2021.3.13 企业级仓库 harbor 高可用方案

1. harbor 产品介绍

Harbor 是 VMware 公司开源的企业级 Docker Registry 项目,其目标是帮助用户迅速搭建一个企业级的 Docker Registry 仓库服务。它以 Docker 公司开源的 Registry 为基础，提供了管理 U。基于角色的访问控制(Role Based AccessControl)、AD/LDAP 集成、以及审计日志(Auditlogging)等企业用户需求的功能。通过添加一些企业必需的功能特性，例如安全、标识和管理等，扩展了开源 Docker Distribution。作为一个企业级私有 Registry 服务器，Harbor 提供了更好的性能和安全，以提升用户使用 Registry 构建和运行环境传输镜像的效率。

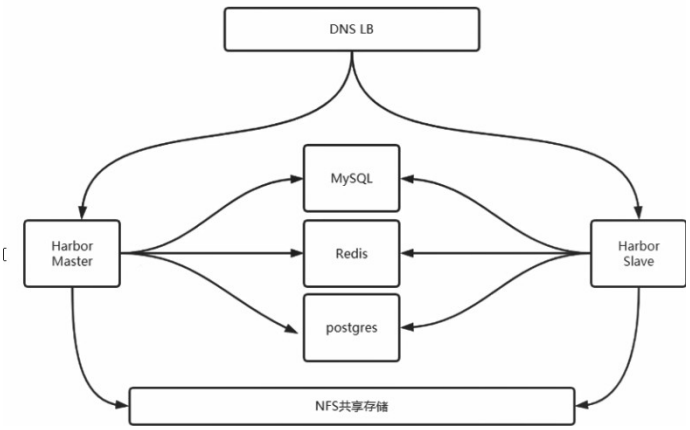
2. 高可用环境规划

注意：每台机器保证 2 核 2G

主机规划

操作系统	主机名	IP 地址	软件
CentOS7.x	master	192.168.200.111	docker-ce、docker-compose、Harbor-offline
CentOS7.x	slave	192.168.200.112	docker-ce、docker-compose、Harbor-offline
CentOS7.x	ldns-nfs	192.168.200.113	docker-ce、mysql、redis、postgres 、nfs、bind

拓扑图

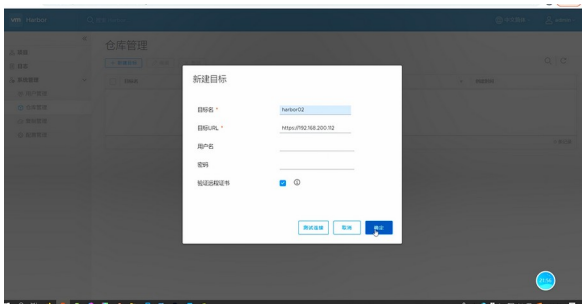


架构隐患：

1. NFS 有单点问题
2. 镜像太多导致占用大量的磁盘空间及镜像访问时磁盘的 IO 过大

解决方案：

1. NFS 做成高和用 NFS+DRBD+Heartbeat
2. 存储使用分布式文件系统（GlusterFS）
3. Harbor 支持主从复制（双向的+keepalived）



具体的自己摸索，和实验没关系。

3. 高可用环境部署

1）基础部署

```
[root@localhost ~]# iptables -F
[root@localhost ~]# setenforce 0
[root@localhost ~]# systemctl stop firewalld

[root@localhost ~]# hostname master
[root@localhost ~]# bash
[root@master ~]#

[root@localhost ~]# hostname slave
[root@localhost ~]# bash
[root@slave ~]#

[root@localhost ~]# hostname ldns-nfs
[root@localhost ~]# bash
```

2）部署 docker-ce 环境

安装 docker-ce（所有主机）

```
wget http://mirrors.aliyun.com/repo/Centos-7.repo
wget http://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo
yum -y install docker-ce
```

3）主从部署 compose

```
[root@master ~]# rz      /docker-compose
[root@master ~]# mv docker-compose  /usr/bin/
[root@master ~]# chmod  +x /usr/bin/docker-compose
[root@master ~]# docker-compose  --version
docker-compose version 1.21.1, build 5a3f1a3
```

