

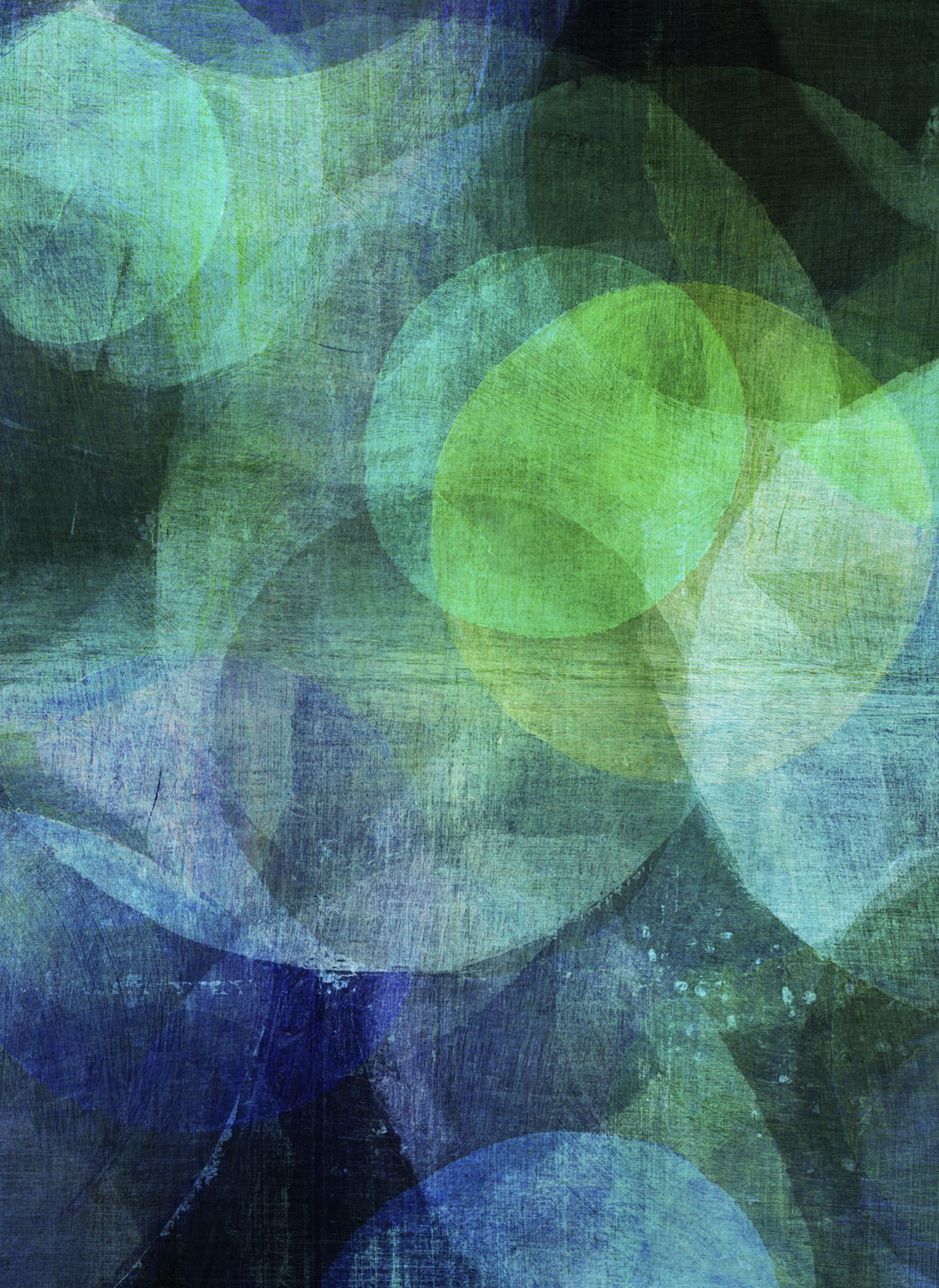
INTRODUCTION TO DOCKER



INTRODUCTION TO DOCKER

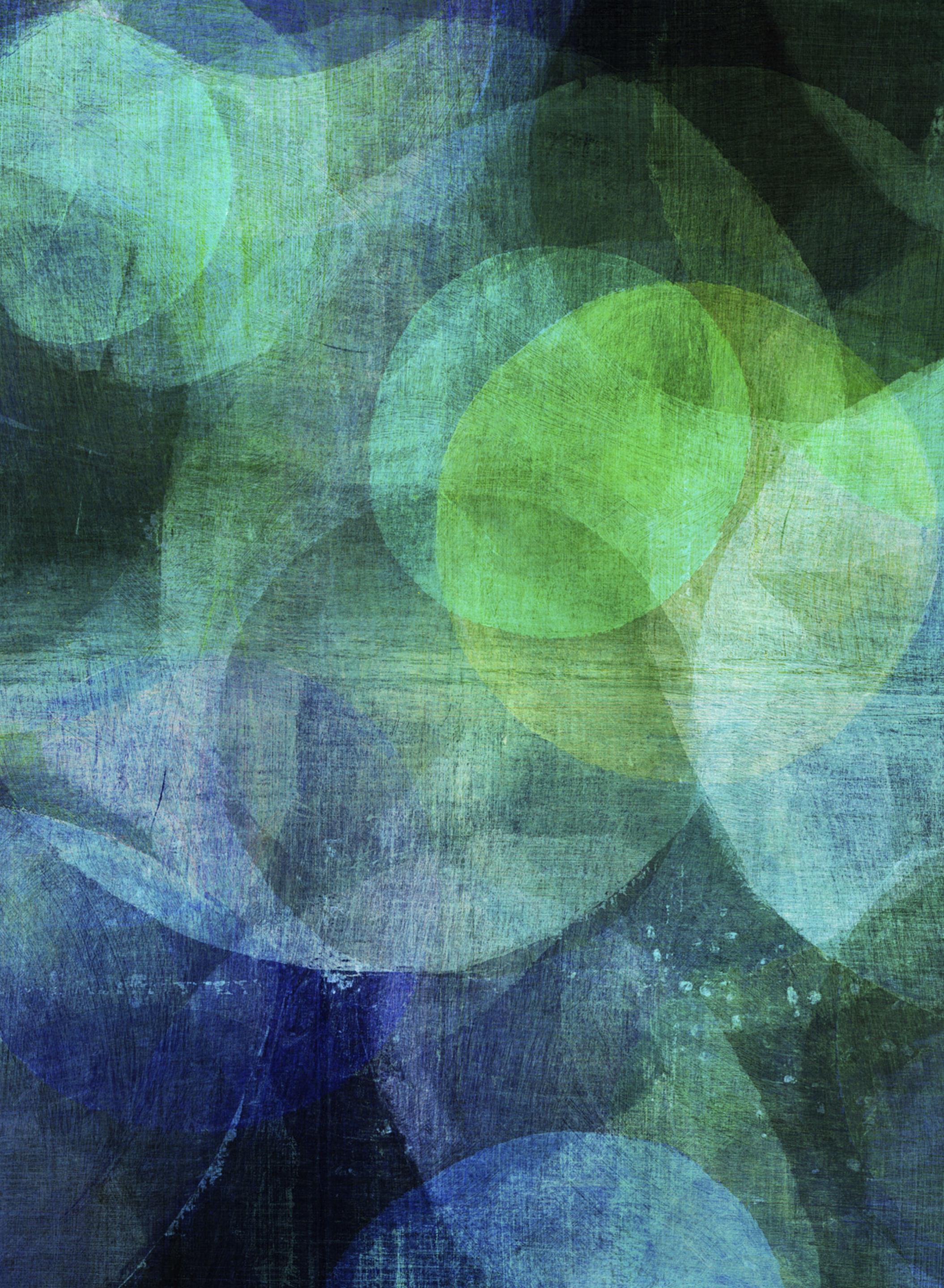
Docker Network and Volumes

AGENDA



AGENDA

- Basic of Container Networking



AGENDA

- Basic of Container Networking
 - Docker Network



AGENDA

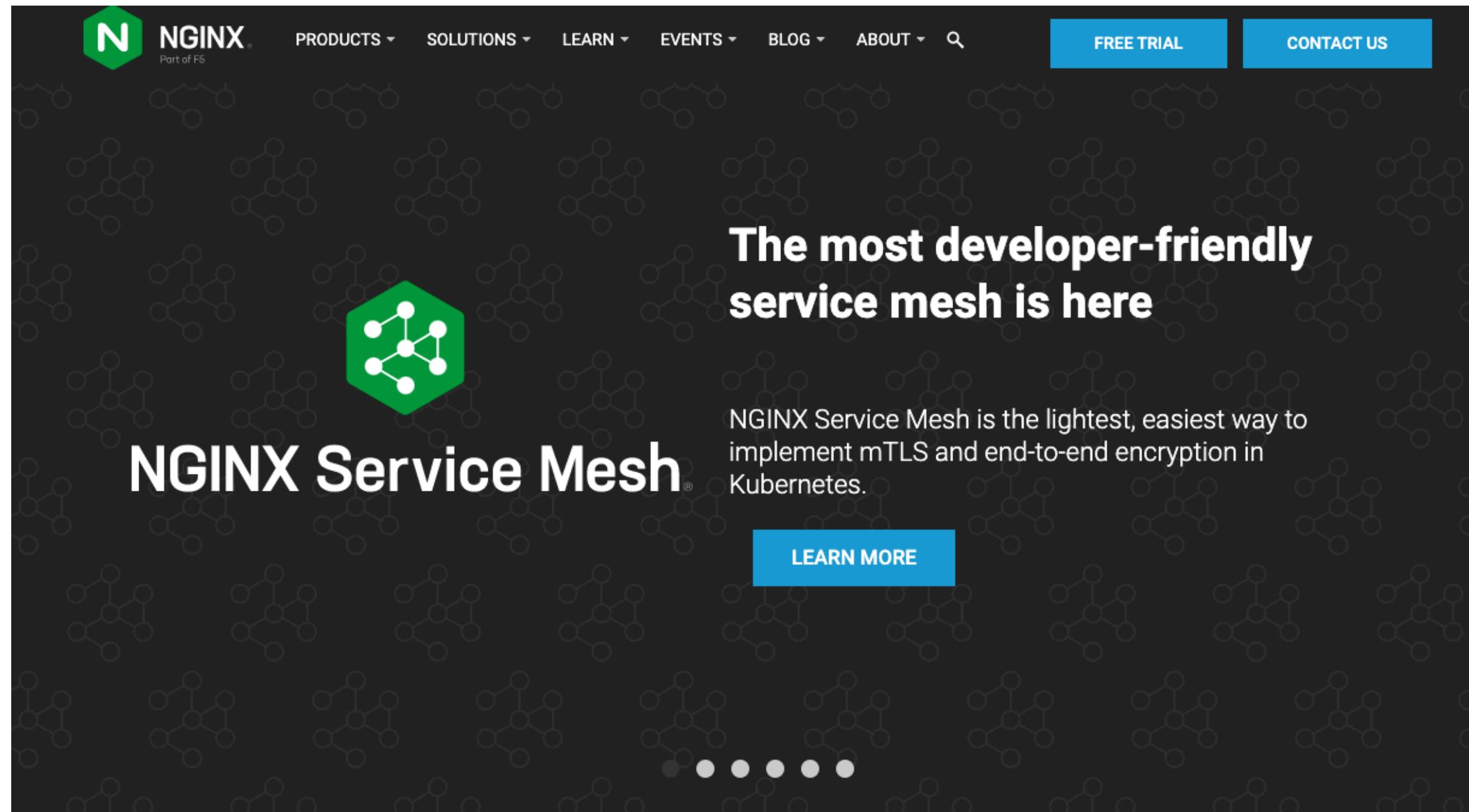
- Basic of Container Networking
 - Docker Network
 - Docker Storage (Volumes)



BASIC OF CONTAINER NETWORKING

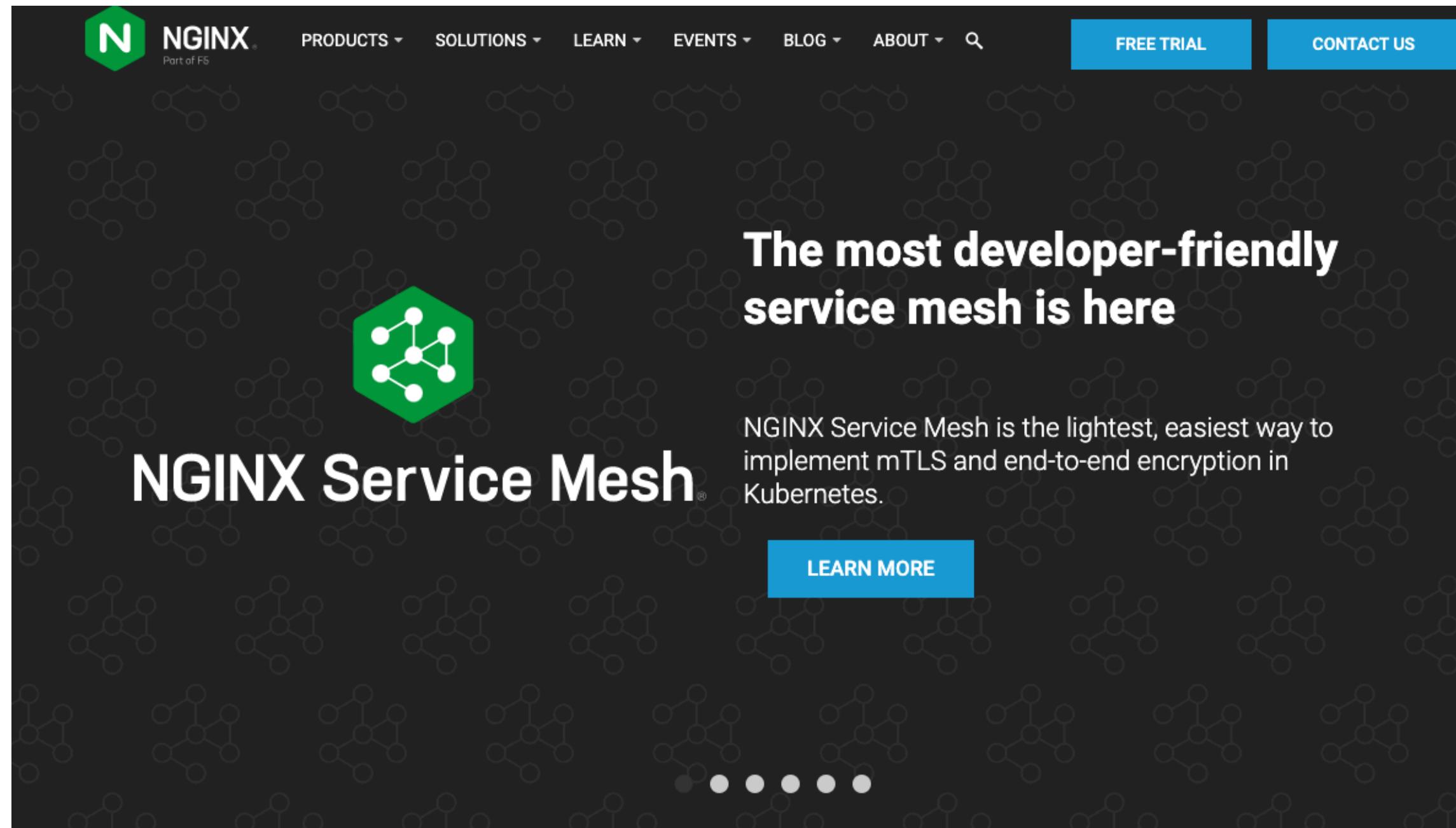
START WEB APPLICATION WITH NGINX

START WEB APPLICATION WITH NGINX

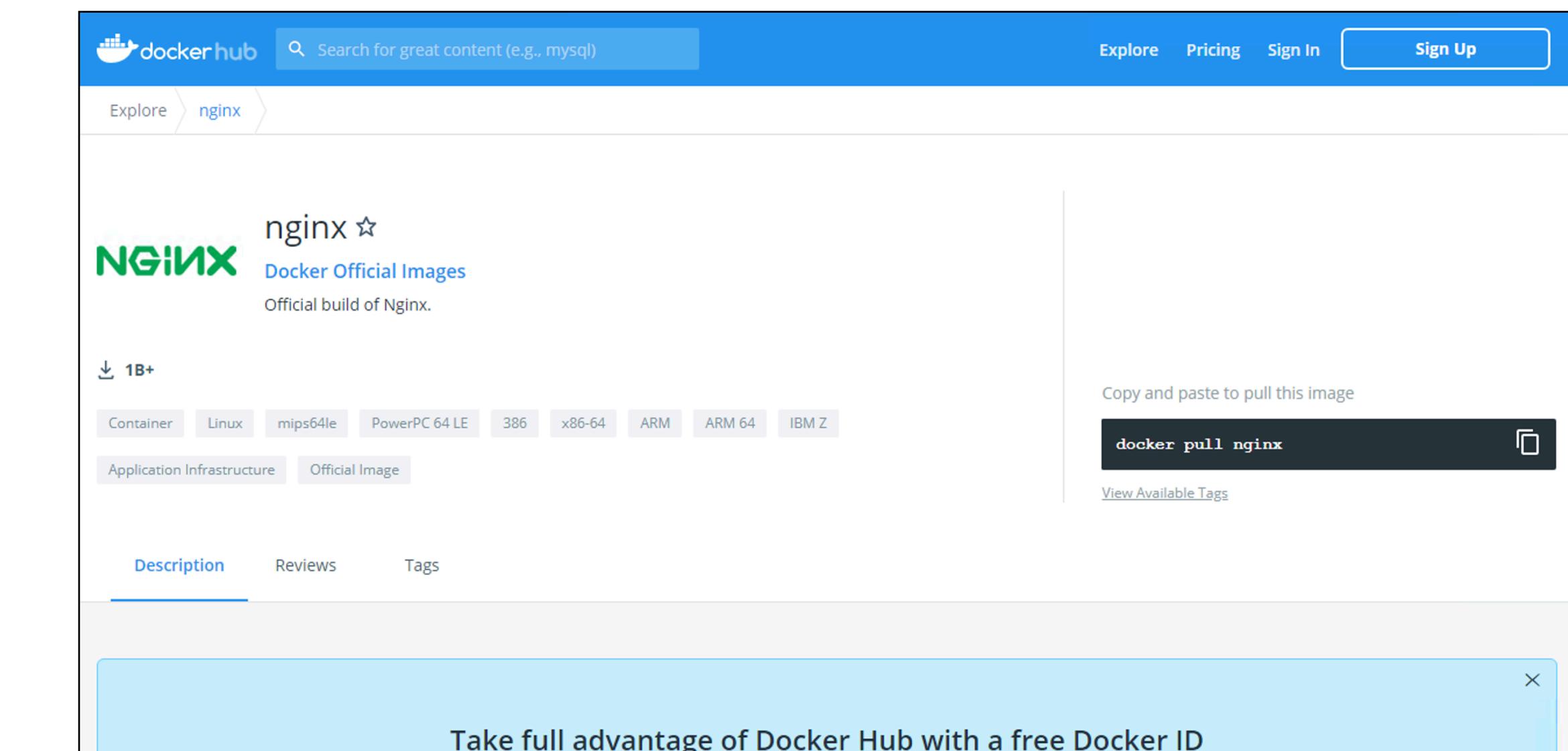


The image shows a screenshot of the NGINX website's landing page for their Service Mesh product. The background features a dark, abstract pattern of interconnected nodes and lines, resembling a network or a molecular structure. In the upper left corner, the NGINX logo is displayed, followed by the text "Part of F5". The top navigation bar includes links for PRODUCTS, SOLUTIONS, LEARN, EVENTS, BLOG, ABOUT, and a search icon. Two prominent blue buttons are located in the top right: "FREE TRIAL" and "CONTACT US". The central headline reads "The most developer-friendly service mesh is here", accompanied by a green hexagonal icon containing a white network graph. Below this, a sub-headline states: "NGINX Service Mesh is the lightest, easiest way to implement mTLS and end-to-end encryption in Kubernetes." A "LEARN MORE" button is positioned at the bottom of this section. At the very bottom of the page, there is a horizontal navigation bar with several small, unlabeled dots.

START WEB APPLICATION WITH NGINX

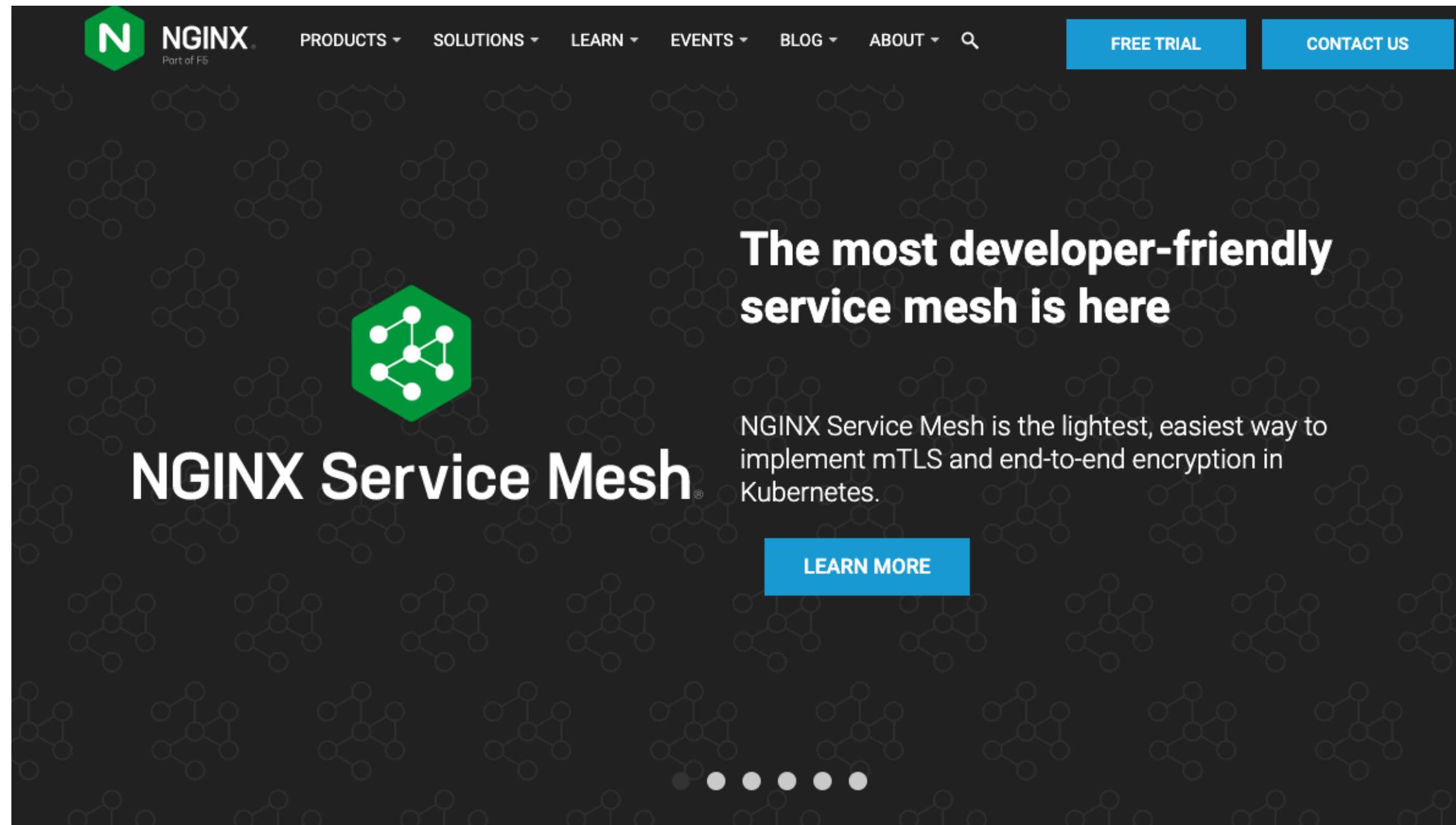


The landing page for NGINX Service Mesh. It features a dark background with a light gray molecular or network-like pattern. In the center, there's a green hexagonal icon containing a white network graph. To the left of the icon, the text "NGINX Service Mesh" is displayed. Above the icon, the text "The most developer-friendly service mesh is here" is written in bold white font. Below the icon, a description reads: "NGINX Service Mesh is the lightest, easiest way to implement mTLS and end-to-end encryption in Kubernetes." A blue "LEARN MORE" button is located at the bottom right of the central area.

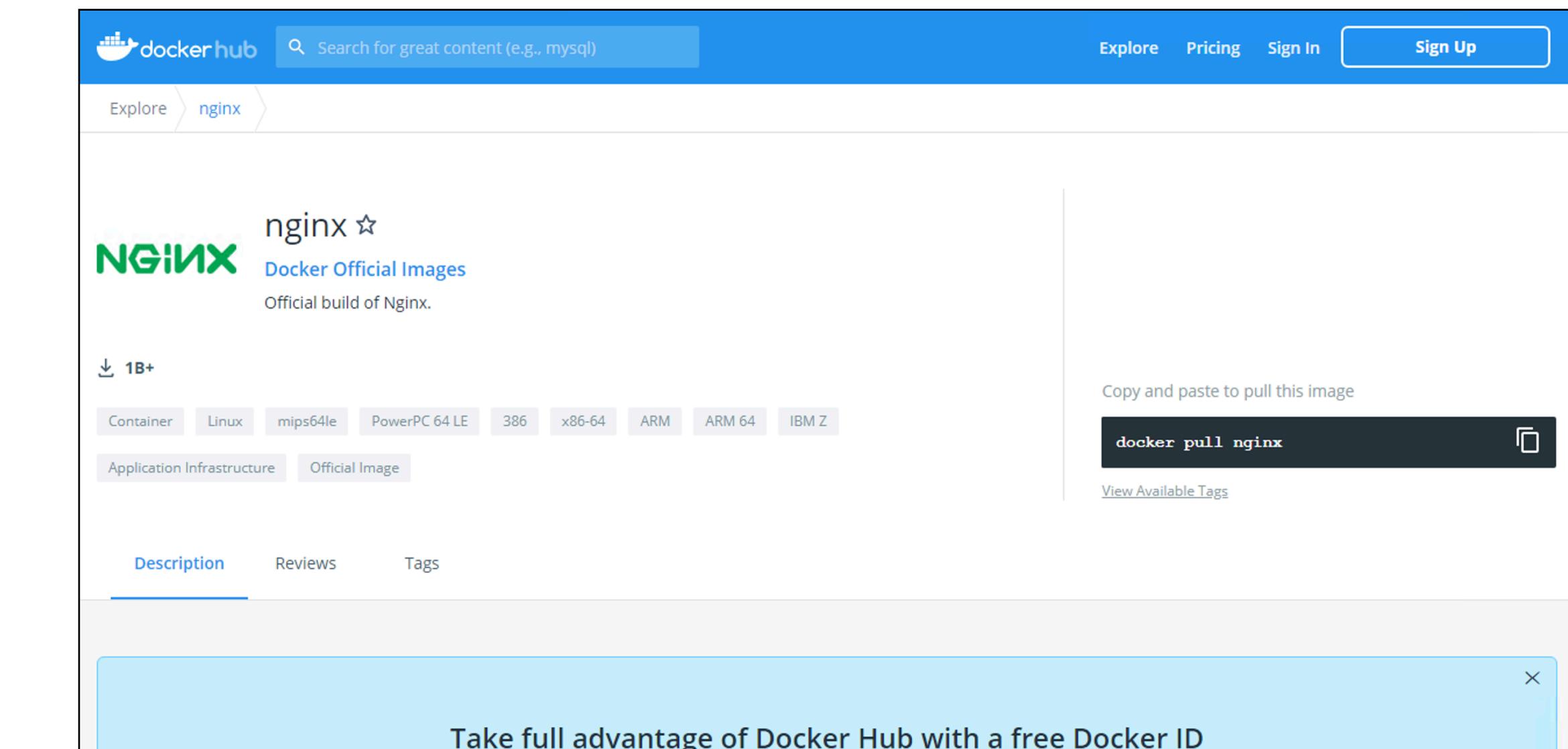


The Docker Hub page for the "nginx" repository. The header includes the Docker Hub logo, a search bar, and navigation links for "Explore", "Pricing", "Sign In", and "Sign Up". The main content shows the "nginx" star icon, the "Docker Official Images" badge, and the text "Official build of Nginx.". It displays "1B+" downloads and a list of supported architectures: Container, Linux, mips64le, PowerPC 64 LE, 386, x86-64, ARM, ARM 64, and IBM Z. Below this, there are tabs for "Description", "Reviews", and "Tags". A prominent blue call-to-action button at the bottom says "Take full advantage of Docker Hub with a free Docker ID".

START WEB APPLICATION WITH NGINX



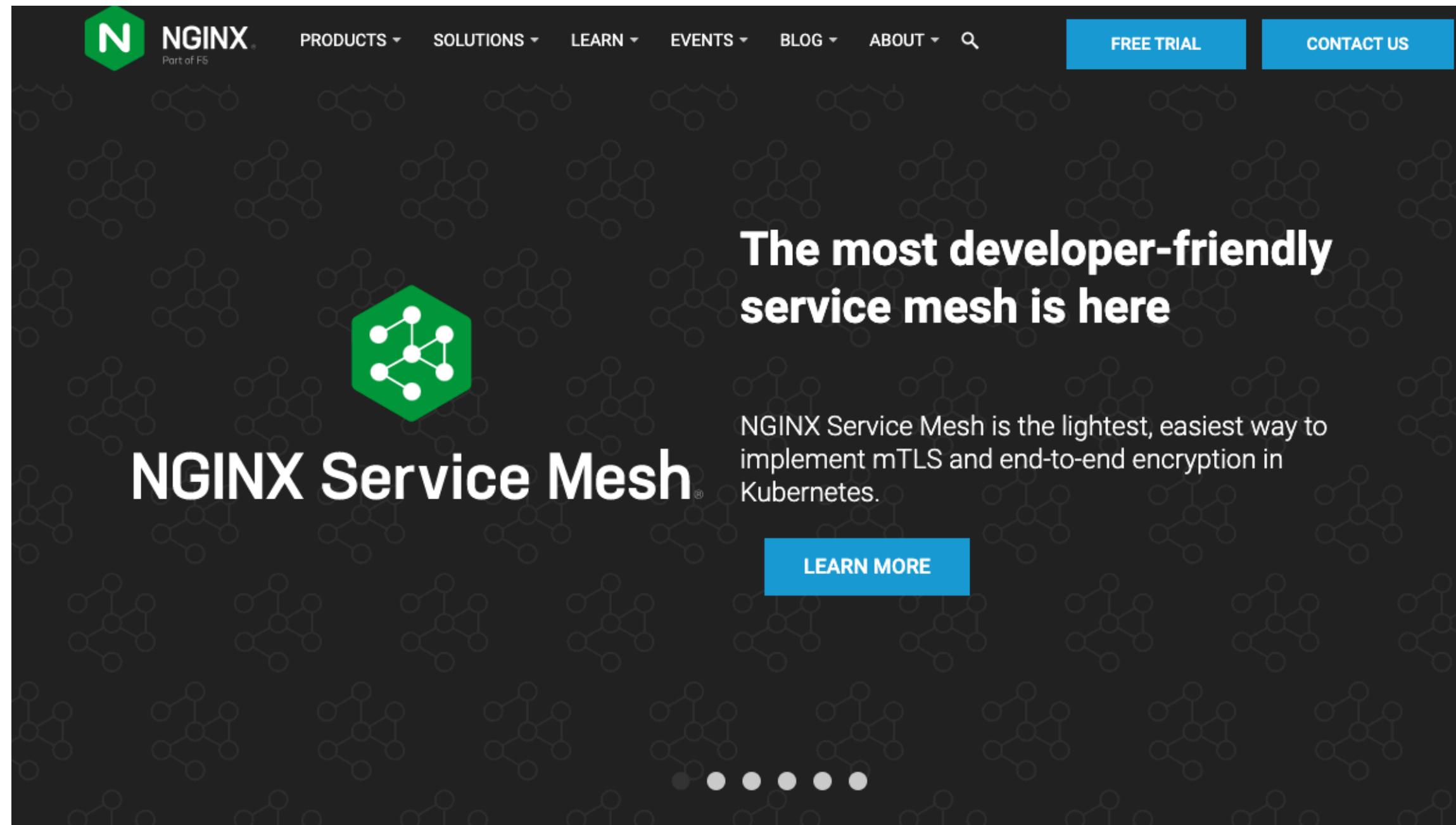
The landing page for NGINX Service Mesh. It features a dark background with a molecular or network-like pattern. The NGINX logo is at the top left. A green hexagonal icon with a network graph inside is positioned on the left. The main headline reads "The most developer-friendly service mesh is here". Below it, a sub-headline says "NGINX Service Mesh is the lightest, easiest way to implement mTLS and end-to-end encryption in Kubernetes." A blue "LEARN MORE" button is at the bottom right.



