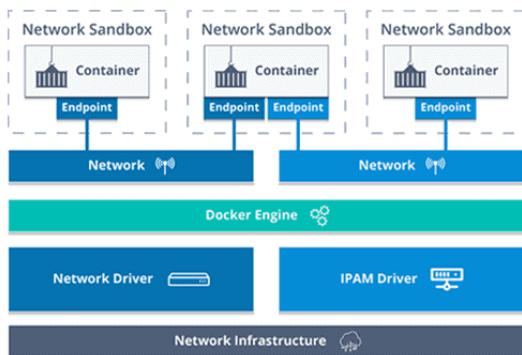


# NETWORKING

## GABRIEL ELIAS GIMENES



---

## Docker Networking For Beginners

---

## PASO 1:

## Vamos a lanzar un contenedor:

```
root@UbuntuServ:/home/vboxuser# docker run -d -p --name web httpd
docker: invalid containerPort: --name

Run 'docker run --help' for more information
root@UbuntuServ:/home/vboxuser# docker run -d -P --name web httpd
99f4f796458bf78e6a46d14c56d7d80c802a9581d9302ee1d732112f3bb
root@UbuntuServ:/home/vboxuser#
```

Vamos a comprobarlo:

IMAGE	ID	DISK USAGE	CONTENT SIZE	EXTRA
gellias/apache:latest	c7c5d67f9165	392MB	182MB	[U]
gabrieleg/apache-passenger:latest	a86118322d329	254MB	80.5MB	[U]
gellias/apache-passenger:latest	b79032f45e8e	254MB	80.5MB	[U]
gellias/apache-passenger:v1	227e1192d706	479MB	130MB	[U]
gellias/apache-passenger:v2	3f72fdcc0eaf	487MB	130MB	[U]

## PASO 2:

Vamos a inspeccionar nuestra imagen con docker inspect web:

### PASO 3:

Ahora hemos hecho un inspector de nuestro network bridge

```
"EnableIPv6": false,
"IPAM": {
    "Driver": "default",
    "Options": null,
    "Config": [
        {
            "Subnet": "172.17.0.0/16",
            "IPRange": "",
            "Gateway": "172.17.0.1"
        }
    ]
},
"Internal": false,
"Attachable": false,
"Ingress": false,
"ConfigFrom": {
    "Network": ""
},
"ConfigOnly": false,
"Options": {
    "com.docker.network.bridge.default_bridge": "true",
    "com.docker.network.bridge.enable_icc": "true",
    "com.docker.network.bridge.enable_ip_masquerade": "true",
    "com.docker.network.bridge.host_binding_ipv4": "0.0.0.0",
    "com.docker.network.bridge.name": "docker0",
    "com.docker.network.driver.mtu": "1500"
},
"Labels": {},
"Containers": {
    "99f41796458bf178df78e6a46d14c56d7d80c802a9581d9302ee1d732112f3bb": {
        "Name": "web",
        "EndpointID": "ec3230d98807c6e7409397c466127224fe8aba6b81152425e5385d2c9ebe083b",
        "MacAddress": "a2:7d:a9:3f:65:89",
        "IPv4Address": "172.17.0.2/16",
        "IPv6Address": ""
    }
},
"Status": {
    "IPAM": {
        "Subnets": {
            "172.17.0.0/16": {
                "IPsInUse": 4,
                "DynamicIPsAvailable": 65532
            }
        }
    }
}
}
```

1 2  
3 4

Num Lock Off

Ahora vamos a crear otro network bridge usando docker network create y el tipo es bridge

```
root@UbuntuServ:/home/vboxuser# docker network create -d bridge miRedLocal
b8c0e50ce5c95537d4c54667ca7ffd0dbd2093051b74e7fe4365dda26abf03d4
root@UbuntuServ:/home/vboxuser#
```

Ahora veremos si se ha creado con docker network ls

```
root@UbuntuServ:/home/vboxuser# docker network create -d bridge miRedLocal
b8c0e50ce5c95537d4c54667ca7ffd0dbd2093051b74e7fe4365dda26abf03d4
root@UbuntuServ:/home/vboxuser# docker network ls
NETWORK ID      NAME      DRIVER      SCOPE
32bb4d8cae23   bridge     bridge     local
f465bcd92eb5   host      host      local
b8c0e50ce5c9   miRedLocal  bridge     local
7e76199d8b9e   none      null      local
root@UbuntuServ:/home/vboxuser#
```

