Docker 安装部署_本地集市

目录

一、docker 部署：

部署 docker 参考链接：https://blog.csdn.net/u010046908/article/details/79553227

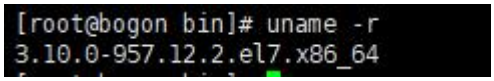
1、操作系统建议：

宿主机操作系统建议 centos7 以上。

2、系统内核版本要求：

docker 要求 CentOS 系统的内核版本高于 3.10，验证服务器 CentOS 版本是否支持 docker，通过“uname -r”命令查看当前的内核版本

例如本地服务器环境：



3、安装 docker 并启动：

yum update -y　　//升级所有包，系统版本和内核，改变软件设置和系统设置

【注】不升级还没遇到问题，保险起见，建议执行该升级命令

yum -y install docker　　//安装 docker

4、启动 docker 服务：

systemctl start docker　　//启动 docker 服务

5、查看 docker 服务状态：

systemctl status docker

```
[root@mpp144 ~]# systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor preset: disabled)
   Active: active (running) since 三 2020-06-24 11:44:54 CST; 2 weeks 1 days ago
     Docs: https://docs.docker.com
 Main PID: 26994 (dockerd)
   Memory: 37.1M
   CGroup: /system.slice/docker.service
           ├─19668 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 5000 -container-ip 172.17.0.2 -containe
           └─26994 /usr/bin/dockerd -g /home/docker -H fd:// --containerd=/run/containerd/containerd.sock

7月 06 19:44:53 mpp144 dockerd[26994]: time="2020-07-06T19:44:53.849343225+08:00" level=info msg="Attempting next...lie
7月 06 19:47:40 mpp144 dockerd[26994]: time="2020-07-06T19:47:40.759980989+08:00" level=info msg="ignoring event"...ele
7月 07 11:58:46 mpp144 dockerd[26994]: time="2020-07-07T11:58:46.183339252+08:00" level=info msg="Layer sha256:21...ed u
7月 07 11:58:54 mpp144 dockerd[26994]: time="2020-07-07T11:58:54.474737721+08:00" level=info msg="Layer sha256:21...ed u
7月 07 11:59:01 mpp144 dockerd[26994]: time="2020-07-07T11:59:01.296765695+08:00" level=info msg="Layer sha256:21...ed u
7月 07 12:00:04 mpp144 dockerd[26994]: time="2020-07-07T12:00:04.303410707+08:00" level=info msg="Attempting next...lie
7月 07 13:33:51 mpp144 dockerd[26994]: time="2020-07-07T13:33:51.797739155+08:00" level=warning msg="Error gettin...lie
7月 07 13:33:51 mpp144 dockerd[26994]: time="2020-07-07T13:33:51.797795055+08:00" level=info msg="Attempting next...lie
7月 07 13:39:03 mpp144 dockerd[26994]: time="2020-07-07T13:39:03.377537291+08:00" level=info msg="ignoring event"...ele
7月 07 14:15:41 mpp144 dockerd[26994]: time="2020-07-07T14:15:41.316779630+08:00" level=info msg="ignoring event"...ele
```

也可以 docker version

```
[root@mpp144 ~]# docker version
Client:
 Version:           18.09.6
 API version:       1.39
 Go version:        go1.10.8
 Git commit:        481bc77156
 Built:             Sat May  4 02:34:58 2019
 OS/Arch:           linux/amd64
 Experimental:      false

Server: Docker Engine - Community
 Engine:
  Version:          18.09.6
  API version:      1.39 (minimum version 1.12)
  Go version:       go1.10.8
  Git commit:       481bc77
  Built:            Sat May  4 02:02:43 2019
  OS/Arch:          linux/amd64
  Experimental:     false
[root@mpp144 ~]#
```

二、镜像拉取：

docker load -i 镜像文件

docker load -i 8.7.tar

--input , -i : 指定导入的文件

```
[root@localhost docker]# docker load -i 8.7.tar
89169d87dbe2: Loading layer [==================================================>] 209.5 MB/209.5 MB
4376126ad640: Loading layer [==================================================>] 119.6 MB/119.6 MB
bc101704fe92: Loading layer [==================================================>] 98.05 MB/98.05 MB
18346fea92b4: Loading layer [==================================================>] 1.807 GB/1.807 GB
113ac67f53ea: Loading layer [==================================================>]   398 MB/398 MB
Loaded image: yh:8.7
```

验证是否导入成功：

docker images -a 列出所有本地镜像

```
[root@localhost docker]# docker images -a
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
yh                  8.7                 e14f108929bd        6 weeks ago         2.61 GB
[root@localhost docker]#
```

