(Senior Academy - IT training center)

The Place You Can Be A Senior

www.seniorsteps.net
https://www.facebook.com/seniorsteps.it
contact us: 0224153419 - 01090873748

فرع مدينة نصر 1 : عمارة 4 – شارع محمد توفيق دياب – عباس العقاد - مدينة نصر – الدورال 1

DevOps Engineer Diploma

DevOps Engineer Diploma

Containers With Docker

Lab 01

Docker Network Configuration and Service Isolation

### Lab Objectives

- Custom Bridge Network Creation
- IP Address Allocation and Management
- Service Discovery and Internal Connectivity
- Multi-Network Container Setup
- Network Isolation and Security Validation

Tasks

To establish a dedicated and isolated **Custom Bridge Network** for a **Human Resources (HR) application stack** running on a single **Docker host**. The primary goal is to ensure **Service Discovery** via **container names** and prevent **IP Address conflicts** with the corporate **VPN network**, which utilizes the **172.17.0.0/16** range.

⍰ **Requirements (Prerequisites)**

1. **Network Type**: **Custom Bridge**
2. **Network Name**: **hr-app-net**
3. **Dedicated Subnet**: **192.168.20.0/24**
4. **Gateway**: **192.168.20.1**
5. **Application Stack**:
   □ **Server**: An **NGINX container** (acting as the web frontend).
   □ **Client/Tester**: An **Alpine container** (for internal diagnostics and connectivity testing).

⍰ **Execution Steps** (To be performed by the user)

1. **Network Creation**: Create the **Custom Bridge Network** using the specified name, subnet, and gateway.
2. **Network Inspection**: Verify the correct subnet and gateway configuration using the **network inspection command**.
3. **Container Deployment**: Run the **NGINX container** (named **nginx-server**) and the **Alpine container** (named **alpine-tester**) and attach both to the newly created **hr-app-net**.
4. **IP Allocation Verification**: Use the **docker inspect command** to confirm that both containers received an IP address within the **192.168.20.x** range.
5. **Service Discovery Test**: Execute a command from inside the **alpine-tester container** to ping the **nginx-server** using its container name.

Expected Outcome
□ Both containers will receive **IPs** from the **192.168.20.0/24 subnet**.
The **ping test** from **alpine-tester** to **nginx-server** using the name will succeed, demonstrating successful **DNS Resolution** and internal connectivity within the isolated subnet.

Demonstrate a **multi-homed container architecture** where a single service (**NGINX Load Balancer**) is connected to two distinct, isolated **Custom Bridge Networks**. This setup is crucial for segregating **client traffic** (**Frontend**) from **internal service communication** (**Backend**) while ensuring the **Load Balancer** acts as the necessary routing bridge.

⍰ Requirements (Prerequisites)

1. **Network 1 (Frontend)**:
   □ **Name**: **frontend-net**
   □ **Dedicated Subnet**: **10.1.1.0/24**
2. **Network 2 (Backend)**:
   □ **Name**: **backend-net**
   □ **Dedicated Subnet**: **10.1.2.0/24**
3. **Containers Configuration**:
   □ **nginx-lb** (**Load Balancer**): Must be connected to both **frontend-net** and **backend-net** (**Multi-Homed**).
   □ **client-tester** (**Client**): Must be connected only to **frontend-net**.
   □ **backend-db** (**Service**): Must be connected only to **backend-net**.

⍰ **Execution Steps** (To be performed by the user)

1. **Create Networks**: Create the two required **Custom Bridge Networks**, **frontend-net** and **backend-net**, specifying their respective subnets.
2. **Deploy Isolated Containers**: Run the **backend-db container** and the **client-tester container**, ensuring each is attached only to its designated network.
3. **Deploy Multi-Homed Container**: Run the **nginx-lb container** and attach it to both **frontend-net** and **backend-net** in a single **docker run command**.
4. **IP Verification (Diagnostic)**: Use the **docker inspect command** on **nginx-lb** to confirm that it has been assigned two distinct **IP addresses** (one from **10.1.1.x** and one from **10.1.2.x**).
5. **Isolation Test (Diagnostic)**: From inside the **client-tester container**, attempt to ping the **backend-db container** using its name.

Expected Outcome

- The **nginx-lb container** will successfully acquire two separate **IP addresses**.
- The **isolation test** (ping from **client-tester** to **backend-db**) must fail because there is no direct network path between **frontend-net** and **backend-net**, validating the isolation.
- (Self-check: A **ping from nginx-lb to backend-db** must succeed).

You are Welcome