

## TALLER REDES EN DOCKER

### Parte A – Red Bridge (local, por defecto)

1. Crear una nueva red tipo **bridge** definiendo una subred y máscara distintas a las que vienen por defecto.

```
C:\Users\pinnc>docker network create --driver bridge --subnet=192.168.100.0/24 --gateway=192.168.100.1 mi-red-bridge
8a5da357e86f018ca46257fd84dc26857f929b2883bb9e0185bc9bbec8fe990c

C:\Users\pinnc>docker network ls
NETWORK ID        NAME                DRIVER  SCOPE
087cd3a8f855     blogwaira_default  bridge  local
5fdeaa0a9670     bridge             bridge  local
7bda2d1bc7e9     host               host    local
8a5da357e86f     mi-red-bridge      bridge  local
bbcd80e5be77     none              null    local
a57821dc0f3d     red1               bridge  local

C:\Users\pinnc>docker network inspect mi-red-bridge
[
  {
    "Name": "mi-red-bridge",
    "Id": "8a5da357e86f018ca46257fd84dc26857f929b2883bb9e0185bc9bbec8fe990c",
    "Created": "2025-09-08T00:47:23.394913016Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv4": true,
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "192.168.100.0/24",
          "Gateway": "192.168.100.1"
        }
      ]
    }
  }
]
```

2. Crear dos contenedores a partir de una misma imagen.

```
C:\Users\pinnc>docker run -dit --name contenedor1 alpine:latest
Unable to find image 'alpine:latest' locally
latest: Pulling from library/alpine
9824c27679d3: Pull complete
Digest: sha256:4bcff63911fcb4448bd4fdaccec207030997caf25e9bea4045fa6c8c44de311d1
Status: Downloaded newer image for alpine:latest
eca8bf9b3d05190c4ed738885cdf1b054a7a6f564f47f3a777700a0d3e44ff33

C:\Users\pinnc>docker run -dit --name contenedor2 alpine:latest
382ffb9a3a3fcd6e2952d7cf21439708953a7ead6aee0eec10f5be86b29430eb

C:\Users\pinnc>docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS
382ffb9a3a3f   alpine:latest  "/bin/sh"               28 seconds ago Up 28 seconds
contenedor2
eca8bf9b3d05   alpine:latest  "/bin/sh"               48 seconds ago Up 48 seconds
contenedor1
1b058fa9da86   mysql:8.0      "docker-entrypoint.s..." 7 weeks ago    Up 12 minutes  33060/tcp, 0.0.0.0:3307->3306/tcp
cp blogwaira-db-1

C:\Users\pinnc>
```

### 3. Conectar uno de los contenedores a la red creada.

```
C:\Users\pinnc>docker network connect mi-red-bridge contenedor1
C:\Users\pinnc>docker network inspect mi-red-bridge
[
  {
    "Name": "mi-red-bridge",
    "Id": "8a5da357e86f018ca46257fd84dc26857f929b2883bb9e0185bc9bbec8fe990c",
    "Created": "2025-09-08T00:47:23.394913016Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv4": true,
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "192.168.100.0/24",
          "Gateway": "192.168.100.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
      "eca8bf9b3d05190c4ed73885cdf1b054a7a6f564f47f3a777700a0d3e44ff33": {
        "Name": "contenedor1",
        "EndpointID": "c43b879d5652e783595d2afdb57fe7a246f8de52883e811e511288e4bb0255b3",
        "MacAddress": "fe:12:4c:ad:b4:7d",
        "IPv4Address": "192.168.100.2/24",
        "IPv6Address": ""
      }
    },
    "Options": {
      "com.docker.network.enable_ipv4": "true",

