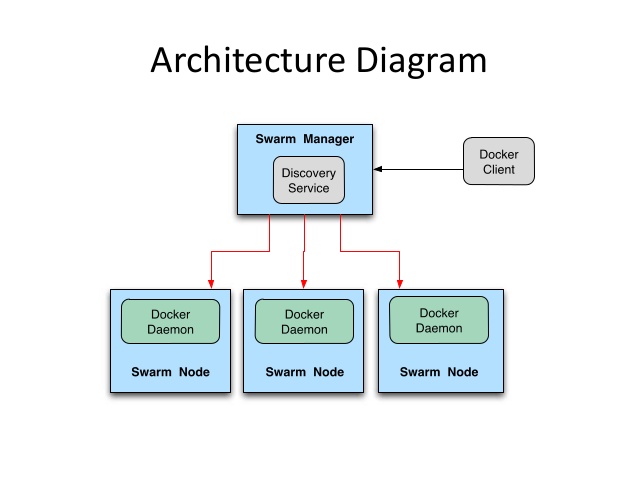
Docker Swarm merupakan Container Orchestration System yang disediakan oleh docker untuk memdistribusikan docker contrainer pada system cluster. Dengan kita memanfaatkan [docker swarm](http://www.dimasrio.com/2017/08/container-orchestration-docker-swarm.html), kita dapat membuat sebuah management yang centralized baik itu create, destroy, scaling up dan scaling down container pada docker swarm manager.

[](https://1.bp.blogspot.com/-eLTSF8gWujQ/WY_VWZFYldI/AAAAAAAAC4U/ogyEPPwpf1UTLbXM-gt3kM3mmScIWRI5wCLcBGAs/s1600/docker-swarm-introduction.jpg)

**Konfigurasi Docker Swarm**

Untuk membuat sebuah docker swarm, diperlukan host yang sudah terinstall docker engine. Pada contoh kali ini saya akan membuat 3 buah docker node, dimana 1 node akan digunakan sebagai manager dan 2 buah node lainnya digunakan untuk swarm worker. Agar lebih mudah membuat docker host kita dapat menggunakan [docker machine](http://www.dimasrio.com/2017/07/install-docker-machine-di-linux.html).

**Setup docker host menggunakan docker machine**

*docker-machine create -d virtualbox manager*  
*docker-machine create -d virtualbox node1*  
*docker-machine create -d virtualbox node2*

Output :

[](https://1.bp.blogspot.com/-kInw9m6mXeI/WY_Zo1GSXlI/AAAAAAAAC4o/2h7GsVp5UpMowxuptekHvJ2cPd8pg706wCLcBGAs/s1600/docker-machine%2Bls.png)

**Setup docker swarm cluster pada node manager**

Login ke dalam docker swarm manager menggunakan docker-machine.

*docker-machine ssh manager*

Create initial sebagai master dengan ip dari host docker swarm manager 192.168.99.100

*docker swarm init --advertise-addr 192.168.99.100*

Setelah di setup, akan muncul token yang digunakan untuk join node ke docker swarm manager.

*docker swarm join --token SWMTKN-1-2rqhmivd1tx2c3pdw50x383tzf0v486kej1or3rk6ecjy31476-3cz5305q7mc2sfnvervpd8bmy 192.168.99.100:2377*

Output :

[](https://2.bp.blogspot.com/-qJcCiPMoXkk/WY_chMPh2iI/AAAAAAAAC40/4fb0THPCJmglKcE2uSnDT-jAy4X9rCrOgCLcBGAs/s1600/docker%2Bswarm%2Bcreate%2Bcluster.png)

**Setup docker worker untuk join ke cluster**

Login ke dalam docker node1 menggunakan docker-machine.

*docker-machine ssh node1*

Jalankan command dari swarm manager pada node 1 dan node 2 untuk join node pada cluster.  
  
Output :

[](https://1.bp.blogspot.com/-L9iwkbumIrY/WY_fNUwzuZI/AAAAAAAAC5A/4uY0HwML-_UbAM1ZRvlpQkK4v_AfYyKFgCLcBGAs/s1600/docker%2Bswarm%2Bworker%2B1.png)

[](https://4.bp.blogspot.com/-y7pk4pySJGM/WY_fTH3zXBI/AAAAAAAAC5E/b1OuLQrT8BYsM3v4O1-W8GpBS9FzLSNMwCLcBGAs/s1600/docker%2Bswarm%2Bworker%2B2.png)

Sempai disini semua node sudah join pada cluster.

[](https://3.bp.blogspot.com/-XAYsK8t6D98/WY_gIdazh8I/AAAAAAAAC5Q/C1kxfidFRaAfN1Sbdi9IVCiDChCYJ1u6ACLcBGAs/s1600/docker%2Bnode%2Bls.png)

**Create service pada docker swarm**

Pada contoh kali ini saya akan membuat sebuah service http menggunakan nginx dengan jumlah container 5.

*docker service create --name web --replicas=5 -p 8080:80 nginx:alpine*

Pastikan container sudah terbuat.