4）搭建 nfs 服务端（nfs 服务器）

```
[root@nfs yum.repos.d]# mkdir  -p /data/nfs
[root@master ~]# rpm -q nfs-utils rpcbind  #查看依赖的软件包
nfs-utils-1.3.0-0.65.el7.x86_64
rpcbind-0.2.0-48.el7.x86_64
[root@master ~]# vim /etc/exports
```

```
/data/nfs 192.168.200.0/24(rw,no_root_squash)
[root@master ~]# systemctl restart  rpcbind  （rpcbind 先启动）
[root@master ~]# systemctl  restart  nfs  （nfs 后启动）
```

5）在 harbor 主和备上创建数据挂载目录，并安装 nfs 节点支持包

Harbor 主备的操作

```
[root@master ~]# rpm -q nfs-utils rpcbind  #查看依赖的软件包
nfs-utils-1.3.0-0.65.el7.x86_64
rpcbind-0.2.0-48.el7.x86_64
[root@slave ~]# mkdir  -p /data/storage
[root@slave ~]# mount  192.168.200.10:/data/nfs  /data/storage/  #挂载共享目录
[root@ldns-nfs ~]# vim /etc/fstab  #配置自动挂载
192.168.200.10:/data/nfs  /data/storage  nfs4  defaults,_netdev  0  0
```

6）nfs 主机准备共用容器

下载一个 redis 镜像

```
[root@master~]# rz / postgres.tar /prometheus.tar /mysql_5.6.tar /redis_alpine.tar
```

导入四个镜像

```
[root@nfs ~]# docker load < mysql_5.6.tar
[root@nfs ~]# docker load < postgres.tar
[root@nfs ~]# docker load < prometheus.tar
[root@nfs ~]# docker load < redis_alpine.tar
```

启动 redis 镜像，端口映射

```
[root@nfs ~]# docker run -dit --name redis_test -p 6379:6379 redis:alpine
e932b9d2c27b2350655f1a3115e1293825ef59a62e975763901307b21926b98e
```

启动镜像 postgres，端口映射

```
[root@nfs ~]# docker run -dit --name postgres_test -p 5432:5432 -e POSTGRES_PASSWORD=123123 postgres
cb3fd75b39c79a08173d267cb3cdb4bed37907ba7bf56d4d131809162305a1e1
```

启动镜像 mysql，端口映射

```
[root@nfs ~]# docker run -dit --name mysql_test -p 3306:3306 -e MYSQL_ROOT_PASSWORD=123123 mysql:5.6 --character-set-server=utf8
cf5aa20bb3d402879d79849e0b882b1cd3a36fc7ddf8219d0de2f4265c23e508
```

最后按照用途，我们分别给数据库改一下名称

```
[root@nfs ~]# docker rename postgres_test clair_db
[root@nfs ~]# docker rename mysql_test harbor_db
[root@nfs ~]# docker rename redis_test session
[root@nfs ~]# docker ps -a
```

4. harbor 配置（master）

1）导入数据

向 mysql_db 容器导入数据表（192.168.200.12），再解压的 harbor 目录里的 ha 目录下的 registry.sql 表导入到我们之前在 NFS 服务端上创建的 mysql 容器里

主和备的操作：注意上传 harbor 版本时选择低版本的（高版本有时不兼容）

```
[root@ma ~]# rz   /harbor-offline-installer-v1.5.0.tgz
[root@ma ~]# tar xf harbor-offline-installer-v1.5.0.tgz
[root@ma ~]# cd  harbor/
[root@ma harbor]# tree ha
ha
├── docker-compose.clair.tpl
├── docker-compose.clair.yml
├── docker-compose.tpl
├── docker-compose.yml        #需要安装的配置文件
├── registry.sql              #需要导入的 mysql 表
└── sample
    ├── active_active
    │   ├── check.sh
    │   └── keepalived_active_active.conf
    └── active_standby
        ├── check_harbor.sh
        └── keepalived_active_standby.conf

3 directories, 9 files
```

远程连接到：192.168.200.10（NFS 服务器端）的 3306 端口，导入表 registry.sql

```
[root@ma harbor]# yum -y install mysql
[root@ma harbor]# systemctl  start  mariadb
[root@ma harbor]# systemctl enable  mariadb
[root@ma harbor]# mysql -uroot -p123123 -h 192.168.200.10 -P3306  #远程登入
MySQL [registry]> source ha/registry.sql    #导入数据（注意需要绝对路径）
```

