LinuxDocker使用手册

[toc]

Docker介绍

Docker的应用场景Web

- 应用的自动化打包和发布。
- 自动化测试和持续集成、发布。
- 在服务型环境中部署和调整数据库或其他的后台应用。
- 从头编译或者扩展现有的OpenShift或Cloud Foundry平台来搭建自己的PaaS环境。

Docker 的优点

- 持续部署与测试
  - Docker可以确保从开发到产品发布整个过程环境的一致性。便于部署和开发测试。
- 多云平台
  - 可移植性
- 环境标准化和版本控制
  - 可以方便的进行版本管理
- 隔离性
- 安全性

Docker 的主要用途

（1）提供一次性的环境。比如，本地测试他人的软件、持续集成的时候提供单元测试和构建的环境。 （2）提供弹性的云服务。因为 Docker 容器可以随开随关，很适合动态扩容和缩容。 （3）组建微服务架构。通过多个容器，一台机器可以跑多个服务，因此在本机就可以模拟出微服务架构

https://blog.csdn.net/u013007900/article/details/62219169
https://blog.csdn.net/xiangxizhishi/article/details/79441391

Docker架构

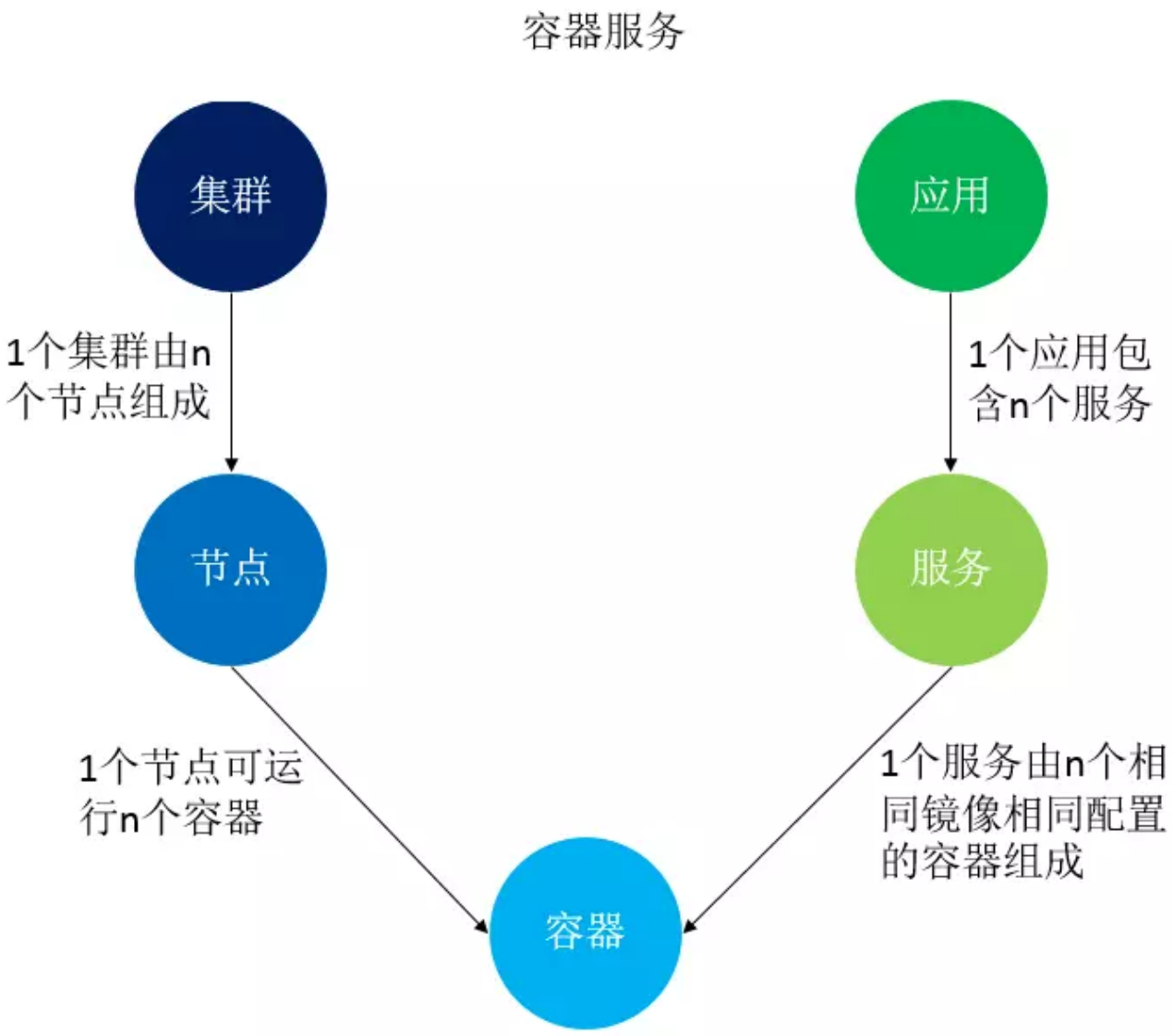
- Docker基本概念https://blog.csdn.net/omnispace/article/details/79778544
- Docker 介绍以及其相关术语、底层原理和技术:https://blog.csdn.net/zxygww/article/details/53709106