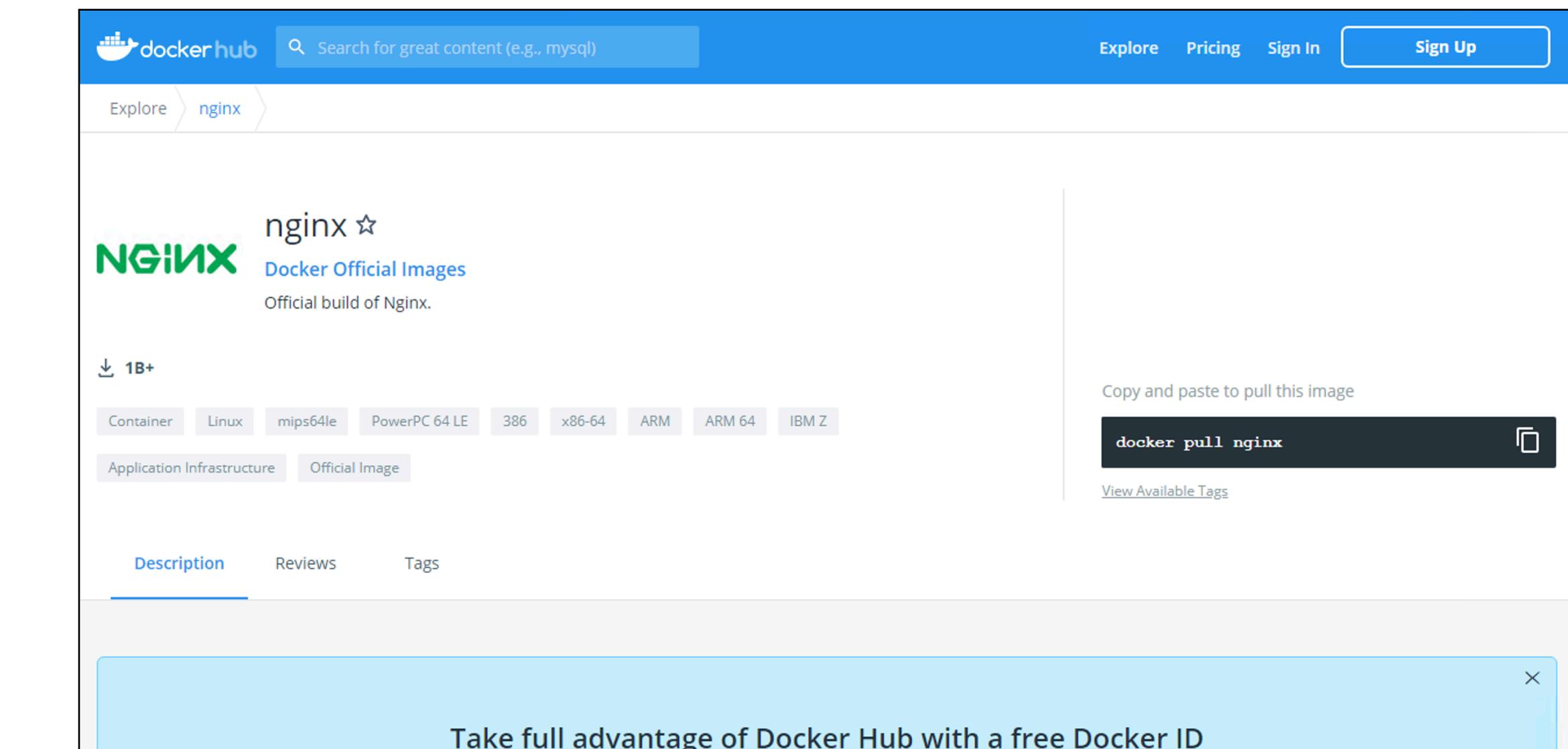
The Docker Hub page for the official NGINX Docker image. The header shows "Explore > nginx". The main card for "nginx" shows it has over 1B+ downloads. It's categorized under "Container", "Linux", "mips64le", "PowerPC 64 LE", "386", "x86-64", "ARM", "ARM 64", and "IBM Z". It's labeled as an "Official Image". A prominent button at the bottom right says "docker pull nginx". A call-to-action banner at the bottom encourages users to "Take full advantage of Docker Hub with a free Docker ID".

```
$ docker container run --name hello-nginx -d nginx
```

START WEB APPLICATION WITH NGINX



The landing page for NGINX Service Mesh. It features a dark background with a molecular or network-like pattern. At the top, there's a navigation bar with links for PRODUCTS, SOLUTIONS, LEARN, EVENTS, BLOG, ABOUT, and a search icon. Below the navigation is a blue button for "FREE TRIAL" and another for "CONTACT US". The main headline reads "The most developer-friendly service mesh is here" in white text. To the left of the text is a green hexagonal icon containing a white network graph. Below the headline, there's a sub-headline: "NGINX Service Mesh is the lightest, easiest way to implement mTLS and end-to-end encryption in Kubernetes." A blue "LEARN MORE" button is located at the bottom right of the main text area.

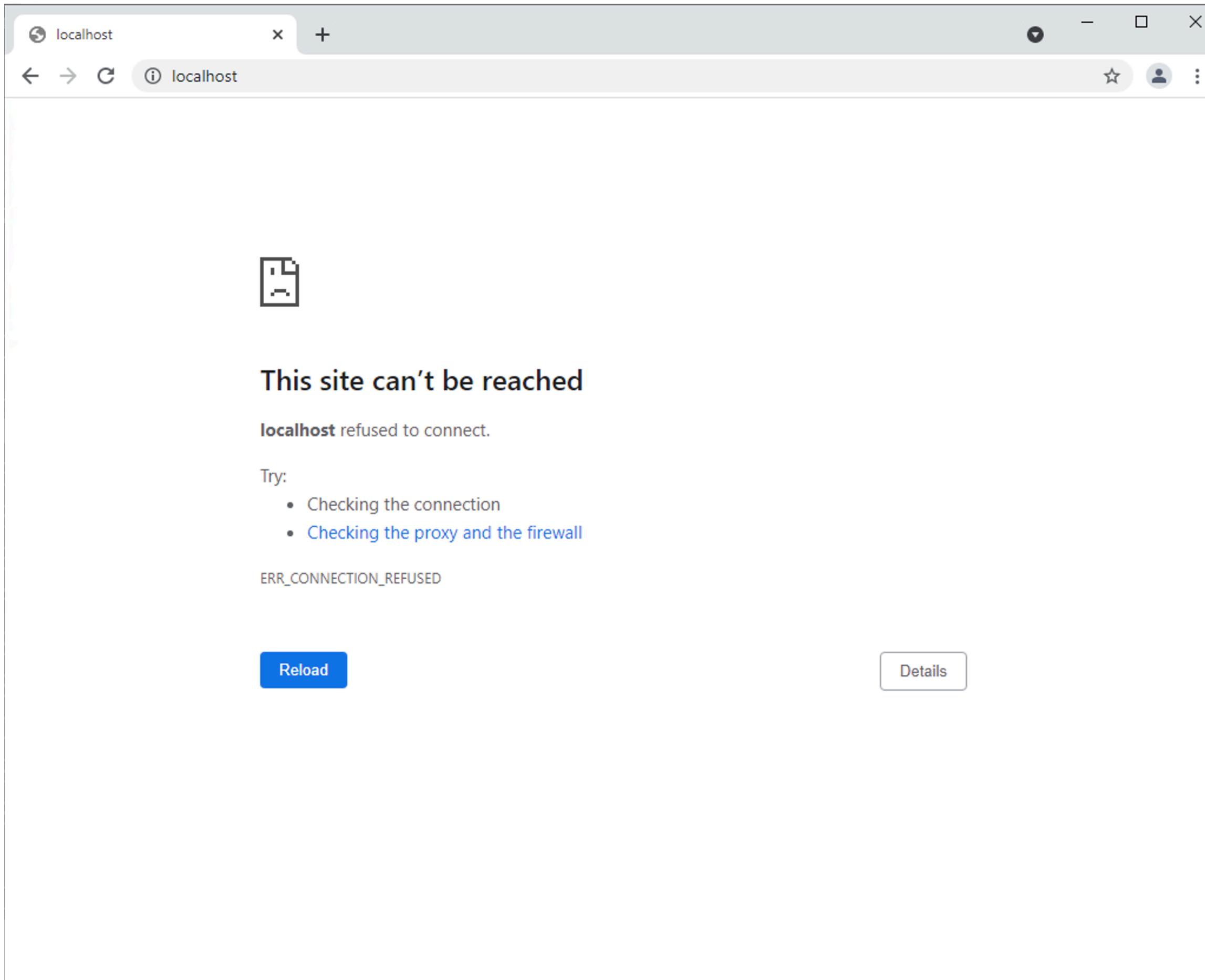


The Docker Hub page for the official NGINX Docker image. The header includes the Docker Hub logo, a search bar, and navigation links for Explore, Pricing, Sign In, and Sign Up. The main section shows the "nginx" repository by "NGINX". It highlights that it's a "Docker Official Images" build. Below that, it shows "1B+" downloads. There are tabs for Container, Linux, mips64le, PowerPC 64 LE, 386, x86-64, ARM, ARM 64, and IBM Z. Underneath, there are tabs for Application Infrastructure and Official Image. A "Description" tab is currently selected. On the right side, there's a "Copy and paste to pull this image" button with the command "docker pull nginx" and a "View Available Tags" link. A blue banner at the bottom encourages users to "Take full advantage of Docker Hub with a free Docker ID".

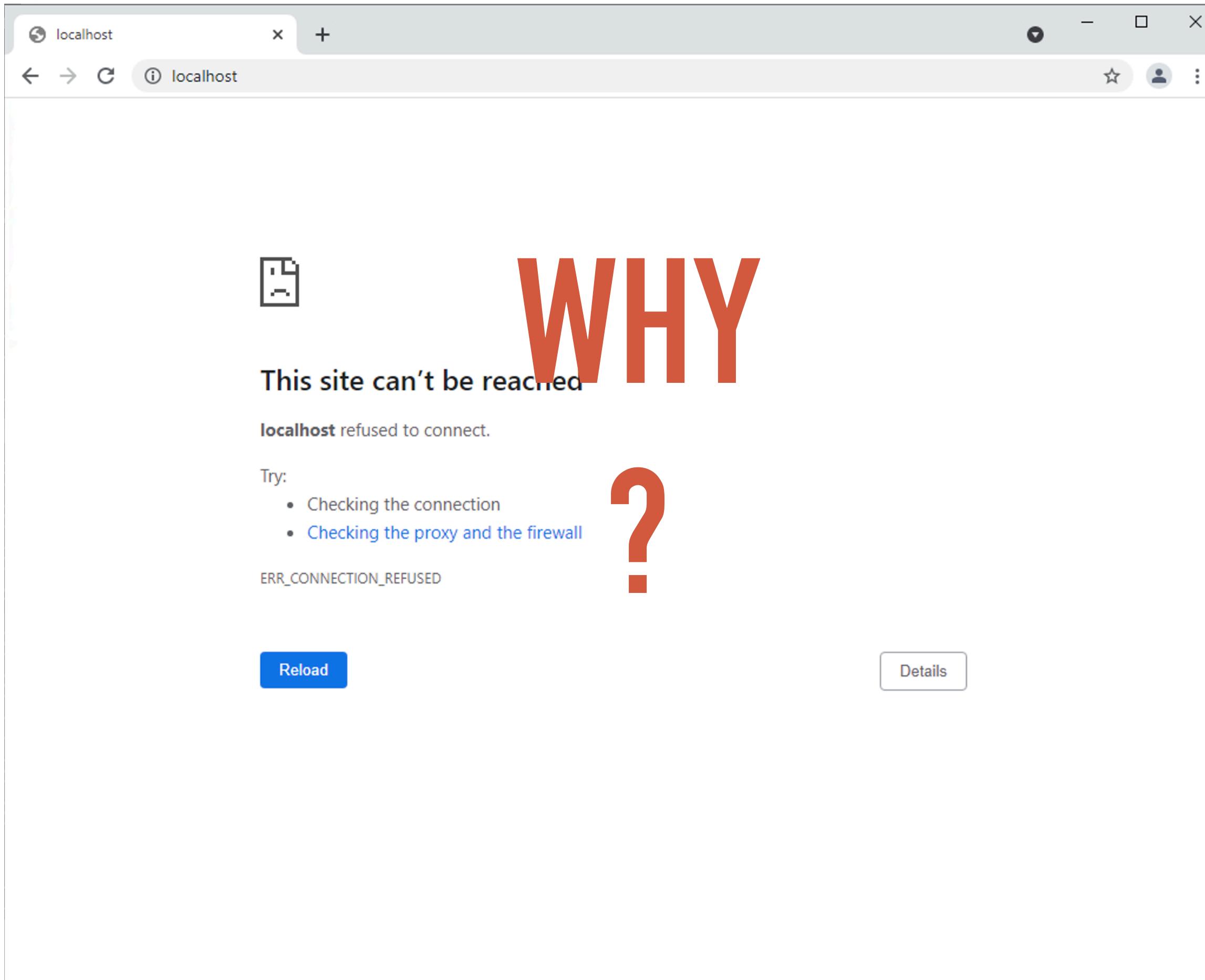
```
$ docker container run --name hello-nginx -d nginx
```

```
74dbf8abb5a5b8533c11c86e1949cf57b5104b2fa721237abfea696c6cc6289
```

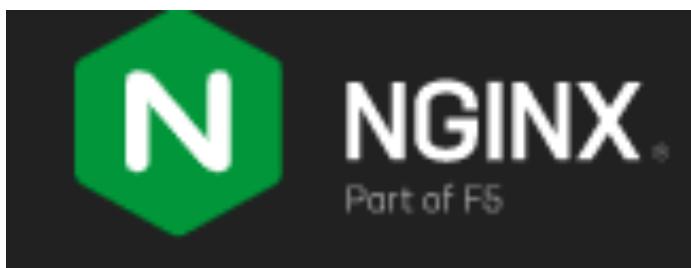
CAN'T CONNECT TO NGINX WITH PORT 80



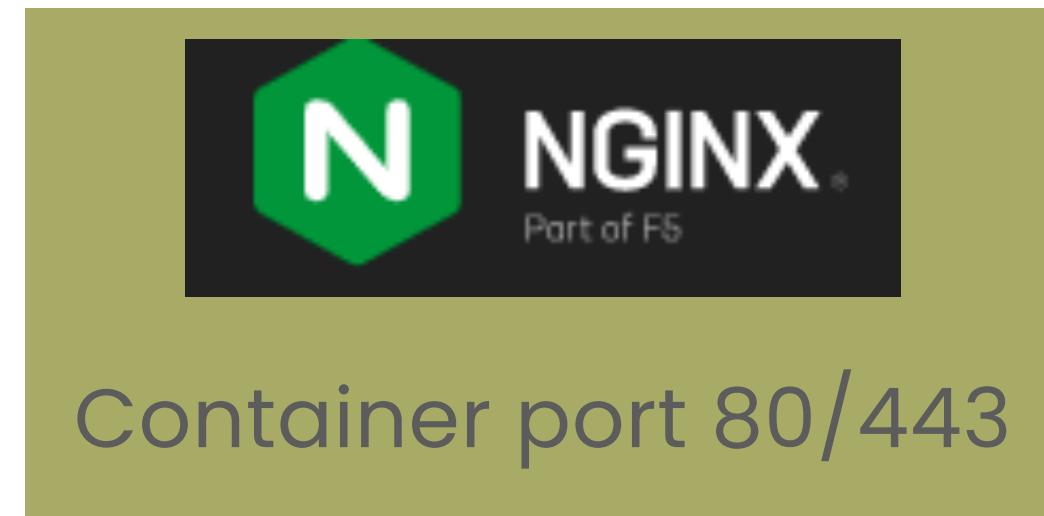
CAN'T CONNECT TO NGINX WITH PORT 80



PORT OF HOST AND CONTAINER



PORT OF HOST AND CONTAINER



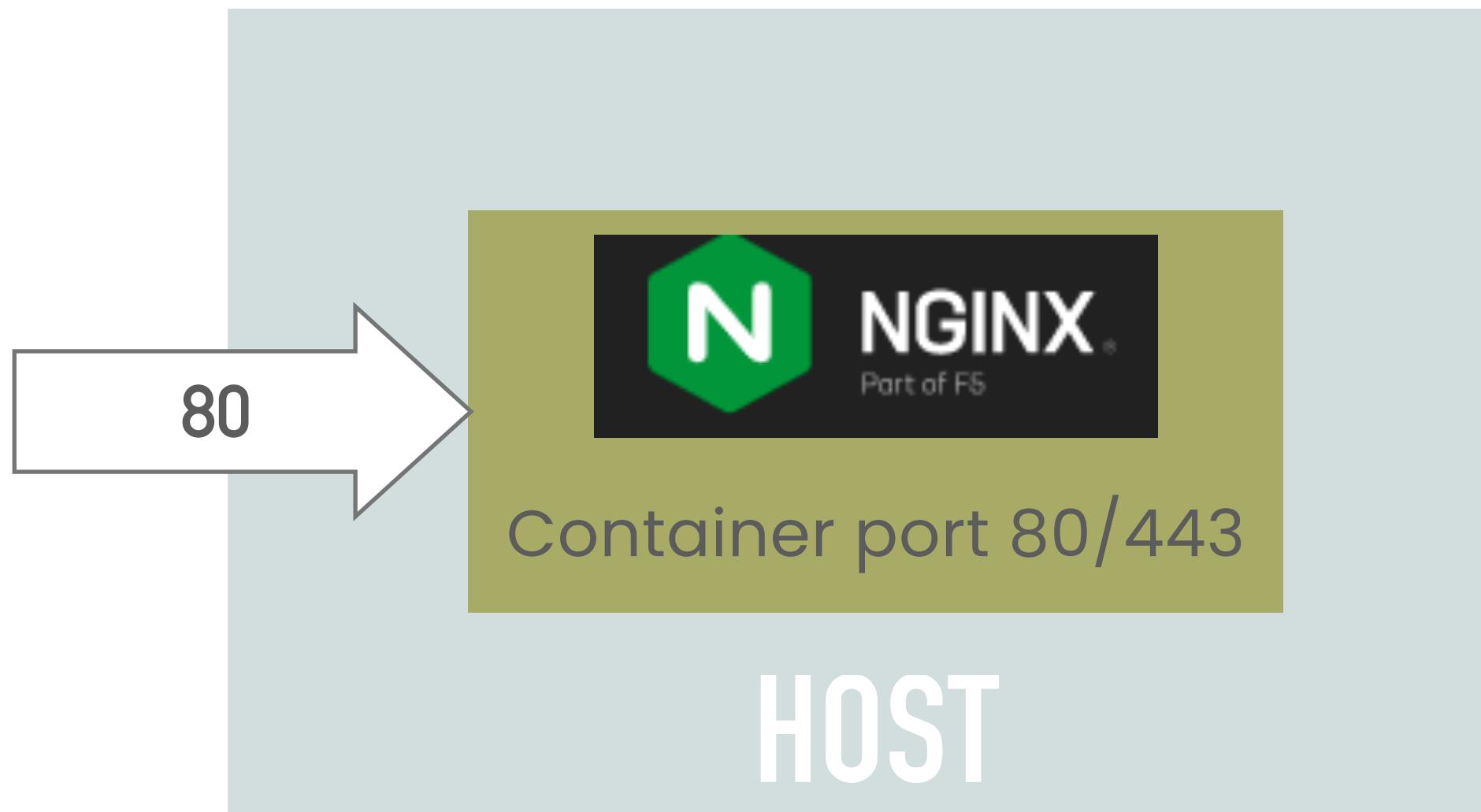
PORT OF HOST AND CONTAINER



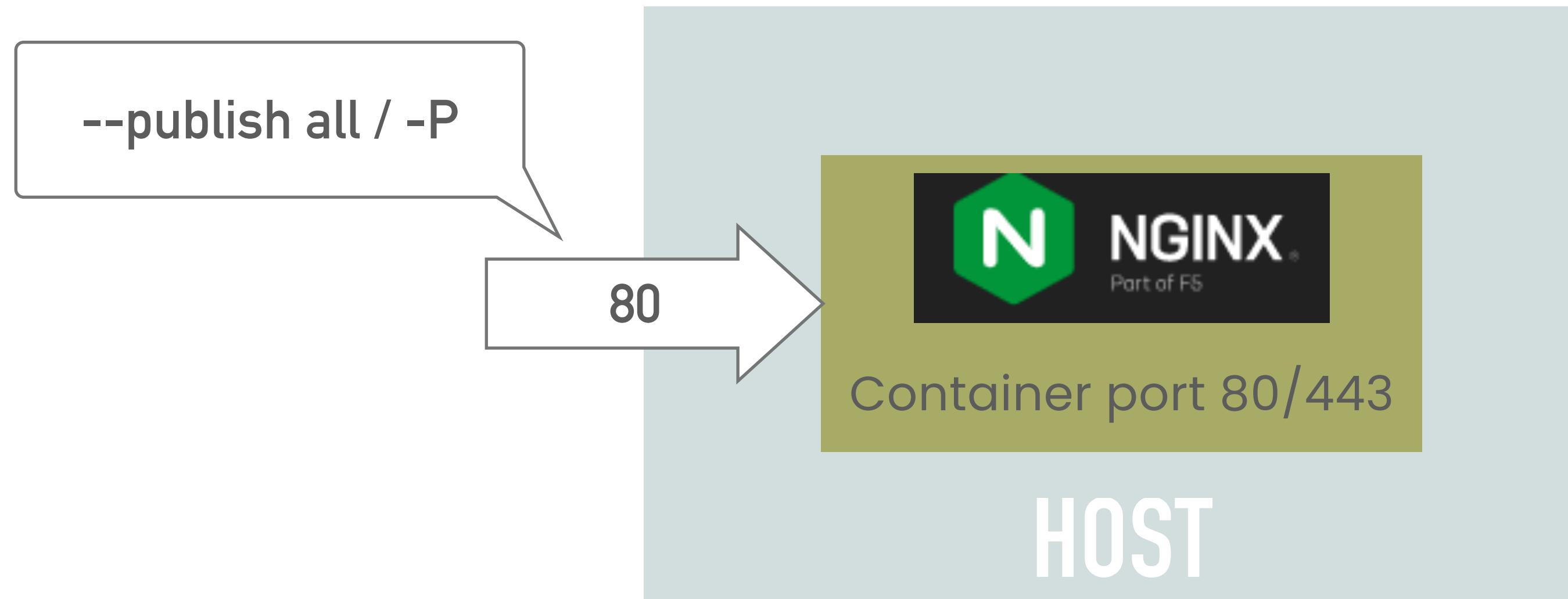
PORT OF HOST AND CONTAINER



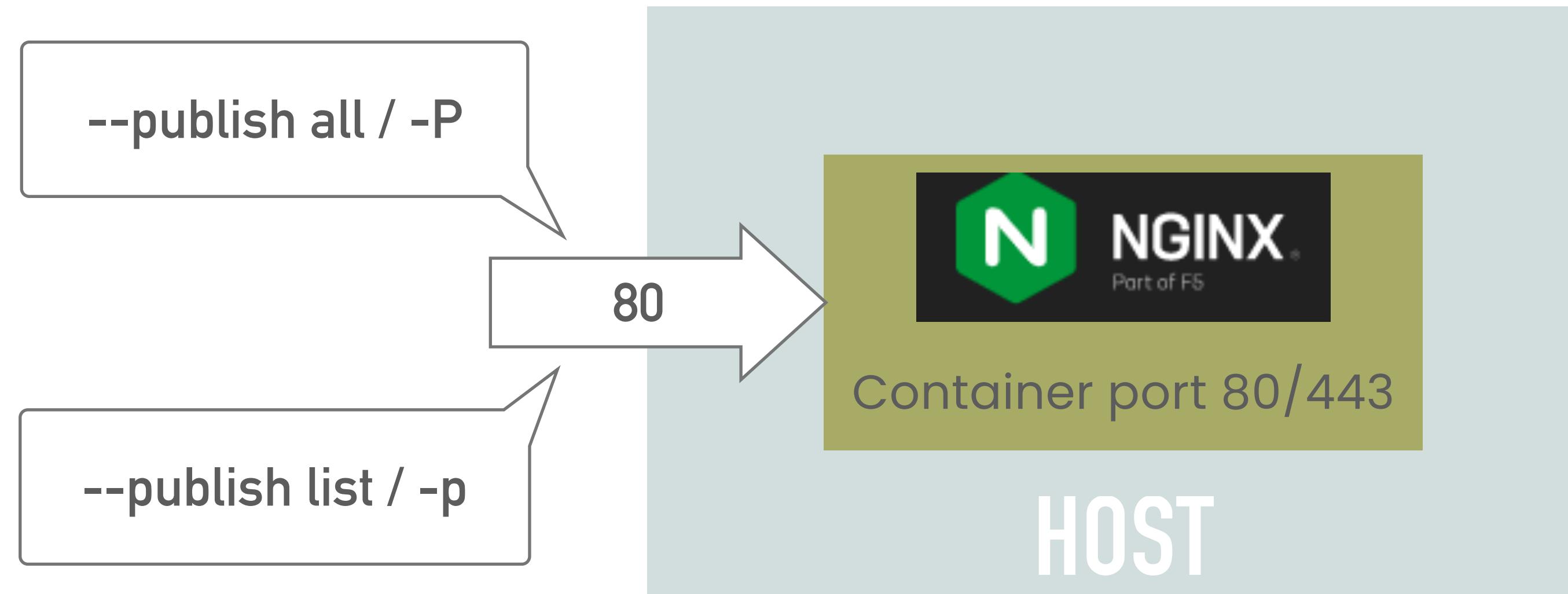
PORT OF HOST AND CONTAINER



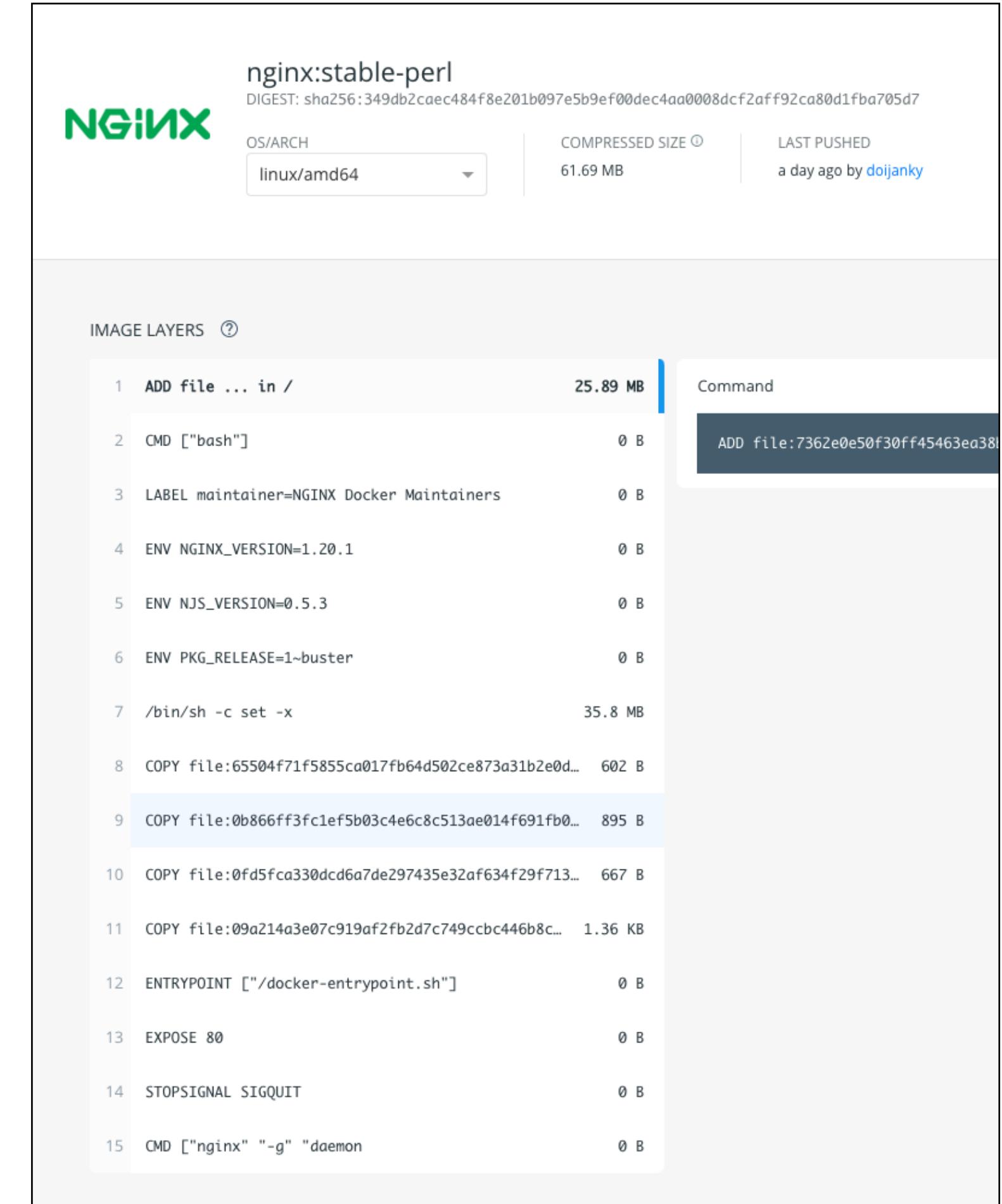
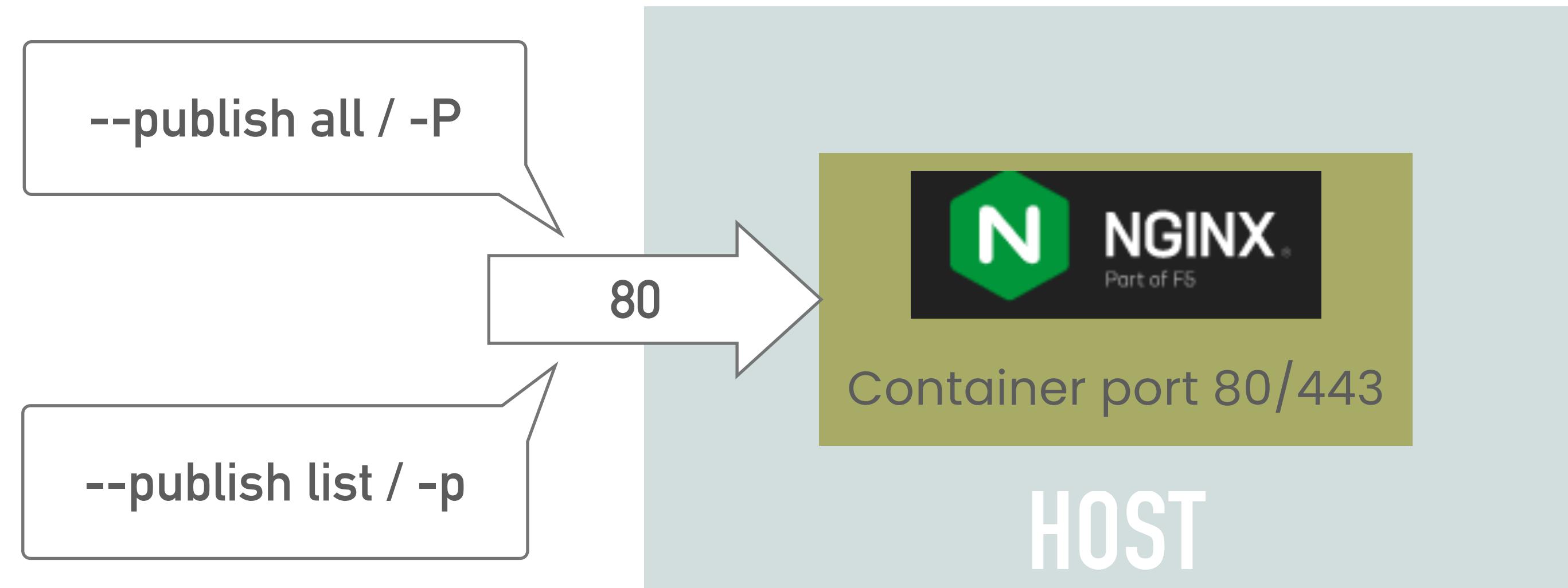
PORT OF HOST AND CONTAINER



