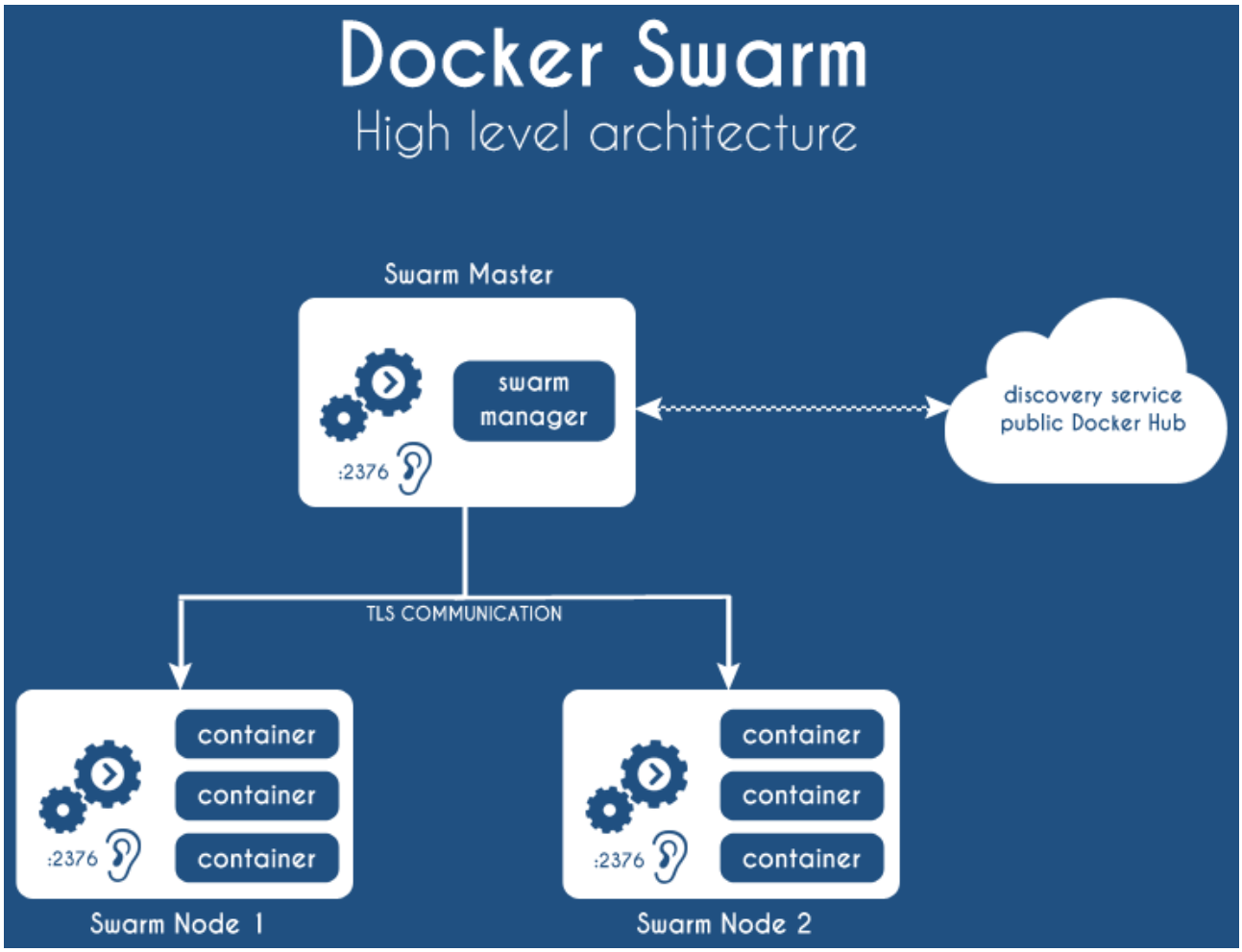
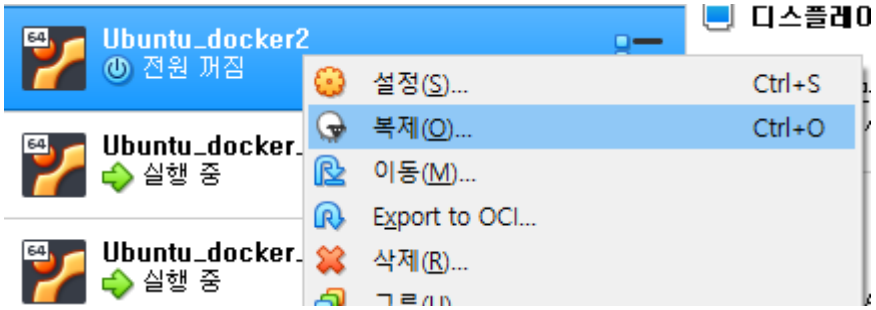
DOCKER SWARM MODE CLUSTER 구축



- 수많은 container 오케스트레이션(Orchestration) 도구 중의 하나로,
- 여러 대의 Docker 호스트들을 마치 하나인 것처럼 만들어주는 Orchestration 도구
- 기본적으로 Docker Swarm 은 Master Node 와 Worker Node 로 시스템을 구성
- Master Node 에서는 클러스터 관리 작업을 하고 클러스터 상태 유지, 스케줄링 서비스, Swarm HTTP API Endpoint 를 제공
- Worker Node 는 container 를 실행하는 역할만 수행
- Container 의 유연성과 확장성, 이동의 편리함을 활용해 기업에서 운영하는 Database, LDAP, Jenkins, Docker Registry, Elasticsearch, Grafana 등의 모든 서비스를 운영, 관리.
- **PaaS** 와 같은 용도로 docker server clustering 구축 시 swarm 이 유용