Docker 使用客户端-服务器 (C/S) 架构模式，使用远程API来管理和创建Docker容器。 Docker 容器通过 Docker 镜像来创建。容器与镜像的关系类似于面向对象编程中的对象与类

Docker术语

集群 一个集群指容器运行所需要的云资源组合，关联了若干服务器节点、负载均衡、专有网络等云资源。 **节点** 一台服务器（可以是虚拟机实例或者物理服务器）已经安装了 Docker Engine，可以用于部署和管理容器；容器服务的 Agent 程序会安装到节点上并注册到一个集群上。集群中的节点数量可以伸缩。 **容器** 一个通过 Docker 镜像创建的运行时实例，一个节点可运行多个容器。 **镜像** Docker 镜像是容器应用打包的标准格

式，在部署容器化应用时可以指定镜像，镜像可以来自于 Docker Hub，阿里云容器 Hub，或者用户的私有 Registry。镜像 ID 可以由镜像所在仓库 URI 和镜像 Tag（缺省为 latest）唯一确认。 **编排模板** 编排模板包含了一组容器服务的定义和其相互关联，可以用于多容器应用的部署和管理。容器服务支持 Docker Compose 模板规范并有所扩展。 **应用** 一个应用可通过单个镜像或一个编排模板创建，每个应用可包含1个或多个服务。 **服务** 一组基于相同镜像和配置定义的容器，作为一个可伸缩的微服务。 **关联关系**



Docker使用

Docker安装

安装说明:

1. CentOS6.10 环境，要求6.8+
2. Docker版本 1.7.1

步骤1：配置下载镜像 `docker.repo` 文件

```
[root@yinsho ~]# cat /etc/yum.repos.d/docker.repo
[dockerrepo]
name=Docker Repository
baseurl=https://yum.dockerproject.org/repo/main/centos/6
```

```
enabled=1
gpgcheck=1
gpgkey=https://yum.dockerproject.org/gpg
```

步骤2：重建元数据

```
yum clean all
yum makecache
```

步骤3：安装docker

```
yum install docker-engine  -y
```

问题：docker-engine conflicts

```
# 如果执行报错 docker-engine conflicts with xxxxx
# 先卸载 docker，再安装 docker-engine
yum remove docker -y
```

步骤4：启动docker服务

```
service docker start
chkconfig docker on
```

安装参考链接： https://blog.csdn.net/abcd_d_/article/details/53996791

yum update -y 如果报错 No module named yum 参考：https://www.cnblogs.com/clover-siyecao/p/5650893.html

rpm -Uvh http://ftp.riken.jp/Linux/fedora/epel/6Server/x86_64/epel-release-6-8.noarch.rpm