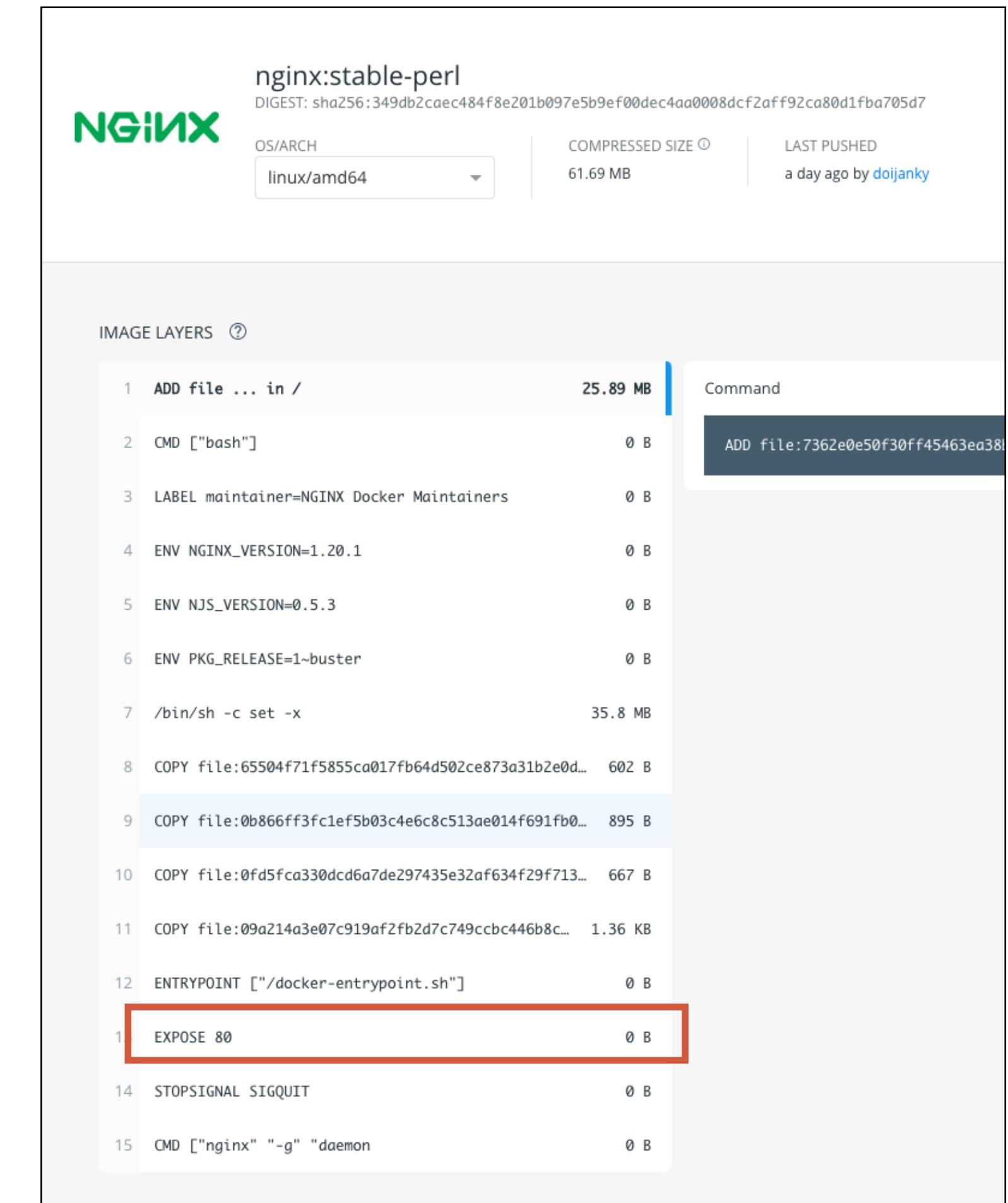
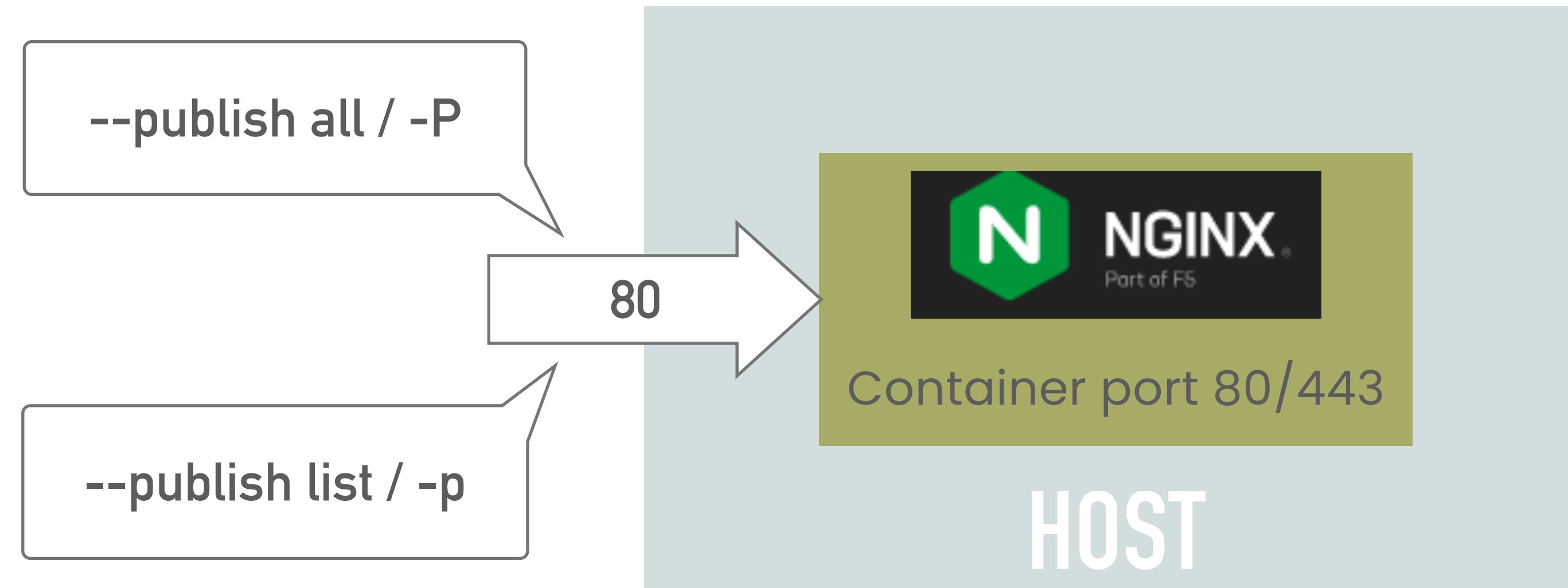
PORT OF HOST AND CONTAINER



PORT OF HOST AND CONTAINER



PORT OF HOST AND CONTAINER



START WEB APPLICATION WITH NGINX

START WEB APPLICATION WITH NGINX

```
$ docker container run --name hello-nginx -d -p 80:80 nginx
```

START WEB APPLICATION WITH NGINX

```
$ docker container run --name hello-nginx -d -p 80:80 nginx
```

OR

START WEB APPLICATION WITH NGINX

```
$ docker container run --name hello-nginx -d -p 80:80 nginx
```

OR

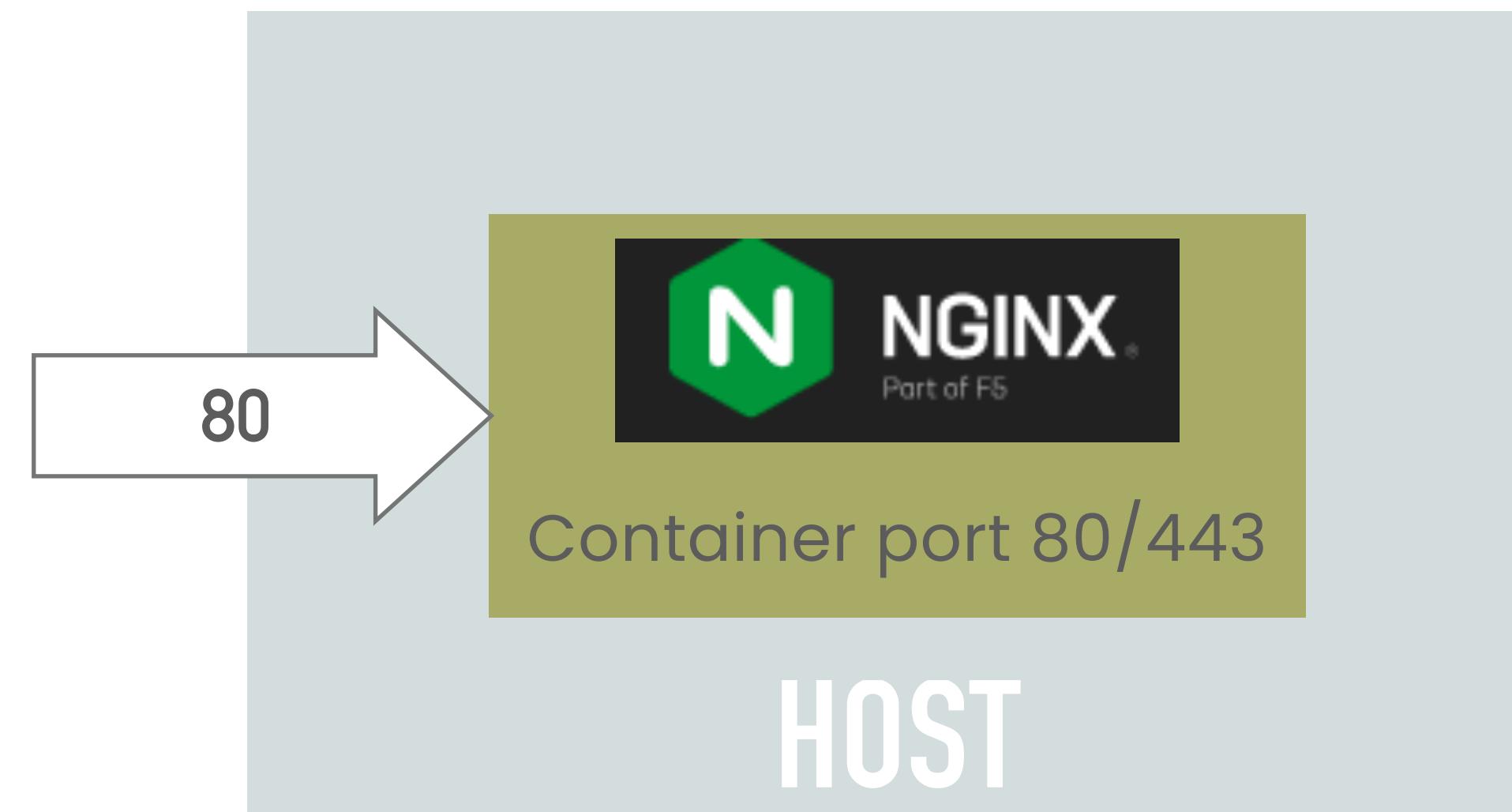
```
$ docker container run --name hello-nginx -d --publish 80:80 nginx
```

START WEB APPLICATION WITH NGINX

```
$ docker container run --name hello-nginx -d -p 80:80 nginx
```

OR

```
$ docker container run --name hello-nginx -d --publish 80:80 nginx
```

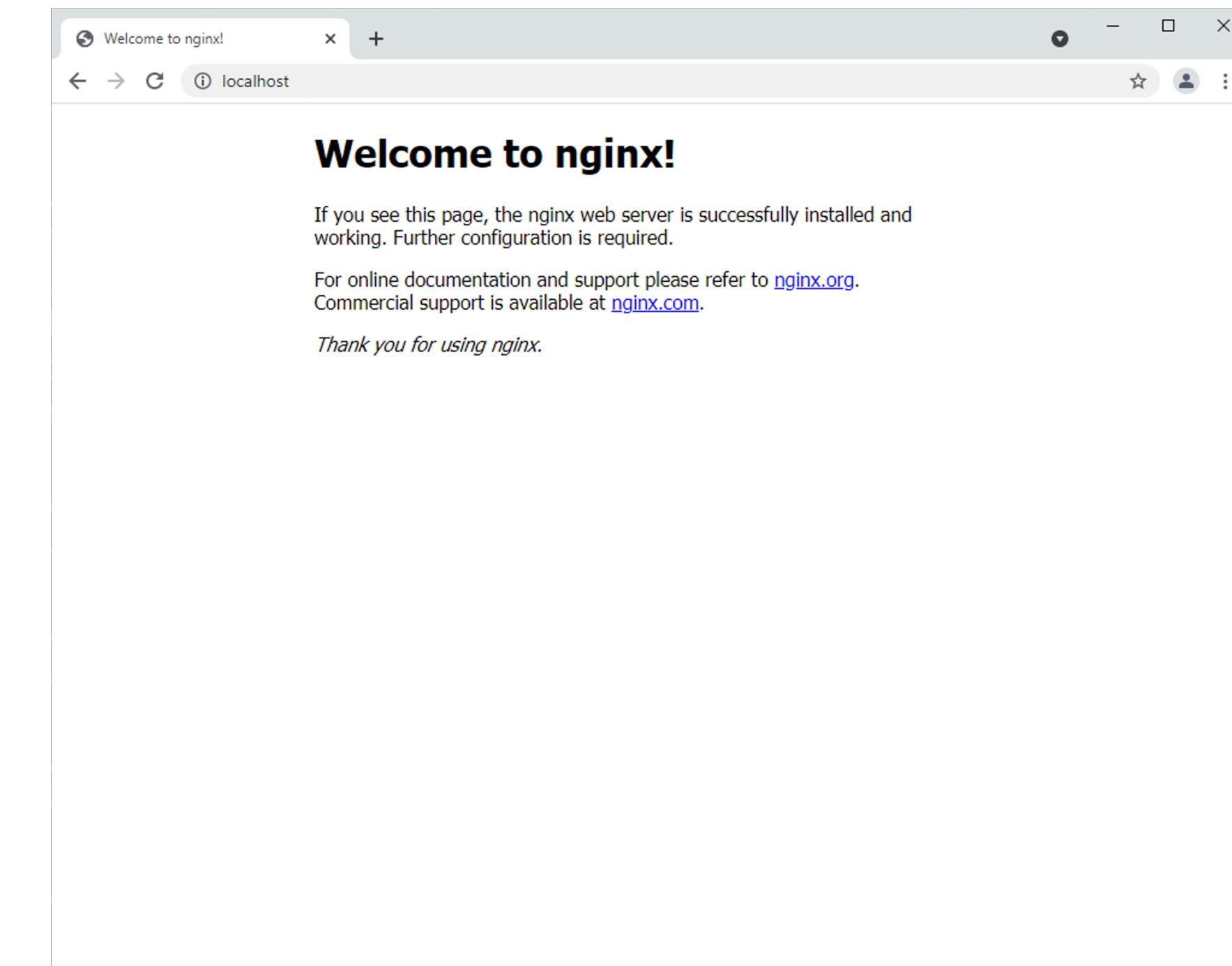
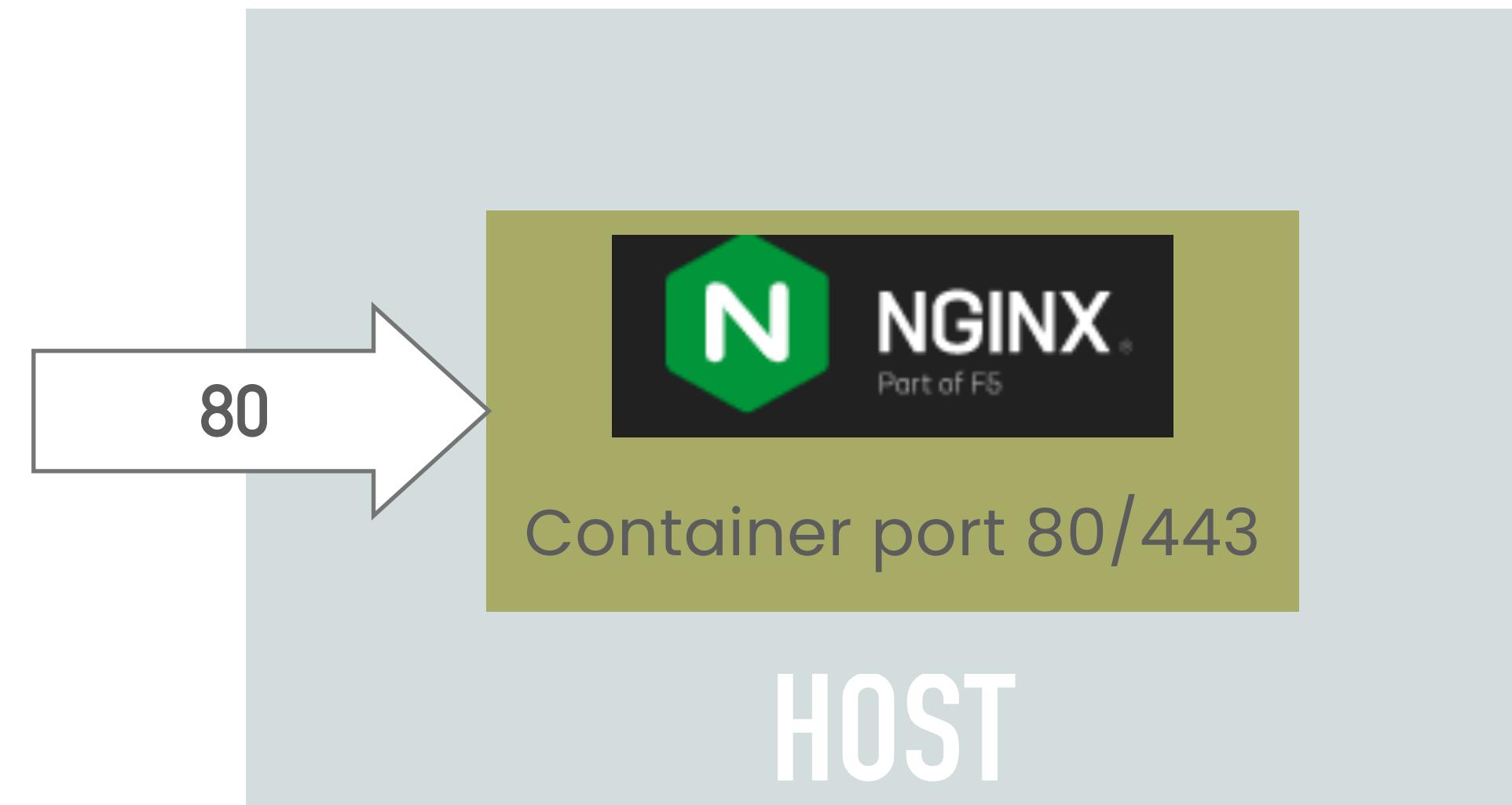


START WEB APPLICATION WITH NGINX

```
$ docker container run --name hello-nginx -d -p 80:80 nginx
```

OR

```
$ docker container run --name hello-nginx -d --publish 80:80 nginx
```

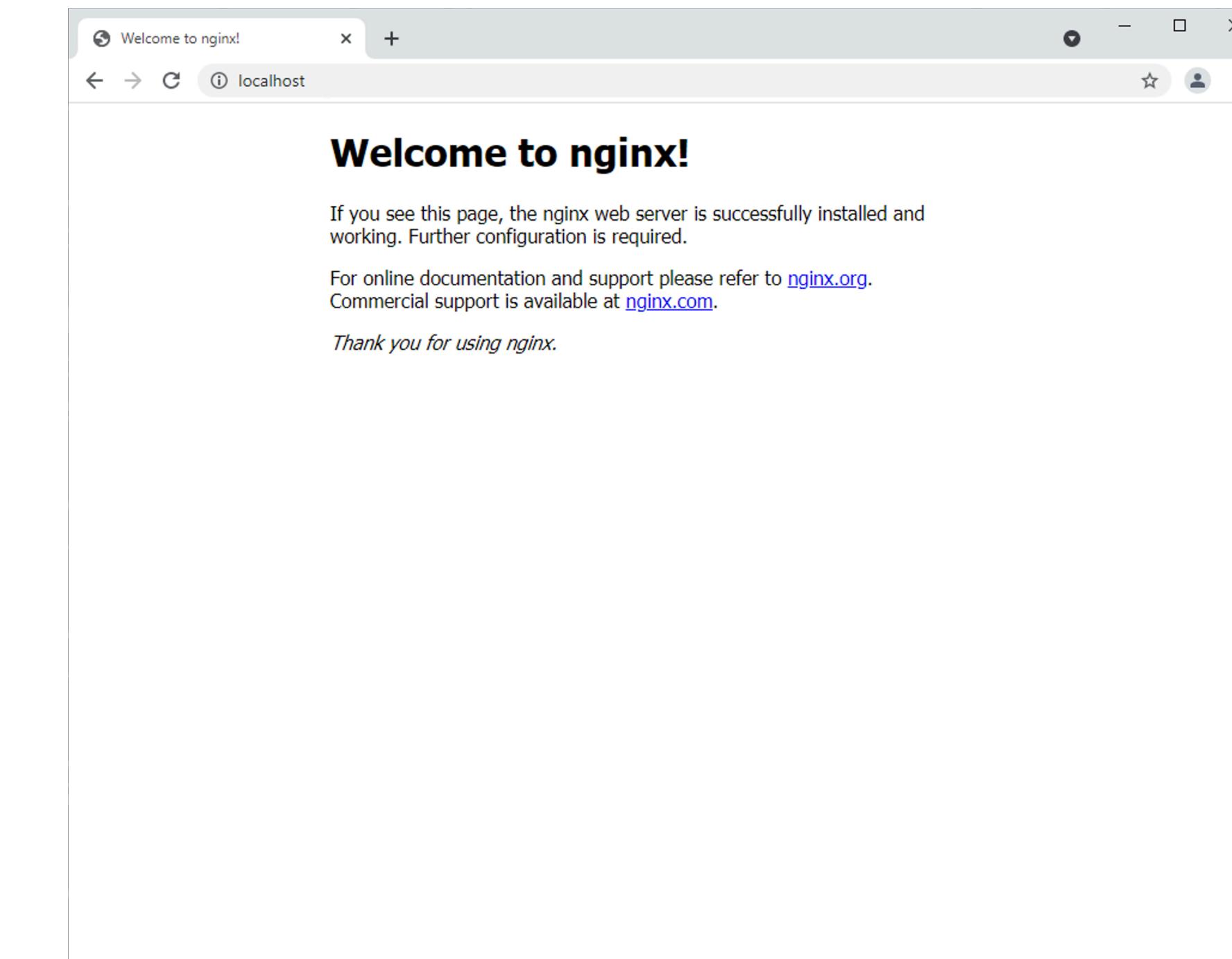
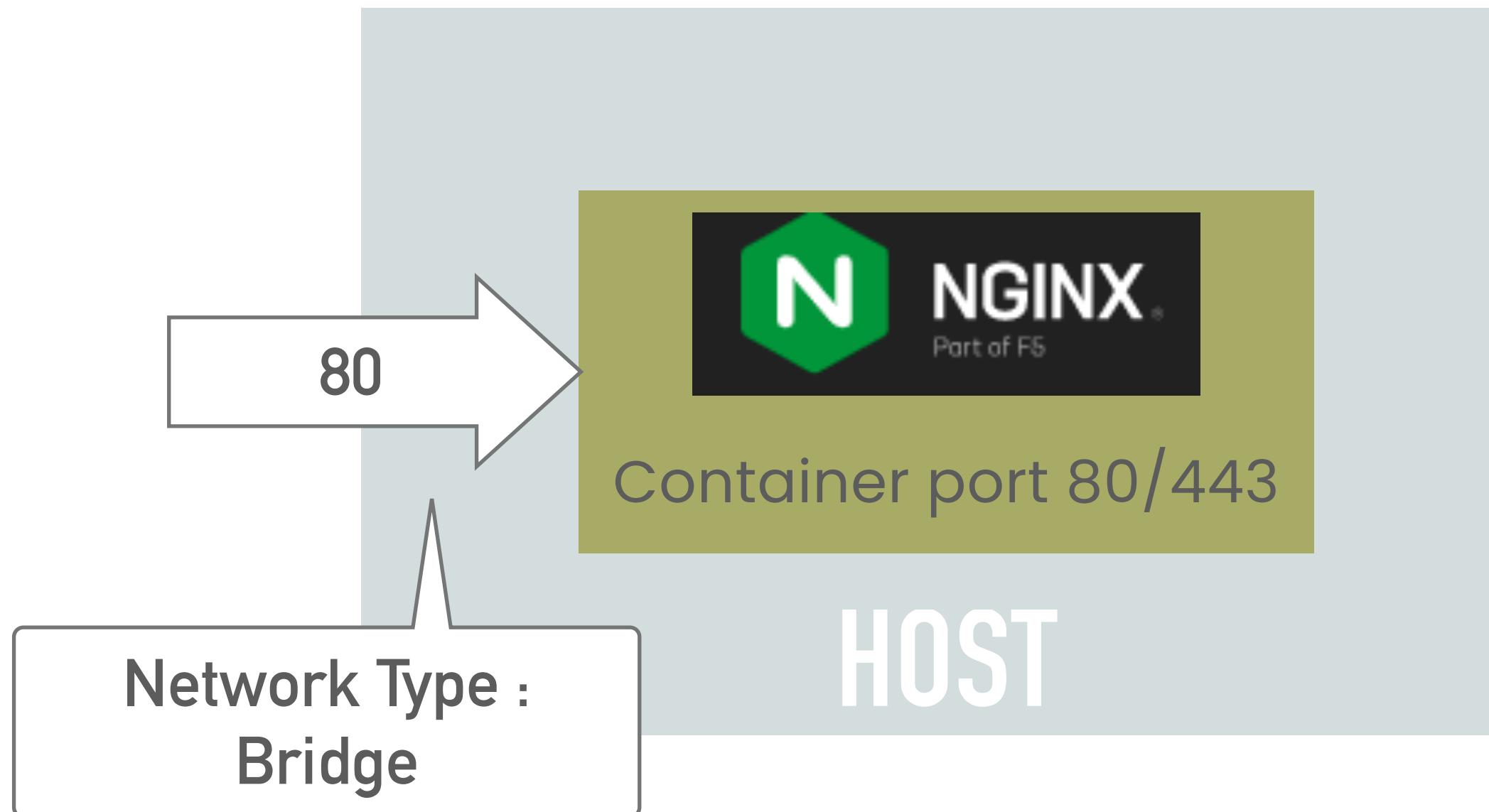


START WEB APPLICATION WITH NGINX

```
$ docker container run --name hello-nginx -d -p 80:80 nginx
```

OR

```
$ docker container run --name hello-nginx -d --publish 80:80 nginx
```

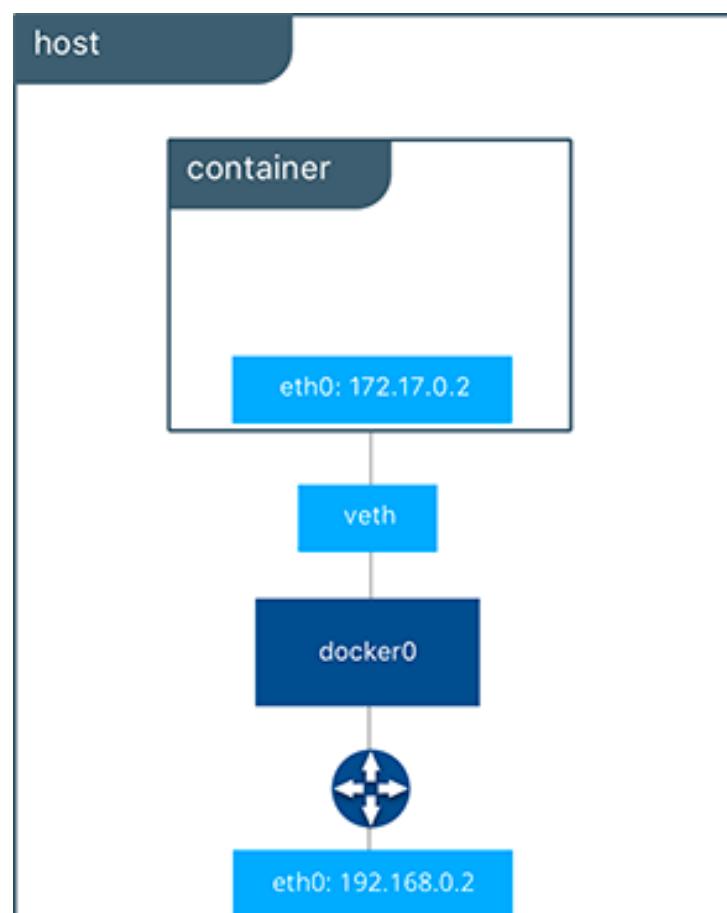


DOCKER NETWORK

DOCKER - NETWORK CLI AND NETWORK DRIVERS

```
$ docker network ls
```

NETWORK ID	NAME	DRIVER	SCOPE
5c3bb2502623	bridge	bridge	local
6877ad312030	host	host	local
d3d51f296ce4	none	null	local



bridge network

- **bridge**: The default network driver.
- **host**: For standalone containers, remove network isolation between the container and the Docker host, and use the host's networking directly.
- **overlay**: Overlay networks connect multiple Docker daemons together and enable swarm services to communicate with each other.
- **macvlan**: Macvlan networks allow you to assign a MAC address to a container, making it appear as a physical device on your network.
- **none**: For this container, disable all networking.