特别提示：如果导入表格出现如下错误

ERROR 1071 (42000) at line 284 in file: 'ha/registry.sql': Specified key was too long; max key length is 767 bytes

这时因为导入的表格建立的索引超过 mysql 默认的上线 767bytes>=154，因此我们需要修改导入的表

```
[root@ma harbor]# vim ha/registry.sql
220   repository varchar(254) NOT NULL,  #把原来的 256 改为 254
291   resource_name varchar(254),      #把原来的 256 改为 254
```

修改完之后再进行表格导入就不会报错了

```
MySQL [(none)]> source ha/registry.sql
MySQL [(none)]> show databases;
```

```
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| registry           |
+--------------------+
4 rows in set (0.00 sec)
```

2）修改 ha/docker-compose.yml 配置文件

```
[root@ma harbor]# vim ha/docker-compose.yml
19    - /data/storage:/storage:z    ：将原来的保存位置改为 nfs 挂载的位置，数据就保存到 nfs 中
```

修改 harbor.cfg 文件

```
[root@ma harbor]# vim harbor.cfg
7   hostname = www.crushlinux.com  #用户名
11  ui_url_protocol = https  #定义的协议
23  ssl_cert = /etc/ssl/harbor/www.crushlinux.com.crt    #证书
24  ssl_cert_key = /etc/ssl/harbor/www.crushlinux.com.key  #证书
68  harbor_admin_password = Harbor12345      #harbor 默认的密码
130 db_host = 192.168.200.10          #nfs 主机（数据库主机）
133 db_password = 123123    #数据库的密码
136 db_port = 3306          #数据库端口
139 db_user = root          #数据库的用户
145 redis_url = 192.168.200.10:6379  #redis 的地址和端口
150 clair_db_host = 192.168.200.10   # postgres 的 IP 地址
154 clair_db_password = 123123     # postgres 的密码
157 clair_db_port = 5432          # postgres 的端口
160 clair_db_username = postgres
163 clair_db = postgres
```

3）为 harbor 配置证书（要么买，要么自己颁发，但是颁发的是别人不承认的）

创建自己的 CA 证书

生成 CA 证书

```
[root@ma harbor]# yum -y install openssl
[root@ma harbor]# mkdir  -p /data/ssl
[root@ma harbor]# cd  /data/ssl
[root@ma  harbor]#openssl  req  -newkey  rsa:4096  -nodes  -sha256  -keyout  ca.key  -x509 -days 365 -out ca.crt
Country Name (2 letter code) [XX]:CN        #国家
State or Province Name (full name) []:biejing   #时间
```

```
Locality Name (eg, city) [Default City]:beijing   #时间
Organization Name (eg, company) [Default Company Ltd]:crushlinux  #名字
Organizational Unit Name (eg, section) []:crushulinux
Common Name (eg, your name or your server's hostname) []:www.crushlinux.com #
域名必须和你写的一样
```

生成证书签名请求

```
[root@master ssl]#
openssl req -newkey rsa:4096 -nodes -sha256 -keyout www.crushlinux.com.key -out
www.crushlinux.com.csr
信息与上相同
```

生成注册主机的证书

```
[root@ma ssl]# openssl x509 -req -days 365 -in www.crushlinux.com.csr -CA ca.crt -CAkey ca.key -
CAcreateserial -out www.crushlinux.com.crt
Signature ok
subject=/C=CN/ST=beijing/L=beijing/O=crushlinux/OU=crushlinux/CN=www.crushlinux.com
Getting CA Private Key
信息和上例相同
```

查看证书情况

```
[root@ma ssl]# ll
总用量 24
-rw-r--r-- 1 root root 2061 3 月  15 10:47 ca.crt
-rw-r--r-- 1 root root 3272 3 月  15 10:47 ca.key
-rw-r--r-- 1 root root   17 3 月  15 10:58 ca.srl
-rw-r--r-- 1 root root 1720 3 月  15 10:54 www.crushlinux.com.csr
-rw-r--r-- 1 root root 3272 3 月  15 10:54 www.crushlinux.com.key
-rw-r--r-- 1 root root 1944 3 月  15 10:58 www.crushlinux.com.crt
```