yum remove docker -y yum install -y docker-io

Docker常用命令

```
# Docker服务启停
service docker restart
# 创建一个容器
docker run -it -v /docker_test:/yufei --name yufei_6 centos
参数说明
-i：允许我们对容器内的 (STDIN) 进行交互
-t：在新容器内指定一个伪终端或终端
-v：是挂在宿机目录， /docker_test是宿机目录，/yufei是当前docker容器的目录，宿机目录必须
```

```
是绝对的。
--name：是给容器起一个名字，可省略，省略的话docker会随机产生一个名字
# 查看docker容器列表(运行中)
docker ps
# 查看所有的docker容器列表
docker ps -a
# 启停容器
docker start yufei_01
docker stop yufei_01
docker restart yufei_01
# 查看容器的日志
docker logs -f yufei_01
# 删除容器，如果容器在运行需要先停止
docker stop yufei_01
docker rm yufei_01
# 删除所有容器
docker rm $(docker ps -a -q)
```

Docker命令大全

- Docker命令大全:http://www.runoob.com/docker/docker-command-manual.html
- 容器生命周期管理
    - docker run 创建一个新的容器并运行一个命令
    - docker restart 重启容器
    - docker kill -s KILL mynginx 杀掉一个运行中的容器。 -s :向容器发送一个信号
    - docker rm ：删除一个或多少容器
    - docker pause :暂停容器中所有的进程。docker unpause :恢复容器中所有的进程。
    - docker create ：创建一个新的容器但不启动它
    - docker exec ：在运行的容器中执行命令
- 容器操作
    - docker ps：列出容器
    - docker inspect：获取容器/镜像的元数据。
    - docker top :查看容器中运行的进程信息，支持 ps 命令参数
    - docker attach :连接到正在运行中的容器
    - docker events：从服务器获取实时事件
    - docker logs：获取容器的日志
    - docker wait：阻塞运行直到容器停止，然后打印出它的退出代码
    - docker export :将文件系统作为一个tar归档文件导出到STDOUT
    - docker port :列出指定的容器的端口映射，或者查找将PRIVATE_PORT NAT到面向公众的端口。
- 容器rootfs命令
    - docker commit :从容器创建一个新的镜像。

    - docker cp :用于容器与主机之间的数据拷贝
    - docker diff：检查容器里文件结构的更改

- 镜像仓库

    - docker login：登陆到一个Docker镜像仓库，如果未指定镜像仓库地址，默认为官方仓库 Docker Hubdocker
    - docker logout：登出一个Docker镜像仓库，如果未指定镜像仓库地址，默认为官方仓库 Docker Hub
    - docker pull：从镜像仓库中拉取或者更新指定镜像
    - docker push：将本地的镜像上传到镜像仓库,要先登陆到镜像仓库
    - docker search：从Docker Hub查找镜像

- 本地镜像管理

    - docker images：列出本地镜像
    - docker rmi：删除本地一个或多少镜像
    - docker tag：标记本地镜像，将其归入某一仓库
    - docker build 命令用于使用 Dockerfile 创建镜像
    - docker history：查看指定镜像的创建历史
    - docker save：将指定镜像保存成 tar 归档文件
    - docker import：从归档文件中创建镜像

- info|version

    - docker info：显示 Docker 系统信息，包括镜像和容器数。
    - docker version :显示 Docker 版本信息

docker命令样例

```
# 启动镜像时，设定docker系统参数 - 修改系统参数 生效
docker run -it -d -p 80:80 -p 3000:3000 -p 8080:8080 -p 9200:9200 -p 5600:5602 -p
5601:5601 --env=vm.max_map_count=262144 fdm_docker_ok /bin/bash

镜像的导入导出
# 导出镜像 images
sudo docker images REPOSITORY TAG IMAGE ID CREATED VIRTUAL SIZE
sudo docker save -o /home/user/images/ubuntu_14.04.tar ubuntu:14.04
# 导入镜像
sudo docker load --input ubuntu_14.04.tar
sudo docker load &lt; ubuntu_14.04.tar


镜像删除
docker rmi images_id


容器模块

查看容器的环境变量
```

```
* 使用docker inspect命令来查看
# docker inspect <CONTAINER-NAME> OR <CONTAINER-ID>
* 使用docker exec -it <CONTAINER-NAME> OR <CONTAINER-ID> env查看

docker镜像启动命令 - 镜像启动每次容器ID都会变更
docker run -it -d -p 50001:22  -p 80:80 -p 3000:3000 -p 8080:8080 -p 9200:9200 -p
5600:5602 -p 5601:5601 --env=vm.max_map_count=262144  fdm_docker /bin/bash

docker 容器启动命令
docker container start 4d15e75d1116

进入docker容器中
docker exec -it fa6e4ac38997 /bin/bash

查看容器ID
# 查看当前运行的容器
docker ps
# 查看历史所有的容器
docker ps -a
可以通过启动历史容器，并进入

保存容器为镜像
docker ps -a
可以通过启动历史容器，并进入


容器的导入导出
# 容器的导入
docker import fdm_docker.tar.gz  fdm_docker
# 将容器保存为镜像
docker commit 8e613c207029 fdm_docker02
```