DOCKER - BRIDGE NETWORK

```
$ docker network inspect bridge
```

DOCKER - BRIDGE NETWORK

```
$ docker network inspect bridge
```

```
C:\Users\karan>docker network inspect bridge
[
  {
    "Name": "bridge",
    "Id": "5c3bb250262393a739a4c9f81b9291859738b94585eed3a0245ebc4d2848a010",
    "Created": "2021-05-26T01:56:57.6721089Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": null,
      "Config": [
        {
          "Subnet": "172.17.0.0/16",
          "Gateway": "172.17.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
      "bd087effleaf79f353c1448924bed9985438382ff822638dc125b8c7b0a3ea83": {
        "Name": "hello-nginx",
        "EndpointID": "306a62f3249ab8cfad056da39ccc81d93f328bf8151640b539e803a4ad32db6c",
        "MacAddress": "02:42:ac:11:00:02",
        "IPv4Address": "172.17.0.2/16",
        "IPv6Address": ""
      }
    },
    "Options": {
      "com.docker.network.bridge.default_bridge": "true",
      "com.docker.network.bridge.enable_icc": "true",
      "com.docker.network.bridge.enable_ip_masquerade": "true"
    }
  }
]
```

DOCKER - BRIDGE NETWORK

```
$ docker network inspect bridge
```

```
C:\Users\karan>docker network inspect bridge
[
  {
    "Name": "bridge",
    "Id": "5c3bb250262393a739a4c9f81b9291859738b94585eed3a0245ebc4d2848a010",
    "Created": "2021-05-26T01:56:57.6721089Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": null,
      "Config": [
        {
          "Subnet": "172.17.0.0/16",
          "Gateway": "172.17.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
      "bd087effleaf79f353c1448924bed9985438382ff822638dc125b8c7b0a3ea83": {
        "Name": "hello-nginx",
        "EndpointID": "306a62f3249ab8cfad056da39ccc81d93f328bf8151640b539e803a4ad32db6c",
        "MacAddress": "02:42:ac:11:00:02",
        "IPv4Address": "172.17.0.2/16",
        "IPv6Address": ""
      }
    },
    "Options": {
      "com.docker.network.bridge.default_bridge": "true",
      "com.docker.network.bridge.enable_icc": "true",
      "com.docker.network.bridge.enable_ip_masquerade": "true"
    }
  }
]
```

DOCKER INTERNAL NETWORK

```
"Config": [
  {
    "Subnet": "172.17.0.0/16",
    "Gateway": "172.17.0.1"
  }
]
```

DOCKER - BRIDGE NETWORK

```
$ docker network inspect bridge
```

```
C:\Users\karan>docker network inspect bridge
[{"Name": "bridge",
 "Id": "5c3bb250262393a739a4c9f81b9291859738b94585eed3a0245ebc4d2848a010",
 "Created": "2021-05-26T01:56:57.6721089Z",
 "Scope": "local",
 "Driver": "bridge",
 "EnableIPv6": false,
 "IPAM": {
   "Driver": "default",
   "Options": null,
   "Config": [
     {
       "Subnet": "172.17.0.0/16",
       "Gateway": "172.17.0.1"
     }
   ]
 },
 "Internal": false,
 "Attachable": false,
 "Ingress": false,
 "ConfigFrom": {
   "Network": ""
 },
 "ConfigOnly": false,
 "Containers": {
   "bd087effleaf79f353c1448924bed99854": {
     "Name": "hello-nginx",
     "EndpointID": "306a62f3249ab8cfad056d",
     "MacAddress": "02:42:ac:11:00:02",
     "IPv4Address": "172.17.0.2/16",
     "IPv6Address": ""
   }
 },
 "Options": {
   "com.docker.network.bridge.default_bridge": "true",
   "com.docker.network.bridge.enable_icc": "true",
   "com.docker.network.bridge.enable_ip_masquerade": "true"
}}
```

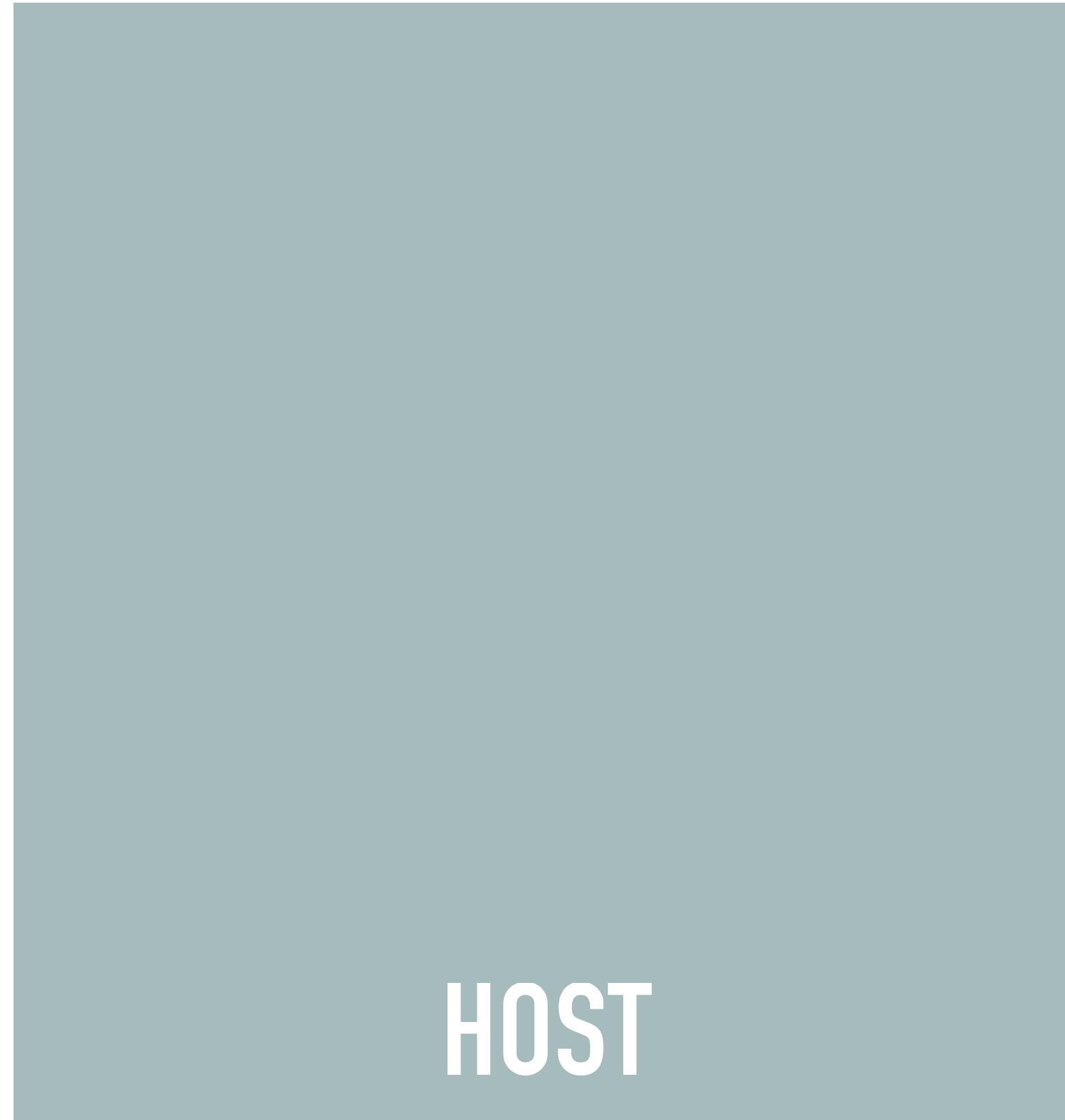
DOCKER INTERNAL NETWORK

```
"Config": [
  {
    "Subnet": "172.17.0.0/16",
    "Gateway": "172.17.0.1"
  }
]
```

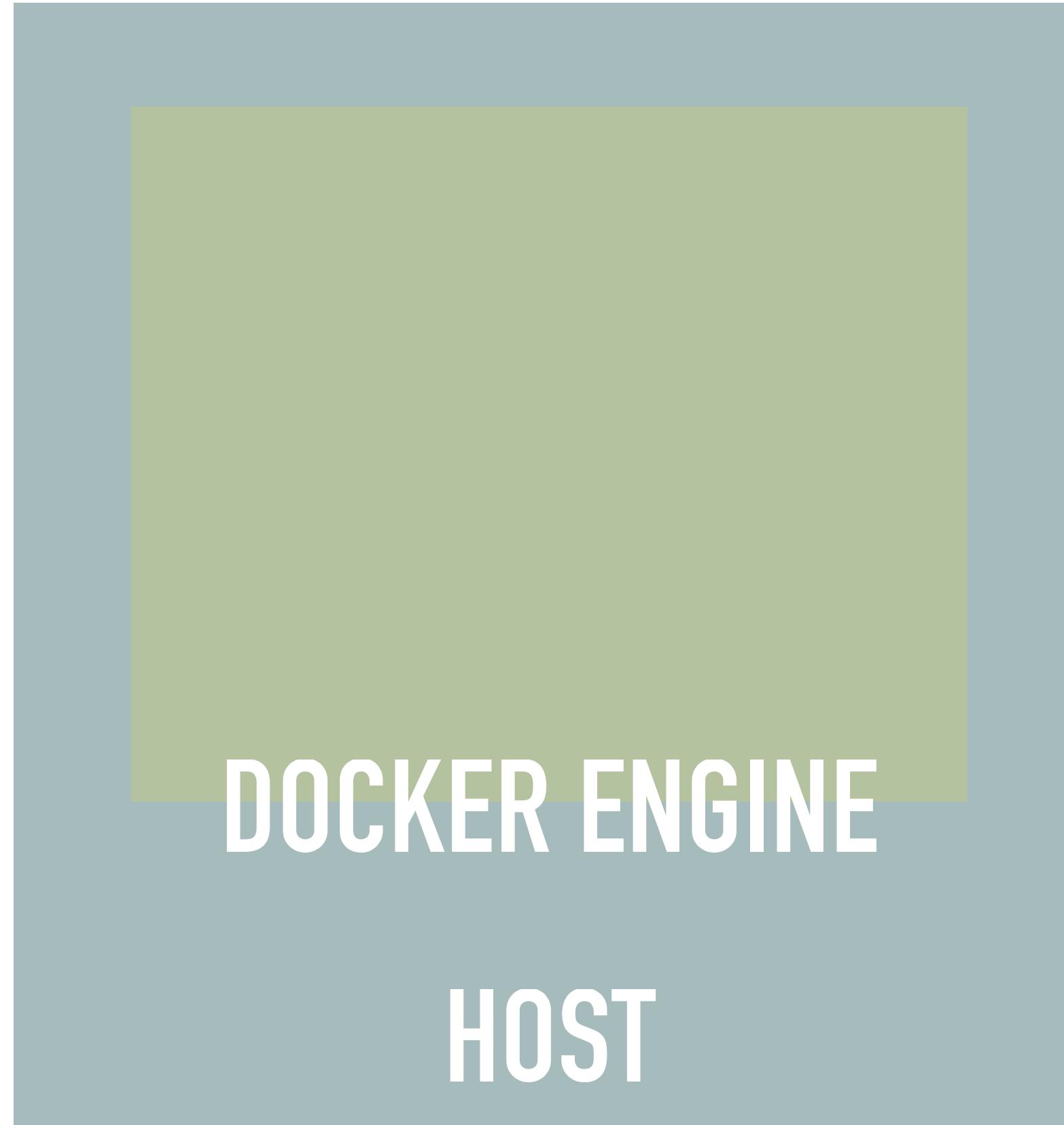
IP-Address of hello-nginx

```
"Name": "hello-nginx",
"EndpointID": "306a62f3249ab8cfad056d",
"MacAddress": "02:42:ac:11:00:02",
"IPv4Address": "172.17.0.2/16",
"IPv6Address": ""
```

