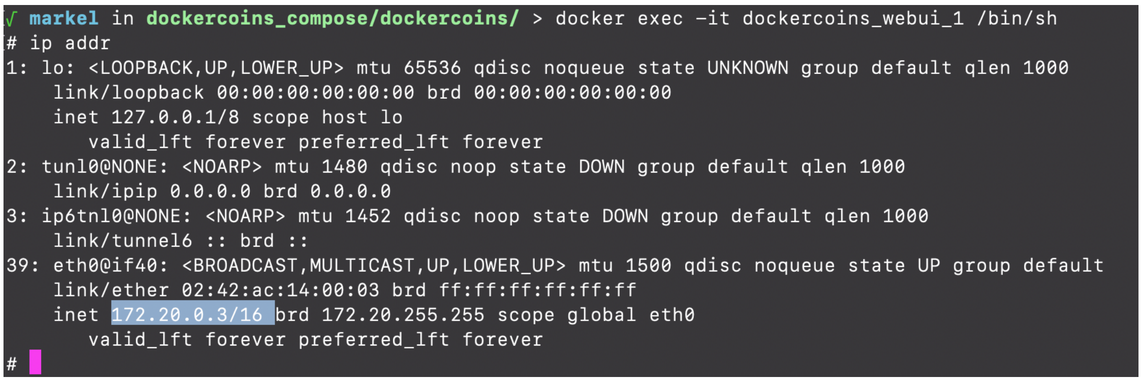
– ¿Hace uso del DNS?

Si hace uso del DNS. Al ser una red diferente a la default los contenedores puedes llegar de uno a otro a través de DNS. Podemos comprobarlo entrando a uno de ellos y resolviendo el nombre de otro contenedor:



```
√ markel in dockercoins_compose/dockercoins/ > docker exec -it dockercoins_hasher_1 /bin/sh
/ # nslookup dockercoins_webui_1
Server:         127.0.0.11
Address:        127.0.0.11:53

Non-authoritative answer:
*** Can't find dockercoins_webui_1: No answer

Non-authoritative answer:
Name:   dockercoins_webui_1
Address: 172.20.0.3

/ #
```

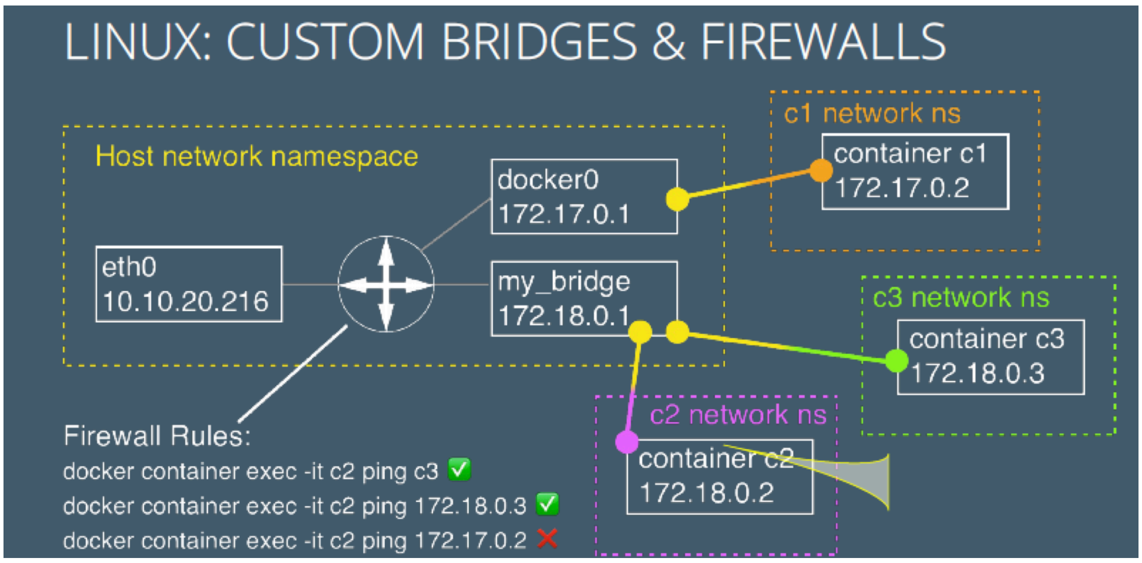
Como podemos ver nos resuelve la IP 172.20.0.3 para el contenedor dockercions_webui_1. Comprobemos que la resolución es correcta:



```
√ markel in dockercoins_compose/dockercoins/ > docker exec -it dockercoins_webui_1 /bin/sh
# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
2: tunl0@NONE: <NOARP> mtu 1480 qdisc noop state DOWN group default qlen 1000
    link/ipip 0.0.0.0 brd 0.0.0.0
3: ip6tnl0@NONE: <NOARP> mtu 1452 qdisc noop state DOWN group default qlen 1000
    link/tunnel6 :: brd ::
39: eth0@if40: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:ac:14:00:03 brd ff:ff:ff:ff:ff:ff
    inet 172.20.0.3/16 brd 172.20.255.255 scope global eth0
       valid_lft forever preferred_lft forever
#
```