Ahora lo que hemos hecho es hacer un inspect de nuestra red creada

```
root@UbuntuServ:/home/vboxuser# docker network ls
NETWORK ID      NAME      DRIVER      SCOPE
32bb4d8cae23   bridge    bridge      local
f465bcd92eb5   host      host       local
b8c0e50ce5c9   miRedLocal bridge      local
7e7619d8b9e    none     null      local
root@UbuntuServ:/home/vboxuser# docker network inspect miRedLocal
[{"Name": "miRedLocal", "Id": "b8c0e50ce5c95537d4c54667ca7ffd0dbd2093051b74e7fe4365dda26abf03d4", "Created": "2026-01-22T11:53:56.725Z", "Scope": "local", "Driver": "bridge", "EnableIPv4": true, "EnableIPv6": false, "IPAM": {"Driver": "default", "Options": {}, "Config": [{"Subnet": "172.18.0.0/16", "IPRange": "", "Gateway": "172.18.0.1"}]}, "Internal": false, "Attachable": false, "Ingress": false, "ConfigFrom": {"Network": ""}, "ConfigOnly": false, "Options": {}, "Labels": {}, "Containers": {}, "Status": {"IPAM": {"Subnets": {"172.18.0.0/16": {"IPsInUse": 3, "DynamicIPsAvailable": 65533}}}}, "EndpointID": "152861b0a9bb8f71f0ddc9d8458e9cf6d7499a59037b149537c0880f3c68929", "MacAddress": "0a:95:da:21:8d:46", "IPPrefixLen": 16, "IPV6Gateway": "", "GlobalIPv6Address": "", "GlobalIPv6PrefixLen": 0, "DNSNames": ["web1", "b42605eb17cc"], "Gateway": "172.18.0.1", "IPAddress": "172.18.0.2", "Platform": {"Architecture": "amd64", "OS": "linux"}}, "ImageManifestDescriptor": {"MediaType": "application/vnd.oci.image.manifest.v1+json", "Digest": "sha256:d17906bc35fb0527e64cf6410704e94cb9cb4657ce690a0107cd5b56943ff9eb0", "Size": 2093, "Annotations": {"com.docker.official-images.bashbrew.arch": "amd64", "org.opencontainers.image.base.digest": "sha256:ef514b33e858a6ddd5a2af2b50f08f7ff2e43726a14d5f53bdb1d75220dfa5fb", "org.opencontainers.image.base.name": "debian:trixie-slim", "org.opencontainers.image.created": "2026-01-13T01:20:52Z", "org.opencontainers.image.revision": "b8bf24dec3fb94efd3d81ac495bea8247d5115d9", "org.opencontainers.image.source": "https://github.com/docker-library/httpd.git#b8bf24dec3fb94efd3d81ac495bea8247d5115d9:2.4", "org.opencontainers.image.url": "https://hub.docker.com/_/httpd", "org.opencontainers.image.version": "2.4.66"}, "Platform": {"Architecture": "amd64", "OS": "linux"}}, "RootFs": {"Type": "oci", "Layers": [{"blob": "sha256:ef514b33e858a6ddd5a2af2b50f08f7ff2e43726a14d5f53bdb1d75220dfa5fb"}]}]
```



#### PASO 4:

Ahora vamos a crear otro contenedor para que se conecte a nuestra red creada

```
root@UbuntuServ:/home/vboxuser# docker run -d --network=miRedLocal --name web1 httpd
b42605eb17cc0202da07f938e8dab46857ed487b8199a04398b2e34154af5405
root@UbuntuServ:/home/vboxuser#
```