```

### 4. Desde un contenedor, intentar la comunicación con el otro (ping).

```
C:\Users\pinnc>docker inspect contenedor2 | grep IPAddress
  "SecondaryIPAddresses": null,
  "IPAddress": "172.17.0.3",
  "IPAddress": "172.17.0.3",

C:\Users\pinnc>docker exec -it contenedor1 sh
/ # ping 172.17.0.3
PING 172.17.0.3 (172.17.0.3): 56 data bytes
64 bytes from 172.17.0.3: seq=0 ttl=64 time=0.179 ms
64 bytes from 172.17.0.3: seq=1 ttl=64 time=0.068 ms
64 bytes from 172.17.0.3: seq=2 ttl=64 time=0.101 ms
64 bytes from 172.17.0.3: seq=3 ttl=64 time=0.105 ms
64 bytes from 172.17.0.3: seq=4 ttl=64 time=0.156 ms
64 bytes from 172.17.0.3: seq=5 ttl=64 time=0.151 ms
64 bytes from 172.17.0.3: seq=6 ttl=64 time=0.182 ms
64 bytes from 172.17.0.3: seq=7 ttl=64 time=0.156 ms
64 bytes from 172.17.0.3: seq=8 ttl=64 time=0.121 ms
64 bytes from 172.17.0.3: seq=9 ttl=64 time=0.117 ms
64 bytes from 172.17.0.3: seq=10 ttl=64 time=0.090 ms
64 bytes from 172.17.0.3: seq=11 ttl=64 time=0.099 ms
64 bytes from 172.17.0.3: seq=12 ttl=64 time=0.110 ms
64 bytes from 172.17.0.3: seq=13 ttl=64 time=0.112 ms
64 bytes from 172.17.0.3: seq=14 ttl=64 time=0.114 ms
64 bytes from 172.17.0.3: seq=15 ttl=64 time=0.119 ms
64 bytes from 172.17.0.3: seq=16 ttl=64 time=0.056 ms
64 bytes from 172.17.0.3: seq=17 ttl=64 time=0.104 ms
64 bytes from 172.17.0.3: seq=18 ttl=64 time=0.104 ms
64 bytes from 172.17.0.3: seq=19 ttl=64 time=0.097 ms
64 bytes from 172.17.0.3: seq=20 ttl=64 time=0.133 ms
```

La comunicación fallar o tiene limitaciones porque están en redes diferentes se queda en un bucle infinito intentado establecer conexión.

5. Conectar el contenedor que estaba fuera de la red a la red creada.

```
C:\Users\pinnc>docker network connect mi-red-bridge contenedor2
C:\Users\pinnc>docker network inspect mi-red-bridge
[
  {
    "Name": "mi-red-bridge",
    "Id": "8a5da357e86f018ca46257fd84dc26857f929b2883bb9e0185bc9bbec8fe990c",
    "Created": "2025-09-08T00:47:23.394913016Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv4": true,
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "192.168.100.0/24",
          "Gateway": "192.168.100.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
      "382ffb9a3a3fcd6e2952d7cf21439708953a7ead6aee0eec10f5be86b29430eb": {
        "Name": "contenedor2",
        "EndpointID": "6b7f32f77ccc1016ad2c912c867127b304e7cfa65a75a70ff25b96af20abf0c0",
        "MacAddress": "fe:38:ee:2f:86:1e",
        "IPv4Address": "192.168.100.3/24",
        "IPv6Address": ""
      },
      "eca8bf9b3d05190c4ed738885cdf1b054a7a6f564f47f3a777700a0d3e44ff33": {
        "Name": "contenedor1",
        "EndpointID": "c43b879d5652e783595d2afdb57fe7a246f8de52883e811e511288e4bb0255b3",
        "MacAddress": "fe:12:4c:ad:b4:7d",
        "IPv4Address": "192.168.100.2/24",
        "IPv6Address": ""
      }
    }
  }
]
```

6. Volver a probar la comunicación y registrar las diferencias observadas.

```
C:\Users\pinnc>docker inspect contenedor2 | grep IPAddress
    "SecondaryIPAddresses": null,
    "IPAddress": "172.17.0.3",
    "IPAddress": "172.17.0.3",
    "IPAddress": "192.168.100.3",

C:\Users\pinnc>docker exec -it contenedor1 sh
/ # ping -c 3 contenedor2
PING contenedor2 (192.168.100.3): 56 data bytes
64 bytes from 192.168.100.3: seq=0 ttl=64 time=0.099 ms
64 bytes from 192.168.100.3: seq=1 ttl=64 time=0.126 ms
64 bytes from 192.168.100.3: seq=2 ttl=64 time=0.071 ms
```