Lanzando el comando ip addr dentro del contenedor vemos que la IP de webui es 172.20.0.3.

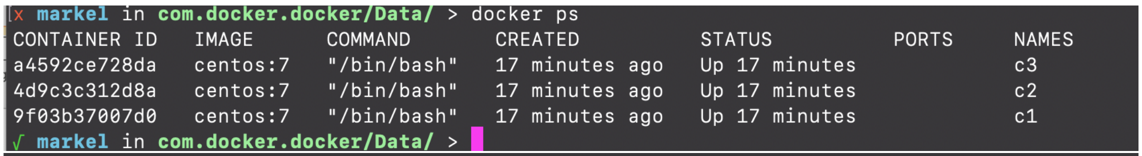
Crea la siguiente arquitectura de redes y contenedores



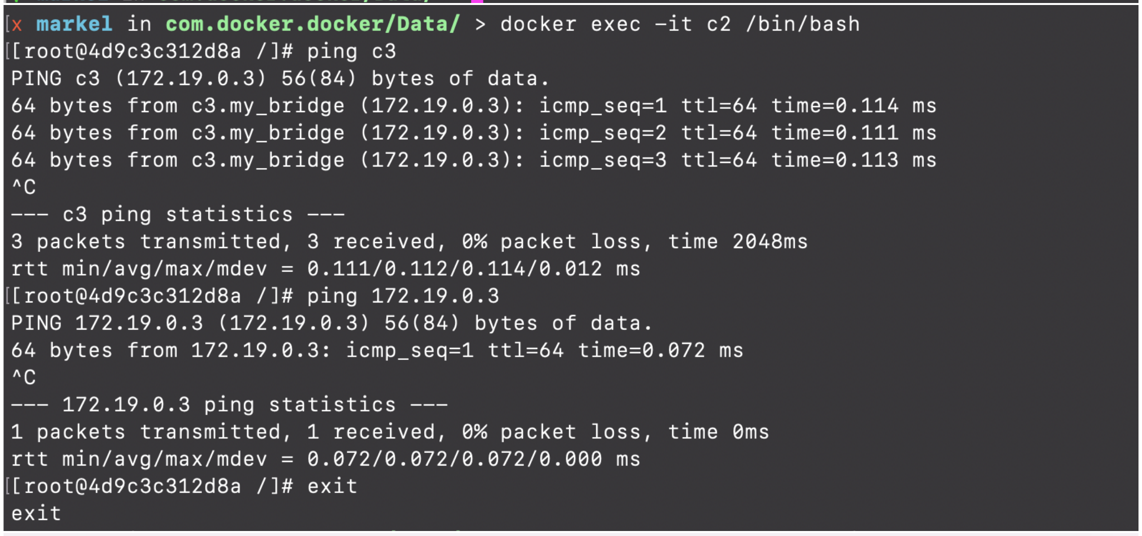
docker network create my_bridge -d bridge
docker run –name=c1 -d -it centos:7 /bin/bash
docker run --net=my_bridge --name=c2 -d -it centos:7 /bin/bash
docker run --net=my_bridge --name=c3 -d -it centos:7 /bin/bash

De esta manera nos quedan dos contenedores (c3 y c2) conectados entre si a través de la red my_bridge. El contenedor c1 al contrario quedará en la red por defecto de docker (docker0).

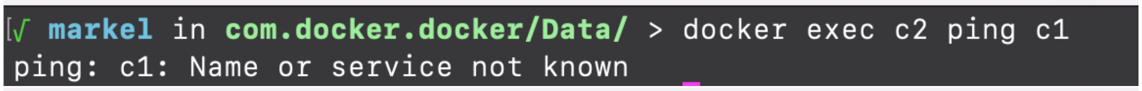
De esta manera al hacer ping de c2 a c3 llegaremos, tanto por nombre como por IP ya que también hace uso del DNS. En cambio, si intentamos hacer ping a c1 no llegaran los paquetes al encontrarse en otra red.