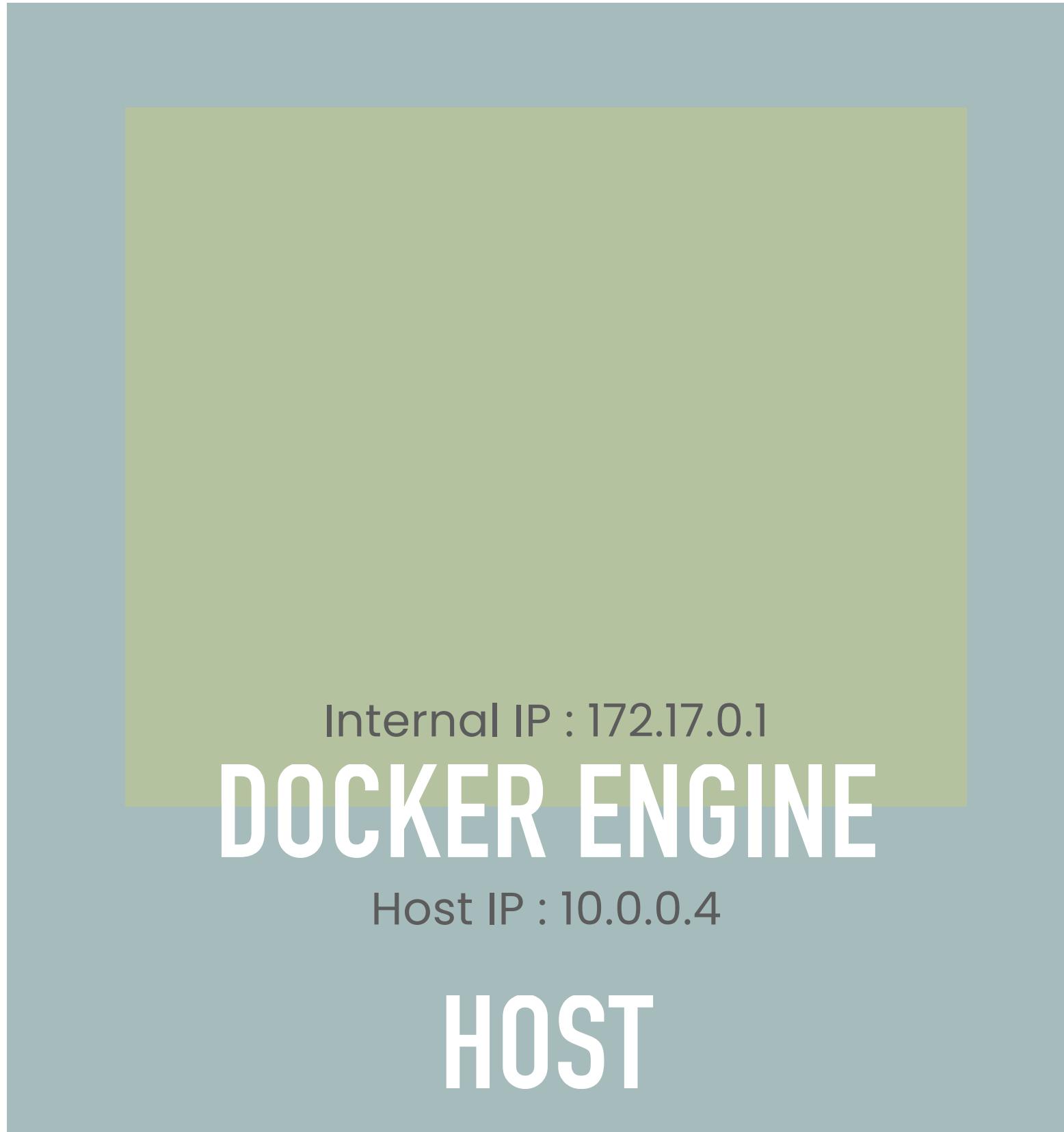
DOCKER - BRIDGE NETWORK



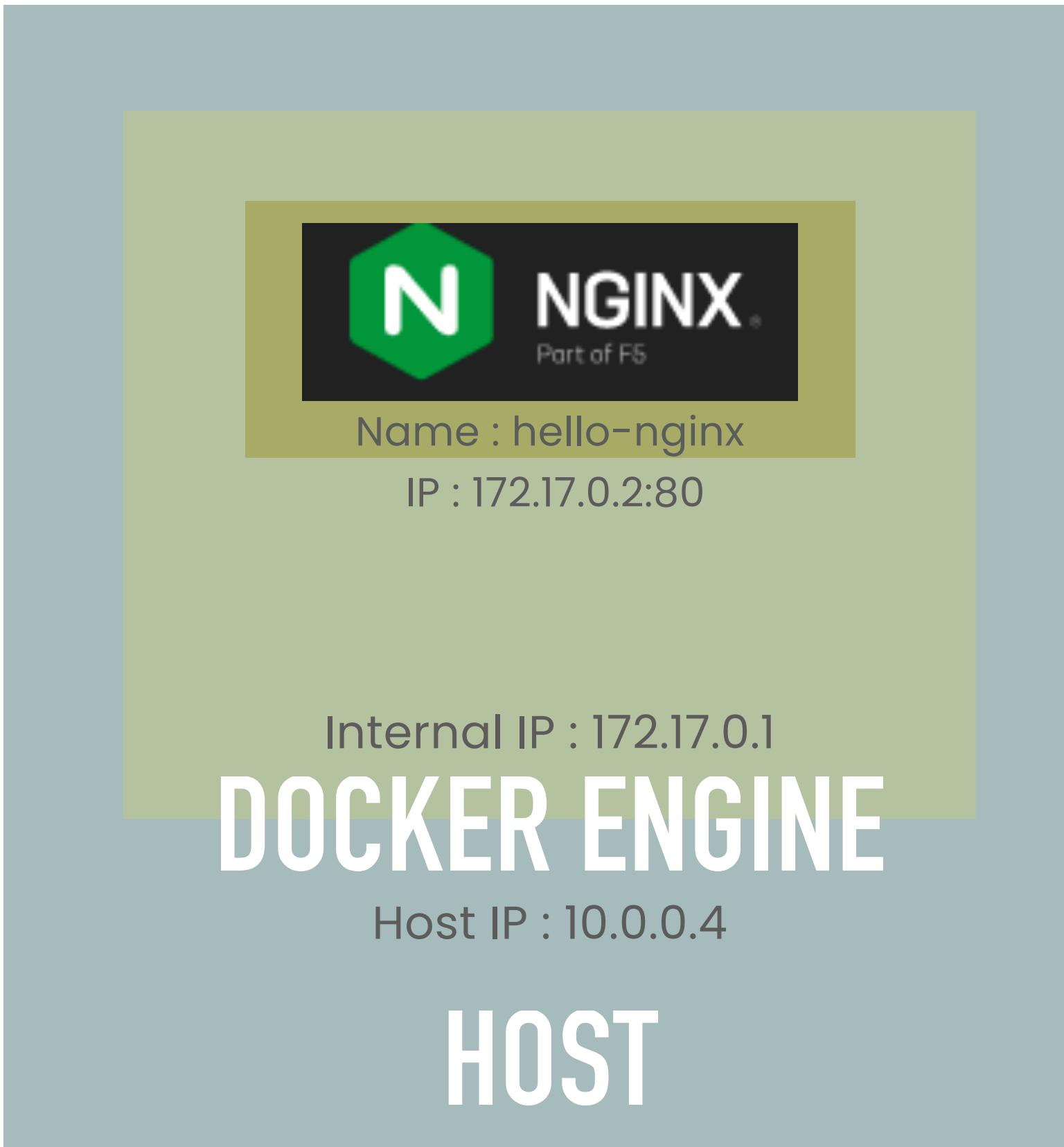
DOCKER - BRIDGE NETWORK



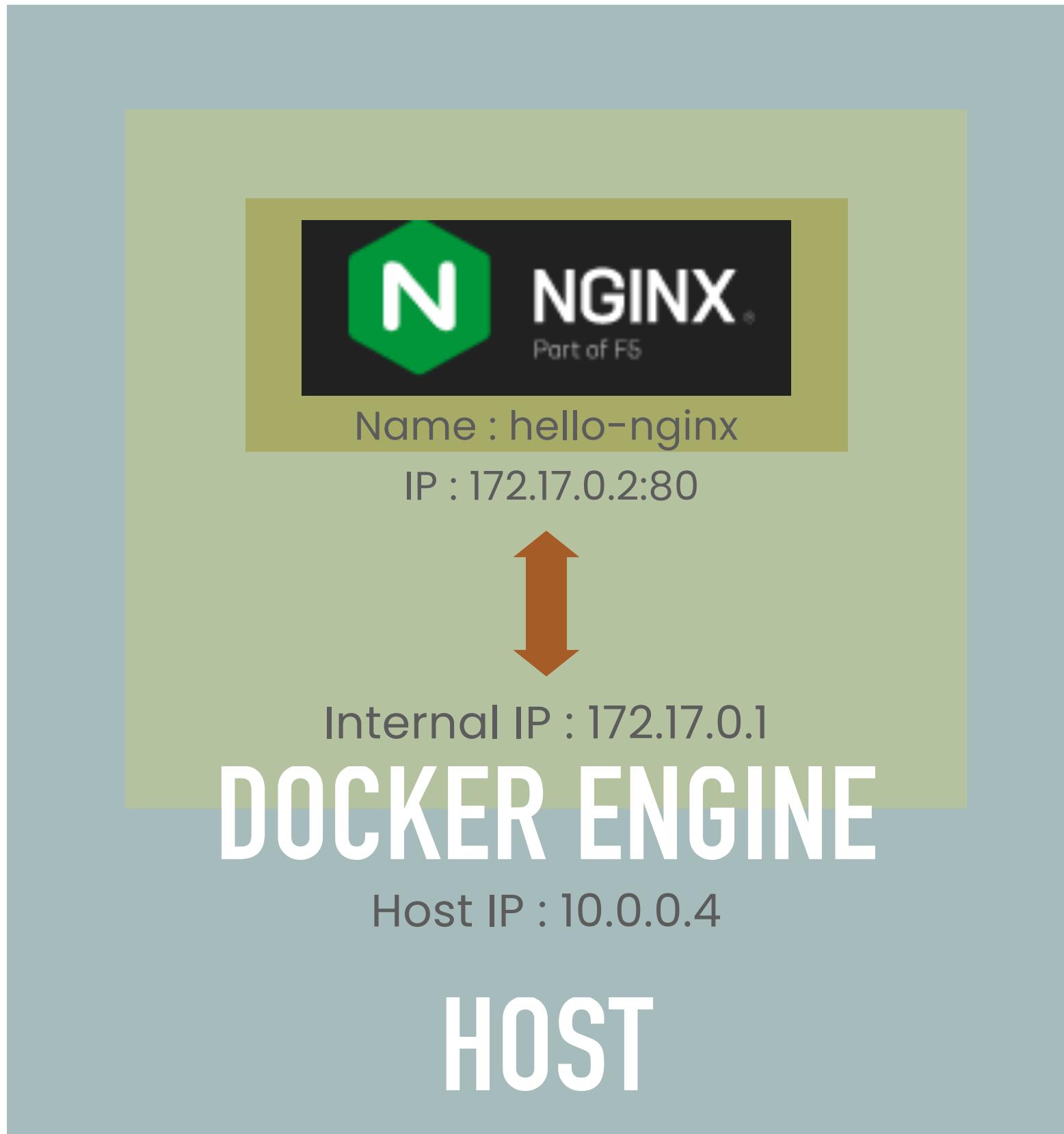
DOCKER - BRIDGE NETWORK



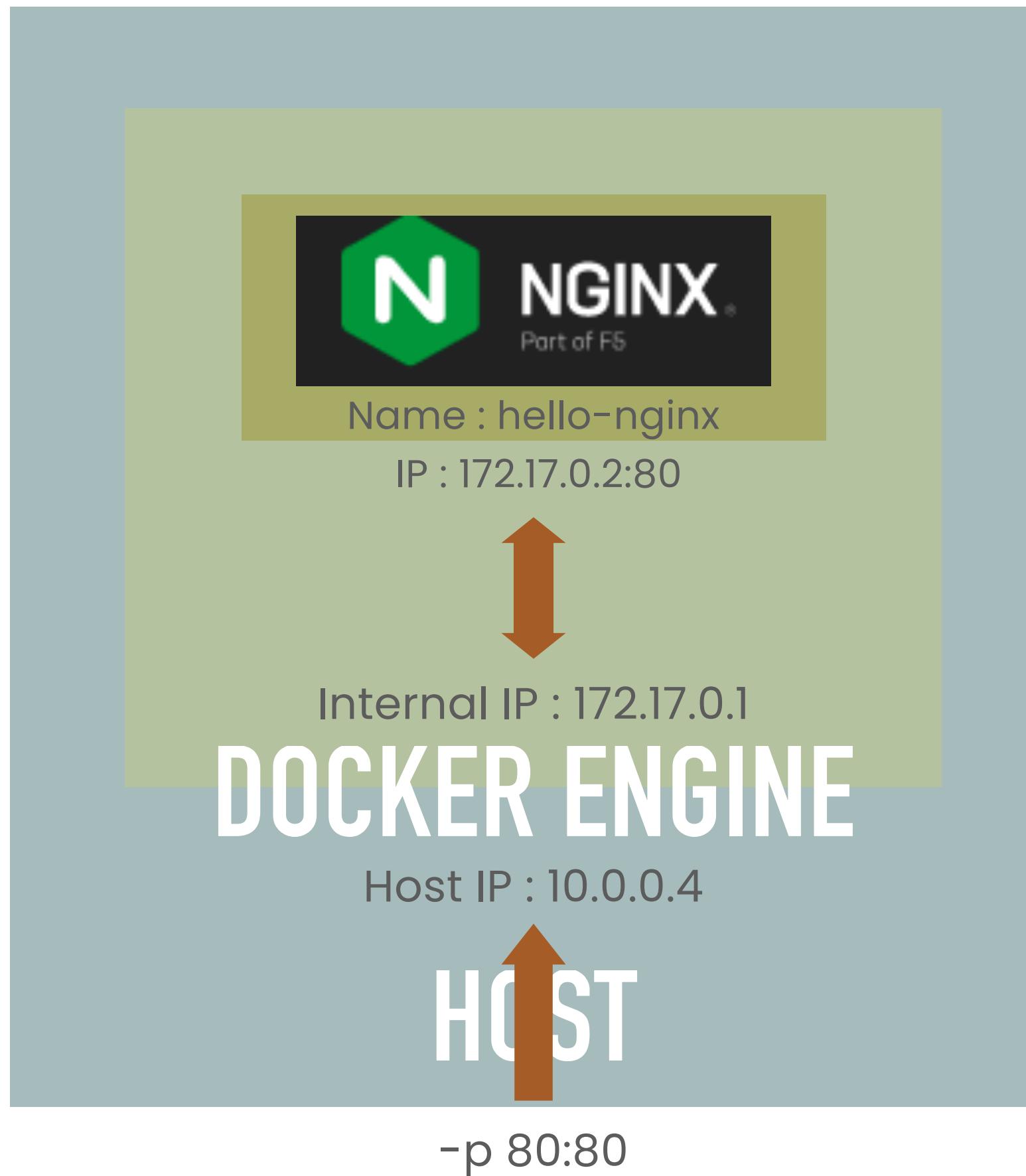
DOCKER - BRIDGE NETWORK



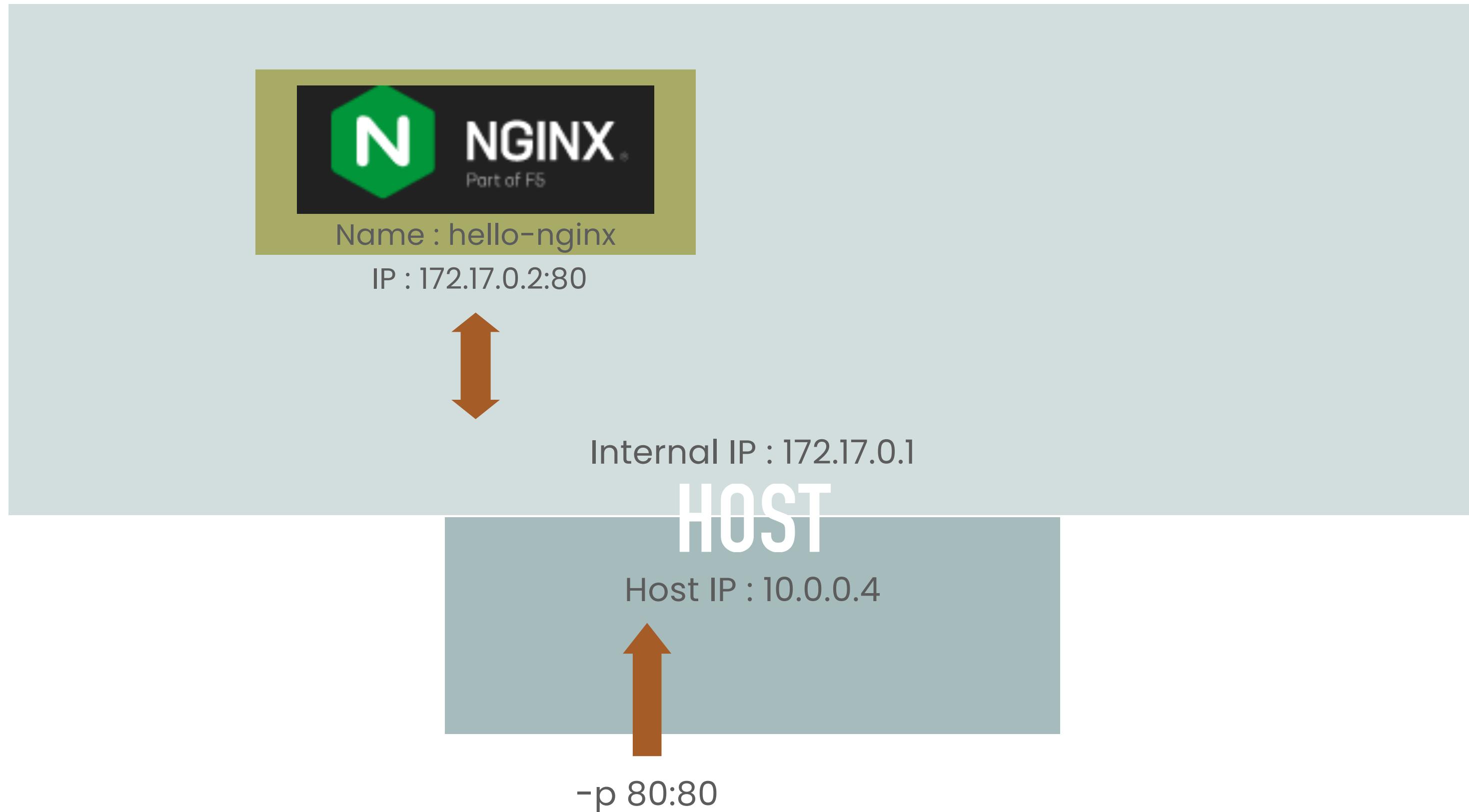
DOCKER - BRIDGE NETWORK



DOCKER - BRIDGE NETWORK

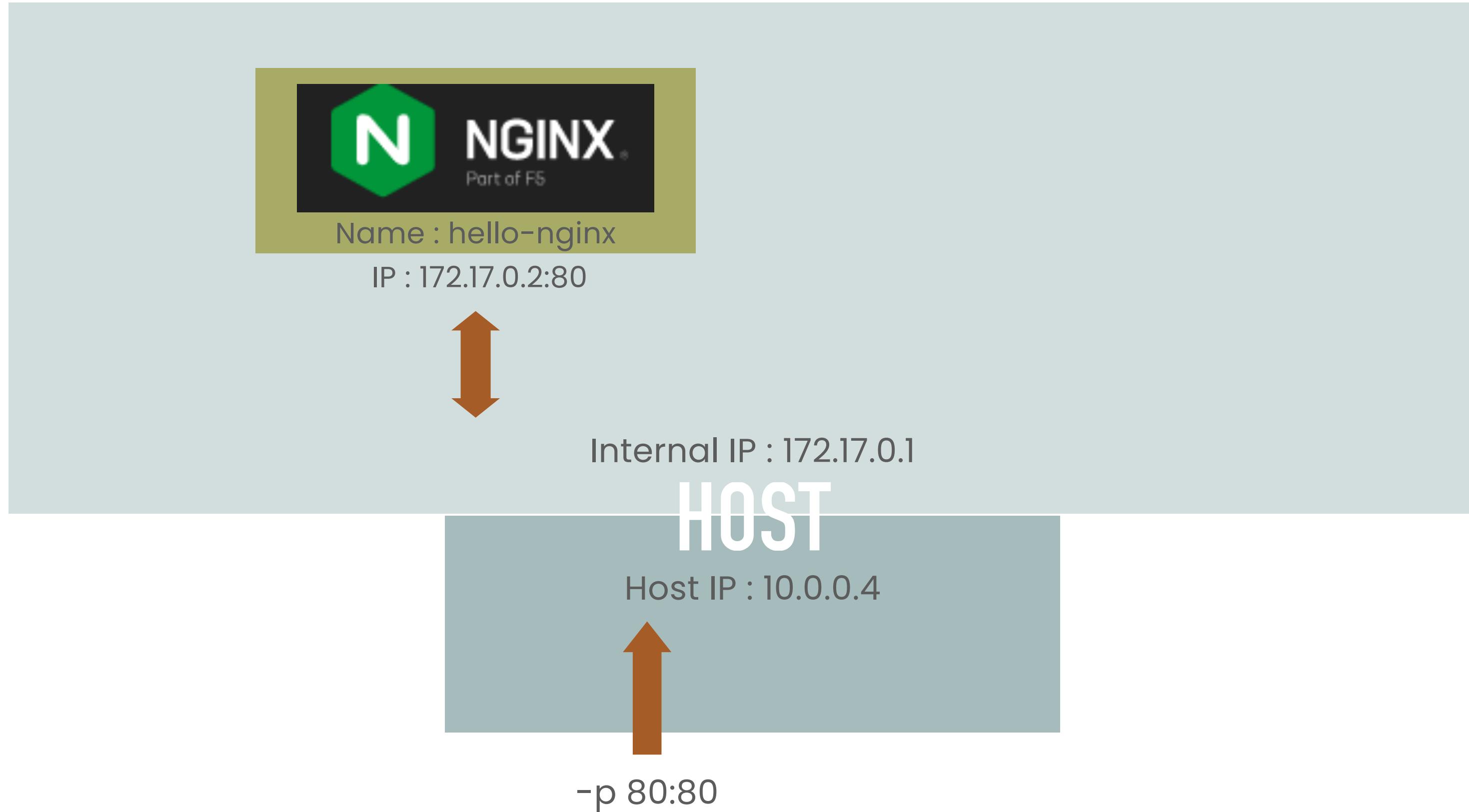


DOCKER - CREATE NEW NGINX CONTAINER



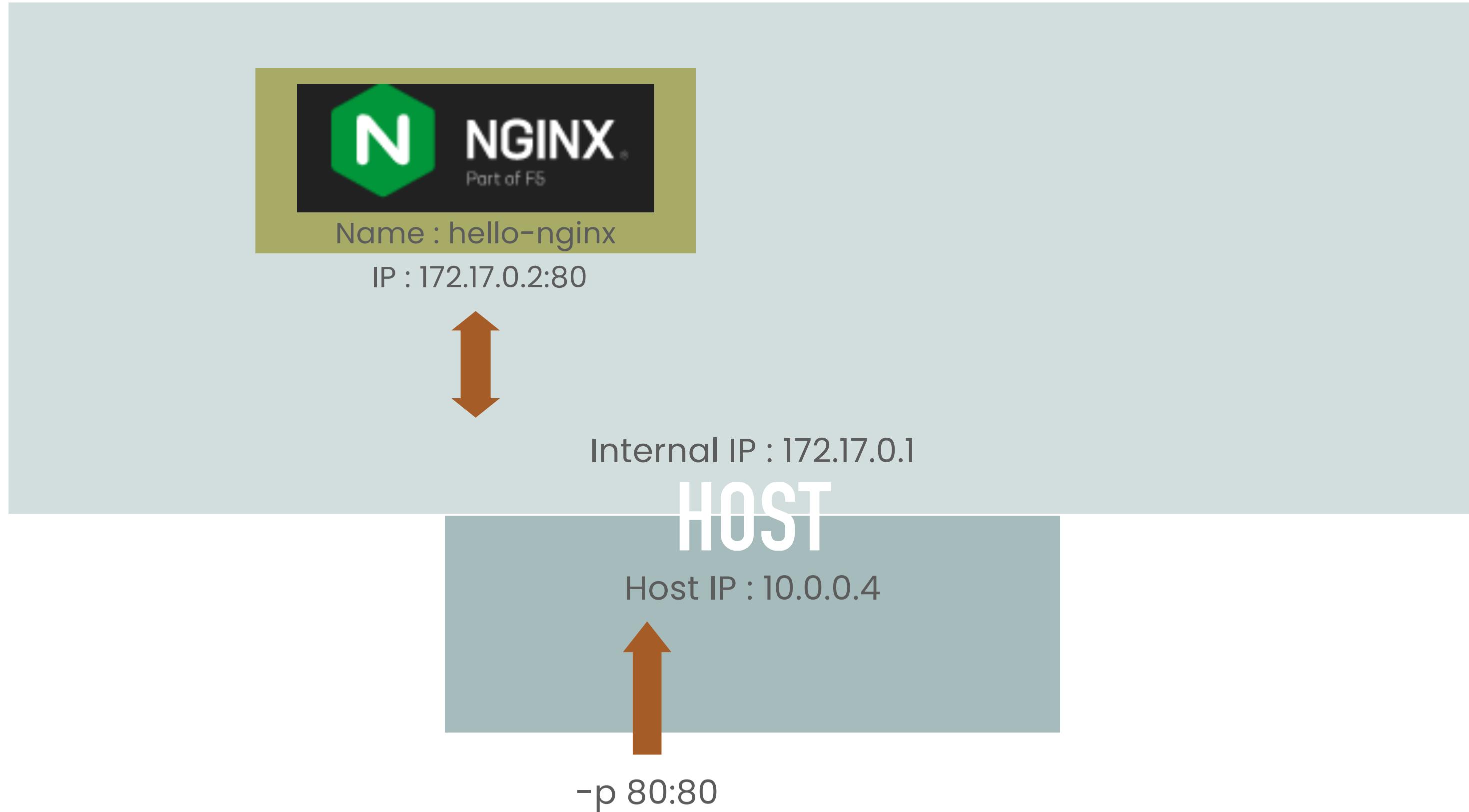
DOCKER - CREATE NEW NGINX CONTAINER

```
$ docker container run --name hello-nginx-2 -d -p 81:80 nginx
```



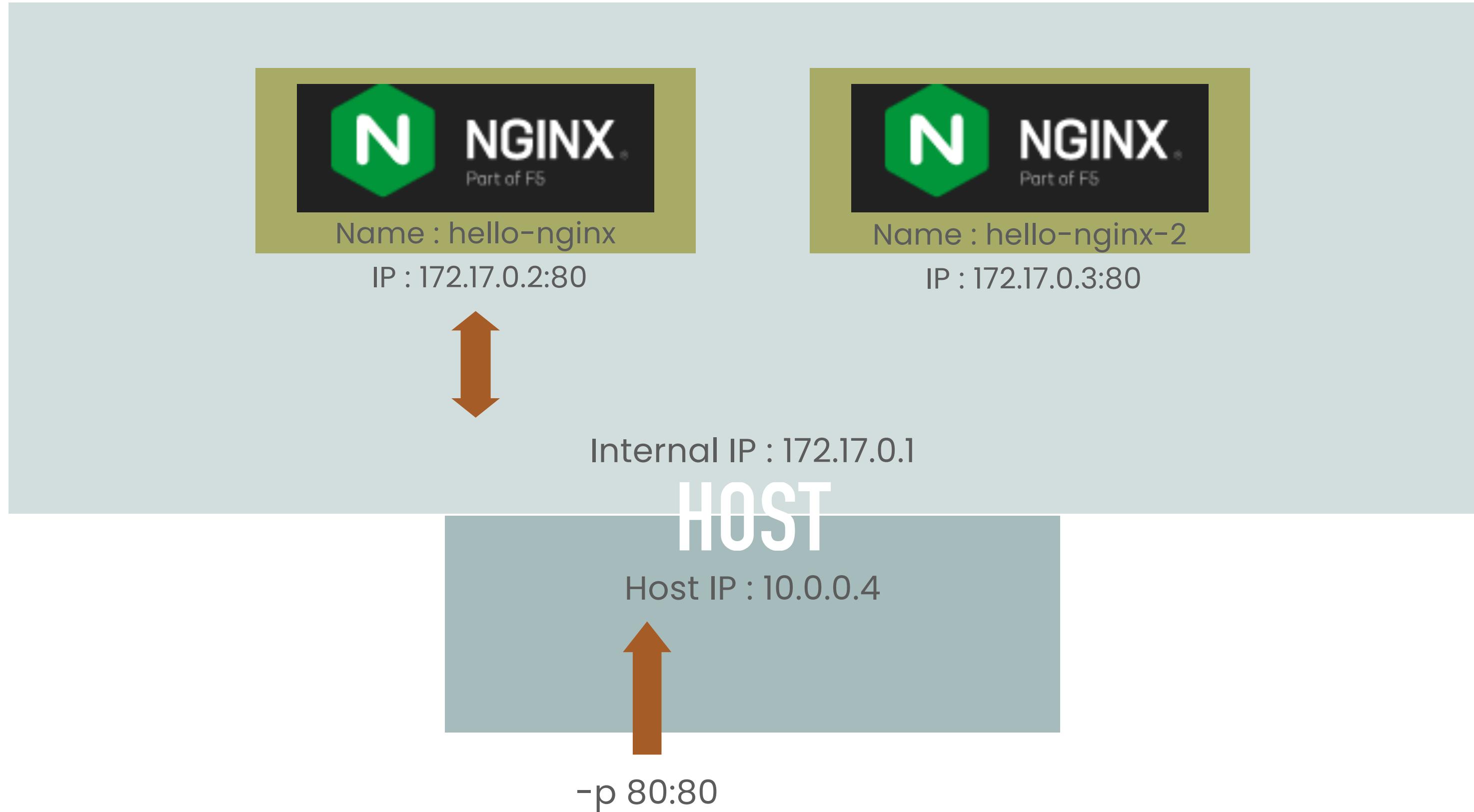
DOCKER - CREATE NEW NGINX CONTAINER

```
$ docker container run --name hello-nginx-2 -d -p 81:80 nginx
```



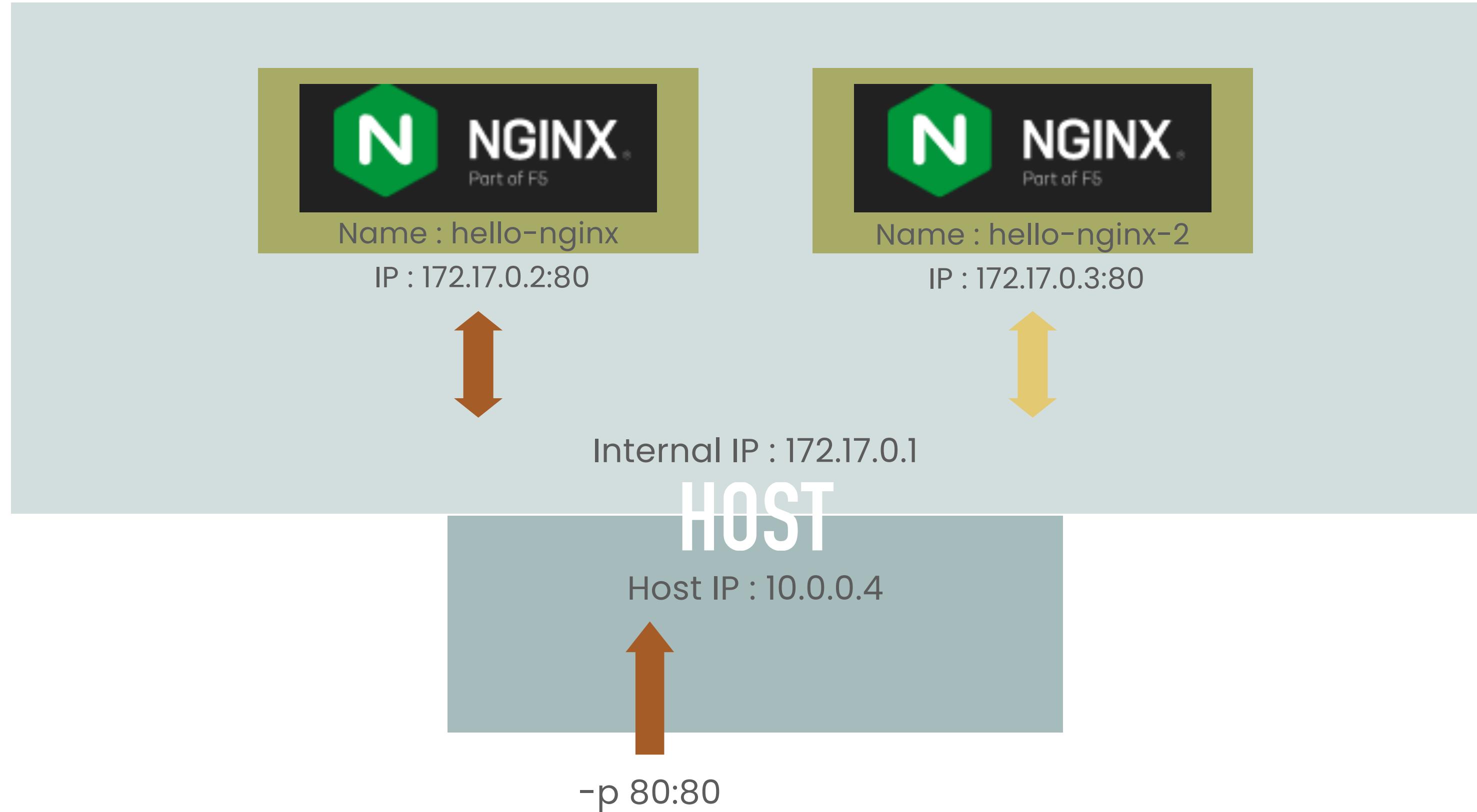
DOCKER - CREATE NEW NGINX CONTAINER

```
$ docker container run --name hello-nginx-2 -d -p 81:80 nginx
```



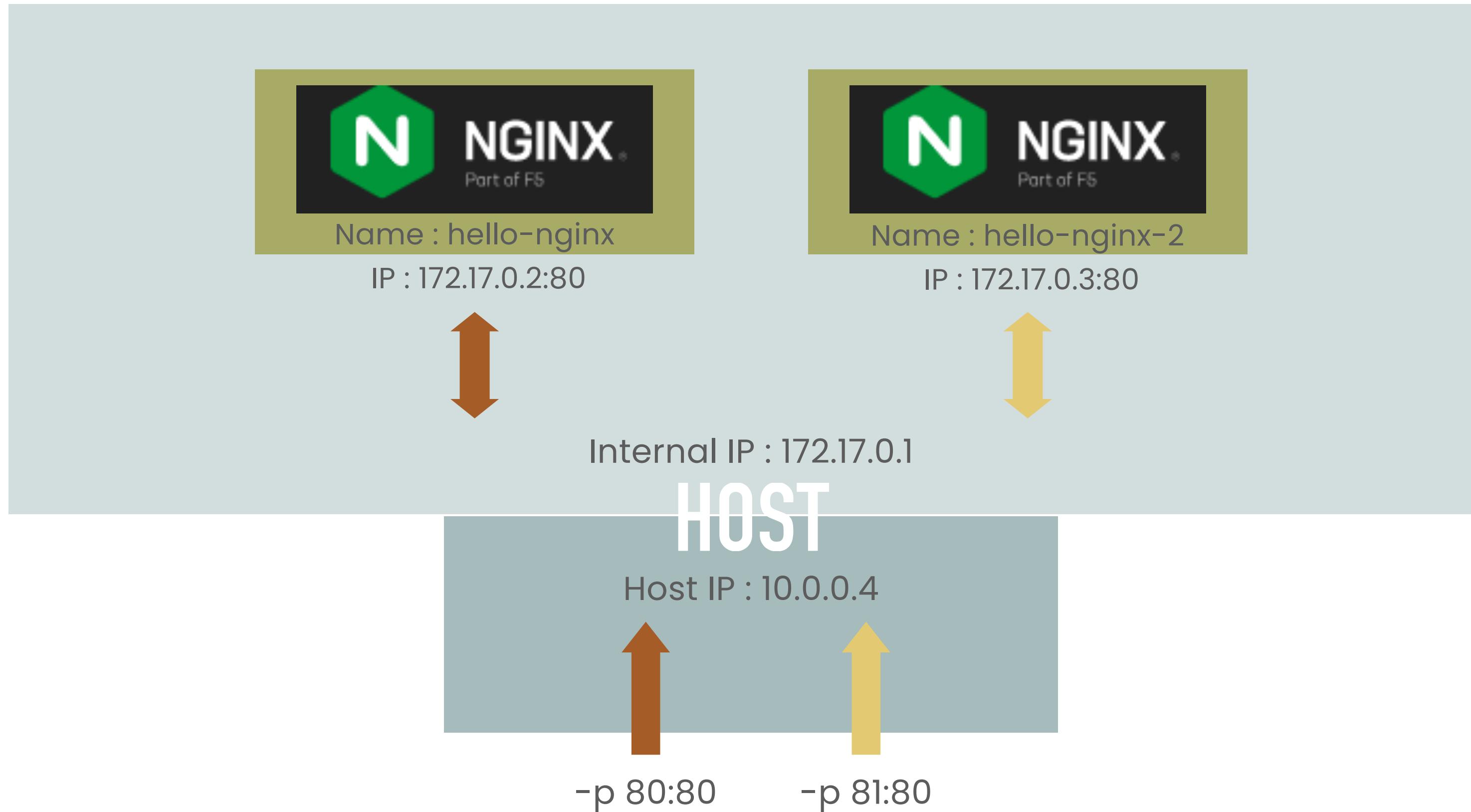
DOCKER - CREATE NEW NGINX CONTAINER

```
$ docker container run --name hello-nginx-2 -d -p 81:80 nginx
```



DOCKER - CREATE NEW NGINX CONTAINER

```
$ docker container run --name hello-nginx-2 -d -p 81:80 nginx
```



CREATE YOUR OWN BRIDGE NETWORK

```
$ docker network inspect bridge
```

CREATE YOUR OWN BRIDGE NETWORK

```
$ docker network inspect bridge
```

```
"IPAM": {
    "Driver": "default",
    "Options": null,
    "Config": [
        {
            "Subnet": "172.17.0.0/16",
            "Gateway": "172.17.0.1"
        }
    ],
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
        "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
        "8c6406e9cb0f8decc57e12c1875ca4cf9d2bc7de833b992e9150f9f555072cf": {
            "Name": "hello-nginx-2",
            "EndpointID": "5f4d9d9b35bab4f6b738881c0036efa91f40ad1ce56294b5a047c450e710cf98",
            "MacAddress": "02:42:ac:11:00:03",
            "IPv4Address": "172.17.0.3/16",
            "IPv6Address": ""
        },
        "bd087effleaf79f353c1448924bed9985438382ff822638dc125b8c7b0a3ea83": {
            "Name": "hello-nginx",
            "EndpointID": "306a62f3249ab8cfad056da39ccc81d93f328bf8151640b539e803a4ad32db6c",
            "MacAddress": "02:42:ac:11:00:02",
            "IPv4Address": "172.17.0.2/16",
            "IPv6Address": ""
        }
    },
    "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.bridge.mtu": "1500"
    }
}
```

CREATE YOUR OWN BRIDGE NETWORK

```
$ docker network inspect bridge
```

```
"IPAM": {
    "Driver": "default",
    "Options": null,
    "Config": [
        {
            "Subnet": "172.17.0.0/16",
            "Gateway": "172.17.0.1"
        }
    ],
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
        "Network": ""
    },
    "ConfigOnly": false,
    "Containers": [
        "8c6406e9cb0f8decc57e12c1875ca4cf9d2bc7de833b992e9150f9f555072cf": {
            "Name": "hello-nginx-2",
            "EndpointID": "5f4d9d9b35bab4f6b738881c0036efa91f40ad1ce56294b5a047c450e710cf98",
            "MacAddress": "02:42:ac:11:00:03",
            "IPv4Address": "172.17.0.3/16",
            "IPv6Address": ""
        },
        "bd087effleaf79f353c1448924bed9985438": {
            "Name": "hello-nginx",
            "EndpointID": "306a62f3249ab8cfad056da39ccc8",
            "MacAddress": "02:42:ac:11:00:02",
            "IPv4Address": "172.17.0.2/16",
            "IPv6Address": ""
        }
    ],
    "Options": {
        "com.docker.network.bridge.default_bridge": "bridge",
        "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.bridge.mtu": "1500"
    }
}
```

| IP-Address of hello-nginx

```
"Name": "hello-nginx",
"EndpointID": "306a62f3249ab8cfad056da39ccc8",
"MacAddress": "02:42:ac:11:00:02",
"IPv4Address": "172.17.0.2/16",
"IPv6Address": ""
```

CREATE YOUR OWN BRIDGE NETWORK

```
$ docker network inspect bridge
```

```
"IPAM": {
    "Driver": "default",
    "Options": null,
    "Config": [
        {
            "Subnet": "172.17.0.0/16",
            "Gateway": "172.17.0.1"
        }
    ],
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
        "Network": ""
    },
    "ConfigOnly": false,
    "Containers": [
        "8c6406e9cb0f8decc57e12c1875ca4cf9d2",
        "Name": "hello-nginx-2",
        "EndpointID": "5f4d9d9b35bab4f6b738881c0036e-",
        "MacAddress": "02:42:ac:11:00:03",
        "IPv4Address": "172.17.0.3/16",
        "IPv6Address": ""
    ],
    "bd087effleaf79f353c1448924bed99854383",
    "Name": "hello-nginx",
    "EndpointID": "306a62f3249ab8cfad056da39ccc8",
    "MacAddress": "02:42:ac:11:00:02",
    "IPv4Address": "172.17.0.2/16",
    "IPv6Address": ""
},
"Options": {
    "com.docker.network.bridge.default_bridge": "bridge",
    "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.bridge.mtu": "1500"
}
```

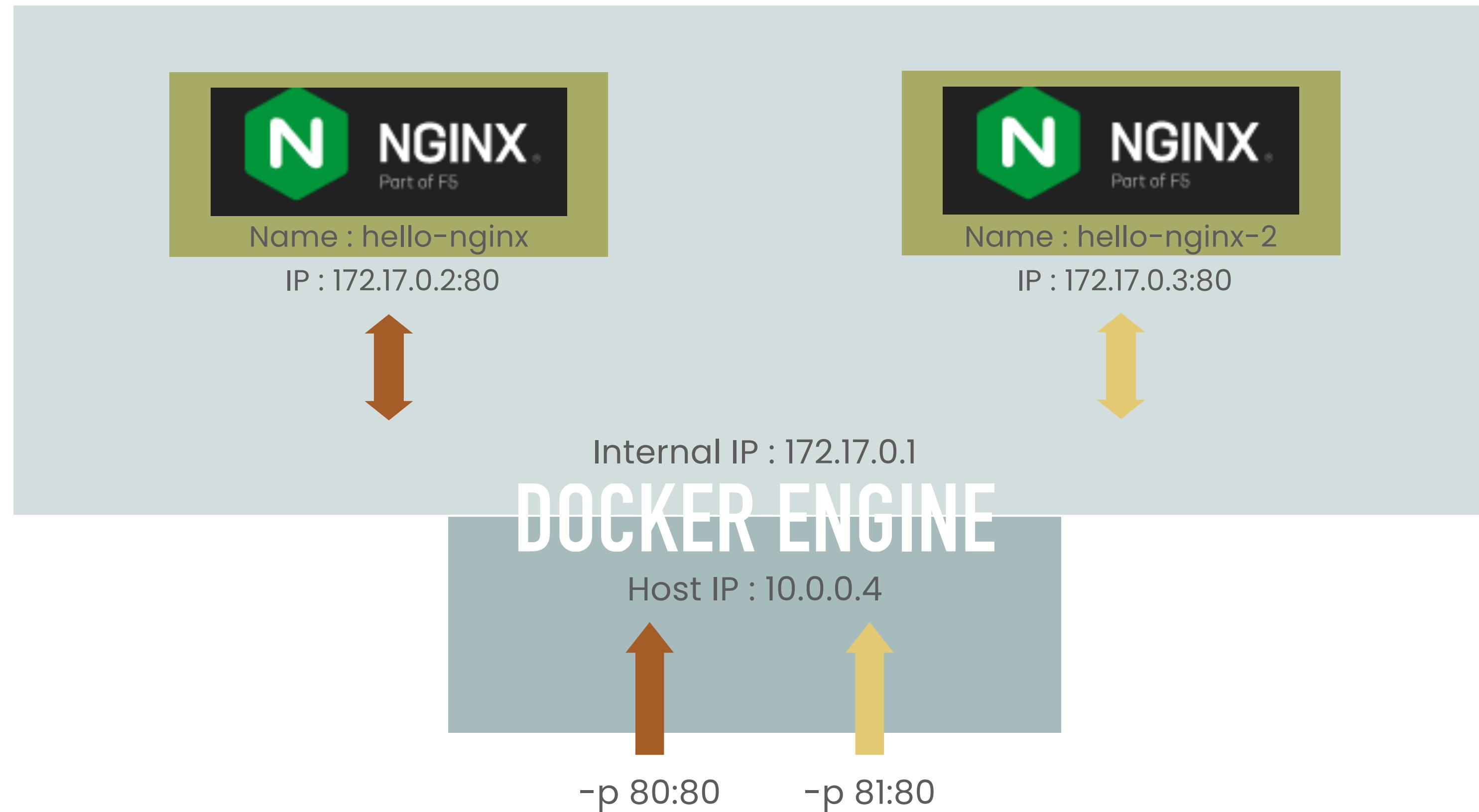
IP-Address of hello-nginx-2

```
"Name": "hello-nginx-2",
"EndpointID": "5f4d9d9b35bab4f6b738881c0036e-",
"MacAddress": "02:42:ac:11:00:03",
"IPv4Address": "172.17.0.3/16",
"IPv6Address": ""
```

IP-Address of hello-nginx

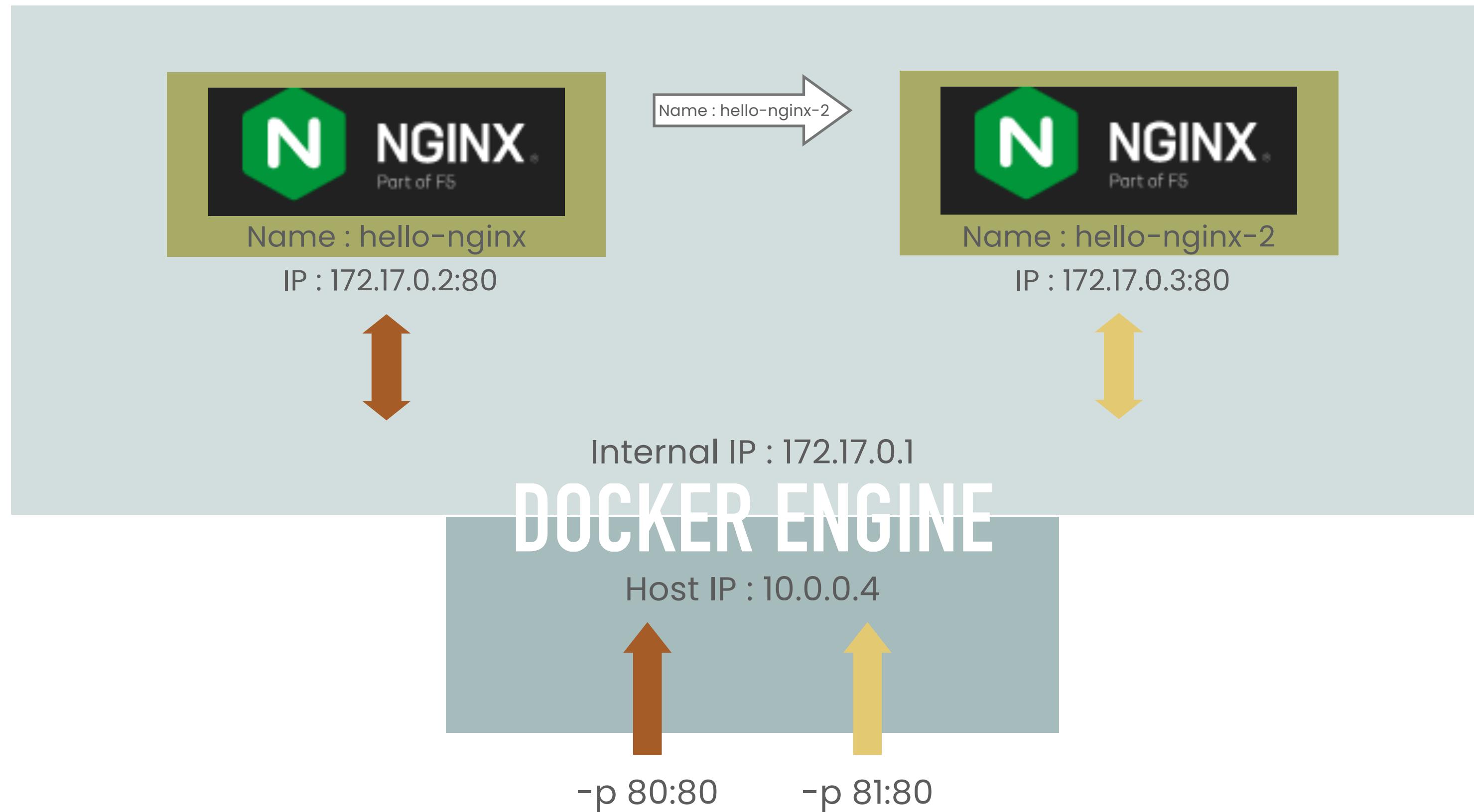
```
"Name": "hello-nginx",
"EndpointID": "306a62f3249ab8cfad056da39ccc8",
"MacAddress": "02:42:ac:11:00:02",
"IPv4Address": "172.17.0.2/16",
"IPv6Address": ""
```