Se establece buena conexión tanto usando la ip como con el nombre del contenedor

## 7. Revisar los parámetros de configuración de la red y los contenedores conectados a ella.

```
C:\Users\pinnc>docker network inspect mi-red-bridge
[
  {
    "Name": "mi-red-bridge",
    "Id": "8a5da357e86f018ca46257fd84dc26857f929b2883bb9e0185bc9bbec8fe990c",
    "Created": "2025-09-08T00:47:23.394913016Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv4": true,
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "192.168.100.0/24",
          "Gateway": "192.168.100.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
      "302f49b9a3a3fcd6e2952d7cf21439708953a7ead6aee0ec10f5be86b29430eb": {
        "Name": "contenedor2",
        "EndpointID": "6b7f32f77ccc1016ad2c912c867127b304e7cfa65a75a70ff25b96af20abf0c0",
        "MacAddress": "fe:38:ee:2f:86:1e",
        "IPv4Address": "192.168.100.3/24",
        "IPv6Address": ""
      },
      "eca8bf9b3d05190c4ed738885cdf1b054a7a6f564f47f3a777700a0d3e44ff33": {
        "Name": "contenedor1",
        "EndpointID": "c43b079d5652e783595d2afdb57fe7a246f8de52883e811e511288e4bb0255b3",
        "MacAddress": "fe:12:4c:ad:b4:7d",
        "IPv4Address": "192.168.100.2/24",
        "IPv6Address": ""
      }
    },
    "Options": {
      "com.docker.network.enable_ipv4": "true",
      "com.docker.network.enable_ipv6": "false"
    },
    "Labels": {}
  }
]
```

## Parte B – Red Host (uso del stack de red del host)

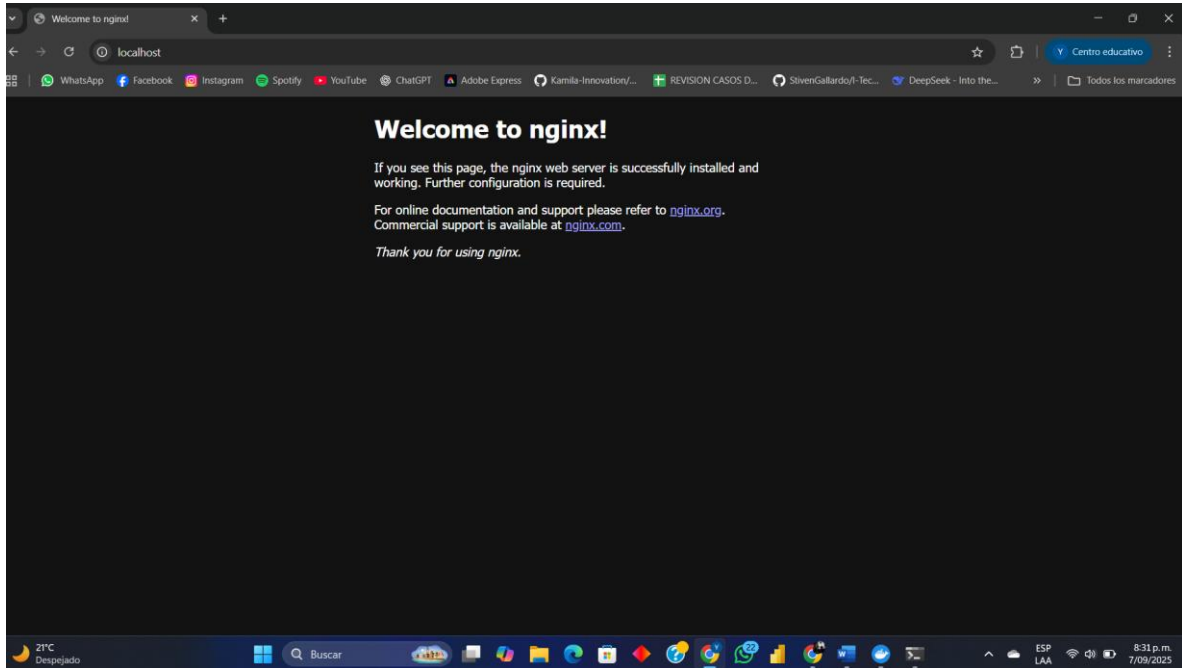
## 8. Crear un contenedor utilizando la red **host**.

```
C:\Users\pinnc>docker run -dit --name contenedor-host --network host nginx:alpine
Unable to find image 'nginx:alpine' locally
alpine: Pulling from library/nginx
cblff4086f82: Pull complete
c9ebe2ff2d2c: Pull complete
6bc572a340ec: Pull complete
403e3f251637: Pull complete
9adfbae99cb7: Pull complete
a992fbc61ecc: Pull complete
7a8a46701e18: Pull complete
Digest: sha256:42a516af16b852e33b7682d5ef8acbd5d13fe08fecadc7ed98605ba5e3b26ab8
Status: Downloaded newer image for nginx:alpine
3579da8396496b7799f39047d9700c4471339dfe1426a5ac8af0b08b6e622b92

C:\Users\pinnc>docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
3579da839649   nginx:alpine  "/docker-entrypoint. ..."  10 seconds ago Up 10 seconds        contenedor-host
382ff9a3a3f    alpine:latest "/bin/sh"              29 minutes ago Up 29 minutes        contenedor2
eca8bf9b3d05   alpine:latest "/bin/sh"              30 minutes ago Up 30 minutes        contenedor1
1b058fa9da86   mysql:8.0     "docker-entrypoint.s ..."  7 weeks ago   Up 42 minutes   33060/tcp, 0.0.0.0:3307->3306/tcp   blogwaira-db-1

C:\Users\pinnc>
```