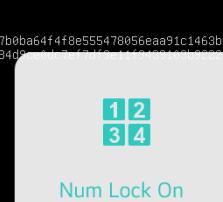


Ahora hemos hecho un inspect sobre nuestra imagen para ver a la ip que está conectada

```
"SandboxID": "a0261a89542e12be6279d6fb249e4526addd9dc1c49ad5f3358a5489cce9cc", "SandboxKey": "/var/run/docker/netns/a0261a89542e", "Ports": {"80/tcp": null}, "Networks": {"miRedLocal": {"IPAMConfig": null, "Links": null, "Aliases": null, "DriverOpts": null, "QoSPriority": 0, "NetworkID": "b8c0e50ce5c95537d4c54667ca7ffd0dbd2093051b74e7fe4365dda26abf03d4", "EndpointID": "152861b0a9bb8f71f0ddc9d8458e9cf6d7499a59037b149537c0880f3c68929", "Gateway": "172.18.0.1", "IPAddress": "172.18.0.2", "MacAddress": "0a:95:da:21:8d:46", "IPPrefixLen": 16, "IPV6Gateway": "", "GlobalIPv6Address": "", "GlobalIPv6PrefixLen": 0, "DNSNames": ["web1", "b42605eb17cc"]}}, "ImageManifestDescriptor": {"MediaType": "application/vnd.oci.image.manifest.v1+json", "Digest": "sha256:d17906bc35fb0527e64cf6410704e94cb9cb4657ce690a0107cd5b56943ff9eb0", "Size": 2093, "Annotations": {"com.docker.official-images.bashbrew.arch": "amd64", "org.opencontainers.image.base.digest": "sha256:ef514b33e858a6ddd5a2af2b50f08f7ff2e43726a14d5f53bdb1d75220dfa5fb", "org.opencontainers.image.base.name": "debian:trixie-slim", "org.opencontainers.image.created": "2026-01-13T01:20:52Z", "org.opencontainers.image.revision": "b8bf24dec3fb94efd3d81ac495bea8247d5115d9", "org.opencontainers.image.source": "https://github.com/docker-library/httpd.git#b8bf24dec3fb94efd3d81ac495bea8247d5115d9:2.4", "org.opencontainers.image.url": "https://hub.docker.com/_/httpd", "org.opencontainers.image.version": "2.4.66"}, "Platform": {"Architecture": "amd64", "OS": "linux"}}, "RootFs": {"Type": "oci", "Layers": [{"blob": "sha256:ef514b33e858a6ddd5a2af2b50f08f7ff2e43726a14d5f53bdb1d75220dfa5fb"}]}
```

Ahora hemos creado otra imagen llamada web2 para ver a qué ip se conecta pero sin especificar en qué red se conecta

```
        "StopSignal": "SIGWINCH"
    },
    "NetworkSettings": {
        "sandboxID": "348094d9f83c76dc79c70f6235b4761a94fffad507a2824034ae7023b2b25d74",
        "sandboxKey": "/var/run/docker/netns/348d94d9f83c",
        "Ports": {
            "80/tcp": null
        }
    },
    "Networks": {
        "bridge": {
            "IPAMConfig": null,
            "Links": null,
            "Aliases": null,
            "DriverOpts": null,
            "GuPriority": 0,
            "NetworkID": "32bb4d8cae2352ceea80dd5461b7b0ba64f4f0e555478056eaa91c1463b8b12d",
            "EndpointID": "df1ba32fc1c6feafedc5d0a51d184d9",
            "Gateway": "172.17.0.1",
            "IPAddress": "172.17.0.3",
            "MacAddress": "6e:da:7d:60:5c:25",
            "IPPrefixLen": 16,
            "IPv6Gateway": "",
            "GlobalIPv6Address": "",
            "GlobalIPv6PrefixLen": 0,
            "DNSNames": null
        }
    }
},
"ImageManifestDescriptor": {
    "mediaType": "application/vnd.oci.image.manifest.v1+json",
    "digest": "sha256:d17906bc39fb027ee64cf6410704e94cb9cb4657ce690a107cd5b56943ff9eb0",
    "size": 2093,
    "annotations": {
        "com.docker.official-images.bashbrew.arch": "amd64",
        "org.opencontainers.image.base.digest": "sha256:e7f514b33e058a6ddd5a2af2b50f08f7ff2e43726a14d5f53bdb1d75220dfa5fb",
        "org.opencontainers.image.name": "debian:trixie-slim",
        "org.opencontainers.image.created": "2026-01-13T01:20:52Z",
        "org.opencontainers.image.revision": "b0bf24dec3fb94efdd01ac495bea0247d5115d9",
        "org.opencontainers.image.source": "https://github.com/docker-library/httpd.git#b0bf24dec3fb94efd3d01ac495bea0247d5115d9:2.4",
        "org.opencontainers.image.url": "https://hub.docker.com/_/httpd",
        "org.opencontainers.image.version": "2.4.66"
    }
},
"platform": {
    "architecture": "amd64",
    "os": "linux"
}
]
root@UbuntuServ:/home/vboxuser#
```



## PASO 6:

Ahora vamos a abrir sesión en web1 con docker exec -ti web1 /bin/bash

```
]  
root@UbuntuServ:/home/vboxuser# docker exec -ti web1 /bin/bash  
root@b42605eb17cc:/usr/local/apache2#
```

Ahora hacemos ping

```
]  
root@UbuntuServ:/home/vboxuser# docker exec -ti web1 /bin/bash  
root@b42605eb17cc:/usr/local/apache2# ping  
bash: ping: command not found  
root@b42605eb17cc:/usr/local/apache2#
```