■ Docker swarm mode cluster 환경 구성

-- 기존 Ubuntu_docker 를 정지 시킨 후 [복제] 수행 (이름변경 및 MAC 초기화)



-- 서버 구성 (Ubuntu 16.04)

1) Hostname: swarm-manager 192.168.56.103
2) Hostname: swarm-worker1 192.168.56.104
3) Hostname: swarm-worker2 192.168.56.105

-- HOSTNAME 변경

```
jeff@jeff-VirtualBox:~$ sudo hostnamectl set-hostname swarm-manager
jeff@jeff-VirtualBox:~$ cat /etc/hostname
swarm-manager
jeff@jeff-VirtualBox:~$ sudo reboot
```

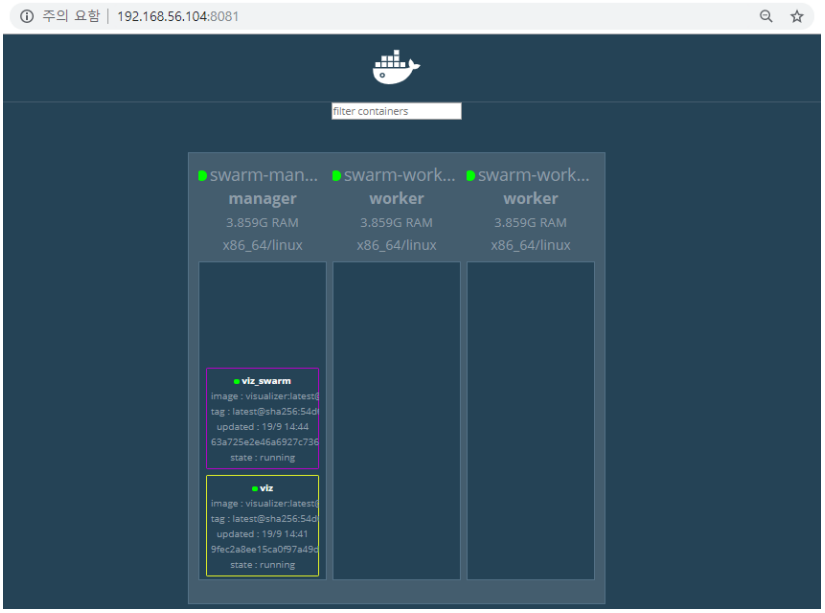
☞ 각각의 hostname 을 위 내용과 같이 변경 후 reboot 수행

■ Docker swarm visualization install

- 여러 호스트와 서비스를 처리 할 때 일종의 시각적 피드백을 갖는 것이 유용
- 그렇기 때문에 클러스터에서 시각화 서비스 실행
- Host OS 의 Docker Engine 을 구동하고, swarm-manager 에서 viz_swarm 서비스를 생성.

```
jeff@swarm-manager:~$ docker service create \
> --name=viz_swarm \
> --publish=8081:8080 \
> --constraint=node.role==manager \
> --mount=type=bind,src=/var/run/docker.sock,dst=/var/run/docker.sock \
> dockersamples/visualizer
klrv5wkiqguv355qq1btjsflq
overall progress: 1 out of 1 tasks
1/1: running
verify: Service converged
```

http://Host_OS_IP:8081



: 위 그림은 아래 swarm worker1,2 를 모두 등록 후 화면임.

■ configure Docker swarm mode cluster

-- swarm manager 설정

```
jeff@swarm-manager:~$ docker swarm init --advertise-addr 192.168.56.103
Swarm initialized: current node (5n8o2k2ennj8vlutyvnb9vfm2) is now a manager.
To add a worker to this swarm, run the following command:
```

init 명령을 통해 manager 역할을 할 서버에서 swarm cluster 시작

--advertise-addr 에는 docker server 가 manager node 에 접근하기 위한 IP 입력 (public IP)

★ docker swarm join --token SWMTKN-1-1v7hhbu51bar1c2trix4jc0hyn0yvjikau4gqgl7hek3drr7rl-7fibqmhdm53lat5173jrncea9 192.168.56.103:2377 ★ # 비밀키 생성이며,

swarm manager 는 default port = 2377, node 간 통신은(7946/tcp, 7946/udp), swarm 이 사용하는 ingress overlay network 는(4789/tcp, 4789/udp) 사용, 사전에 firewall 에 open 주의.

```
To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.
```

-- swarm worker1 설정(swarm-manager 에서 token copy)

```
jeff@swarm-worker1:~$ docker swarm join --token SWMTKN-1-1v7hhbu51bar1c2trix4jc0hyn0yvjikau4gqgl7hek3drr7rl-7fibqmhdm53lat5173jrncea9 192.168.56.103:2377
This node joined a swarm as a worker.
```

-- swarm worker2 설정(swarm-manager 에서 token copy)

```
jeff@swarm-worke2:~$ docker swarm join --token SWMTKN-1-1v7hhbu51bar1c2trix4jc0hyn0yvjikau4gqgl7hek3drr7rl-7fibqmhdm53lat5173jrncea9 192.168.56.103:2377
This node joined a swarm as a worker.
```

(주의) token 은 외부에 노출되지 않도록 관리 필요. 공개 시 누구든 swarm cluster 에 연결 가능.