9. Verificar cómo se comporta respecto a la interfaz de red del host (por ejemplo, levantar un servidor web dentro del contenedor y acceder desde el navegador del host).



10. Comparar la salida de ifconfig/ip a dentro del contenedor con la del host.

### Instalación net-tools

```
C:\Users\pinn>docker exec -it contenedor-host sh
# ifconfig
sh: 1: ifconfig: not found
# ip addr show
sh: 2: ip: not found
# apt update && apt install -y net-tools iproute2
Get:1 http://deb.debian.org/debian bookworm InRelease [151 kB]
Get:2 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Get:3 http://deb.debian.org/debian-security bookworm-security InRelease [48.0 kB]
Get:4 http://deb.debian.org/debian bookworm/main amd64 Packages [8791 kB]
Get:5 http://deb.debian.org/debian bookworm-updates/main amd64 Packages [6924 B]
Get:6 http://deb.debian.org/debian-security bookworm-security/main amd64 Packages [277 kB]
Fetched 9330 kB in 2s (5931 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
29 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libatlas libbpf1 libcap2 libcap2-bin libelf1 libmnl0 libpam-cap libtirpc-common libtirpc3 libxtables12
Suggested packages:
  iproute2-doc python3:any
The following NEW packages will be installed:
  iproute2 libatlas libbpf1 libcap2-bin libelf1 libmnl0 libpam-cap libtirpc-common libtirpc3 libxtables12 net-tools
The following packages will be upgraded:
  libcap2
1 upgraded, 11 newly installed, 0 to remove and 28 not upgraded.
Need to get 1896 kB of archives.
After this operation, 6773 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian bookworm/main amd64 libcap2 amd64 1:2.66-4+deb12u2 [27.2 kB]
Get:2 http://deb.debian.org/debian bookworm/main amd64 libelf1 amd64 0.188-2.1 [174 kB]
Get:3 http://deb.debian.org/debian bookworm/main amd64 libbpf1 amd64 1:1.1.2-0+deb12u1 [145 kB]
Get:4 http://deb.debian.org/debian bookworm/main amd64 libmnl0 amd64 1.0.4-3 [12.5 kB]
Get:5 http://deb.debian.org/debian bookworm/main amd64 libtirpc-common all 1.3.3+ds-1 [14.0 kB]
Get:6 http://deb.debian.org/debian bookworm/main amd64 libtirpc3 amd64 1.3.3+ds-1 [85.2 kB]
Get:7 http://deb.debian.org/debian bookworm/main amd64 libxtables12 amd64 1.8.9-2 [38.8 kB]
Get:8 http://deb.debian.org/debian bookworm/main amd64 libcap2-bin amd64 1:2.66-4+deb12u2 [34.9 kB]
Get:9 http://deb.debian.org/debian bookworm/main amd64 iproute2 amd64 6.1.0-3 [1046 kB]
Get:10 http://deb.debian.org/debian bookworm/main amd64 libatlas amd64 1:2.5.1-4+b2 [68.3 kB]
Get:11 http://deb.debian.org/debian bookworm/main amd64 libpam-cap amd64 1:2.66-4+deb12u2 [14.7 kB]
Get:12 http://deb.debian.org/debian bookworm/main amd64 net-tools amd64 2.10-0.1+deb12u2 [243 kB]
Fetched 1896 kB in 8s (2886 kB/s)
```

