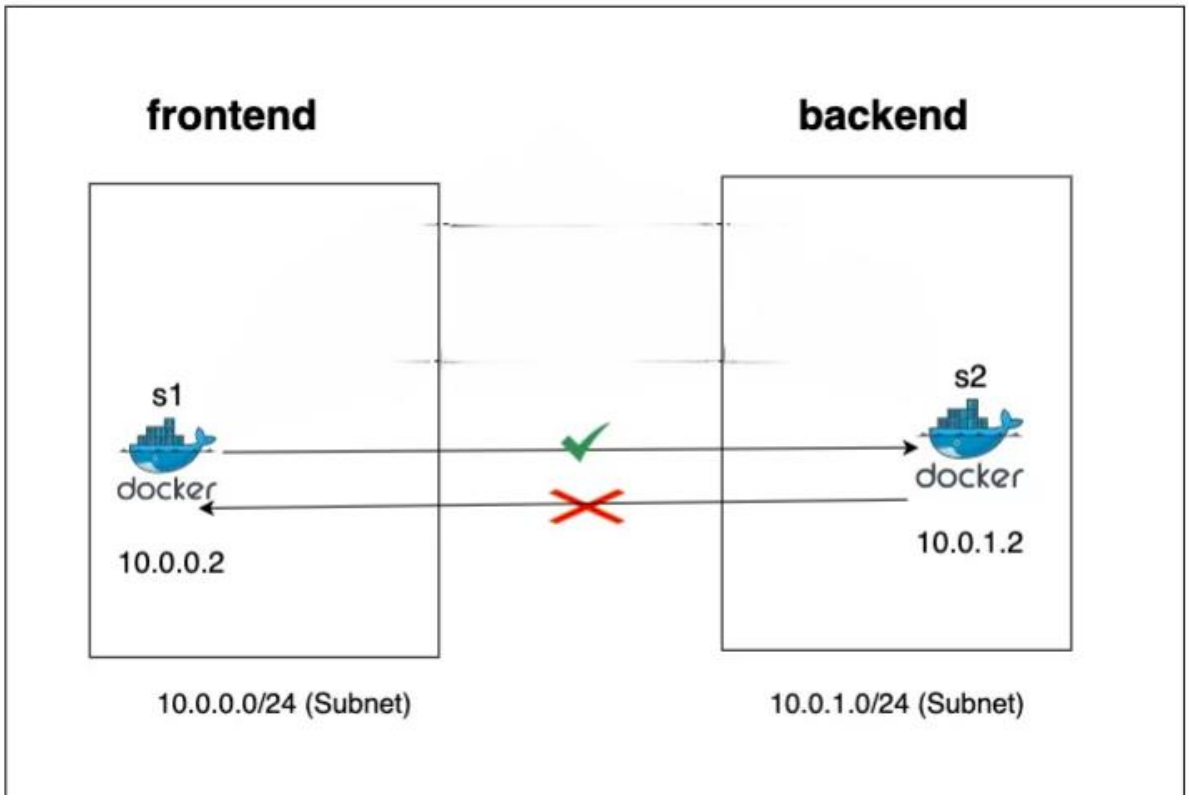
Establish the connectivity between the two Container in different network's



Step 1: Create a Two network

```
controlplane $ docker network create mynet1
a4427bdbcbb327bf7d043e8dbb2b72bc4897179b538425abefe276f87e358057
controlplane $ docker network create mynet2
9652e580fddd304546d9b418369e9b2c0ffcc0b00755128344ecd81189144ee2
controlplane $
```

Step 2: Check the created network

```
controlplane $ docker network create mynet1
a4427bdbcbb327bf7d043e8dbb2b72bc4897179b538425abefe276f87e358057
controlplane $ docker network create mynet2
9652e580fddd304546d9b418369e9b2c0ffcc0b00755128344ecd81189144ee2
controlplane $ docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
677bb6f029f1   bridge    bridge    local
2f4bb03e8f1d   host      host      local
a4427bdbcbb3   mynet1    bridge    local
9652e580fddd   mynet2    bridge    local
280ff08fd1d7   none      null      local
controlplane $
```