*docker service ls*

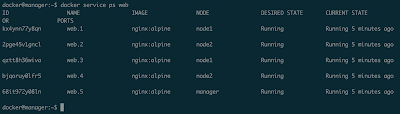
Output :

[](https://3.bp.blogspot.com/-Cfvf2ntNEuQ/WZAEdA-9C_I/AAAAAAAAC5g/bWWKYsDx_F0P76ud3qgVR1ieVq6EJ7U7ACLcBGAs/s1600/docker%2Bswarm%2Bls.png)

Kita juga dapat melihat proses orchestration untuk service web yang running pada node lain.

*docker service ps web*

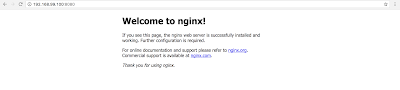
Output :

[](https://2.bp.blogspot.com/-TF7PZwfQu50/WZAFMk3jonI/AAAAAAAAC5o/oubqslcScLYHThfwQcgzPLCh-wyEV7z6wCLcBGAs/s1600/docker%2Bservice%2Bps.png)

Sampai disini web service sudah running, pastikan sudah dapat di akses via browser.

*http://192.168.99.100:8080*

Output:

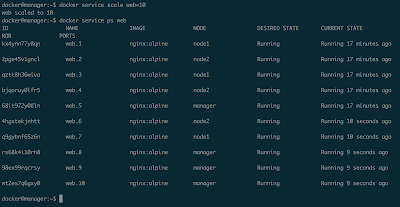
[](https://4.bp.blogspot.com/-DGo7YgI4KQU/WZAGb4QTETI/AAAAAAAAC50/hZuSscjNjlwKXkVII8915vzJF0YDl1UEgCLcBGAs/s1600/webservice%2Bnginx.png)

**Scaling up dan Scaling down**

Seperti yang sudah saya sebutkan di atas, salah satu fitur dari docker swarm adalah scaling. Pada contoh kali ini, saya akan scale jumlah container untuk service web.

*docker service scale web=10*

Output :

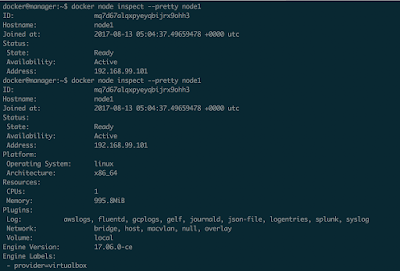
[](https://3.bp.blogspot.com/-DBasZgrdi5o/WZAHnu4BxSI/AAAAAAAAC6A/p8UqSebmk5E_RjtfVPmr7wWyGanYduaaQCLcBGAs/s1600/scaling%2Bup%2Bdocker%2Bswarm.png)

**Inspect node via swarm manager**

Pada docker swarm manager kita dapat melihat informasi mengenai node worker.

*docker node inspect --pretty node1*

Output :

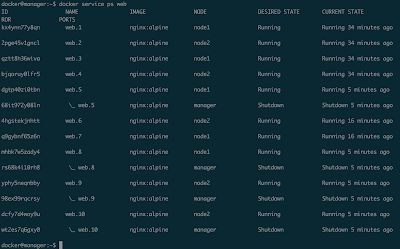
[](https://2.bp.blogspot.com/-Iqe9jbNPXZY/WZAJHFXoE2I/AAAAAAAAC6M/5HMr0oUpD5wp5A72qFPTABbjR5WHNGhTQCLcBGAs/s1600/node%2Binspect.png)

**Setup mode drain untuk maintenance**

Jika kita ingin melakukan update pada host tanpa mengganggu container yang sedang running dapat menggunakan mode drain. Mode drain akan memindahkan seluruh container ke host yang sedang aktif. Pada contoh kali ini saya kan mencoba setup mode drain pada docker swarm manager.

*docker node update --availability drain manager*

Output:

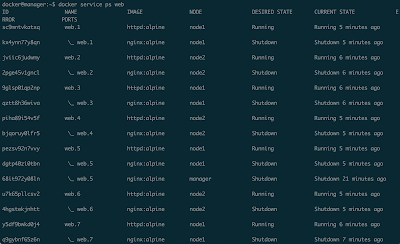
[](https://3.bp.blogspot.com/-w_WECdrOrpI/WZALnCmiVbI/AAAAAAAAC6Y/laAln5A0gN0tGf9XKD5e1enUPME5G0IxwCLcBGAs/s1600/docker%2Bdrain%2Bmode.png)

Dari gambar diatas, terlihat container yang running pada swarm manager sudah pindah ke node1 dan node2.

Selanjutnya kita dapat update image melalui swarm manager. Sebagai contoh saya akan update image dari nginx ke httpd.

*docker service update --image httpd:alpine web*

Output:

[](https://1.bp.blogspot.com/-MmomFAhzFCs/WZAPS_XdgfI/AAAAAAAAC6k/bdT17IqgcOASSAO2WHlVNP6FqN6aU3GhwCLcBGAs/s1600/Update%2Bimage%2Bdocker%2Bswarm.png)

Dari gambar diatas container sudah menggunakan images httpd.

**Remove service**

Jika kita ingin menghapus service dapat menggunakan perintah berikut.

*docker service rm web*

Demikian tutorial docker kali ini mengenai [container orchestration dengan docker swarm](http://www.dimasrio.com/2017/08/container-orchestration-docker-swarm.html). Docker swarm sangat powerfull, akan tetapi jika sobat ingin mencoba orchestration yang lebih menarik bisa mencoba [Kubernetes](https://kubernetes.io/).