

EXPERIMENT-3

Aim: *Creating Networks Between Containers using Networks*

- To start with we create the network with our predefined name.

\$ docker network create backend-network

```
5068352cba46f264089330f5d0e29f2aa14d44b4ba98007d3bf268f04b4046de
root@ubuntu-xenial:~# docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
5068352cba46        backend-network     bridge              local
4a554a927f85        bridge             bridge              local
a78f5ee130bc        host               host                local
f5864c242fb8        none               null                local
root@ubuntu-xenial:~#
```

- When we launch new containers, we can use the “*--net*” attribute to assign which network they should be connected to.

\$ docker run -d --name=redis --net=backend-network redis

```
root@ubuntu-xenial:~# docker run -d --name=redis --net=backend-network redis
68b02b783878ce8430839b1627121c576a9515ab143c8352d63ddcd5d51fa977
root@ubuntu-xenial:~# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
68b02b783878        redis              "docker-entrypoint.s..." 12 seconds ago      Up 11 seconds      6379/tcp           redis
root@ubuntu-xenial:~#
```

- Create another redis container with name “*redis_1*” under the “*backend-network*” network.

```
root@ubuntu-xenial:~# docker run -d --name=redis_1 --net=backend-network redis
cbb7251dc409b2cc4b89032b538b0f0fd76ddcd29323ccf35930d515f6687c23
root@ubuntu-xenial:~# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
cbb7251dc409        redis              "docker-entrypoint.s..." 7 seconds ago       Up 6 seconds       6379/tcp           redis_1
68b02b783878        redis              "docker-entrypoint.s..." 2 hours ago         Up 2 hours         6379/tcp           redis
root@ubuntu-xenial:~#
```

\$ docker run -d --name=redis_1 --net=backend-network redis

- Now ping the redis (*redis_1*) container from the newly created alpine (*alpine_1*) container with in the same network (*backend-network*).

```
$ docker run --name=alpine_1 --net=backend-network alpine ping redis_1
```

```
root@ubuntu-xenial:~# docker run --name=alpine_1 --net=backend-network alpine ping redis_1
PING redis_1 (172.18.0.3): 56 data bytes
64 bytes from 172.18.0.3: seq=0 ttl=64 time=0.119 ms
64 bytes from 172.18.0.3: seq=1 ttl=64 time=0.080 ms
64 bytes from 172.18.0.3: seq=2 ttl=64 time=0.071 ms
64 bytes from 172.18.0.3: seq=3 ttl=64 time=0.247 ms
64 bytes from 172.18.0.3: seq=4 ttl=64 time=0.135 ms
64 bytes from 172.18.0.3: seq=5 ttl=64 time=0.076 ms
64 bytes from 172.18.0.3: seq=6 ttl=64 time=0.076 ms
64 bytes from 172.18.0.3: seq=7 ttl=64 time=0.076 ms
64 bytes from 172.18.0.3: seq=8 ttl=64 time=0.094 ms
64 bytes from 172.18.0.3: seq=9 ttl=64 time=0.145 ms
64 bytes from 172.18.0.3: seq=10 ttl=64 time=0.088 ms
64 bytes from 172.18.0.3: seq=11 ttl=64 time=0.076 ms
64 bytes from 172.18.0.3: seq=12 ttl=64 time=0.089 ms
^C
--- redis_1 ping statistics ---
13 packets transmitted, 13 packets received, 0% packet loss
round-trip min/avg/max = 0.071/0.105/0.247 ms
root@ubuntu-xenial:~#
```

- Create another network (frontend-network) and connect the redis (*redis_1*) container with is network.

```
$ docker network connect frontend-network redis_1
```

```
[root@ubuntu-xenial:~# docker network create frontend-network
d0ac8a30b641d2e6c293c0bb3bbb61ac779d7bc2ded83e2a5ef174dd8bb181b6
root@ubuntu-xenial:~# docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
5068352cba46        backend-network     bridge              local
4a554a927f85        bridge              bridge              local
d0ac8a30b641        frontend-network    bridge              local
a78f5ee130bc        host                host                local
f5864c242fb8        none                null                local
root@ubuntu-xenial:~#
root@ubuntu-xenial:~#
root@ubuntu-xenial:~# docker network connect forntend-network redis_1
Error response from daemon: network forntend-network not found
root@ubuntu-xenial:~# docker network connect frontend-network redis_1
root@ubuntu-xenial:~#
```