Step 3: Pull the two httpd and Busybox images.

```
controlplane $ docker pull docker.io/httpd
Using default tag: latest
latest: Pulling from library/httpd
2cc3ae149d28: Pull complete
840d8df643b2: Pull complete
4f4fb700ef54: Pull complete
9d1465828338: Pull complete
4a16a983b278: Pull complete
9129890c4c50: Pull complete
Digest: sha256:10182d88d7fbc5161ae0f6f758cba7adc56d4aae2dc950e51d72c0cf68967cea
Status: Downloaded newer image for httpd:latest
docker.io/library/httpd:latest
controlplane $ docker pull docker.io/busybox
Using default tag: latest
latest: Pulling from library/busybox
ec562eabd705: Pull complete
Digest: sha256:9ae97d36d26566ff84e8893c64a6dc4fe8ca6d1144bf5b87b2b85a32def253c7
Status: Downloaded newer image for busybox:latest
docker.io/library/busybox:latest
controlplane $ docker images
REPOSITORY   TAG       IMAGE ID       CREATED         SIZE
httpd        latest    bfe6700e6779   2 months ago    147MB
busybox      latest    65ad0d468eb1   13 months ago   4.26MB
controlplane $
```

Step 4: Run the container with define both difference IP addresses

```
controlplane $ docker run --name=c1 --network=mynet1 -d httpd
fdcce74d70c0b8de9fdf8e465209b3af50057ffdc848c516474addbf48c2cfd2
controlplane $ docker run --name=c2 --network=mynet2 -dit docker.io/busybox
f24b5ac4af7883dd06d36bb1174229961f136984c304705c78d250d1f083d855
controlplane $ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS     NAM
ES
f24b5ac4af78   busybox   "sh"                     5 seconds ago    Up 3 seconds              c2
fdcce74d70c0   httpd     "httpd-foreground"       40 seconds ago   Up 38 seconds   80/tcp    c1
controlplane $
```

Step 5: Check the IP address of container 1

```
                "mynet1": {
                    "IPAMConfig": null,
                    "Links": null,
                    "Aliases": [
                        "fdcce74d70c0"
                    ],
                    "NetworkID": "a4427bdbcbb327bf7d043e8dbb2b72bc4897179b538425abefe276f87e
358057",
                    "EndpointID": "30e5d194fa77087c609c12cf2a7fc699f0283a2f19cab37d8f8109afc
2926fe0",
                    "Gateway": "172.18.0.1",
                    "IPAddress": "172.18.0.2",
                    "IPPrefixLen": 16,
                    "IPv6Gateway": "",
                    "GlobalIPv6Address": "",
                    "GlobalIPv6PrefixLen": 0,
                    "MacAddress": "02:42:ac:12:00:02",
                    "DriverOpts": null
                }
            }
        }
    }
]
controlplane $
```

Step 6: Check the IP address of container 2

```
                "mynet2": {
                    "IPAMConfig": null,
                    "Links": null,
                    "Aliases": [
                        "f24b5ac4af78"
                    ],
                    "NetworkID": "9652e580fddd304546d9b418369e9b2c0ffcc0b00755128344ecd81189
144ee2",
                    "EndpointID": "25e487349a0f9dcb8bd445e471103d235dbe58d02d5420127c97695ed
1c90723",
                    "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
                }
            }
        }
    }
]
```

Step 7: Execute the container 2 and ping to the container 1 which is in the different network, then it's not received the packet

```
controlplane $ docker exec -it c2 bin/sh
/ # ping 172.18.0.2
PING 172.18.0.2 (172.18.0.2): 56 data bytes
^C
--- 172.18.0.2 ping statistics ---
16 packets transmitted, 0 packets received, 100% packet loss
/ #
```

Step 8: Connect the container 2 with container one network

```
controlplane $ docker network connect mynet1 c2
controlplane $
```

Step 9: Check the Result.

- Execute the container 2 and ping to the container 1

```
controlplane $ docker exec -it c2 bin/sh
/ # ping 172.18.0.2
PING 172.18.0.2 (172.18.0.2): 56 data bytes
64 bytes from 172.18.0.2: seq=0 ttl=64 time=0.877 ms
64 bytes from 172.18.0.2: seq=1 ttl=64 time=0.095 ms
64 bytes from 172.18.0.2: seq=2 ttl=64 time=0.083 ms
64 bytes from 172.18.0.2: seq=3 ttl=64 time=0.094 ms
64 bytes from 172.18.0.2: seq=4 ttl=64 time=0.091 ms
^C
--- 172.18.0.2 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 0.083/0.248/0.877 ms
/ #
```