## Uso ifconfig

```
Símbolo del sistema - docker  X  +  v

# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 172.17.0.4  netmask 255.255.0.0  broadcast 172.17.255.255
    ether 1e:67:18:00:09:40  txqueuelen 0  (Ethernet)
    RX packets 7976  bytes 11770237 (11.2 MiB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 2785  bytes 189042 (184.6 KiB)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
    inet 127.0.0.1  netmask 255.0.0.0
    inet6 ::1  prefixlen 128  scopeid 0x10<host>
    loop txqueuelen 1000  (Local Loopback)
    RX packets 0  bytes 0 (0.0 B)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 0  bytes 0 (0.0 B)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

# |
```

11. Responder: ¿qué implicaciones de seguridad y de rendimiento tiene este modo?

- Comparte el stack de red del host completamente
- Mejor rendimiento, mayores riesgos de seguridad
- Sin aislamiento de red
- Posibles conflictos de puertos entre contenedores
- Mejor rendimiento de red

## Parte C – Red None (aislamiento total)

12. Crear un contenedor con la red **none**.

```
C:\Users\pinnc>docker run -dit --name contenedor-aislado --network none alpine:latest
6a0ac56a04cd98784b1ec0a9e34f3f266bafeb6422d2aa91fcb55422735e3b0f

C:\Users\pinnc>docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
6a0ac56a04cd   alpine:latest  "/bin/sh"               10 seconds ago Up 9 seconds                               contenedor-aisla
do
edf2a8165ef6   nginx         "/docker-entrypoint..." 18 minutes ago Up 18 minutes  0.0.0.0:80->80/tcp                 contenedor-host
382ffb9a3a3f   alpine:latest  "/bin/sh"               56 minutes ago Up 56 minutes                               contenedor2
eca8bf9b3d05   alpine:latest  "/bin/sh"               57 minutes ago Up 57 minutes                               contenedor1
1b058fa9da86   mysql:8.0     "docker-entrypoint.s..." 7 weeks ago   Up About an hour  33060/tcp, 0.0.0.0:3307->3306/tcp    blogwaira-db-1

C:\Users\pinnc>
```

13. Comprobar la ausencia de interfaces de red configuradas más allá de loopback.

```
C:\Users\pinnc>docker exec -it contenedor-aislado sh
/ # ifconfig
lo      Link encap:Local Loopback
        inet addr:127.0.0.1  Mask:255.0.0.0
        inet6 addr: ::1/128 Scope:Host
        UP LOOPBACK RUNNING  MTU:65536  Metric:1
        RX packets:0 errors:0 dropped:0 overruns:0 frame:0
        TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

/ # ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN 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
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
/ # |
```

14. Intentar hacer ping a otro contenedor o al host. ¿Qué sucede?

```
C:\Users\pinnc>docker exec -it contenedor-aislado sh
/ # ping -c 2 8.8.8.8
PING 8.8.8.8 (8.8.8.8): 56 data bytes
ping: sendto: Network unreachable
/ # ping -c 1 127.0.0.1
PING 127.0.0.1 (127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: seq=0 ttl=64 time=0.641 ms

--- 127.0.0.1 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.641/0.641/0.641 ms
/ # ping -c 2 contenedor1
ping: bad address 'contenedor1'
/ # |
```

15. Reflexionar: ¿para qué casos de uso podría servir este tipo de red?

- Para tareas específicas sin requerimientos de red
- Solo para compartir volúmenes
- Ideal para procesamiento sin conectividad
- Cuando se requiere aislamiento total de red
- Para probar aplicaciones sin dependencias de red

Enviar al correo [nicolas.jurado@itp.edu.co](mailto:nicolas.jurado@itp.edu.co)

Fecha máxima de entrega martes 9 de septiembre