Docker

Docker basic

Docker Compose

Docker Machine

Docker Swarm Classic

Docker Swarm Mode

Docker Swarm management

Docker volume orchestration

Docker Swarm on AWS

Stateful load-balancing in swarm

Centralized logging in swarm

Metrics and Monitoring in swarm

Auto-scaling swarm

Kafka with ELK on swarm

Java EE application with docker
Itt egy tipikus, produkciós docker architektúrát mutatunk be egy két lábas JBoss cluster-el, de swarm nélkül

Java EE application with swarm
Egy lehetséges production grade swarm architektúra telepített Java EE alkalmazás kialakítását mutatjuk be.

Contents

Docker on Fedora 31

https://www.reddit.com/r/linuxquestions/comments/dn2psl/upgraded_to_fedora_31_docker_will_not_work/
https://fedoraproject.org/wiki/Changes/CGroupsV2<br A Fedora31-ben bevezették a CGroupsV2-t amit a docker még nem követett le, ezért a docker a CGroupsV2-vel nem fog m?ködni, ki kell kapcsolni.

1-

vim /etc/default/grub 2- Add Line below in GRUB_CMDLINE_LINUX systemd.unified_cgroup_hierarchy=0

```
GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
GRUB_CMDLINE_LINUX="resume=/dev/mapper/fedora_localhost--live-swap rd.lvm.lv=fedora_localhost-live/root rd.luks.uuid=luks-42aca868-45a4-438e-
GRUB_DISABLE_RECOVERY="true"
GRUB_ENABLE_BLSCFG=true
```

3- Then :

```
# grub2-mkconfig
```

4- Restart your PC

Swarm Classic VS Swarm mode

Docker has been innovating at quite a dramatic pace, and focussing on making their technology easier to deploy, and applicable for a wider range of use cases. One of the features that has received the highest level of focus is Clustering/Orchestration. In Docker language, that means Swarm.

source: https://www.linkedin.com/pulse/docker-swarm-vs-mode-neil-cresswell/

Swarm classic

Prior to Docker 1.12 Swarm (Classic) existed as a standalone product, it relied on a complicated setup of external service discovery systems (eg consul) and a dedicated set of containers which ran as the swarm controllers. Load balancing network traffic across containers required external load balancers, and these needed to be integrated with service discovery to function correctly. Standalone Docker hosts were members of a swarm cluster, and the swarm controllers presented the pooled capacity from all hosts as a single ?virtual? docker host. By presenting the swarm cluster as a virtual docker host meant that the way you interacted with Swarm was exactly the same way you interacted with a standalone host (docker run, docker ps, docker images, docker volumes), you just directed the commands (using ?H=tcp://) at the swarm master IP:Port instead of individual swarm nodes.

A Docker 1.12-es verziója el?tt a Swarm (Classic) egy külön álló termék volt, nem volt része a docker engine-nek. A swarm-ot a docker engine-en futó swarm konténerekkel kellett létrehozni. Vo

Swarm mode

Since releasing Docker 1.12, and embedding Swarm Mode (I really wish they had called it something else to minimise confusion) into the core Docker engine, the functionality and management of swarm has altered dramatically. No longer does the cluster (pool of resources) emulate a virtual docker host, and no longer can you run standard docker engine commands against the swarm cluster, you now need to use specific commands (service create, service inspect, service ps, service ls, service scale etc). If you run Docker engine commands (docker ps) what is returned is a list of containers running on the Docker Swarm Master HOST (not the cluster). If you want to interact with containers that make up a swarm ?service?, you need to take multiple steps (service ps, to show the containers, and which host they are on, then change the focus of your docker commands to that host, connect to that host, and then issue the docker commands to manage the containers on that specific host/swarm member).

The key point of SwarmMode is that it is an overlay engine for running SERVICES, not Containers. In fact, a service actually comprises a number of tasks, with a task being a container and any commands to execute within the container (but a task might also be a VM in the future).

One of the major enhancements in Swarm mode is the load balancing, which is now built-in; now when you publish service, exposed ports will automatically be load balanced across the containers (tasks though, remember) that comprise that service. You don?t need to configure any additional load balancing. This change makes it incredibly easy to, say for instance, scale a nginx service from 1 worker task (container) to 10.

So, if you are using Swarm mode in Docker 1.12, you need to stop thinking about Containers (and trying to interact with the containers that make up a service) and rather, manage the service and tasks.

In Portainer.io, we exhibit the same behaviour as above, so if you click on ?containers? you will only see the container