보안 측면에서 문제가 될 수 있어 주기적으로 swarm cluster token 변경 필요 (--rotate)

-- swarm manager 에서 node 확인

```
jeff@swarm-manager:~$ docker node ls
ID                            HOSTNAME            STATUS              AVAILABILITY        MANAGER STATUS      ENGINE
VERSION
5n8o2k2ennj8vlutyvnb9vfm2 *   swarm-manager       Ready               Active              Leader
18.09.6
krsh0pfvs1vi8seqk4dr0pz91     swarm-worke2        Ready               Active
18.09.6
fj0q2n27uqxnrauuj3gkqlmfq     swarm-worker1       Ready               Active
18.09.6


jeff@swarm-manager:~$ docker info
Containers: 4
 Running: 0
 Paused: 0
 Stopped: 4
Images: 2
Server Version: 18.09.6
Storage Driver: overlay2
 Backing Filesystem: extfs
 Supports d_type: true
 Native Overlay Diff: false
Logging Driver: json-file
Cgroup Driver: cgroupfs
Plugins:
 Volume: local
 Network: bridge host macvlan null overlay
 Log: awslogs fluentd gcplogs gelf journald json-file local logentries splunk sy    slog
Swarm: active★★★
 NodeID: 5n8o2k2ennj8vlutyvnb9vfm2
 Is Manager: true
 ClusterID: oin36r7jlamfa0zglk5070vwt
 Managers: 1 ★★
 Nodes: 3           ★★
 Default Address Pool: 10.0.0.0/8
 SubnetSize: 24
 Orchestration:
  Task History Retention Limit: 5
 Raft:
  …
```

--[참고]-- 삭제는 leave ----------

```
jeff@swarm-worker1:~$ docker swarm leave
jeff@swarm-manager:~$ docker swarm leave –force
```

-- cluster 에 새로운 Node 추가 시 사용되는 토큰 갱신 (--rotate)

```
# worker node 에 추가하기 위한 token 확인
jeff@swarm-manager:~$ docker swarm join-token manager
To add a manager to this swarm, run the following command:

    docker swarm join --token
SWMTKN-1-1v7hhbu51bar1c2trix4jc0hyn0yvjikau4gqgl7hek3drr7rl-0lyoo7v3cqexbejcmhtj3w7vj
192.168.56.103:2377

# token update (manager node 에서만 수행 가능)
jeff@swarm-manager:~$ docker swarm join-token --rotate manager
Successfully rotated manager join token.

To add a manager to this swarm, run the following command:

    docker swarm join --token
SWMTKN-1-1v7hhbu51bar1c2trix4jc0hyn0yvjikau4gqgl7hek3drr7rl-8imcmzoekleukd0b1gumo0yfj
192.168.56.103:2377

jeff@swarm-manager:~$ docker info
```

-- swarm cluster 에서 node 상태 변경

```
# worker node 삭제(해제)하면 manager node 는 해당 worker node 의 상태를 Down 으로 변경, 삭제 안함
jeff@swarm-worker1:~$ docker swarm leave

jeff@swarm-manager:~$ docker node ls

# worker node 완전 삭제
jeff@swarm-manager:~$ docker node rm swarm-worker1 (hostname)
jeff@swarm-manager:~$ docker node ls
```

```
# manager node 삭제(해제), manager node 가 1 개인 경우 삭제 시 더 이상 swarm cluster 사용못함
jeff@swarm-manager:~$ docker swarm leave --force

# worker node → manager node 로 변경
jeff@swarm-manager:~$ docker node promote swarm-worker1

# manager node → worker node 로 변경
jeff@swarm-manager:~$ docker node demote swarm-worker1
```

■ Docker swarm mode cluster 서비스

1) Swarm mode 에서 제어하는 단위는 container 가 아닌 서비스(service)
2) 서비스는 같은 이미지에서 생성된 container 의 집합
3) 서비스 제어하면 해당 서비스 내의 container 에 동일 명령이 수행됨
4) 서비스내의 container 는 1 개 이상 존재하며, worker 와 manager Node 에 할당
   이러한 container 들을 Task 라고 함.

-- [실습 1] ubuntu 서비스 생성 ----------

: ubuntu:14.04 이미지로 서비스 내의 container 를 생성하며 container 시작 시 2 초마다 실행할 명령어로 "hello world"를 출력하는 shell 설정

(주의) 서비스 내의 container 는 -d(detached) mode 로 동작하는 이미지를 사용해야 함.

```
jeff@swarm-manager:~$ docker service create \
> ubuntu:14.04 \
> /bin/sh -c "while true; do echo hello world; sleep 2; done"
j2rgjs9dfqko0x11h59dddmms
overall progress: 1 out of 1 tasks
1/1: running
verify: Service converged

jeff@swarm-manager:~$ docker service ls
ID                  NAME                MODE                REPLICAS            IMAGE
PORTS
j2rgjs9dfqko        eloquent_gagarin    replicated          1/1                 ubuntu:14.04

jeff@swarm-manager:~$ docker service ps eloquent_gagarin
ID                  NAME                IMAGE               NODE                DESIRED STATE
CURRENT STATE           ERROR               PORTS
q6prs3t58ad2        eloquent_gagarin.1  ubuntu:14.04        swarm-manager       Running
Running 3 minutes ago
```

service 내에서 발생하는 정보(log) 확인

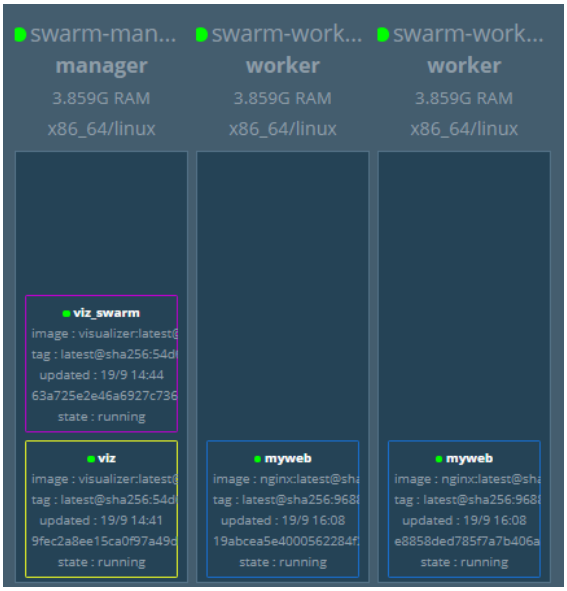
```
jeff@swarm-manager:~$ docker service logs q6prs3t58ad2
hello world
hello world
...

jeff@swarm-manager:~$ docker service rm eloquent_gagarin
eloquent_gagarin
jeff@swarm-manager:~$ docker service ls
```