- Now inspect the “*redis_1*” container to check the network connectivity.

```
$ docker inspect redis_1
```

```
    "Gateway": "",
    "GlobalIPv6Address": "",
    "GlobalIPv6PrefixLen": 0,
    "IPAddress": "",
    "IPPrefixLen": 0,
    "IPv6Gateway": "",
    "MacAddress": "",
    "Networks": {
      "backend-network": {
        "IPAMConfig": null,
        "Links": null,
        "Aliases": [
          "cbb7251dc409"
        ],
        "NetworkID": "5068352cba46f264089330f5d0e29f2aa14d44b4ba98007d3bf268f04b4046de",
        "EndpointID": "54c5ecdba06dfa211767c85f4c00de61096d2773d8f3a3efc32544a02f2f7dc1"
      },
      "frontend-network": {
        "IPAMConfig": {},
        "Links": null,
        "Aliases": [
          "cbb7251dc409"
        ],
        "NetworkID": "d0ac8a30b641d2e6c293c0bb3bbb61ac779d7bc2ded83e2a5ef174dd8bb181b6",
        "EndpointID": "beac76e831b2ffcd8ff95774b247cedf3b4328a052a3e8a47664c44f9cb90e54"
      },
      "Gateway": "172.19.0.1",
      "IPAddress": "172.19.0.2",
      "IPPrefixLen": 16,
      "IPv6Gateway": ""
    }
  }
```

- Now disconnect the “*redis_1*” container from the “*backend-network*” network.

```
$ docker network disconnect backend-network redis_1
```

```
$ docker inspect redis_1
```

```
"NetworkSettings": {
  "Bridge": "",
  "SandboxID": "c60ff396504d3d6ad410fb5a89bf0152ab277cdabc2555000e73428ef67d4f09",
  "HairpinMode": false,
  "LinkLocalIPv6Address": "",
  "LinkLocalIPv6PrefixLen": 0,
  "Ports": {
    "6379/tcp": null
  },
  "SandboxKey": "/var/run/docker/netns/c60ff396504d",
  "SecondaryIPAddresses": null,
  "SecondaryIPv6Addresses": null,
  "EndpointID": "",
  "Gateway": "",
  "GlobalIPv6Address": "",
  "GlobalIPv6PrefixLen": 0,
  "IPAddress": "",
  "IPPrefixLen": 0,
  "IPv6Gateway": "",
  "MacAddress": "",
  "Networks": {
    "frontend-network": {
      "IPAMConfig": {},
      "Links": null,
      "Aliases": [
        "cbb7251dc409"
      ],
      "NetworkID": "d0ac8a30b641d2e6c293c0bb3bbb61ac779d7bc2ded83e2a5ef174dd8bb181b6",
      "EndpointID": "beac76e831b2ffcd8ff95774b247cedf3b4328a052a3e8a47664c44f9cb90e54"
    },
    "backend-network": {
      "IPAMConfig": {},
      "Links": null,
      "Aliases": null,
      "NetworkID": "d0ac8a30b641d2e6c293c0bb3bbb61ac779d7bc2ded83e2a5ef174dd8bb181b6",
      "EndpointID": "beac76e831b2ffcd8ff95774b247cedf3b4328a052a3e8a47664c44f9cb90e54",
      "Gateway": "172.19.0.1",
      "IPAddress": "172.19.0.2",
      "IPPrefixLen": 16,
      "IPv6Gateway": "",
      "GlobalIPv6Address": "",
      "GlobalIPv6PrefixLen": 0,
      "MacAddress": "02:42:ac:13:00:02",
      "DriverOpts": null
    }
  }
}
```