其他模块

```
其他模块
使用xshell登录docker -- 方式1 进入docker虚拟机
ssh 192.168.99.100 # docker的IP ，通过查看docker虚拟机的ip登入docker界面
用户名默认是：docker
密码默认：tcuser
端口：22

# 涉及安装openssh-server
http://blog.csdn.net/vincent2610/article/details/52490397
yum install -y openssh-server
vi /etc/ssh/sshd_config
将PermitRootLogin的值从withoutPassword改为yes
登出容器，并将容器保存为新的镜像。
关闭原有容器，用新镜像生成新的容器
```

使用xshell登录docker -- 方式2 docker进入容器
1.安装配置好sshd，并进入后重启服务。
2.docker run 通过 -p 50001:22，将22端口映射到50001
3.打开cmd，查看windwosIP，例如 192.168.43.25
4.ssh 192.168.43.25 50001
或者 ssh 192.168.43.25 -p 50001
即可登录进入容器中

配置容器系统参数 - 需要从docker上配置
sysctl: setting key "vm.max_map_count": Read-only file system 问题
参考链接：https://stackoverflow.com/questions/41064572/docker-elk-vm-max-map-count
说明：由于docker是最高层级，容器是最低层级，部分系统参数需要从docker中修改，否则权限不足
解决方法：
```
docker-machine create -d virtualbox default # 创建默认虚拟机，涉及需要开启windows功能Hyper-V
docker-machine start 机器名称 # 出现蓝屏问题，暂时未解决 PASS
docker-machine ssh
sudo sysctl -w vm.max_map_count=262144
```
配置容器系统参数 - 需要从docker上配置 -- 问题1：登陆docker界面，但是docker中virtualbox不存在。

```
# 查看已有的docker-machine机器名称
docker-machine ls
# 进入docker
docker-machine ssh 机器名称ID
```

错误：Error: No machine name(s) specified and no "default" machine exists
错误原因：本机没有machine，需要创建
```
# 创建docker机器
docker-machine create -d virtualbox default 机器名称
```

错误：Error with pre-create check: "This computer is running Hyper-V. VirtualBox won't boot a 64bits VM when Hyper-V is activated. Either use Hyper-V as a driver, or disable the Hyper-V hypervisor. (To skip this check, use --virtualbox-no-vtx-check)
错误原因：docker的virtualbox和已有的虚拟机VMware或virtualBox冲突
参考链接：http://blog.csdn.net/qwsamxy/article/details/50533007/
解决方法：
```
bcdedit /set hypervisorlaunchtype off
bcdedit /set hypervisorlaunchtype auto
```

```
bcdedit /copy {current} /d "Windows 10 (开启 Hyper-V)"
bcdedit /set {XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX} hypervisorlaunchtype auto
```


切换用户执行脚本
```
su - test -c "pwd"
```

```
删除images后，释放空间：（会删除未使用的的容器和已删除的镜像-慎重）
docker system prune -a



## docker-ce容器管理页面

参考链接：https://www.cnblogs.com/myzony/p/9071210.html

pass
CentOS7可用？
```

参考链接

docker基础命令：https://www.server110.com/docker/201411/11122.html docker run 参数：http://www.runoob.com/docker/docker-run-command.html docker官方英文文档：https://docs.docker.com/ docker中文文档网站：http://www.docker.org.cn/ 第一本docker书籍：https://download.csdn.net/download/qq_21165007/10276074

docker问题记录

问题：Repository dgraph/dgraph already being pulled by another client. Waiting.