-- [실습 2] Nginx 웹서버 서비스 생성 ----------

: docker service create 시 --replicas 옵션 추가하고 외부 노출

: 2 개의 replica container 정의하고 80:80 으로 연결, 이름을 "myweb" 으로 지정

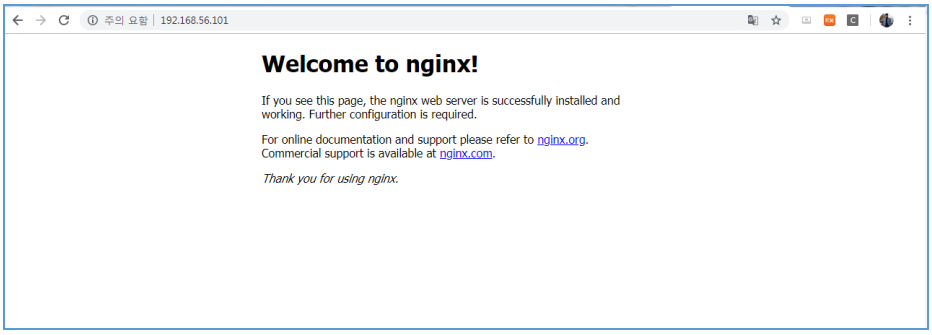
```
jeff@swarm-manager:~$ docker service create --name myweb \
> --replicas 2 \
> -p 80:80 \
> nginx

s5itb78wn2edrypcva9qn0d9m
overall progress: 2 out of 2 tasks
1/2: running   ★
2/2: running   ★
verify: Service converged
```



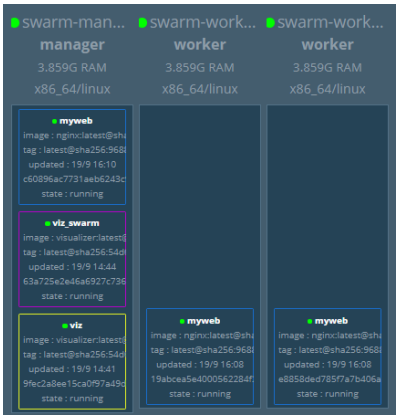

```
jeff@swarm-manager:~$ docker service ps myweb
ID                  NAME                IMAGE               NODE                DESIRED STATE
CURRENT STATE               ERROR               PORTS
iidjb91mr6kt        myweb.1             nginx:latest        swarm-worker1       Running
Running about a minute ago
p8zzmjfkbwau        myweb.2             nginx:latest        swarm-manager       Running
Running about a minute ago
```

생성 완료 후 Swarm 클러스터 내의 Node 중 하나를 선택해 80 번 포트로 접근 가능확인, 설치가 진행되지 않았던 worker1 의 주소로 접근

http://192.168.56.104:80 >>>>> Welcome to nginx!



-- Node 확장




```
jeff@swarm-manager:~$ docker service scale myweb=3
myweb scaled to 3
overall progress: 3 out of 3 tasks
1/3: running
2/3: running
3/3: running
verify: Service converged
```

```
jeff@swarm-manager:~$ docker service ps myweb
ID                  NAME                IMAGE               NODE                DESIRED STATE
CURRENT STATE           ERROR               PORTS
iidjb91mr6kt        myweb.1             nginx:latest        swarm-worker1       Running
Running 4 minutes ago
p8zzmjfkbwau        myweb.2             nginx:latest        swarm-manager       Running
Running 4 minutes ago
72oebt4egf2d        myweb.3             nginx:latest        swarm-worke2        Running
Running 7 seconds ago
```

설치가 진행되지 않았던 worker2 의 주소로 접근

http://192.168.56.105:80 >>>>> Welcome to nginx!

-- worker1, 2 에서 container 확인

```
jeff@swarm-worker1:~$ docker container ls
CONTAINER ID        IMAGE               COMMAND                  CREATED
STATUS              PORTS               NAMES
66012101991d        nginx:latest        "nginx -g 'daemon of…"   5 minutes
ago       Up 5 minutes        80/tcp
myweb.1.iidjb91mr6ktt3hio71wsfdo1
```

 # container 는 swarm-manager node 에 하나를 생성하고 복제 연결되어있음을 확인.

[추가 실습]

```
jeff@swarm-manager:~$ docker service scale myweb=4
```

#4개로 설정 시 node는 3개 지만 4개로 복제 가능, 나머지 하나는 추가로 임의의 node에 추가로 생성

#실제 요청이 어느 host의 어떤 node로 접근하든 4개의 container중 1개로 re-direction되서 지정

-- [실습 3] Nginx 웹서버 Global 서비스 생성 --

- 실습 2 번과 다르게 global mode 는 docker swarm cluster 내에서 사용할 수 있는 모든 Node 에 container 를 반드시 하나씩 생성

- 따라서, global mode 로 생성한 서비스는 별도의 --replicas 를 지정하지 않음.