Nos da error ya que no tenemos el comando ping l, o tenemos que instalar con apt-get update && apt-get install iputils-ping

```
root@UbuntuServ:/home/vboxuser# docker exec -ti web1 /bin/bash  
root@b42605eb17cc:/usr/local/apache2# ping  
bash: ping: command not found  
root@b42605eb17cc:/usr/local/apache2# apt-get update && apt-get install iputils-ping  
Hit:1 http://deb.debian.org/debian trixie InRelease  
Get:2 http://deb.debian.org/debian trixie-updates InRelease [47.3 kB]  
Get:3 http://deb.debian.org/debian-security trixie-security InRelease [43.4 kB]  
Get:4 http://deb.debian.org/debian trixie/main amd64 Packages [9670 kB]
```

## PASO 7:

Para que se puedan ver ambos containers, hay que conectar sus redes. Para ello vamos a conectar web2 a miRedLocal con el comando docker network connect miRedLocal web2.

```
Setting up iputils-ping (3:20240905-3) ...
root@b42605eb17cc:/usr/local/apache2# exit
exit
root@UbuntuServ:/home/vboxuser# docker network connect miRedLocal web2
root@UbuntuServ:/home/vboxuser#
```

Ahora con docker inspect web2 probaremos que es accesible

```
{
    "GlobalIPv6PrefixLen": 0,
    "DNSNames": null
},
"miRedLocal": {
    "IPAMConfig": {
        "IPv4Address": "",
        "IPv6Address": ""
    },
    "Links": null,
    "Aliases": [],
    "DriverOpts": {},
    "GuPriority": 0,
    "NetworkID": "b8c0e50ce5c95537d4c54667ca7ff0dbd2093051b74e7fe4365dda26abf03d4",
    "EndpointID": "14fe56b35f1368835d363e2ac42664374af75488e4fac25f0b5c6320527194ea",
    "Gateway": "172.18.0.1",
    "IPAddress": "172.18.0.3",
    "MacAddress": "7aa:e0:7e:b0:a7:13",
    "IPPrefixLen": 16,
    "IPv6Gateway": "",
    "GlobalIPv6Address": "",
    "GlobalIPv6PrefixLen": 0,
    "DNSNames": [
        "web2",
        "3440e03921e8"
    ]
},
"ImageManifestDescriptor": {
    "mediaType": "application/vnd.oci.image.manifest.v1+json",
    "digest": "sha256:d17906bc35fb0527e64cf6410704e94cb9cb4657ce690a107cd5b56943ff9eb0",
    "size": 2093,
    "annotations": {
        "com.docker.official-images.bashbrew.arch": "amd64",
        "org.opencontainers.image.base.digest": "sha256:ef514b33e858a6ddd5a2af2b50f08f7ff2e43726a14d5f53bdb1d75220dfa5fb",
        "org.opencontainers.image.base.name": "debian:trixie-slim",
        "org.opencontainers.image.created": "2026-01-13T01:20:52Z",
        "org.opencontainers.image.revision": "b8bf24dec3fb94efd3d81ac495bea8247d5115d9",
        "org.opencontainers.image.source": "https://github.com/docker-library/httpd.git#b8bf24dec3fb94efd3d81ac495bea8247d5115d9:2.4",
        "org.opencontainers.image.url": "https://hub.docker.com/_/httpd",
        "org.opencontainers.image.version": "2.4.66"
    },
    "platform": {
        "architecture": "amd64",
        "os": "linux"
    }
}
}
```

Ahora lo comprobaremos haciendo un ping a la ip

```
] 
root@UbuntuServ:/home/vboxuser# ping 172.17.0.3
PING 172.17.0.3 (172.17.0.3) 56(84) bytes of data.
64 bytes from 172.17.0.3: icmp_seq=1 ttl=64 time=0.179 ms
64 bytes from 172.17.0.3: icmp_seq=2 ttl=64 time=0.058 ms
64 bytes from 172.17.0.3: icmp_seq=3 ttl=64 time=0.070 ms
64 bytes from 172.17.0.3: icmp_seq=4 ttl=64 time=0.314 ms
64 bytes from 172.17.0.3: icmp_seq=5 ttl=64 time=0.048 ms
64 bytes from 172.17.0.3: icmp_seq=6 ttl=64 time=0.098 ms
64 bytes from 172.17.0.3: icmp_seq=7 ttl=64 time=0.042 ms
64 bytes from 172.17.0.3: icmp_seq=8 ttl=64 time=0.083 ms
```