解决方法：重启服务

```
[root@WOdocker pull dgraph/dgraph:latest
Repository dgraph/dgraph already being pulled by another client. Waiting.

[root@WOM ~]#
[root@WOM ~]# service docker restart
停止 docker:                                               [确定]
Starting docker:                                          [确定]
[root@WOM ~]#
[root@WOM ~]# docker pull dgraph/dgraph:latest




[root@WOM ~]# docker pull dgraph/dgraph:latest
latest: Pulling from dgraph/dgraph

f2b818b26f75: Pulling fs layer
c87298e9b6ec: Pulling fs layer
d29d3718cea9: Pulling fs layer
55982ec1ed3b: Pulling fs layer
a5019d93caef: Pulling fs layer
cea85299b18b: Pulling fs layer
4696dbf656b6: Pulling fs layer
c2c5bc4dfb3f: Pulling fs layer
bd8f9f1a25f5: Pulling fs layer
```

```
ca927ff9c37d: Pulling fs layer
8e51752bd503: Pulling fs layer
c58a4ff12da9: Pulling fs layer
Pulling repository dgraph/dgraph
Tag latest not found in repository dgraph/dgraph # 1.说明标签错误
[root@WOM ~]# docker search dgraph  # 2.搜索这个镜像，看是否可以找到标签
NAME                                       DESCRIPTION
         STARS     OFFICIAL   AUTOMATED
dgraph/dgraph                              Docker image for Dgraph
(https://github.co...   22
arubeh/dgraph
         1
jalberto/dgraph
         0
ibbd/dgraph                                IBBD
Dgraph                                     0                    [OK]
dancompton/dgraph
         0
euforia/dgraph
         0
dotf/dgraph
         0
jonrmayer/dgraph                           added another volume to
dgraph                  0                    [OK]
dotf/dgraphzero
         0
priyanshujain/dgraph
         0
alexmilowski/dgraph
         0
doc2run/dgraph                             Dgraph
DB                                       0
demandjump/dgraph
         0
bnjainonday198v/dgraphicsdesignsoftw4282   3D Graphics Design Software Download #
Fre...   0
akshaydeo/mnet                             Kafka, Cassandra, Redis, DGraph,
Postgresql     0
taylorsmithgg/dgraph
         0
qnib/plain-dgraph                          Plain image holding Dgraph, a graph
databa...   0                    [OK]
# 3.去官网查看标签  查询URL为:  https://hub.docker.com/r/【镜像名】/tags/
# https://hub.docker.com/r/dgraph/dgraph/tags/
```

问题:Segmentation Fault or Critical Error encountered

提示：Segmentation Fault or Critical Error encountered. Dumping core and aborting. Aborted 解答：安装错误安装docker了，应该安装docker-io

问题:docker-io-1.7.1-2.el6.x86_64

提示：Transaction Check Error: file /usr/bin/docker from install of docker-io-1.7.1-2.el6.x86_64 conflicts with file from package docker-1.5-5.el6.x86_64 解答：这个是因为先装了docker，再装docker-io后的结果，解决方法是yum remove docker后再yum install docker-io即可。

问题:/var/run/docker.sock: no such file or directory

提示：Get http:///var/run/docker.sock/v1.19/images/search?term=centos: dial unix /var/run/docker.sock: no such file or directory. Are you trying to connect to a ? 解答：docker没有启动，/etc/init.d/docker start

参考资源

Docker 是一个开源的应用容器引擎，基于 Go 语言 并遵从Apache2.0协议开源。 Docker 可以让开发者打包他们的应用以及依赖包到一个轻量级、可移植的容器中，然后发布到任何流行的 Linux 机器上，也可以实现虚拟化。 容器是完全使用沙箱机制，相互之间不会有任何接口（类似 iPhone 的 app）,更重要的是容器性能开销极低。

Docker参考链接：

- Docker官方中文网：http://www.docker.org.cn/
- Docker官网：https://www.docker.com/
- Docker菜鸟教程：http://www.runoob.com/docker/docker-tutorial.html