- global mode 는 warm cluster 를 모니터링 하기 위한 agent container 등을 생성 시 유용

```
jeff@swarm-manager:~$ docker service create --name global_myweb \
> --mode global \
> nginx
rq38314669hev2zl66u7w94np
overall progress: 3 out of 3 tasks
fj0q2n27uqxn: running
5n8o2k2ennj8: running
krsh0pfvs1vi: running
verify: Service converged
```

```
jeff@swarm-manager:~$ docker service ls
ID                  NAME                MODE                REPLICAS            IMAGE               PORTS
rq38314669he        global_myweb        global              3/3                 nginx:latest
s5itb78wn2ed        myweb               replicated          3/3                 nginx:latest        *:80->80/tcp
```

```
jeff@swarm-manager:~$ docker service ps global_myweb
ID                  NAME                                    IMAGE               NODE
DESIRED STATE       CURRENT STATE           ERROR               PORTS
qqp4up976utr        global_myweb.5n8o2k2ennj8vlutyvnb9vfm2  nginx:latest        swarm-manager
Running             Running 29 seconds ago
zvgwb5jtyqf3        global_myweb.krsh0pfvs1vi8seqk4dr0pz91  nginx:latest        swarm-worke2
Running             Running 29 seconds ago
hozrzyyjs0ek        global_myweb.fj0q2n27uqxnrauuj3gkqlmfq  nginx:latest        swarm-worker1
Running             Running 29 seconds ago
```

■ Docker swarm mode cluster 서비스 장애복구

: 복제 mode 로 설정된 서비스의 container 가 정지하거나 특정 Node 가 다운되면 Swarm 매니저는 새로운 container 를 생성해 자동으로 이를 복구수행.

```
jeff@swarm-manager:~$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
PORTS               NAMES
d5347353ef63        nginx:latest        "nginx -g 'daemon of…"   2 minutes ago       Up 2 minutes
80/tcp              global_myweb.5n8o2k2ennj8vlutyvnb9vfm2.qqp4up976utrimok3aordyv1c
20a06ee4fc7d        nginx:latest        "nginx -g 'daemon of…"   12 minutes ago      Up 12 minutes
80/tcp              myweb.2.p8zzmjfkbwauajq2bzszz4yy1
```

```
# 실습을 위해 myweb 서비스 중 하나를 삭제하여 장애 유발
jeff@swarm-manager:~$ docker service ps myweb
ID                  NAME                IMAGE               NODE                DESIRED STATE
CURRENT STATE           ERROR               PORTS
iidjb91mr6kt        myweb.1             nginx:latest        swarm-worker1       Running
Running 13 minutes ago
p8zzmjfkbwau        myweb.2             nginx:latest        swarm-manager       Running
Running 13 minutes ago
72oebt4egf2d        myweb.3             nginx:latest        swarm-worke2        Running
Running 9 minutes ago
                                                [NAME].[ID]
jeff@swarm-manager:~$ docker rm -f myweb.2.p8zzmjfkbwauajq2bzszz4yy1
myweb.2.p8zzmjfkbwauajq2bzszz4yy1

jeff@swarm-manager:~$ docker service ps myweb
ID                  NAME                IMAGE               NODE                DESIRED STATE
CURRENT STATE           ERROR                       PORTS
iidjb91mr6kt        myweb.1             nginx:latest        swarm-worker1       Running
Running 14 minutes ago
c54xvhhw77ef        myweb.2             nginx:latest        swarm-manager       Ready
Ready 3 seconds ago
p8zzmjfkbwau         \_ myweb.2         nginx:latest        swarm-manager       Shutdown
Failed 3 seconds ago     "task: non-zero exit (137)"
72oebt4egf2d        myweb.3             nginx:latest        swarm-worke2        Running
Running 10 minutes ago

# 특정 node 가 down 되었을 때도 위와 같이 동작, worker1 Node 에서 서비스 중단
jeff@swarm-worker1:~$ sudo service docker stop
sudo: unable to resolve host swarm-worker1
```

```
jeff@swarm-manager:~$ docker node ls
ID                            HOSTNAME            STATUS              AVAILABILITY        MANAGER
STATUS      ENGINE VERSION
5n8o2k2ennj8vlutyvnb9vfm2 *   swarm-manager       Ready               Active              Leader
18.09.6
krsh0pfvs1vi8seqk4dr0pz91     swarm-worke2        Ready               Active
18.09.6
fj0q2n27uqxnrauuj3gkqlmfq     swarm-worker1       Down                Active
18.09.6
```

```
# down 된 swarm-worker1 node 의 상태가 shutdown 됐으며,
# 이를 복구하기 위해 swarm-manager node 에 서비스 생성됨을 확인
jeff@swarm-manager:~$ docker service ps myweb
ID                  NAME                IMAGE               NODE                DESIRED STATE
CURRENT STATE               ERROR                         PORTS
xyy48soh1i3w        myweb.1             nginx:latest        swarm-manager       Running
Running about a minute ago
iidjb91mr6kt         \_ myweb.1         nginx:latest        swarm-worker1       Shutdown
Running about a minute ago
p6bltbqjujc6        myweb.2             nginx:latest        swarm-manager       Running
Running about a minute ago
c54xvhhw77ef         \_ myweb.2         nginx:latest        swarm-manager       Shutdown
Complete about a minute ago
p8zzmjfkbwau         \_ myweb.2         nginx:latest        swarm-manager       Shutdown
Failed 3 minutes ago          "task: non-zero exit (137)"
72oebt4egf2d        myweb.3             nginx:latest        swarm-worke2        Running
Running about a minute ago
```

