Docker swarm

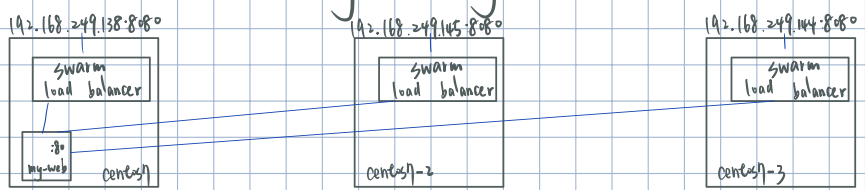
centos7

```
# docker node ls                    查看 node 有沒有 ready
        ready
# systemctl status httpd
                        (dead)
# docker start [id]
#  docker ps
```

chrome 192.168.249.138:8888

test1：有3台虛擬機，然後在一個 service，在每台 docker swarm 的結點上，掛負載均衡器 (load balancer)

連第一台 ip，請求可能會分到 my-web1 or my-web2，連第二、三台 ip 也會是這樣，就算第三台沒有任何容器，請求一樣能發送到 my-web1 or my-web2



centos7

```
# docker service create --name my-web --publish published=8080, target=80 --replicas 1 httpd
# docker service ls
```

chrome 192.168.249.138:8080
192.168.249.145:8080
192.168.249.144:8080
都会成功

test2

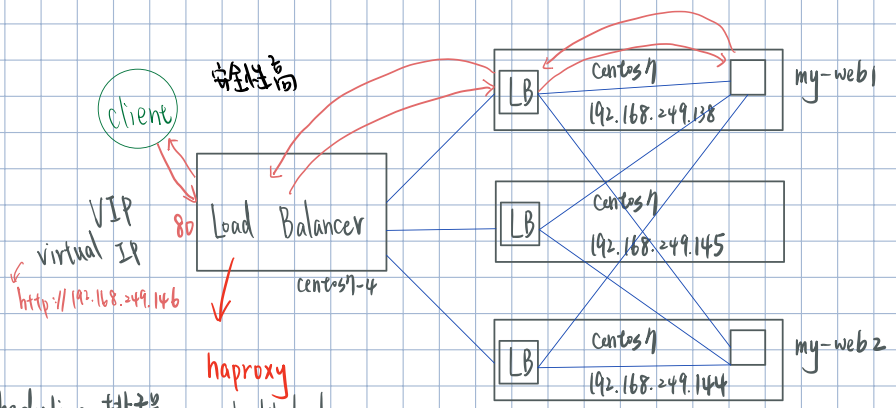
centos7

```
# docker service scale my-web=2
# docker ps
   Container ID
# docker exec -it [id] bash
   # cd htdocs/
   # ls
   # echo "centos7" > index.html
   # cat index.html
```

centos7-3

```
# docker ps
   Container ID
# docker exec -it [id] bash
   # cd htdocs/
   # ls
   # echo "centos7-3" > index.html
   # cat index.html
```

chrome
192.168.249.138:8080 centos7
192.168.249.145:8080 centos7 or centos7-3
192.168.249.144:8080 centos7-3



scheduling 排程

RR

WRR 权重

LC least connecting 最少連線數

healthcheck

假設 机器 断電,故障

就不会把請求丟給 故障的那台

centos7-4

```
# ifconfig
# yum install epel-release
# yum install haproxy
# mv /etc/haproxy/haproxy.cfg /etc/haproxy/haproxy.cfg.bak
# vim /etc/haproxy/haproxy.cfg
:
```

```
blance  roundrobin          RR
server  centos7   192.168.249.138:8080  check
server  centos7-2 192.168.249.145:8080  check
server  centos7-3 192.168.249.144:8080  check
```

```
# systemctl start haproxy
# systemctl status haproxy
```

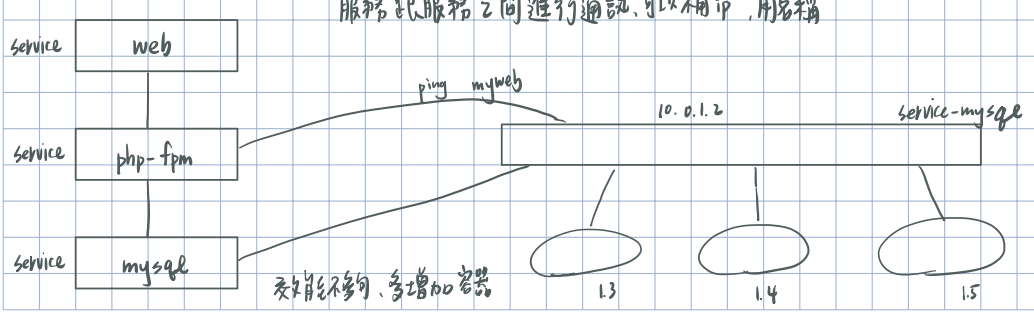
chrome 192.168.249.146 centos7 or centos7-3

多跑幾次,就跟一樣

---

test3 : docker swarm 環境中, service 之間的通信服務發現 (service discovery)

服務跟服務之間進行通訊、可以不用 ip ,用名稱



效能不夠、多增加容器

centos7

```
# docker network ls
# docker network create --driver overlay myapp_net
# docker network ls
```

centos7-2、7-3

```
# docker network ls
```

```
# docker service ls
# docker service rm my-web
```

```
# docker service create --name myweb --replicas 3 --network myapp_net httpd
# docker service ls
```

centos7-2、centos7-3

```
# docker network ls
```

myapp_net

centos7

```
# docker service create --name util --network myapp_net busybox sleep 10000000
# docker service ls
# docker inspect id
```

centos7-2

```
# docker ps
```

Container ID　util

```
# docker exec -itd [ID] /bash/sh
/# ping -c 3 myweb
```

10.0.1.2

```
/# ping 10.0.1.3
   10.0.1.4
   10.0.1.5
```

→ 看myweb的ip

centos7、centos7-2、centos7-3

```
# docker ps
# docker inspect [ID]
```

變成1台時，也可以用名稱ping

centos7

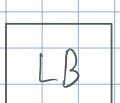
```
# docker service scale myweb=1
```

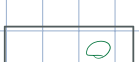
centos7-2

```
/# ping -c 3 myweb
```

10.0.1.2

```
# docker service create --name util --network myapp_net busybox sleep 10000000
```





變成5台時，也可以用名稱ping

centos7

```
# docker service scale myweb=5
```

centos7-2

```
/# ping -c 3 myweb
```

10.0.1.2