DOCKER - NO DNS FOR DEFAULT BRIDGE NETWORK



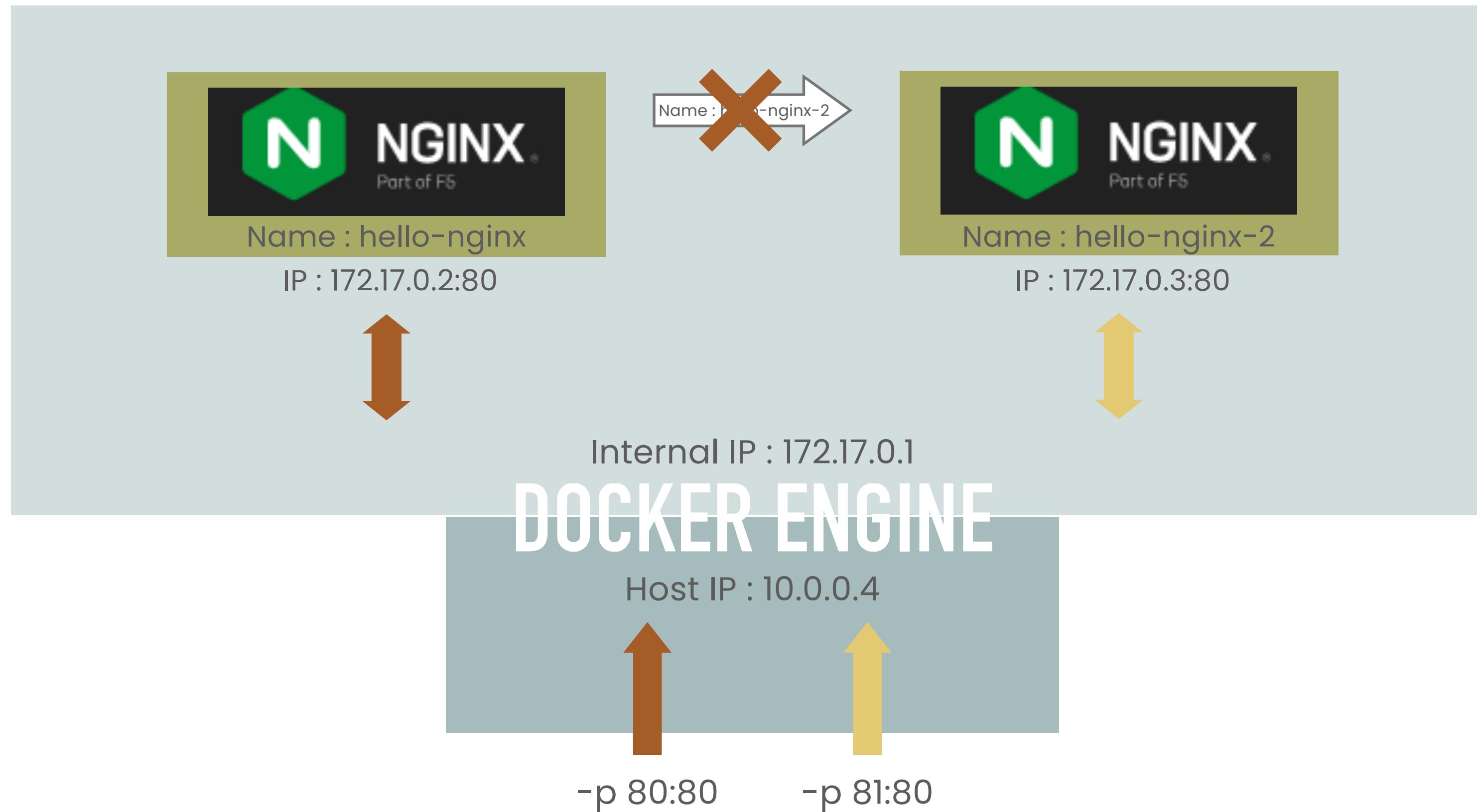
DOCKER - NO DNS FOR DEFAULT BRIDGE NETWORK

```
$ docker container exec -it hello-nginx curl hello-nginx-2
```



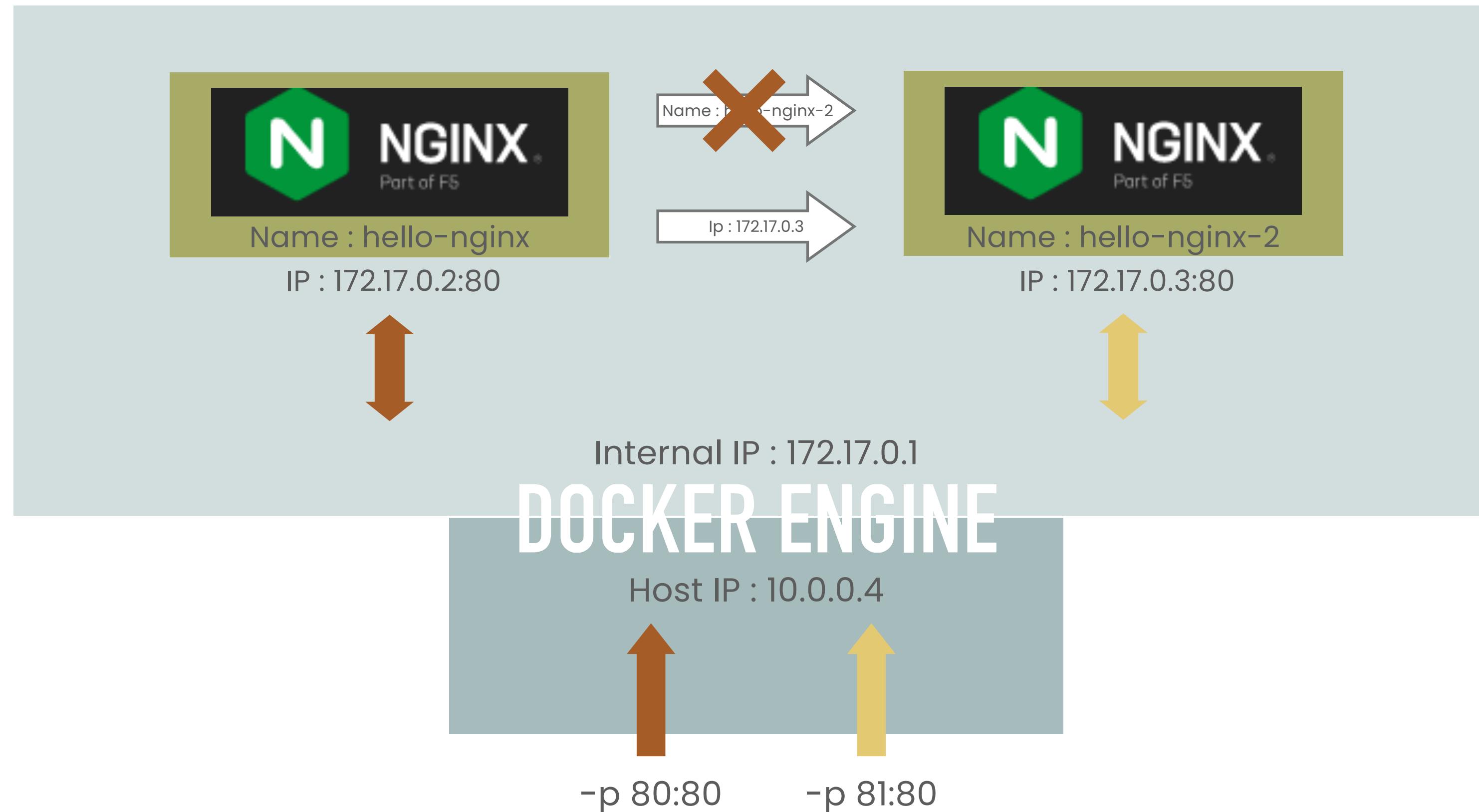
DOCKER - NO DNS FOR DEFAULT BRIDGE NETWORK

```
$ docker container exec -it hello-nginx curl hello-nginx-2
```



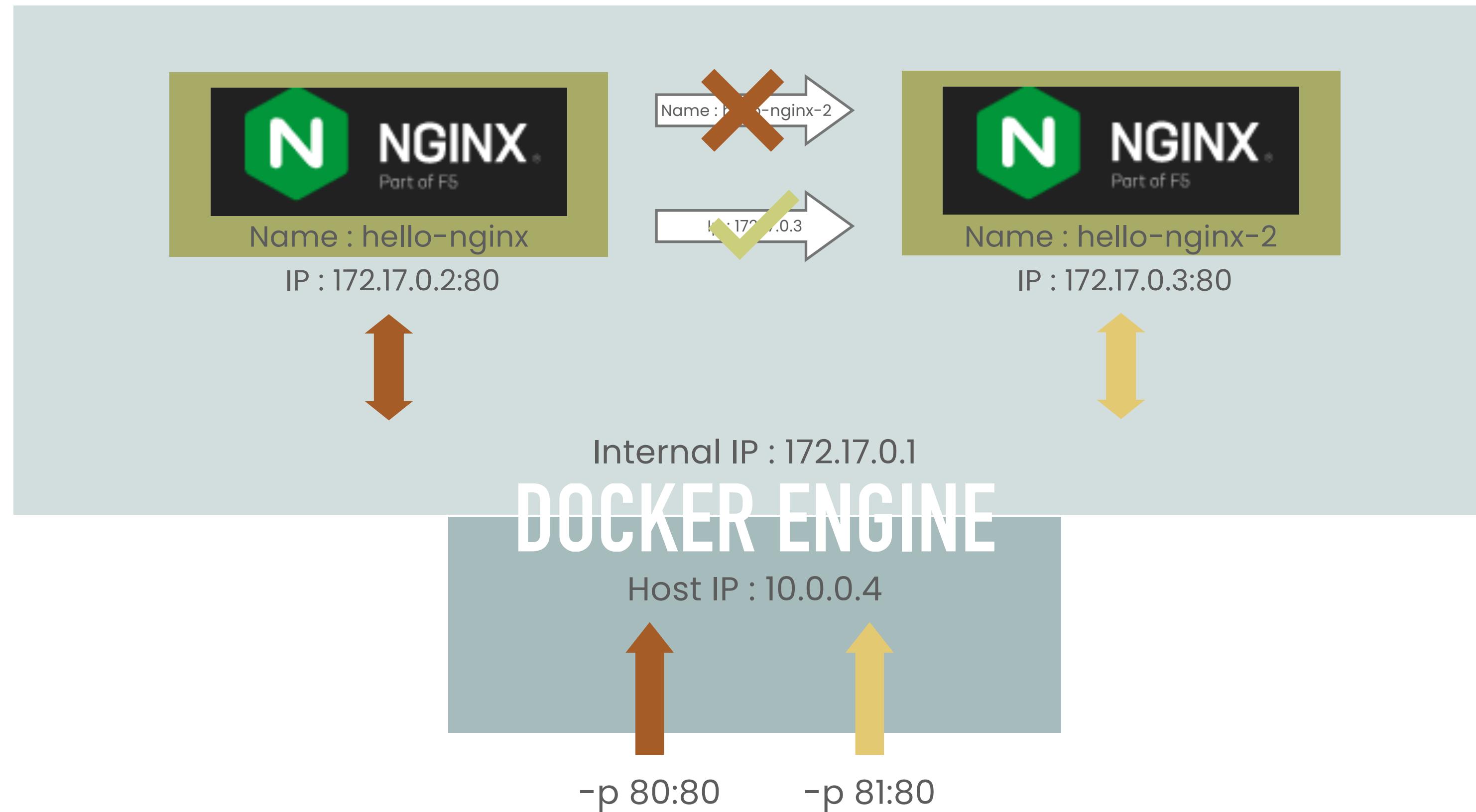
DOCKER - NO DNS FOR DEFAULT BRIDGE NETWORK

```
$ docker container exec -it hello-nginx curl 172.17.0.3
```



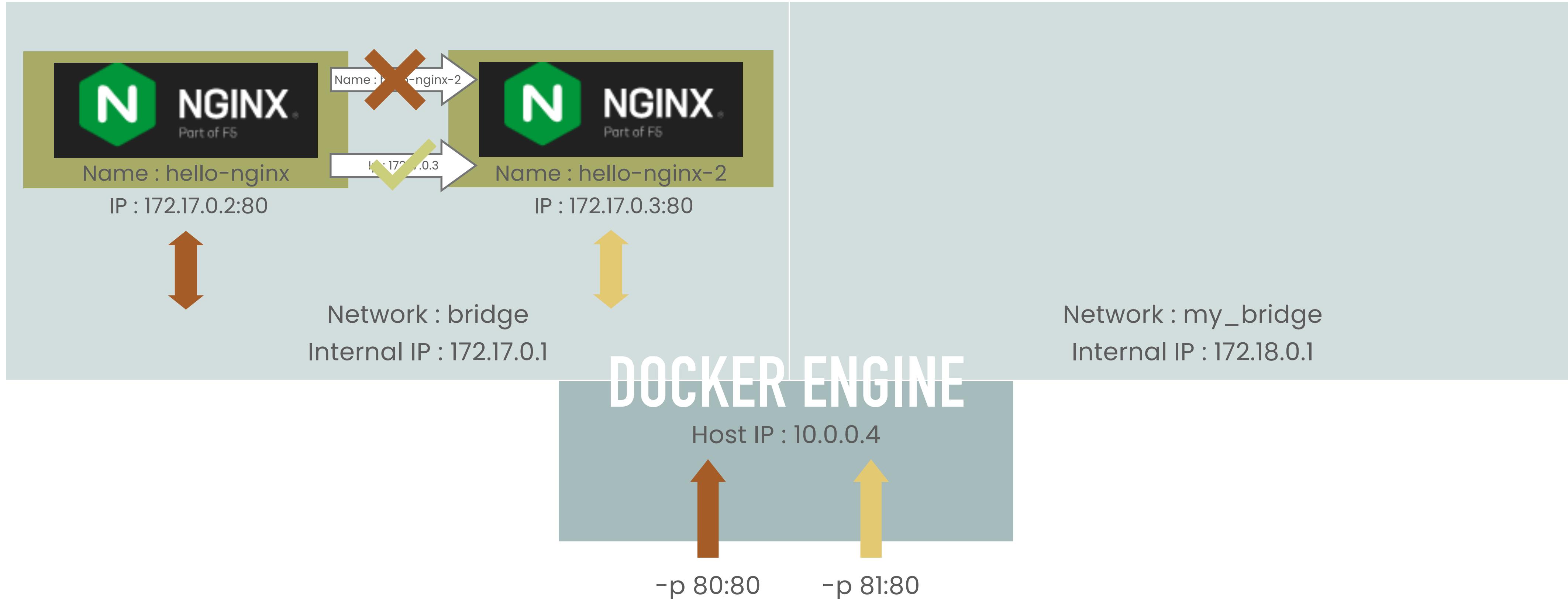
DOCKER - NO DNS FOR DEFAULT BRIDGE NETWORK

```
$ docker container exec -it hello-nginx curl 172.17.0.3 ✓
```



CREATE YOUR OWN (USER-DEFINED) BRIDGE NETWORK

```
$ docker network create -d bridge my_bridge
```



CREATE YOUR OWN (USER-DEFINED) BRIDGE NETWORK

CREATE YOUR OWN (USER-DEFINED) BRIDGE NETWORK

```
$ docker network ls
```

NETWORK ID	NAME	DRIVER	SCOPE
5c3bb2502623	bridge	bridge	local
6877ad312030	host	host	local
900ed84b8b07	my_bridge	bridge	local
d3d51f296ce4	none	null	local

CREATE YOUR OWN (USER-DEFINED) BRIDGE NETWORK

```
$ docker network inspect my_bridge
```

```
$ docker network ls
```

NETWORK ID	NAME	DRIVER	SCOPE
5c3bb2502623	bridge	bridge	local
6877ad312030	host	host	local
900ed84b8b07	my_bridge	bridge	local
d3d51f296ce4	none	null	local

```
C:\Users\karan>docker network inspect my_bridge
[
  {
    "Name": "my_bridge",
    "Id": "900ed84b8b071491d929553deaa44844d88a859c2e79f0b0e203bf33f3f26d6b",
    "Created": "2021-05-26T18:20:04.8395706Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "172.18.0.0/16",
          "Gateway": "172.18.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {},
    "Options": {},
    "Labels": {}
  }
]
```

DOCKER INTERNAL NETWORK

```
"Config": [
  {
    "Subnet": "172.18.0.0/16",
    "Gateway": "172.18.0.1"
  }
]
```

ADD CONTAINER TO USER-DEFINID BRIDGE NETWORK

ADD CONTAINER TO USER-DEFINID BRIDGE NETWORK

```
$ docker container run --net=my_bridge --name hello-nginx-3 -d -p 83:80 nginx
```

ADD CONTAINER TO USER-DEFINER BRIDGE NETWORK

```
$ docker container run --net=my_bridge --name hello-nginx-3 -d -p 83:80 nginx
```

```
$ docker container run --net=my_bridge --name hello-nginx-4 -d -p 84:80 nginx
```

ADD CONTAINER TO USER-DEFINID BRIDGE NETWORK

```
$ docker container run --net=my_bridge --name hello-nginx-3 -d -p 83:80 nginx
```

```
$ docker container run --net=my_bridge --name hello-nginx-4 -d -p 84:80 nginx
```

```
$ docker network inspect my_bridge
```

```
C:\Users\karan>docker network inspect my_bridge
...
...
...
{
  "ConfigOnly": false,
  "Containers": {
    "22da2499c01e42465eedef4b6fb12826e866845c2efaaa978040b3de8a7aa13d": {
      "Name": "hello-nginx-4",
      "EndpointID": "9d7bd857111806849d7730460ccfcffa700bfd1e79f06f87f625be45513432e",
      "MacAddress": "02:42:ac:12:00:03",
      "IPv4Address": "172.18.0.3/16",
      "IPv6Address": ""
    },
    "416bdb1eb345380ad67605d9c1b51c666110ef65edfe4be54d6ea91d32770687": {
      "Name": "hello-nginx-3",
      "EndpointID": "4b1b43a31cf13c3c655254569a8a754190297997352f550f606ae4e5e80c185b",
      "MacAddress": "02:42:ac:12:00:02",
      "IPv4Address": "172.18.0.2/16",
      "IPv6Address": ""
    }
  },
  "Options": {},
  "Labels": {}
}
```

ADD CONTAINER TO USER-DEFINED BRIDGE NETWORK

```
$ docker container run --net=my_bridge --name hello-nginx-3 -d -p 83:80 nginx
```

```
$ docker container run --net=my_bridge --name hello-nginx-4 -d -p 84:80 nginx
```

```
$ docker network inspect my_bridge
```

```
C:\Users\karan>docker network inspect my_bridge
...
...
...
{
  "ConfigOnly": false,
  "Containers": {
    "22da2499c01e42465eedef4b6fb12826e866845c2efaaa978040b3de8a7aa13d": {
      "Name": "hello-nginx-4",
      "EndpointID": "9d7bd857111806849d7730460ccfcffa700bfd1e79f06f87f625be45513432e",
      "MacAddress": "02:42:ac:12:00:03",
      "IPv4Address": "172.18.0.3/16",
      "IPv6Address": ""
    },
    "416bdb1eb345380ad67605d9c1b51c666110ef65edfe4be54d6ea91d32770687": {
      "Name": "hello-nginx-3",
      "EndpointID": "4b1b43a31cf13c3c655254569a8a754190297997352f550f606ae4e5e80c185b",
      "MacAddress": "02:42:ac:12:00:02",
      "IPv4Address": "172.18.0.2/16",
      "IPv6Address": ""
    }
  },
  "Options": {},
  "Labels": {}
}
```

ADD CONTAINER TO USER-DEFINED BRIDGE NETWORK

```
$ docker container run --net=my_bridge --name hello-nginx-3 -d -p 83:80 nginx
```

```
$ docker container run --net=my_bridge --name hello-nginx-4 -d -p 84:80 nginx
```

```
$ docker network inspect my_bridge
```

```
C:\Users\karan>docker network inspect my_bridge
...
...
...
{
  "ConfigOnly": false,
  "Containers": {
    "22da2499c01e42465eedef4b6fb12826e866845c2efaaa978040b3de8a7aa13d": {
      "Name": "hello-nginx-4",
      "EndpointID": "9d7bd857111806849d7730460ccfcffa700bfd1e79f06f87f625be45513432e",
      "MacAddress": "02:42:ac:12:00:03",
      "IPv4Address": "172.18.0.3/16",
      "IPv6Address": ""
    },
    "416bdb1eb345380ad67605d9c1b51c666110ef65edfe4be54d6ea91d32770687": {
      "Name": "hello-nginx-3",
      "EndpointID": "4b1b43a31cf13c3c655254569a8a754190297997352f550f606ae4e5e80c185b",
      "MacAddress": "02:42:ac:12:00:02",
      "IPv4Address": "172.18.0.2/16",
      "IPv6Address": ""
    }
  },
  "Options": {},
  "Labels": {}
}
```

ADD CONTAINER TO USER-DEFINED BRIDGE NETWORK

```
$ docker container run --net=my_bridge --name hello-nginx-3 -d -p 83:80 nginx
```

```
$ docker container run --net=my_bridge --name hello-nginx-4 -d -p 84:80 nginx
```

```
$ docker network inspect my_bridge
```

```
C:\Users\karan>docker network inspect my_bridge
...
...
...
{
  "ConfigOnly": false,
  "Containers": {
    "22da2499c01e42465eedef4b6fb12826e866845c2efaaa978040b3de8a7aa13d": {
      "Name": "hello-nginx-4",
      "EndpointID": "9d7bd857111806849d7730460ccfcffa700bfd1e79f06f87f625be45513432e",
      "MacAddress": "02:42:ac:12:00:03",
      "IPv4Address": "172.18.0.3/16",
      "IPv6Address": ""
    },
    "416bdb1eb345380ad67605d9c1b51c666110ef65edfe4be54d6ea91d32770687": {
      "Name": "hello-nginx-3",
      "EndpointID": "4b1b43a31cf13c3c655254569a8a754190297997352f550f606ae4e5e80c185b",
      "MacAddress": "02:42:ac:12:00:02",
      "IPv4Address": "172.18.0.2/16",
      "IPv6Address": ""
    }
  },
  "Options": {},
  "Labels": {}
}
```

```
$ docker network inspect bridge
```

```
C:\Users\karan>docker network inspect bridge
...
...
...
{
  "ConfigOnly": false,
  "Containers": {
    "8c6406e9cb0f8decc57e12c1875ca4cf9d2bc7de833b992e9150f9f555072cf": {
      "Name": "hello-nginx-2",
      "EndpointID": "5f4d9d9b35bab4f6b738881c0036efa91f40ad1ce56294b5a047c450e710cf98",
      "MacAddress": "02:42:ac:11:00:03",
      "IPv4Address": "172.17.0.3/16",
      "IPv6Address": ""
    },
    "bd087effleaf79f353c1448924bed9985438382ff822638dc125b8c7b0a3ea83": {
      "Name": "hello-nginx",
      "EndpointID": "306a62f3249ab8cfad056da39ccc81d93f328bf8151640b539e803a4ad32db6c",
      "MacAddress": "02:42:ac:11:00:02",
      "IPv4Address": "172.17.0.2/16",
      "IPv6Address": ""
    }
  },
  "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",
    ...
  }
}
```

ADD CONTAINER TO USER-DEFINED BRIDGE NETWORK

```
$ docker container run --net=my_bridge --name hello-nginx-3 -d -p 83:80 nginx
```

```
$ docker container run --net=my_bridge --name hello-nginx-4 -d -p 84:80 nginx
```

```
$ docker network inspect my_bridge
```

```
C:\Users\karan>docker network inspect my_bridge
...
...
...
{
  "ConfigOnly": false,
  "Containers": {
    "22da2499c01e42465eedef4b6fb12826e866845c2efaaa978040b3de8a7aa13d": {
      "Name": "hello-nginx-4",
      "EndpointID": "9d7bd857111806849d7730460ccfcffa700bfd1e79f06f87f625be45513432e",
      "MacAddress": "02:42:ac:12:00:03",
      "IPv4Address": "172.18.0.3/16",
      "IPv6Address": ""
    },
    "416bdb1eb345380ad67605d9c1b51c666110ef65edfe4be54d6ea91d32770687": {
      "Name": "hello-nginx-3",
      "EndpointID": "4b1b43a31cf13c3c655254569a8a754190297997352f550f606ae4e5e80c185b",
      "MacAddress": "02:42:ac:12:00:02",
      "IPv4Address": "172.18.0.2/16",
      "IPv6Address": ""
    }
  },
  "Options": {},
  "Labels": {}
}
```

```
$ docker network inspect bridge
```

```
C:\Users\karan>docker network inspect bridge
...
...
...
{
  "ConfigOnly": false,
  "Containers": {
    "8c6406e9cb0f8decc57e12c1875ca4cf9d2bc7de833b992e9150f9f555072cf": {
      "Name": "hello-nginx-2",
      "EndpointID": "5f4d9d9b35bab4f6b738881c0036efa91f40ad1ce56294b5a047c450e710cf98",
      "MacAddress": "02:42:ac:11:00:03",
      "IPv4Address": "172.17.0.3/16",
      "IPv6Address": ""
    },
    "bd087effleaf79f353c1448924bed9985438382ff822638dc125b8c7b0a3ea83": {
      "Name": "hello-nginx",
      "EndpointID": "306a62f3249ab8cfad056da39ccc81d93f328bf8151640b539e803a4ad32db6c",
      "MacAddress": "02:42:ac:11:00:02",
      "IPv4Address": "172.17.0.2/16",
      "IPv6Address": ""
    }
  },
  "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",
    ...
  }
}
```

ADD CONTAINER TO USER-DEFINED BRIDGE NETWORK

```
$ docker container run --net=my_bridge --name hello-nginx-3 -d -p 83:80 nginx
```

```
$ docker container run --net=my_bridge --name hello-nginx-4 -d -p 84:80 nginx
```

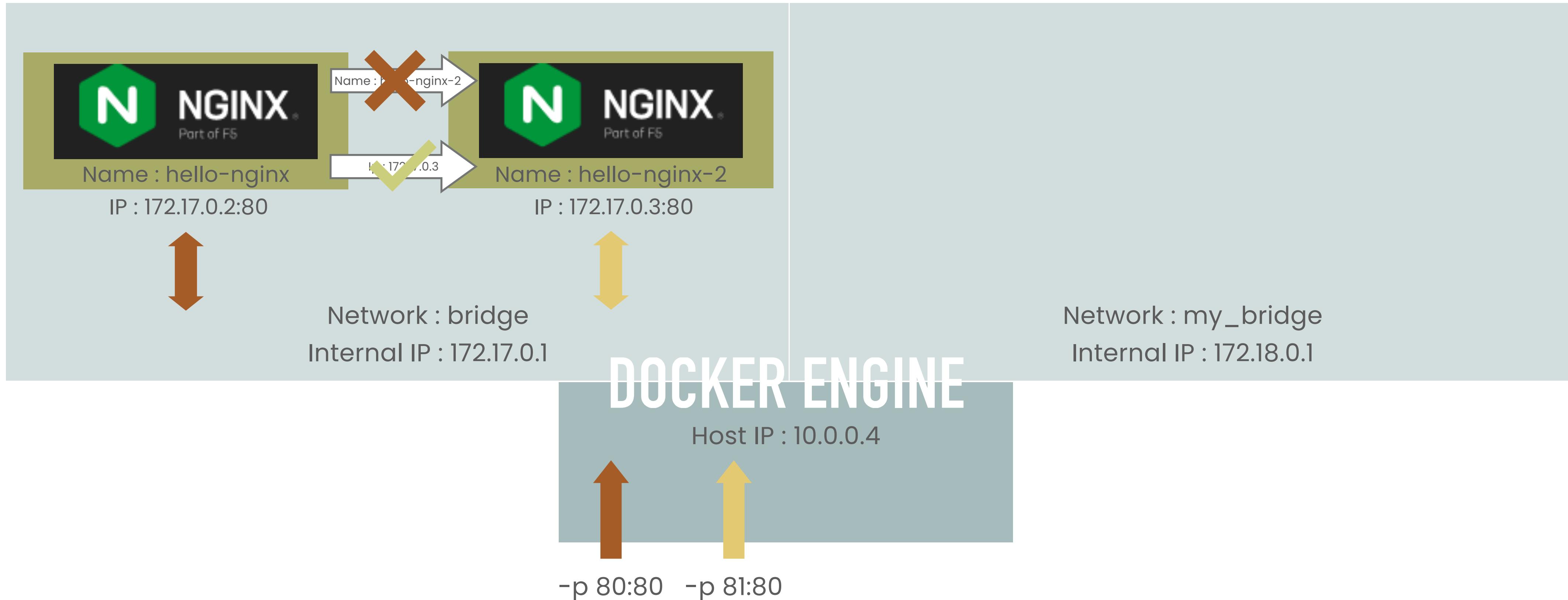
```
$ docker network inspect my_bridge
```

```
C:\Users\karan>docker network inspect my_bridge
...
...
...
{
  "ConfigOnly": false,
  "Containers": {
    "22da2499c01e42465eedef4b6fb12826e866845c2efaaa978040b3de8a7aa13d": {
      "Name": "hello-nginx-4",
      "EndpointID": "9d7bd857111806849d7730460ccfcffa700bfd1e79f06f87f625be45513432e",
      "MacAddress": "02:42:ac:12:00:03",
      "IPv4Address": "172.18.0.3/16",
      "IPv6Address": ""
    },
    "416bdb1eb345380ad67605d9c1b51c666110ef65edfe4be54d6ea91d32770687": {
      "Name": "hello-nginx-3",
      "EndpointID": "4b1b43a31cf13c3c655254569a8a754190297997352f550f606ae4e5e80c185b",
      "MacAddress": "02:42:ac:12:00:02",
      "IPv4Address": "172.18.0.2/16",
      "IPv6Address": ""
    }
  },
  "Options": {},
  "Labels": {}
}
```