-- worker1 Node 에서 서비스 시작

```
jeff@swarm-worker1:~$ sudo service docker start
sudo: unable to resolve host swarm-worker1
```

-- worker1 Node 에서 서비스 시작한 뒤 서비스 상태를 보면 정상적으로 돌아 오지 않음

-- 이는 문제가 된 Node 가 서비스 시작이 되면 자동 rebalance 가 일어나지 않음을 뜻함

-- 이를 맞추기 위해 수동으로 scale 를 줄였다 늘였다 해줘야 함.

```
jeff@swarm-manager:~$ docker service ps myweb
ID                  NAME                IMAGE               NODE                DESIRED STATE
CURRENT STATE           ERROR                         PORTS
xyy48soh1i3w        myweb.1             nginx:latest        swarm-manager       Running
Running 2 minutes ago
iidjb91mr6kt         \_ myweb.1         nginx:latest        swarm-worker1       Shutdown
Shutdown 30 seconds ago
p6bltbqjujc6        myweb.2             nginx:latest        swarm-manager       Running
Running 3 minutes ago
c54xvhhw77ef         \_ myweb.2         nginx:latest        swarm-manager       Shutdown
Complete 3 minutes ago
p8zzmjfkbwau         \_ myweb.2         nginx:latest        swarm-manager       Shutdown
Failed 4 minutes ago      "task: non-zero exit (137)"
72oebt4egf2d        myweb.3             nginx:latest        swarm-worke2        Running
Running 3 minutes ago
```

```
jeff@swarm-manager:~$ docker service scale myweb=1
myweb scaled to 1
overall progress: 1 out of 1 tasks
1/1:
verify: Service converged
```

```
jeff@swarm-manager:~$ docker service scale myweb=3
myweb scaled to 3
overall progress: 3 out of 3 tasks
1/3: running
2/3: running
3/3: running
verify: Service converged
```

```
jeff@swarm-manager:~$ docker service ps myweb
ID                  NAME                IMAGE               NODE                DESIRED STATE
CURRENT STATE               ERROR                         PORTS

xyy48soh1i3w        myweb.1             nginx:latest        swarm-manager       Running
Running 2 minutes ago

iidjb91mr6kt         \_ myweb.1         nginx:latest        swarm-worker1       Shutdown
Shutdown about a minute ago

q8p8zyp7xv65        myweb.2             nginx:latest        swarm-worke2        Running
Running 9 seconds ago

c54xvhhw77ef         \_ myweb.2         nginx:latest        swarm-manager       Shutdown
Complete 3 minutes ago

p8zzmjfkbwau         \_ myweb.2         nginx:latest        swarm-manager       Shutdown
Failed 5 minutes ago          "task: non-zero exit (137)"

w0ohk1i14ldb        myweb.3             nginx:latest        swarm-worker1       Running
Running 9 seconds ago
```

■ Docker swarm mode cluster 서비스 [Rolling update]

swarm mode 는 rolling update 자체 지원, 간단하게 사용 가능

rolling update 사용을 위한 서비스 생성

```
jeff@swarm-manager:~$ docker service create --name rollup_myweb --replicas 3 nginx:1.10
s7g7wttv26bgqofgsu9iz0idq
overall progress: 3 out of 3 tasks
1/3: running
2/3: running
3/3: running
verify: Service converged
```

```
jeff@swarm-manager:~$ docker service ps rollup_myweb
ID                  NAME                IMAGE               NODE                DESIRED STATE
CURRENT STATE           ERROR               PORTS
vgppvzs3q2ij        rollup_myweb.1      nginx:1.10          swarm-worker1       Running
Running 23 seconds ago
tgv84d3hxqkw        rollup_myweb.2      nginx:1.10          swarm-worke2        Running
Running 23 seconds ago
xljai0o03nzi        rollup_myweb.3      nginx:1.10          swarm-manager       Running
Running 23 seconds ago
```

docker service update 를 통해 image update 수행

```
jeff@swarm-manager:~$ docker service update --image nginx:1.11 rollup_myweb
overall progress: 3 out of 3 tasks
1/3: running
2/3: running
3/3: running
verify: Service converged
```

```
jeff@swarm-manager:~$ docker service ps rollup_myweb
ID                  NAME                IMAGE               NODE                DESIRED STATE
CURRENT STATE           ERROR               PORTS
4m1wkr7n56cy        rollup_myweb.1      nginx:1.11          swarm-worker1       Running
Running 8 seconds ago
vgppvzs3q2ij         \_ rollup_myweb.1   nginx:1.10          swarm-worker1       Shutdown
Shutdown 11 seconds ago
7327umk3lue2        rollup_myweb.2      nginx:1.11          swarm-worke2        Running
Running 13 seconds ago
tgv84d3hxqkw         \_ rollup_myweb.2   nginx:1.10          swarm-worke2        Shutdown
Shutdown 17 seconds ago
```

```
yt6hsr2ozqdi        rollup_myweb.3       nginx:1.11          swarm-manager       Running
Running 18 seconds ago
xljai0o03nzi         \_ rollup_myweb.3   nginx:1.10          swarm-manager       Shutdown
Shutdown 22 seconds ago
```

