CO515 Advanced Computer Networks: Selected Topics
Docker Networking Lab 01

Docker

Docker, is a software framework for building, running, and managing containers on servers and the cloud

Docker Networking

For Docker containers to communicate with each other and the outside world via the host machine, there has to be a layer of networking involved. Docker supports different types of networks, each fit for certain use cases.

For example, building an application which runs on a single Docker container will have a different network setup as compared to a web application with a cluster with database, application and load balancers which span multiple containers that need to communicate with each other. Additionally, clients from the outside world will need to access the web application container.

Getting Started with the Lab

Setup Docker environment in your PCs. Refer the following link for Docker official site, https://www.docker.com/products/docker-desktop

Lab Task 01

1. **Getting Familiar with basic docker networking commands**
   - 1.1. Run docker network command in the first terminal. Observe the output.
   - 1.2. Use subcommand on docker network command from the list of commands to
     - 1.2.1. Obtain the list of available networks
     - 1.2.2. Obtain the configuration detail of the available networks. List the details which can be obtained using that command.
   - 1.3. Use docker info command and observe the output.

Bridge Networking

Docker has different networking drivers:

- bridge: the default network driver,
- host: for standalone containers, ● overlay: used in Docker swarm,
- macvlan: works on MAC addresses assigned to containers,
- none: networking is disabled,
- Network plugins: Third-party network plugins with Docker.

Here in this lab we will mainly focus on bridge networking driver since it allows us to connect two or more containers together and it gives us the most flexibility/functionality in terms of networking.

All networks created with the bridge driver are based on a Linux bridge (a virtual switch)

Lab Task 02

1. **Every clean installation of Docker comes with a pre-built network called bridge.**
   - 1.1. You can verify this with the docker network ls command.
   - 1.2. Install the brctl command and use it to list the Linux bridges on your Docker host. You can do this by running sudo apt-get install bridge-utils
   - 1.3. Use brctl show command to list the bridges on your Docker host. The output above shows a single Linux bridge called docker0. This is the bridge that was automatically created for the bridge network. You can see that it has no interfaces currently connected to it.
   - 1.4. use the ip a command to view details of the docker0 bridge.

2. **Connecting a Container**

   The bridge network is the default network for new containers. This means that unless you specify a different network, all new containers will be connected to the bridge network.
   - 2.1. Create a new container by running docker run -dt ubuntu sleep infinity
   - 2.2. Verify that the new container is up by running docker ps
   - 2.3. As no network was specified on the docker run command, the container will be added to the bridge network. Run the brctl show command again.
   - 2.4. Notice how the docker0 bridge now has an interface connected.Inspect the bridge network again using appropriate command. What is the ip address of the new container?

3. Test network connectivity

3.1. Ping the IP address of the container from the shell prompt of your Docker host by running ping -c5 <IPv4 Address> .Observe the replies.

3.2. Verify that the container can connect to the outside world by ping to www.google.com.

3.2.1. Get the ID of the container using docker ps

3.2.2. Run a shell inside that ubuntu container, by running docker exec -it <CONTAINER ID> /bin/bash

3.2.3. Run apt-get update && apt-get install -y iputils-ping to install the ping program

3.2.4. ping www.google.com by running ping -c5 www.google.com

3.2.5. Run exit to disconnect the shell from the container

3.2.6. Stop this container by cleaning things up from this test, by running docker stop <CONTAINER ID>

4. Configure NAT for external connectivity

In this step we'll start a new NGINX container and map port 8080 on the Docker host to port 80 inside of the container. This means that traffic that hits the Docker host on port 8080 will be passed on to port 80 inside the container. NOTE: If you start a new container from the official NGINX image without specifying a command to run, the container will run a basic web server on port 80.

4.1. Start a new container based off the official NGINX image by running docker run --name web1 -d -p 8080:80 nginx

4.2. Review the container status and port mappings by running docker ps

Submission

Submit a report named Lab01_EYYXXX.pdf (XXX is your E Number) with the screenshots and explanations necessary for all the questions in the lab tasks