Docker networking Example

08:55 AM

Networking using the host network

Estimated reading time: 2 minutes

This series of tutorials deals with networking standalone containers which bind directly to the Docker host's network, with no network isolation. For other networking topics, see the overview.

Goal

The goal of this tutorial is to start a nginx container which binds directly to port 80 on the Docker host. From a networking point of view, this is the same level of isolation as if the nginx process were running directly on the Docker host and not in a container. However, in all other ways, such as storage, process namespace, and user namespace, the nginx process is isolated from the host.

Prerequisites

- This procedure requires port 80 to be available on the Docker host. To make Nginx listen on a different port, see the documentation for the nginx image
- The host networking driver only works on Linux hosts, and is not supported on Docker Desktop for Mac, Docker Desktop for Windows, or Docker EE for Windows Server.

Procedure

1. Create and start the container as a detached process. The --rm option means to remove the container once it exits/stops. The -d flag means to start the container detached (in the background).
   $docker run --rm-d--networkhost --namemy_nginx nginx

2. Access Nginx by browsing to http://localhost:80/.
3. Examine your network stack using the following commands:
   - Examine all network interfaces and verify that a new one was not created.
     $ip addr show
   - Verify which process is bound to port 80, using the netstat command. You need to use sudo because the process is owned by the Docker daemon user and you otherwise won't be able to see its name or PID.

```
$sudo netstat -tulpn| grep:80
```

4. Stop the container. It will be removed automatically as it was started using the `--rm` option.

   ```
   docker container stop my_nginx
   ```

https://docs.docker.com/network/network-tutorial-standalone/