-- update 주기, update 를 동시에 진행할 container 수, update 실패 시 대처 등을 설정

>> container 복제를 10 초 단위로 업데이트

>> update 작업을 한번에 2 개의 container 에 수행

>> 미 설정 시 주기 없이 차례대로 container 를 한 개씩 update 함.

```
jeff@swarm-manager:~$ docker service create --replicas 4 \
> --name rollup_myweb2 \
> --update-delay 10s \
> --update-parallelism 2 \
> nginx:1.10
h7s4nd1jgwu1zrc1cjruvur7q
overall progress: 4 out of 4 tasks
1/4: running
2/4: running
3/4: running
4/4: running
verify: Service converged
```

```
jeff@swarm-manager:~$ docker service inspect --pretty rollup_myweb2
ID:             h7s4nd1jgwu1zrc1cjruvur7q
Name:           rollup_myweb2
Service Mode:   Replicated
 Replicas:      4
Placement:
UpdateConfig:
 Parallelism:   2
 Delay:         10s
 On failure:    pause ★★★
 Monitoring Period: 5s
 Max failure ratio: 0
 Update order:      stop-first
…
 Rollback order:    stop-first
ContainerSpec:
```

```
 Image:         nginx:1.10@sha256:6202beb06ea61f44179e02ca965e8e13b961d12640101fca213efbfd145d7575
 Init:          false
Resources:
Endpoint Mode:  vip
```

```
jeff@swarm-manager:~$ docker service ps rollup_myweb2
ID                  NAME                IMAGE               NODE                DESIRED STATE
CURRENT STATE               ERROR               PORTS
js3tzovrz5co        rollup_myweb2.1     nginx:1.10          swarm-worker1       Running
Running about a minute ago
xcsr2j6vhkgv        rollup_myweb2.2     nginx:1.10          swarm-worker1       Running
Running about a minute ago
n18uajzc126r        rollup_myweb2.3     nginx:1.10          swarm-worke2        Running
Running about a minute ago
dh53f4dx8muj        rollup_myweb2.4     nginx:1.10          swarm-manager       Running
Running about a minute ago
```

-- 장애 대처 방안

On failure: pause 항목은 업데이트 도중 오류가 발생하면 rolling update 를 중지하는 것을 의미

서비스 생성 시 --update-failure-action continue 지정 시 오류가 발생해도 계속 rolling update 진행

```
jeff@swarm-manager:~$ docker service create --name myweb3 --replicas 3
--update-failure-action continue nginx:1.10
x399m2av62o3ecev91cd19bt1
overall progress: 3 out of 3 tasks
1/3: running
2/3: running
3/3: running
verify: Service converged

jeff@swarm-manager:~$ docker service inspect --pretty myweb3
ID:             x399m2av62o3ecev91cd19bt1
Name:           myweb3
Service Mode:   Replicated
 Replicas:      3
Placement:
UpdateConfig:
 Parallelism:   1
 On failure:    continue      ★★★
 Monitoring Period: 5s
 Max failure ratio: 0
 Update order:      stop-first
RollbackConfig:
 Parallelism:   1
 On failure:    pause
 Monitoring Period: 5s
 Max failure ratio: 0
 Rollback order:    stop-first
ContainerSpec:
 Image:         nginx:1.10@sha256:6202beb06ea61f44179e02ca965e8e13b961d12640101fca213efbfd145d7575
 Init:          false
Resources:
Endpoint Mode:  vip
```

● Docker swarm network

```
jeff@swarm-manager:~$ docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
37117fc4df70        bridge              bridge              local
35ecd33cc994        docker_gwbridge     bridge              local    ★★★
e9a91d7b5860        host                host                local
t9hq4e76jb3d        ingress             overlay             swarm    ★★★
9a67edfef71e        none                null                local
```

- Docker Swarm 은 두 가지 종류의 Traffic 생성

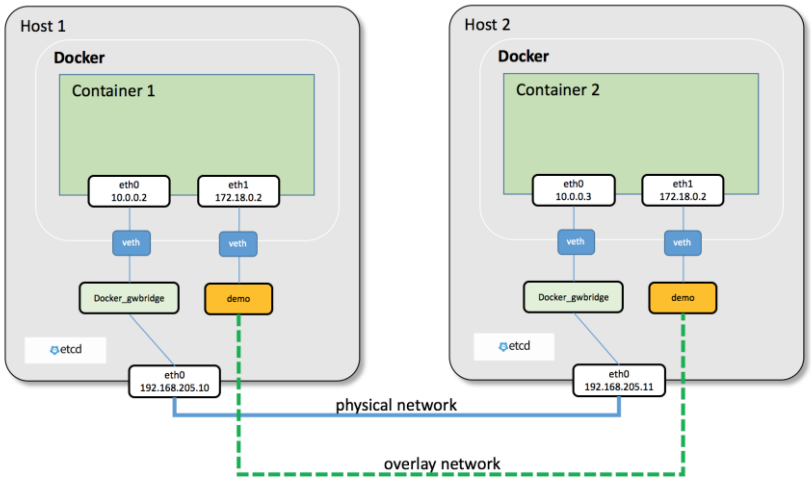
- 제어 및 관리 영역 Traffic: Docker Swarm 에 대한 참가 및 탈퇴 요청과 같은 Docker Swarm 의 관리 Message 가 포함. Traffic 은 항상 암호화.

- Application Data 영역 Traffic: Container 및 외부 Client 와의 Traffic.