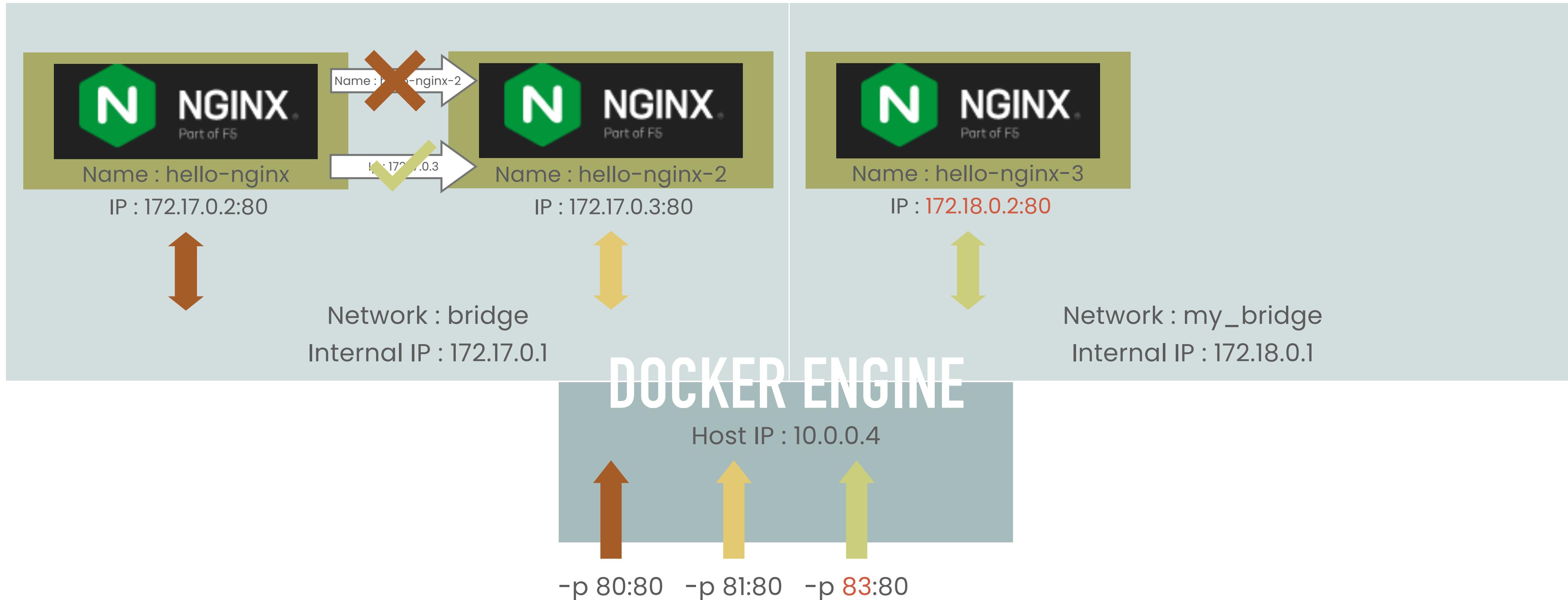
```
$ docker network inspect bridge
```

```
C:\Users\karan>docker network inspect bridge
...
...
...
{
  "ConfigOnly": false,
  "Containers": {
    "8c6406e9cb0f8decc57e12c1875ca4cf9d2bc7de833b992e9150f9f555072cf": {
      "Name": "hello-nginx-2",
      "EndpointID": "5f4d9d9b35bab4f6b738881c0036efa91f40ad1ce56294b5a047c450e710cf98",
      "MacAddress": "02:42:ac:11:00:03",
      "IPv4Address": "172.17.0.3/16",
      "IPv6Address": ""
    },
    "bd087effleaf79f353c1448924bed9985438382ff822638dc125b8c7b0a3ea83": {
      "Name": "hello-nginx",
      "EndpointID": "306a62f3249ab8cfad056dd39ccc81d93f328bf8151640b539e803a4ad32db6c",
      "MacAddress": "02:42:ac:11:00:02",
      "IPv4Address": "172.17.0.2/16",
      "IPv6Address": ""
    }
  },
  "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",
    ...
  }
}
```

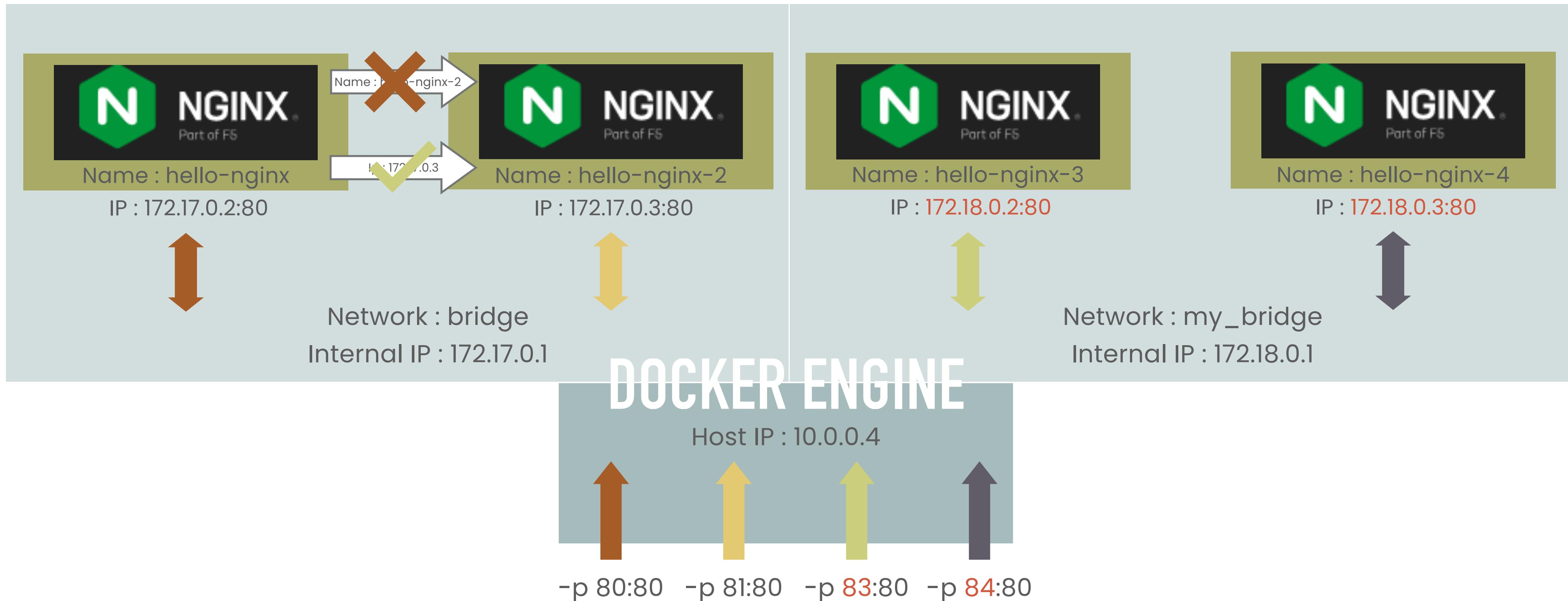
ADD CONTAINER TO USER-DEFINED BRIDGE NETWORK



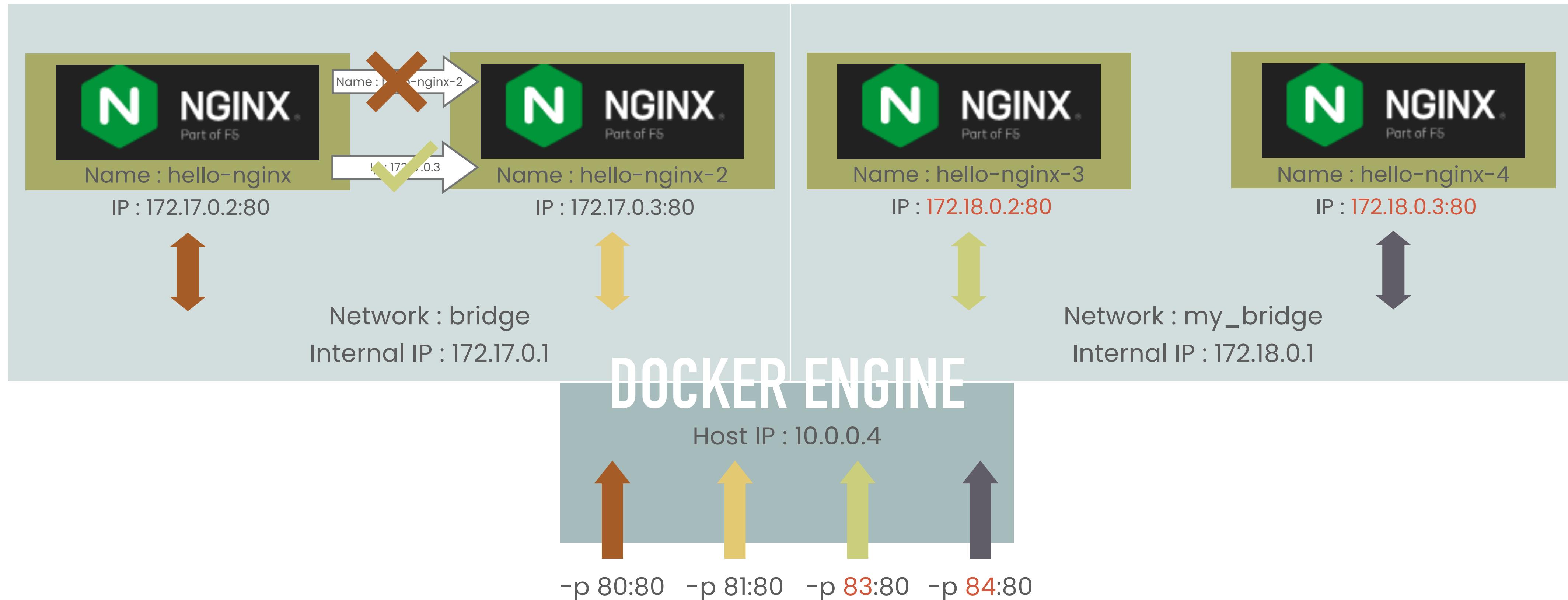
ADD CONTAINER TO USER-DEFINED BRIDGE NETWORK



ADD CONTAINER TO USER-DEFINED BRIDGE NETWORK

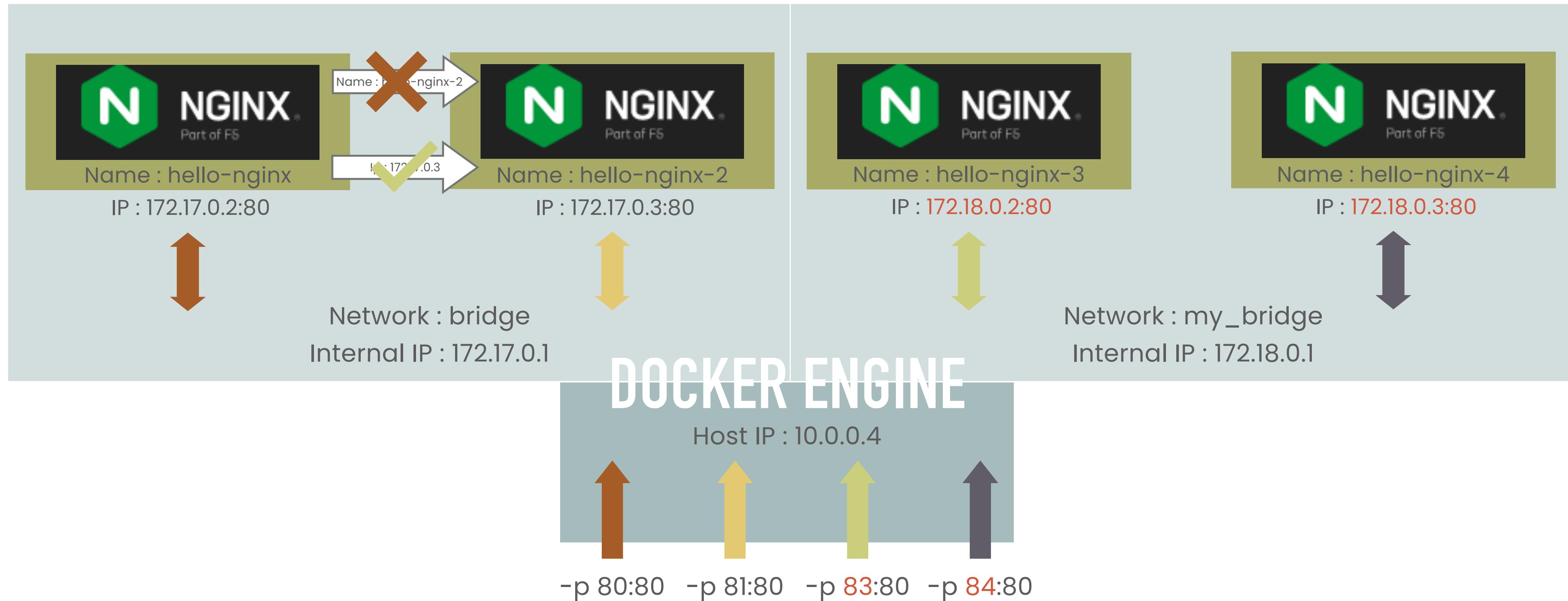


DNS IN USER-DEFINED BRIDGE NETWORK



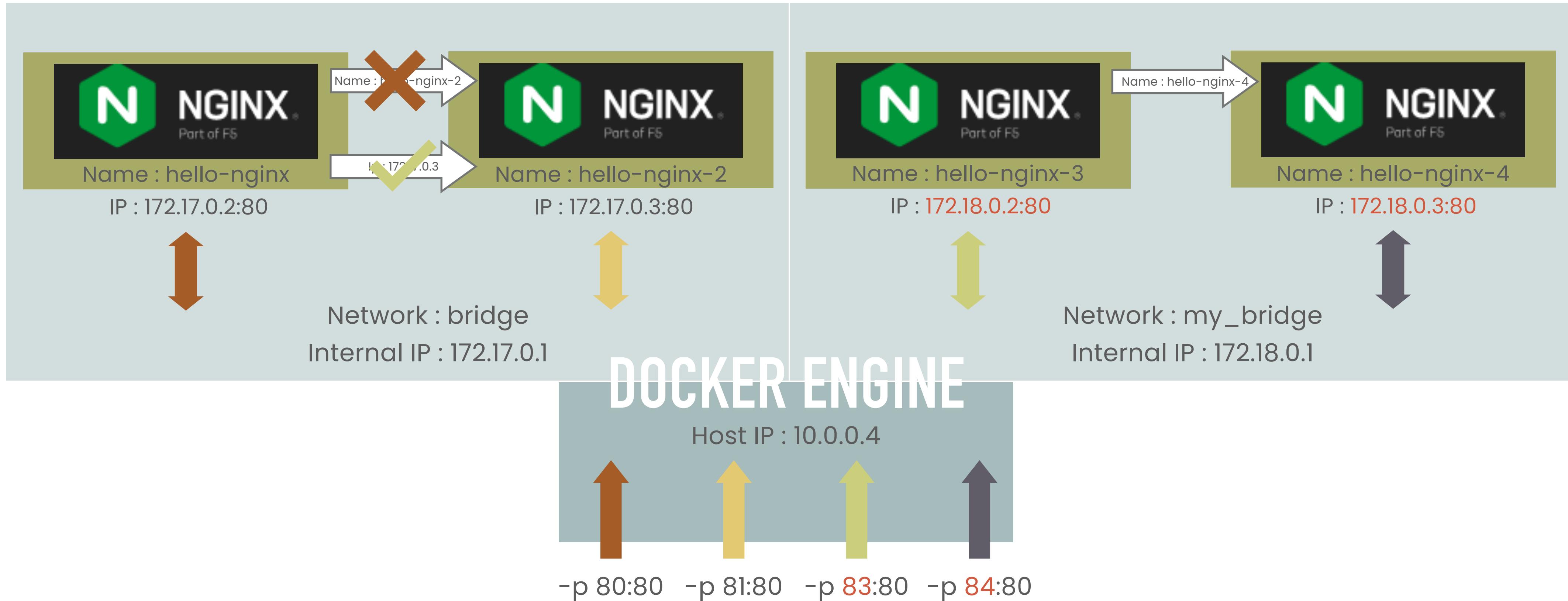
DNS IN USER-DEFINED BRIDGE NETWORK

```
$ docker container exec -it hello-nginx-3 curl hello-nginx-4
```



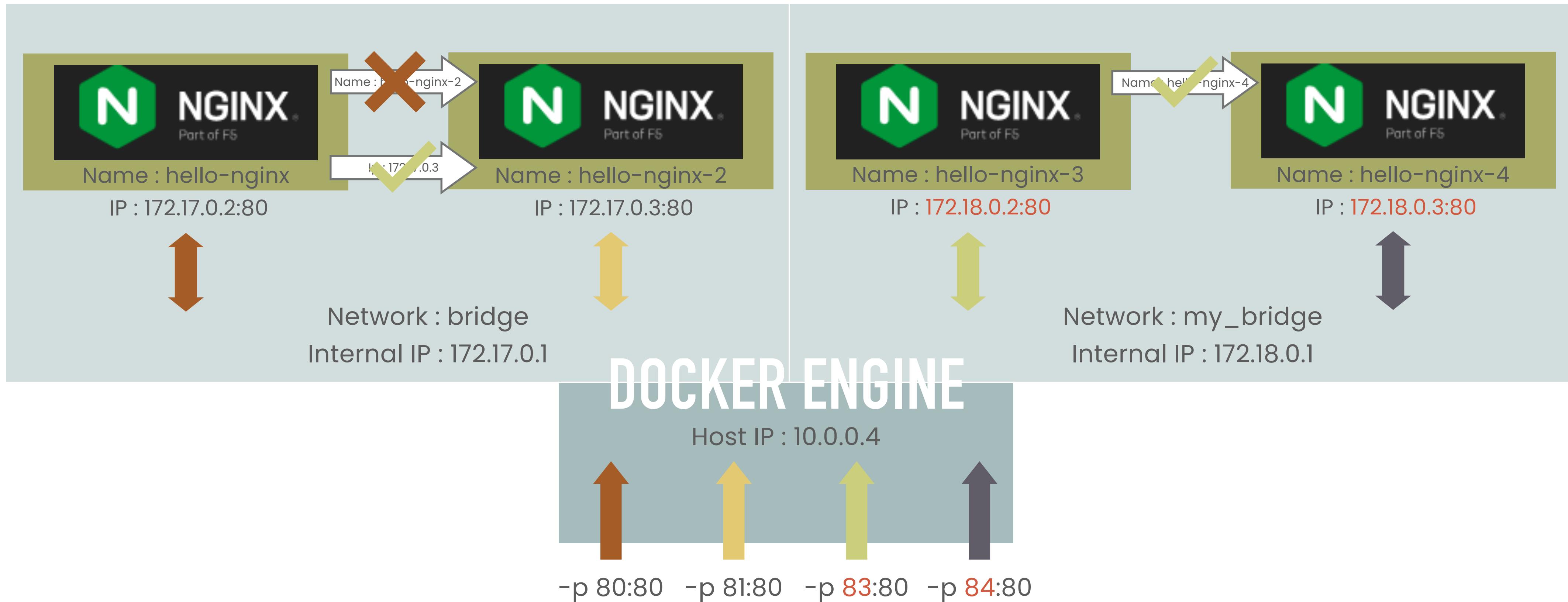
DNS IN USER-DEFINED BRIDGE NETWORK

```
$ docker container exec -it hello-nginx-3 curl hello-nginx-4
```



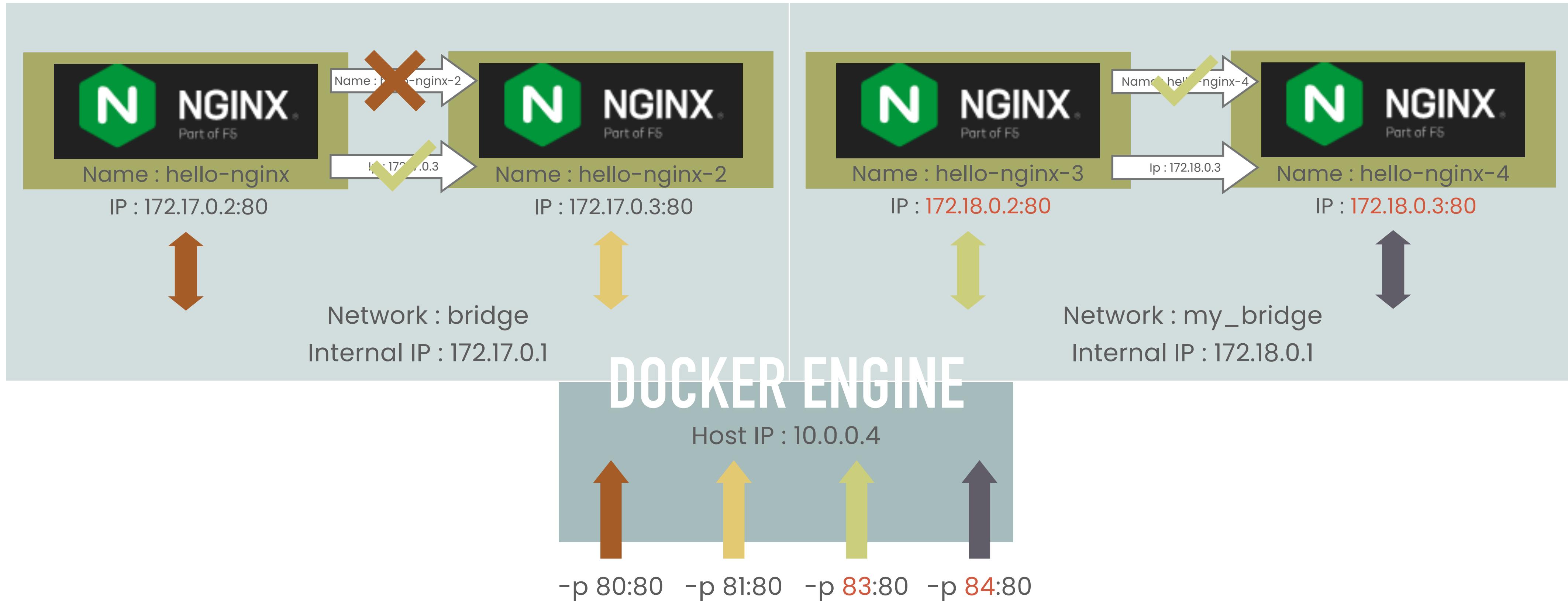
DNS IN USER-DEFINED BRIDGE NETWORK

```
$ docker container exec -it hello-nginx-3 curl hello-nginx-4
```



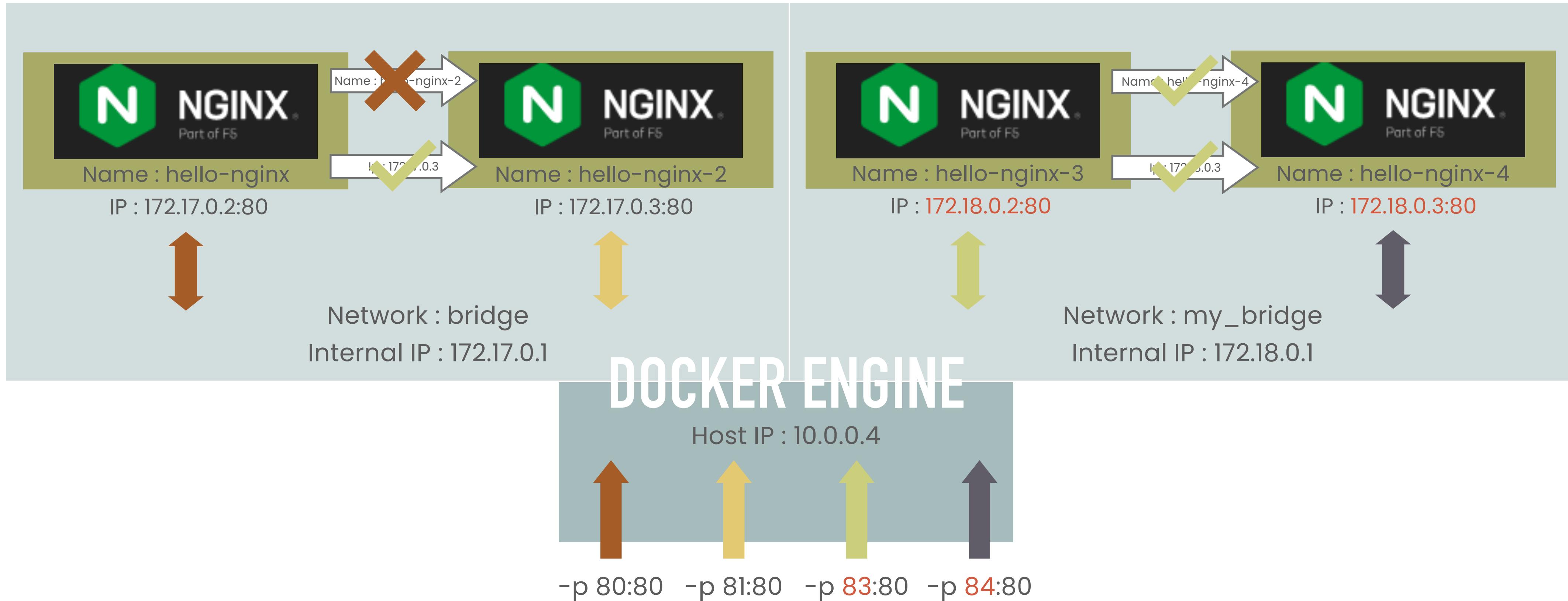
DNS IN USER-DEFINED BRIDGE NETWORK

```
$ docker container exec -it hello-nginx-3 curl hello-nginx-4
```

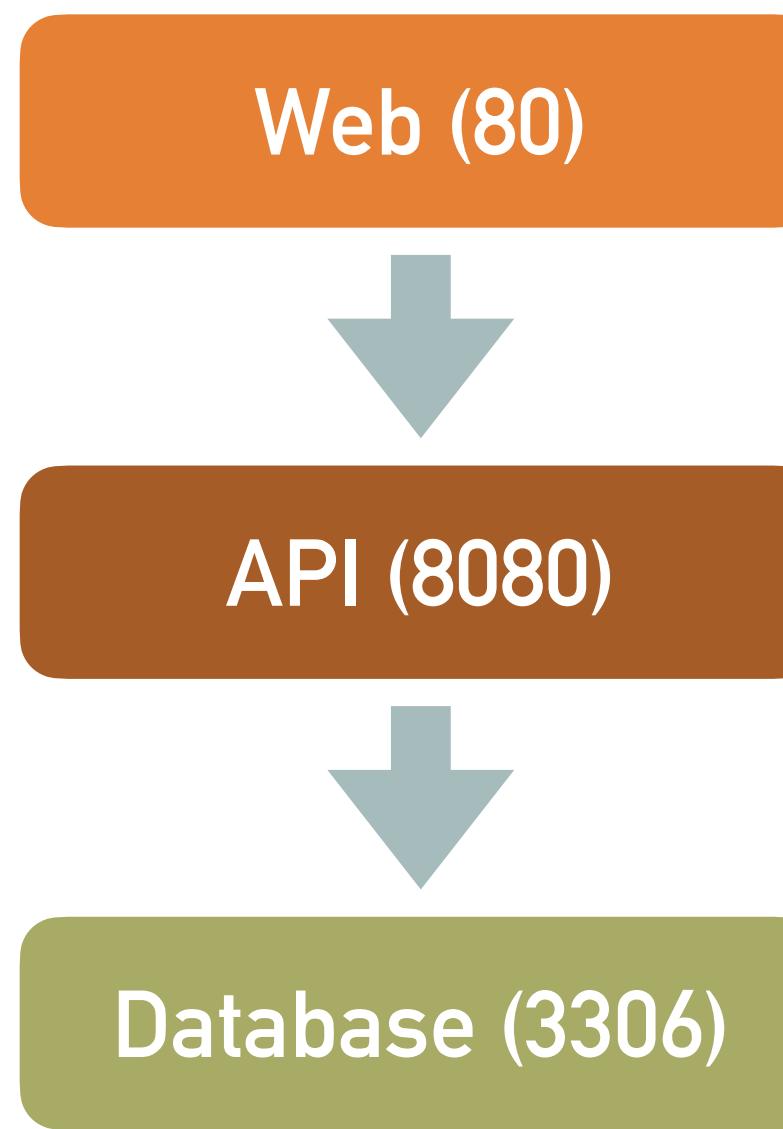


DNS IN USER-DEFINED BRIDGE NETWORK

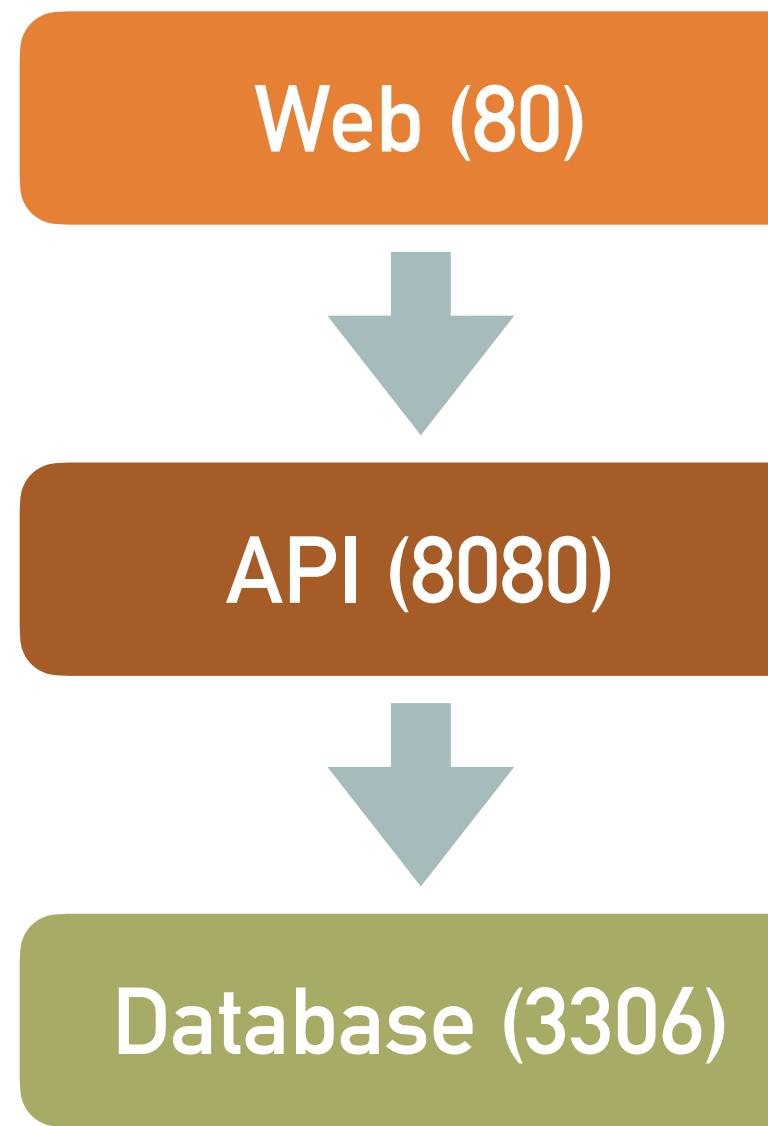
```
$ docker container exec -it hello-nginx-3 curl hello-nginx-4
```



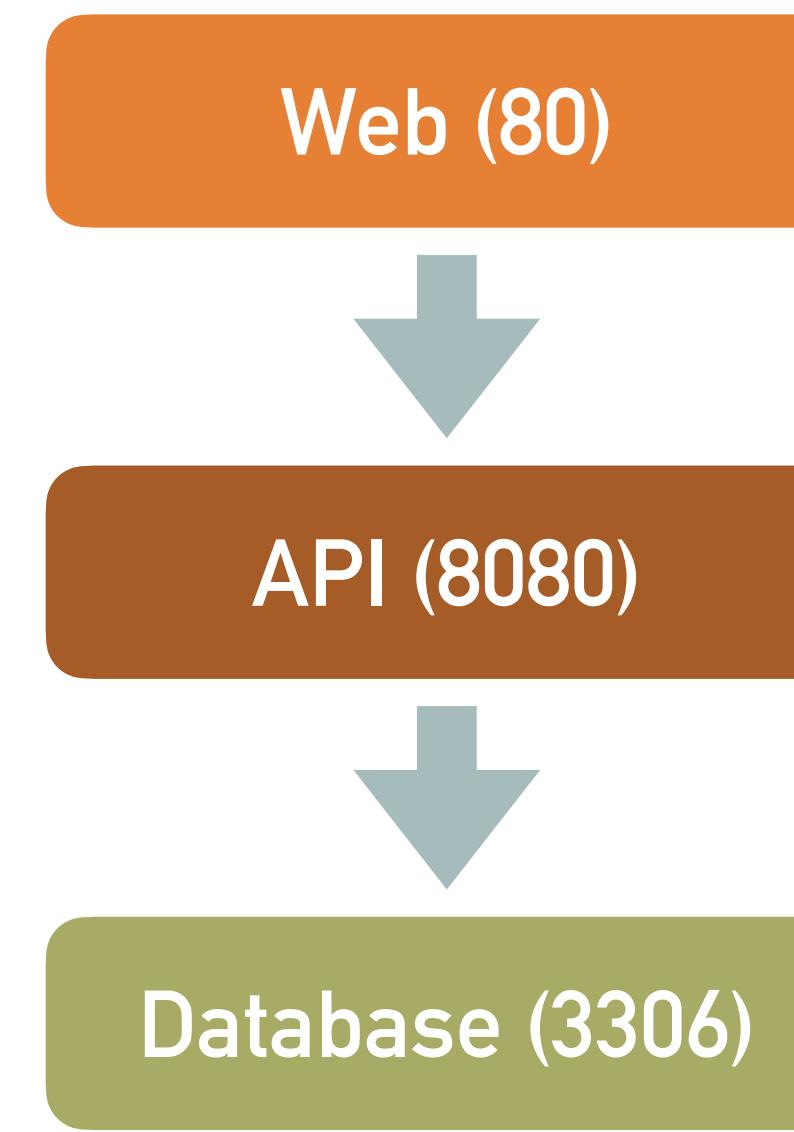
EXERCISE : HOW CAN I SETUP THIS ARCHITECTURE IN DOCKER ?



