**Balanceig de càrrega i arquitectura de serveis amb Docker Swarm**

**Objectius de la pràctica**

* Entendre com es distribueixen les peticions entre múltiples rèpliques d’un servei.
* Implementar un balanceig de càrrega amb NGINX en entorn Docker Swarm.
* Visualitzar la rotació de peticions amb respostes diferenciades.
* Fer proves de tolerància a fallades.

**1. Inicialització de Docker Swarm**

**docker swarm init**

**2. Creació de serveis backend personalitzats**

1. Crea tres directoris: web1/, web2/, web3/
2. A cada directori, crea un fitxer index.html amb contingut diferent:

**Exemple (web1/index.html):**

**<h1>Benvinguts al servidor 1</h1>**

1. A cada carpeta, afegeix aquest Dockerfile:

**FROM nginx:alpine**

**COPY index.html /usr/share/nginx/html/index.html**

1. Construeix les imatges (Compte amb els directoris):

**docker build -t web1-img ./web1**

**docker build -t web2-img ./web2**

**docker build -t web3-img ./web3**

**3. Creació de la xarxa virtual**

**docker network create --driver overlay webnet**

**4. Desplegament dels serveis backend**

**docker service create --name web1 --network webnet web1-img**

**docker service create --name web2 --network webnet web2-img**

**docker service create --name web3 --network webnet web3-img**

**5. Creació del balancer NGINX**

1. Crea un directori **balancer/**
2. A dins, crea un fitxer **nginx.conf**:

events {}

http {

upstream backend {

server web1:80;

server web2:80;

server web3:80;

}

server {

listen 80;

location / {

proxy\_pass http://backend;

}

}

}

1. Crea el Dockerfile del balancer:

**FROM nginx:alpine**

**RUN rm /etc/nginx/conf.d/default.conf**

**COPY nginx.conf /etc/nginx/nginx.conf**

1. Construeix la imatge:

**docker build -t balancer-img ./balancer**

1. Desplega el servei (És una sola ordre):

**docker service create \**

**--name balancer \**

**--publish 8080:80 \**

**--network webnet \**

**balancer-img**

**6. Proves amb curl (forçant IPv4)**

**curl -4 http://localhost:8080**

Repetint diverses vegades has de veure respostes com:

**<h1>Benvinguts al servidor 1</h1>**

**<h1>Benvinguts al servidor 2</h1>**

**<h1>Benvinguts al servidor 3</h1>**

S’ha de veure que va rotant el servidor que dona resposta a la petició web.

**7. Proves addicionals**

7.1 Peticions múltiples:

for i in {1..10}; do curl -4 http://localhost:8080; sleep 1; done

7.2 Round Robin:

Comprovar que es roten les peticions fent servir un algorisme de Round Robin

**8. Prova de tolerància a fallades (Swarm en acció)**

Objectiu: demostrar com Docker Swarm recupera automàticament serveis fallits.

1. Llista les tasques actuals del servei:

**docker service ps web2**

1. Mata la rèplica activa:

**docker ps | grep web2**

**docker rm -f <ID>**

1. Comprova la reacció de Swarm:

**docker service ps web2**

Observa que:

* L’antiga rèplica apareix com Shutdown
* Una nova rèplica ha estat creada i està en estat Running

1. Torna a fer:

**curl -4 http://localhost:8080**

Has de seguir rebent respostes de web2, demostrant que Swarm ha recuperat automàticament el servei.

**9. Neteja de l’entorn**

docker service rm balancer web1 web2 web3

docker network rm webnet