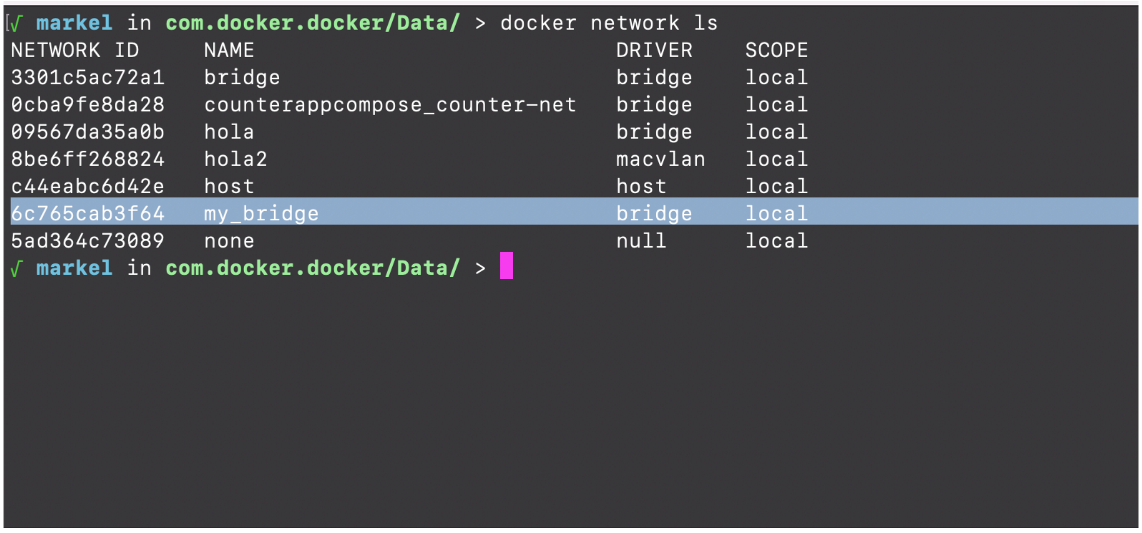
```
x markel in com.docker.docker/Data/ > docker ps
CONTAINER ID   IMAGE      COMMAND       CREATED          STATUS          PORTS     NAMES
a4592ce728da   centos:7   "/bin/bash"   17 minutes ago   Up 17 minutes             c3
4d9c3c312d8a   centos:7   "/bin/bash"   17 minutes ago   Up 17 minutes             c2
9f03b37007d0   centos:7   "/bin/bash"   17 minutes ago   Up 17 minutes             c1
√ markel in com.docker.docker/Data/ >
```



```
x markel in com.docker.docker/Data/ > docker exec -it c2 /bin/bash
[root@4d9c3c312d8a /]# ping c3
PING c3 (172.19.0.3) 56(84) bytes of data.
64 bytes from c3.my_bridge (172.19.0.3): icmp_seq=1 ttl=64 time=0.114 ms
64 bytes from c3.my_bridge (172.19.0.3): icmp_seq=2 ttl=64 time=0.111 ms
64 bytes from c3.my_bridge (172.19.0.3): icmp_seq=3 ttl=64 time=0.113 ms
^C
--- c3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2048ms
rtt min/avg/max/mdev = 0.111/0.112/0.114/0.012 ms
[root@4d9c3c312d8a /]# ping 172.19.0.3
PING 172.19.0.3 (172.19.0.3) 56(84) bytes of data.
64 bytes from 172.19.0.3: icmp_seq=1 ttl=64 time=0.072 ms
^C
--- 172.19.0.3 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.072/0.072/0.072/0.000 ms
[root@4d9c3c312d8a /]# exit
exit
```



```
√ markel in com.docker.docker/Data/ > docker exec c2 ping c1
ping: c1: Name or service not known
```



```
√ markel in com.docker.docker/Data/ > docker network ls
NETWORK ID     NAME                            DRIVER    SCOPE
3301c5ac72a1   bridge                          bridge    local
0cba9fe8da28   counterappcompose_counter-net   bridge    local
09567da35a0b   hola                            bridge    local
8be6ff268824   hola2                           macvlan   local
c44eabc6d42e   host                            host      local
6c765cab3f64   my_bridge                       bridge    local
5ad364c73089   none                            null      local
√ markel in com.docker.docker/Data/ >
```