删除镜像 image

docker rmi 镜像 ID

```
[root@localhost bihome]# docker rmi 9269470fcc4f
Untagged: yh:9.1x
Deleted: sha256:9269470fcc4f51d0acdfa2a21495993d56d76bf7ca98d3b57b861e499fee80f3
Deleted: sha256:1be8acc8c162018a1704362b02f196b6bbae46b64ec5a07cf3c3c4f714f17c23
Deleted: sha256:10d131e9198009d2938c1b5087d6854351b4f66afd0277527871e0258f3a5259
Deleted: sha256:5ec991237ffeb3abbeff6f462573f80a2d3c2dbaf4a1a036dd0d622b99291de9
Deleted: sha256:1e49cbade1b770e01a33a17f4d7d382dd536aa53274adc86703033805d081750
Deleted: sha256:89169d87dbe2b72ba42bfbb3579c957322baca28e03a1e558076542a1c1b2b4a
[root@localhost bihome]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
[root@localhost bihome]#
```

三、配置容器：

1、创建并启动容器：

docker run -d --name client（容器名） --net=host -m=2048m --privileged=true -v /home/docker/bihome:/home/yh/Yonghong8.7/Yonghong/bihome yh:8.7

例子：

docker run -d --name yh87 --net=host -m=500m --privileged=true -v /home/docker/bihome:/home/yh/Yonghong8.7/Yonghong/bihome yh:8.7

```
[root@localhost docker]# docker run -d --name yh87 --net=host -m=500m --privileged=true -v /home/docker/bihome:/home/yh/Yonghong8.7/Yonghong/bihome yh:8.7
8475c7d765c84635fa596b0c04f5ef4dacfdefac39a8065bd2005baeae17c81e
```

【注】红色的部分可以自定义修改：

-d: 后台运行容器，并返回容器 ID；

--name client（容器名） ： 设置容器名称

--net="bridge": 指定容器的网络连接类型，支持 bridge/host/none/container:四种类型;

-m=2048m　　:　　分配给容器的最大内存

--privileged 容器将拥有访问主机所有设备的权限

-v　　用于把容器的数据卷映射到宿主机上，用法如：docker run -itd -v /data:/data centos bash 表示把容器的/data 目录映射到宿主机的/data 目录，左边是宿主机的目录，右边是容器里的目录

/home/docker/bihome　　:　　宿主机的 bihome 文件夹路径

yh:8.7　　: docker 镜像

验证是否创建成功：

docker ps -a ： 列出所有的 container（包含历史，即运行过的 container）

```
[root@localhost docker]# docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS               NAMES
8475c7d765c8        yh:8.7              "bash /home/yh/Yon..."   7 seconds ago       Up 6 seconds                            yh87
[root@localhost docker]#
```

出现 NAMES 为新创建容器名的即为创建成功

查看容器相关命令：

---

docker ps ： 列出当前所有正在运行的 container

docker ps -l ： 列出最近一次启动的 container

docker ps -q ： 列出最近一次运行的 container ID

删除容器

docker rm 容器 ID

```
[root@localhost docker]# docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS                      PORTS               NAMES
a877e4286f33        yh:8.7              "bash /home/yh/Yon..."   6 minutes ago       Exited (143) 11 seconds ago                     yh87
[root@localhost docker]# docker rm a877e4286f33
a877e4286f33
[root@localhost docker]# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
[root@localhost docker]#
```

2、停止 docker 容器：

docker stop 容器 ID 或容器名

例子：

docker stop yh87

```
[root@localhost dashboard]# docker stop yh87
yh87
[root@localhost dashboard]#
```

3、将已准备好的 bihome 文件夹内容复制到宿主机的 bihome 中

4、启动 docker 容器：

docker start 容器 ID 或容器名

```
[root@localhost bihome]# docker start yh87
yh87
```

docker 常规操作——启动、停止、重启容器方法参考链接：

https://blog.csdn.net/Michel4Liu/article/details/80889977

启动：docker start 容器 ID 或容器名

停止：docker stop 容器 ID 或容器名

重启：docker restart 容器 ID 或容器名 ；不管容器是否启动，直接重启容器

退出容器但不关闭容器：exit

【注】启动容器后，会自动启动永洪产品 tomcat

5、浏览器访问容器中的服务器：

四、修改容器 bihome 中配置文件：

1、进入容器：

docker exec -it 容器名或容器 ID bash

2、进入到镜像永洪路径 bihome 中：

cd /home/yh/Yonghong8.7/Yonghong/bihome

3、编辑配置文件：

vi bi.properties

dc.io.ip=<当前主机 ip>

vi global_bi.properties

dc.node.naming=<命名节点的主机 ip>

4、重启 docker 容器：

docker restart 容器 ID 或容器名

五、升级替换产品 jar 包：

docker cp product.jar client:/home/yh/Yonghong8.7/Yonghong/product

六、常见问题：

1、重启 docker，提示 dial tcp xxx.xxx.xxx.xxx:5000: connect: network is unreachable

解决方法：编辑文件/etc/network/interfaces 在最后添加 dns-nameservers 8.8.8.8

然后 systemctl restart networking.service。

2、是否可以在一台宿主机部署集群?

答案是肯定的，但是需要注意的是在拉取镜像（步骤三-1）时，需要注意容器名称及宿主机 bihome 存储路径不能冲突。

3、修改宿主机或容器中的 bihome 信息，重启容器后修改后的信息会恢复原始状态吗?

修改 bihome 信息是双向同步的，重启容器后即生效。