docker

Docker Machine

- docker-machine
  - create --driver <drivername> \ <machine-name> — *creates a new docker host*
  - start <machine-name> — *starts a stopped machine*
  - ssh — *runs a command or interactive ssh session on the machine*
  - stop <machine-name>
  - inspect — *displays machine configuration details*
  - ls — *list all machines you've created and their status*
  - ip <machine-name> — *returns a paecific host's IP add*
  - env <machine-name> — *outputs the configuration command you should use*
  - rm <machine-name> — *removes a machine*
  - upgrade — *upgrades a machine's Docker client to the lastest stable release*
- Example
  - docker-machine create \ --driver virtualbox \ <machine-name> — VirtualBox — *local VM*
  - docker-machine create \ --driver digitalocean \ --digitalocean-access-token xxxx \ <machine-name> — Digital Ocean — *cloud provider*

Docker Cloud

Universal Control Plane

Docker Toolbox

Component Projects

Docker Engine

- *start* — $ docker run
  - -d — deamon
  - -i — interactive
  - -t — tty
  - -p — [hostPortNum]:[ContainerPortNum]
  - H — host:Port
  - command test example
    - $ docker run ubuntu /bin/echo 'Hello world' Hello world
    - $ docker run -t -i ubuntu /bin/bash root@af8bae53bdd3:/#
- log
- Get a new image — $ docker pull <image-name>
- Finding Images — $ docker seach <image-name>
- Remove Images
  - $ docker rm `docker ps -aq`
  - $ docker rmi -f `docker images | awk 'NR == 2 {print $3}'`
- Creating Images
  - Update and committing an image — $ docker commit -m "your messages" -a "commiter name" \ <container id> <direcotry>/<name>:<tag>
  - Building from a Dockerfile
  - need samples! https://docs.docker.com/engine/userguide/containers/dockerimages
- ps
  - -q — show id only
  - -a — all-images
- attach
  - ID
  - コンテナへSSH接続
- *stop*
  - **$ docker stop <id>**
  - **$ docker stop `docker ps | awk 'NR ==2 {print $1}'`**

Docker Swarm

Docker Compose

Docker Hub

Docker Trusted Registry

Glossary

Install

- CentOS7
  - Prerequisites — kernel 3.10 at minimum
  - *1. Update CentOS* **$ sudo yum update**
  - Install
    - 2 **$ sudo tee /etc/yum.repos.d/docker.repo <<-'EOF' [dockerrepo name=Docker Repository baseurl=https://yum.dockerproject.org/repo/main/centos/$releasever/ enabled=1 gpgcheck=1 gpgkey=https://yum.dockerproject.org/gpg EOF**
    - 3 **$ sudo yum install docker-engine**
    - 4 $ sudo service docker start
    - 5 test $ sudo docker run hello-world
  - Create a docker group
  - Start the docker deamon at boot — $sudo chkconfig docker on
  - Uninstall
    - 1 **$ sudo yum list installed | grep docker**
    - 2 $ sudo yum remove docker-engine.x86_64
    - 3 delete all images, containers, and volumes $ rm -rf /var/lib/docker

Need Comments on first line of each boxes!

About

- What is Docker?
- Components
  - Docker — the open source containerization platform
  - Docker Hub — *Saas platform for sharing and managing Docker containers*
- Architecture
  - Docker deamon — *runs on a host machine, user doesn't directly interact with the deamon but instead through client*
  - Client — *the primary user interface to Docker, accept commands from the user and communicates back and forth with Docker deamon*
  - inside Docker
    - Images — *read only template*
    - Containers — *similar to directory, holds everything that is needed for an app to run. Each container is created from a Docker image.*
  - Registries — *public or private stores from which you upload or download images*
- Process
  - Pulls the ubuntu image
  - Creates a new container
  - Allocates a filesystem and mounts a read-write layer
  - Allocates a network bridge interface
  - Set up an IP adress
  - Excute a process that you specify
  - Caputures and provides application output
- Underlying technology
  - Namespaces — *provide the isolated workspace for container, each of them runs in its own namespace and doesn't access outside of it*
    - Example namespaces
      - pid — *process isolation (PID: Process ID)*
      - net — *managing network interface*
      - icp — *managing access to IPC resource (ICP: InterProcess Communication)*
      - mnt — *managing mount-points*
      - uts — *isolating kernel and version identifiers (UTS: Unix Timesharing System)*
  - Control Groups (cgroups) — *allows Docker to share available hardware resources to containers and if required set up limits and costraints. For example, limiting memory available to a specific containter.*
  - Union file systems(UnionFS) — operate by creating layers, making them very lightweight and fast. Docker can make use of several UnionFS: AUFS, btrfs, vfs, and DeviceMapper
  - Container format — The default format is libcontainer, in future, may support BSD Jails or Solaris Zones.