● Docker network 종류

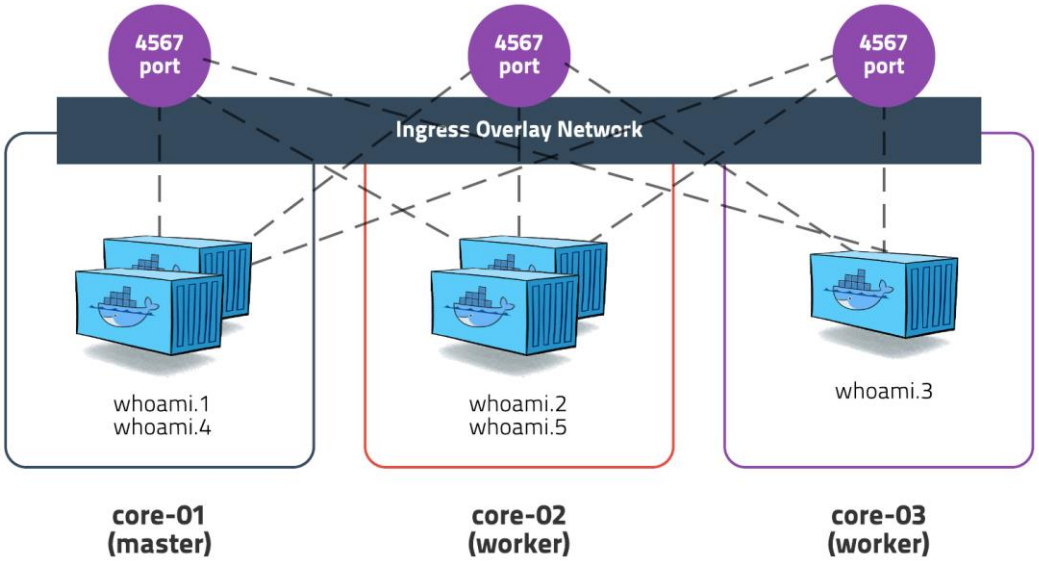
- Overlay Network

- Docker Swarm 에 참여하는 Docker Daemon 간의 통신관리
- 독립 실행 Container 의 Network 를 생성하는 방법과 동일한 방식 생성
- 기존에 생성된 Overlay Network 에 Service 를 연결시켜 Service 간 통신을 활성화 가능
- Overlay Network Driver 사용



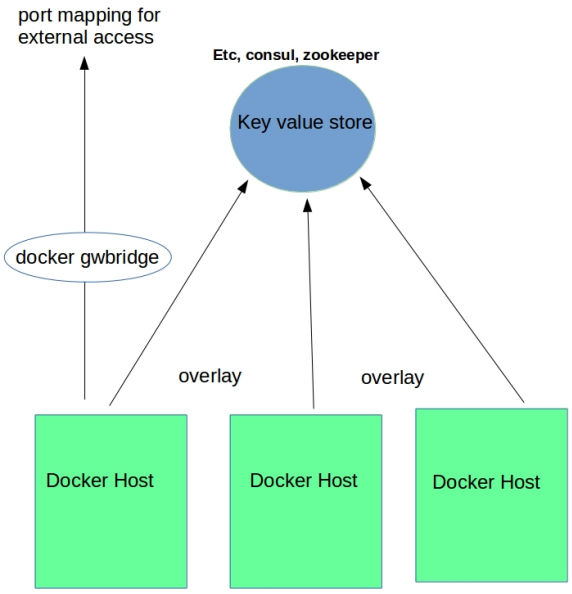
- Ingress Network

- Service 의 Node 간 Load Balancing 수행
- Docker Swarm 의 모든 Node 가 노출된 Port 로 요청을 받게 되면, 해당 요청을 IPVS 라는 모듈로 전달
- IPVS 는 해당 Service 에 참여하는 모든 IP 주소를 추적하고 그 중 하나를 선택한 뒤, 요청을 해당 경로로 Routing
- Ingress Network 는 Docker Swarm 을 init 하거나 Join 할 때 자동 생성



- docker_gwbridge

- docker_gwbridge 는 Overlay Network(Ingress Network 포함)를 개별 Docker Daemon 의 물리적 Network 에 연결하는 Bridge Network
- 기본적으로, Service 가 실행 중인 각각의 Container 는 로컬 Docker Daemon Host 의 docker_gwbridge Network 에 연결
- docker_gwbridge Network 는 Docker Swarm 을 init 하거나 Join 할 때 자동 생성



● Service Discovery (service 를 찾아주는 기능 → DNS 를 통한 서비스 검색 기능)

- Docker 는 사용자가 정의한 Bridge, Overlay 및 MACVLAN Network 들에게 Host 내의 모든 Container 의 위치를 제공하는 내부 DNS Server 를 보유
- 각 Docker Container(또는 Docker Swarm 의 Task)에 존재하는 DNS Resolver 가, DNS 쿼리를 DNS Server 역할을 하는 Docker Engine 으로 전달 한 뒤
- Docker Engine 은 DNS 쿼리가 요청한 Container 가 Network 내에 포함되어있는지 확인
- Docker Engine 은 key-value 저장소에서 Container, Task 또는 Service 이름과 일치하는 IP 주소를 조회하고, 해당 IP 또는 Service Virtual IP(VIP)를 요청자에게 반환
- 이렇게 Docker 는 내장 DNS 를 사용하여, Single Docker Engine 에서 실행되는 Container 및 Docker Swarm 에서 실행되는 Task 에 대한 Service Discovery 기능을 제공
- Service Discovery 는 Network 범위 내에서 동작
- 동일한 Network 에 있는 Container 나 Task 만 내장 DNS 기능을 사용할 수 있음을 의미
- 따라서, 동일한 Network 에 있지 않은 Container 는 서로의 주소를 확인할 수 없음
- 또한, 특정 Network 에 Container 또는 Task 가 있는 Node 만 해당 Network 의 DNS 항목들을 저장
- 이러한 특징들이 Docker 의 보안 및 성능을 향상
- 만약 대상 Container 또는 Service 가 원본 Container 와 동일한 Network 에 속하지 않는다면, Docker Engine 은 구성된 기본 **DNS Server** 로 DNS 쿼리를 전달