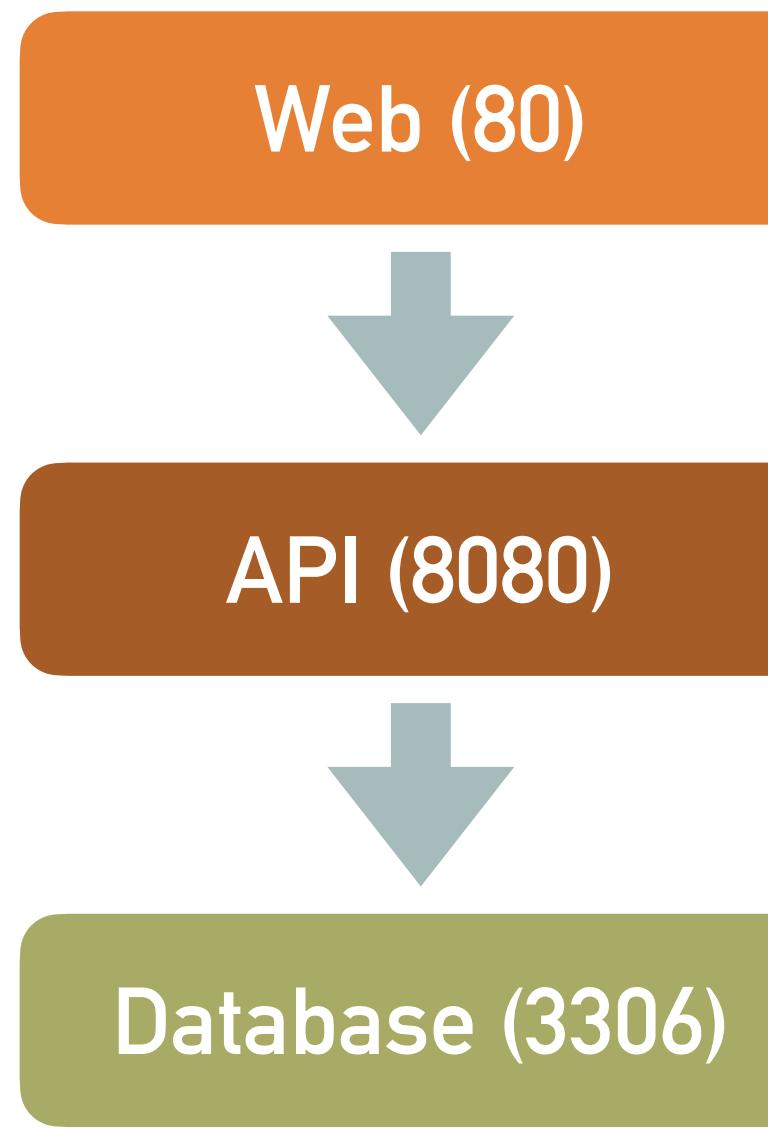
EXERCISE : HOW CAN I SETUP THIS ARCHITECTURE IN DOCKER ?



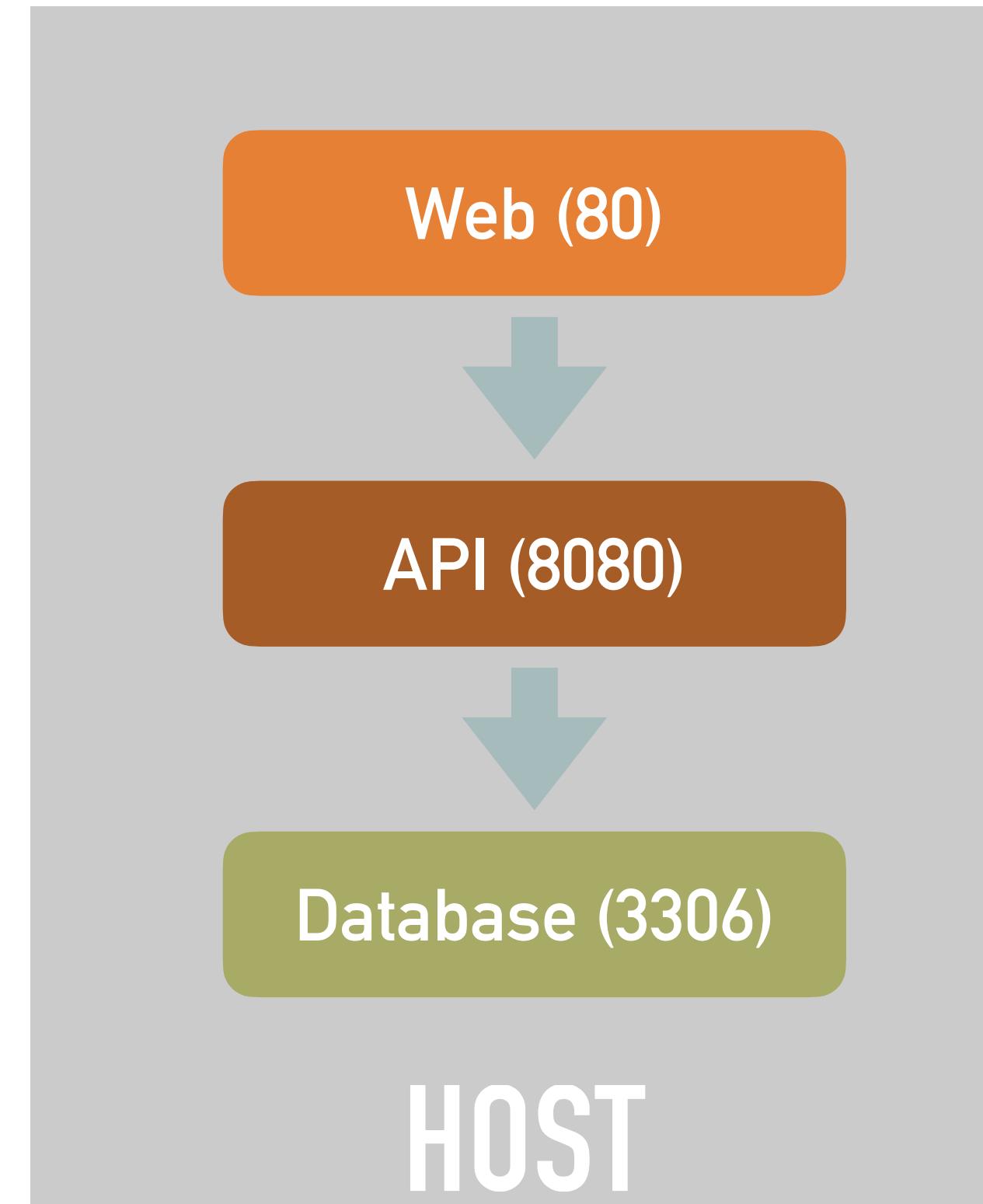
EXERCISE : HOW CAN I SETUP THIS ARCHITECTURE IN DOCKER ?



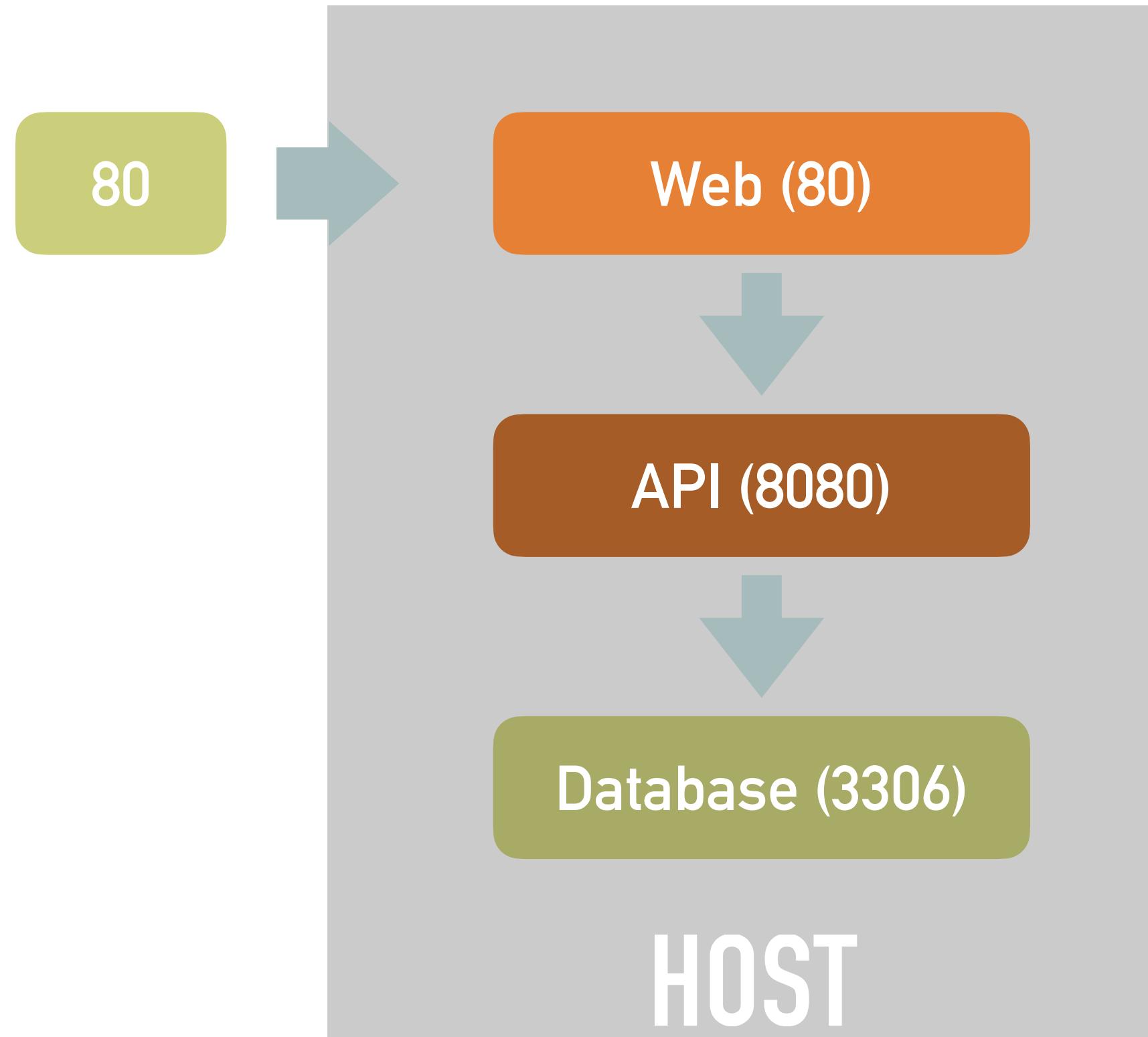
EXERCISE : HOW CAN I SETUP THIS ARCHITECTURE IN DOCKER ?



EXERCISE : HOW CAN I SETUP THIS ARCHITECTURE IN DOCKER ?



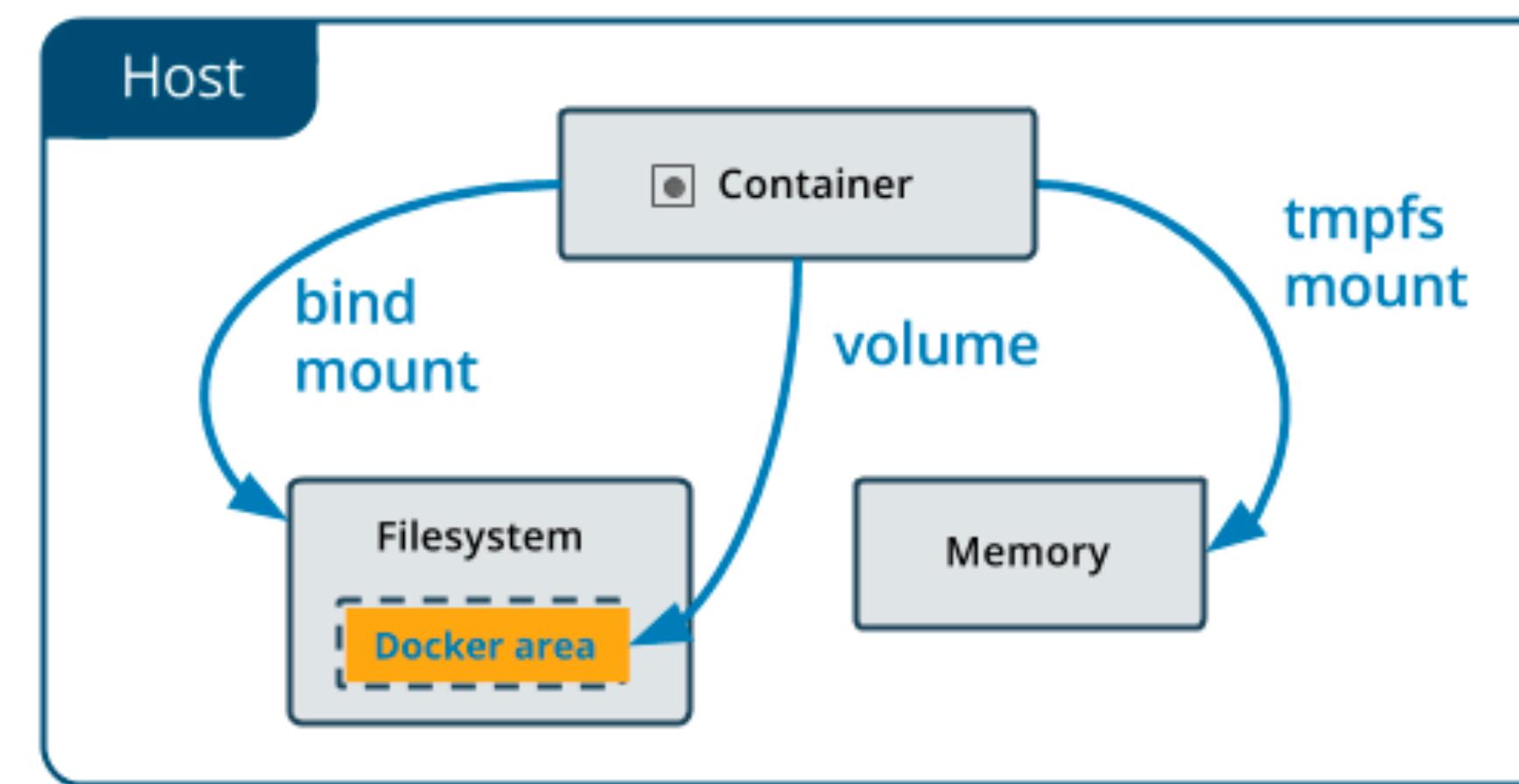
EXERCISE : HOW CAN I SETUP THIS ARCHITECTURE IN DOCKER ?



WORKING WITH VOLUMES

DOCKER VOLUMES

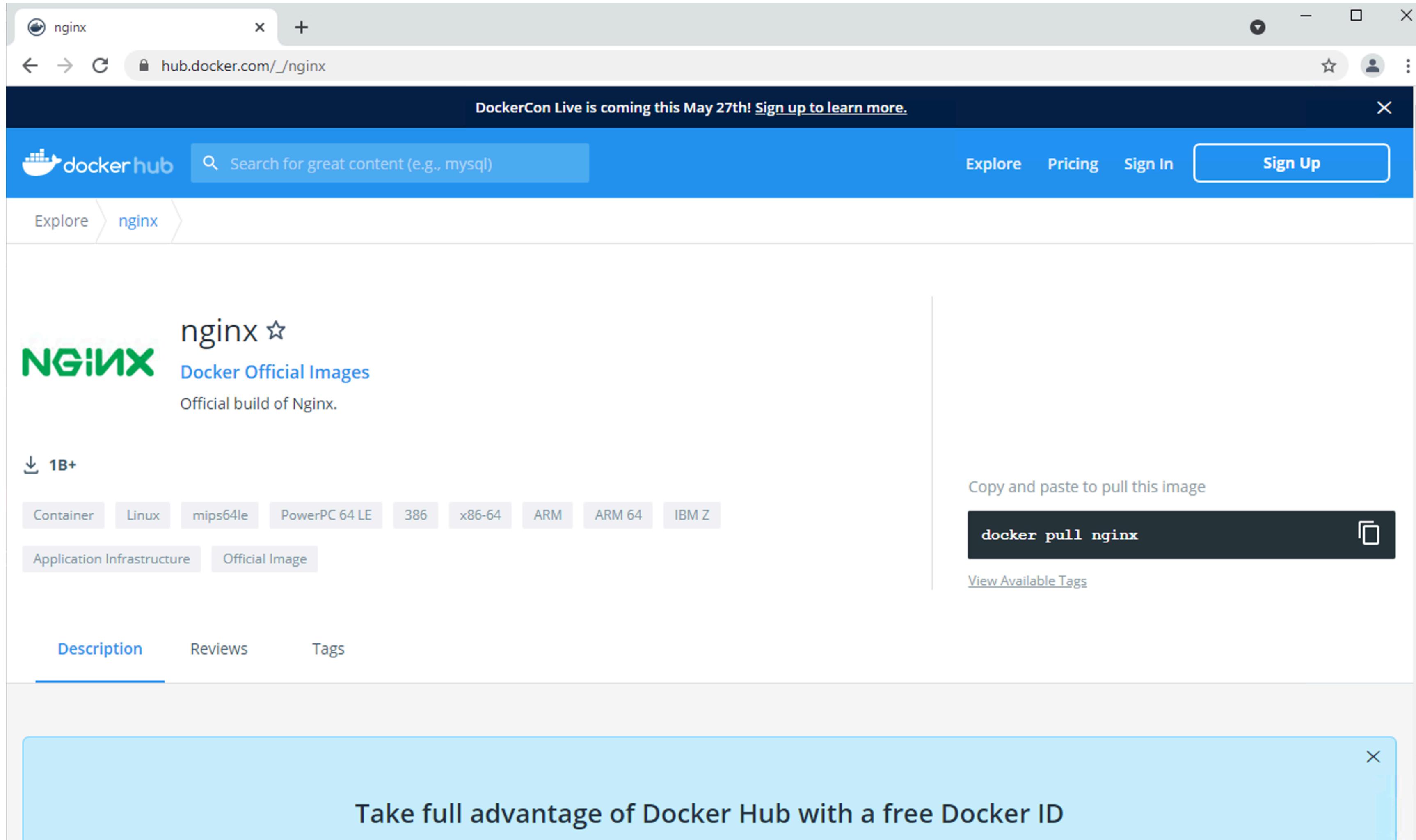
Volumes are the preferred mechanism **for persisting data generated by and used by Docker containers**. While **bind mounts** are **dependent on** the directory structure and OS of the **host machine**, volumes are completely managed by Docker.



ADVANTAGES OF VOLUMES

- Volumes are easier to back up or migrate than bind mounts.
- You can manage volumes using Docker CLI commands or the Docker API.
- Volumes work on both Linux and Windows containers.
- Volumes can be more safely shared among multiple containers.
- Volume drivers let you store volumes on remote hosts or cloud providers, to encrypt the contents of volumes, or to add other functionality.
- New volumes can have their content pre-populated by a container.
- Volumes on Docker Desktop have much higher performance than bind mounts from Mac and Windows hosts.

NGINX WITH VOLUMES



HOW TO USE NGINX DOCKER IMAGE



How to use this image

Hosting some simple static content

```
$ docker run --name some-nginx -v /some/content:/usr/share/nginx/html:ro -d nginx
```

Alternatively, a simple `Dockerfile` can be used to generate a new image that includes the necessary content (which is a much cleaner solution than the bind mount above):

```
FROM nginx
COPY static-html-directory /usr/share/nginx/html
```

Place this file in the same directory as your directory of content ("static-html-directory"), run `docker build -t some-content-nginx .`, then start your container:

```
$ docker run --name some-nginx -d some-content-nginx
```

Exposing external port

```
$ docker run --name some-nginx -d -p 8080:80 some-content-nginx
```

Then you can hit `http://localhost:8080` or `http://host-ip:8080` in your browser.

VOLUME - HOST AND CONTAINER

VOLUME - HOST AND CONTAINER



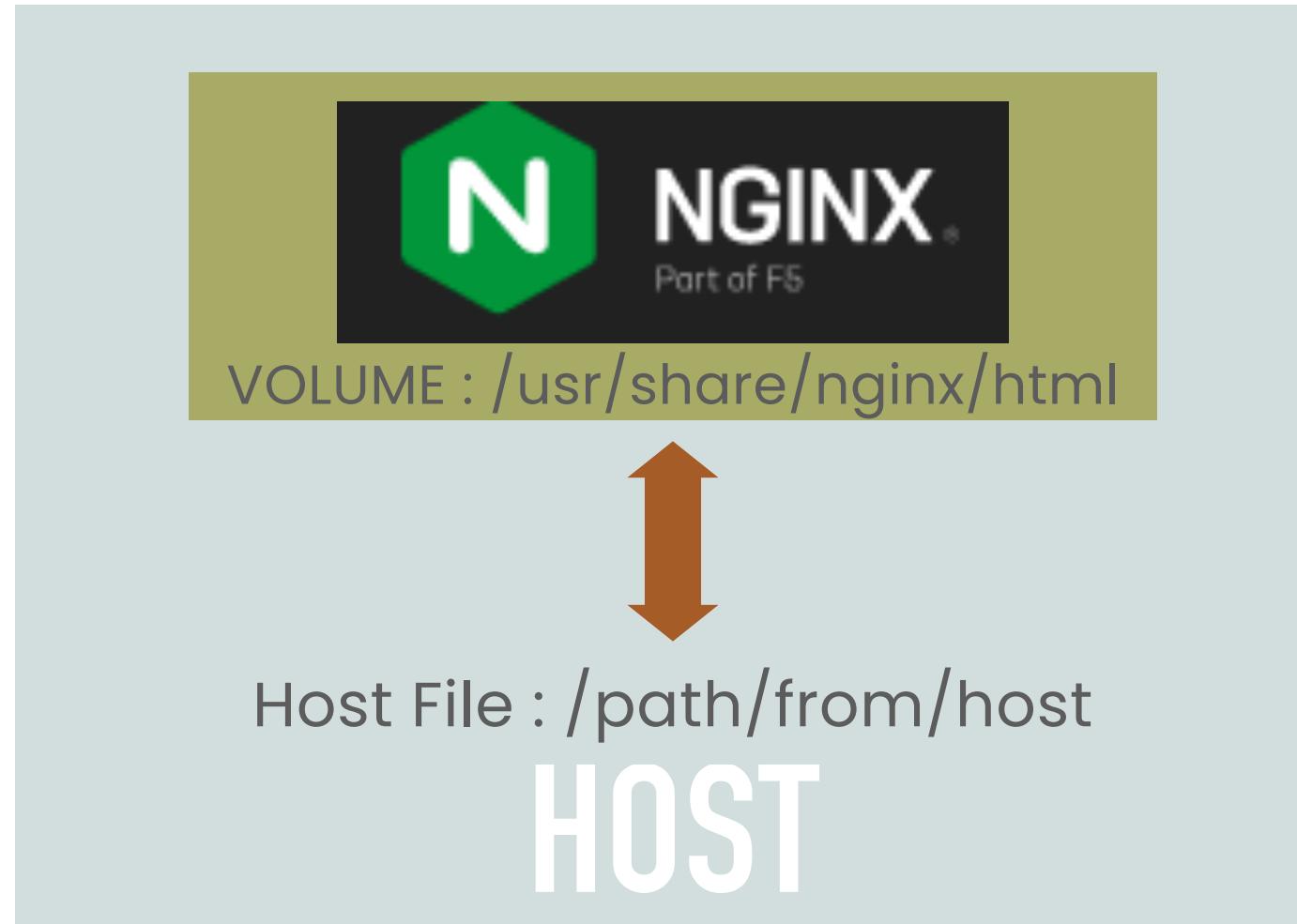
VOLUME - HOST AND CONTAINER



VOLUME - HOST AND CONTAINER



VOLUME - HOST AND CONTAINER



CREATE STATIC WEBSITE FOR NGINX

CREATE STATIC WEBSITE FOR NGINX

```
$ mkdir web
```

CREATE STATIC WEBSITE FOR NGINX

```
$ mkdir web
```

```
$ docker container run --name hello-nginx -d -p 80:80  
  -v ${PWD}/web:/usr/share/nginx/html  
  nginx
```

CREATE STATIC WEBSITE FOR NGINX

```
$ mkdir web
```

```
$ docker container run --name hello-nginx -d -p 80:80  
    -v ${PWD}/web:/usr/share/nginx/html  
nginx
```

Only for Powershell

CREATE STATIC WEBSITE FOR NGINX

```
$ mkdir web
```

```
$ docker container run --name hello-nginx -d -p 80:80  
  -v ${PWD}/web:/usr/share/nginx/html  
nginx
```

Only for Powershell

CREATE STATIC WEBSITE FOR NGINX

```
$ mkdir web
```

```
$ docker container run --name hello-nginx -d -p 80:80  
-v ${PWD}/web:/usr/share/nginx/html  
nginx
```

Only for Powershell

Host File/Directory

CREATE STATIC WEBSITE FOR NGINX

```
$ mkdir web
```

```
$ docker container run --name hello-nginx -d -p 80:80  
-v ${PWD}/web:/usr/share/nginx/html  
nginx
```

Only for Powershell

Host File/Directory

CREATE STATIC WEBSITE FOR NGINX

```
$ mkdir web
```

```
$ docker container run --name hello-nginx -d -p 80:80  
-v ${PWD}/web:/usr/share/nginx/html  
nginx
```

Only for Powershell

Host File/Directory

Container
File/Directory

CREATE STATIC WEBSITE FOR NGINX

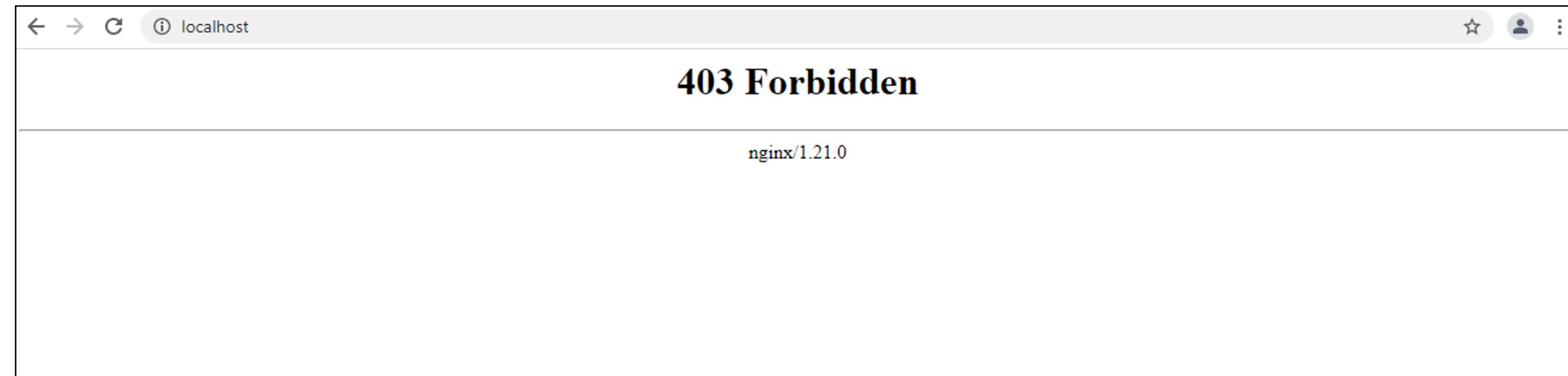
```
$ mkdir web
```

```
$ docker container run --name hello-nginx -d -p 80:80  
-v ${PWD}/web:/usr/share/nginx/html  
nginx
```

Only for Powershell

Host File/Directory

Container
File/Directory



CREATE STATIC WEBSITE FOR NGINX

CREATE STATIC WEBSITE FOR NGINX

```
$cd web  
$echo "Hello World" > index.html
```

CREATE STATIC WEBSITE FOR NGINX

```
$cd web  
$echo "Hello World" > index.html
```



THANK YOU