4）信任自签发的域名证书

由于 CA 证书是我们自己签发的，linux 操作系统是不信任的，因此我们需要把证书加入到系统的信任证书中

```
[root@ma ssl]# cp www.crushlinux.com.crt /etc/pki/ca-trust/source/anchors/
```

让系统即刻生效（也就是更新）

```
[root@ma ssl]# update-ca-trust extract
[root@ma ssl]# update-ca-trust enable
```

创建 harbor 的证书目录，并复制

```
[root@ma ssl]# mkdir -p /etc/ssl/harbor
[root@ma ssl]# cp www.crushlinux.com.key /etc/ssl/harbor/
[root@ma ssl]# cp www.crushlinux.com.crt /etc/ssl/harbor/
[root@ma ssl]# ls /etc/ssl/harbor/
www.crushlinux.com.key  www.crushlinx.com.crt
```

再这一步 harbor 的准备环境就结束了

5）部署并启动 harbor

```
[root@ma ssl]# cd /root/harbor/
[root@ma harbor]# ./install.sh --with-clair --ha
```

因为使用自定义的存储路径，安装过程中需要输入两次 yes 确认

```
Creating network "harbor_harbor-clair"
Creating harbor-log ... done
Creating registry           ... done
Creating clair              ... done
Creating harbor-adminserver ... done
Creating harbor-ui          ... done
Creating nginx              ... done
Creating harbor-jobservice  ... done
```

浏览器进行访问测试：192.168.200.10

用户：admin

密码：Harbor12345

本地测试

5. 客户端连接测试（nfs 主机）

```
[root@ma harbor]# vim /etc/hosts
192.168.200.10 www.crushlinux.com
```

如果使用 docker 客户端测试，那么就要为 docker 客户端认证

Master 客户端操作

```
[root@master harbor]# cd /data/ssl
[root@master ssl]#
scp www.crushlinux.com.crt 192.168.200.12:/etc/pki/ca-trust/source/anchors/ #被认证的主机
```

nfs 主机

```
[root@nfs nfs]# update-ca-trust extract
[root@nfs nfs]# update-ca-trust enable
[root@nfs nfs]# systemctl  restart docker
[root@nfs nfs]# docker  start  容器 ID
```

```
[root@nfs nfs]# docker   ps  -a

测试连接：
[root@nfs anchors]# docker login www.crushlinux.com
输入用户：
输入密码：
Login Succeeded     //登录成功
[root@nfs ~]# docker images    #先查看容器
[root@nfs ~]# docker tag redis:alpine www.crushlinux.com/library/redis  #给 redis 改个名
[root@nfs ~]# docker push www.crushlinux.com/library/redis

刷新浏览器 harbor 仓库可以看到 library/redis
```

注意:如果重启 docker 之后，容器就会随之关闭，这里需要启动所有关闭的容器

6. 部署 harbor slave 备库

上传 harbor-offline-installer-v1.5.0.tgz

```
[root@ma ~]# rz    /harbor-offline-installer-v1.5.0.tgz
[root@slave ~]# tar xf harbor-offline-installer-v1.5.0.tgz
[root@slave harbor]# cd  harbor/
[root@slave harbor]# vim ha/docker-compose.yml
19      - /data/storage:/storage:z
```

修改 vim harbor.cfg 文件

```
[root@slave harbor]# vim harbor.cfg
7 hostname = www.crushlinux.com
11 ui_url_protocol = https
23 ssl_cert = /etc/ssl/harbor/www.crushlinux.com.crt
24 ssl_cert_key = /etc/ssl/harbor/www.crushlinux.com.key
68 harbor_admin_password = 12345
130 db_host = 192.168.200.12
133 db_password = 123123
136 db_port = 3306
139 db_user = root
145 redis_url = 192.168.200.12:6379
150 clair_db_host = 192.168.200.12
154 clair_db_password = 123123
157 clair_db_port = 5432
163 clair_db = postgres
```

共用一个证书

```
[root@slave ~]# mkdir -p /etc/ssl/harbor
```

```
[root@slave ~]# cd /etc/ssl/harbor/
[root@slave harbor]#
scp 192.168.200.10:/etc/ssl/harbor/www.crushlinux.com.* ./
[root@slave harbor]# ll
总用量 8
-rw-r--r-- 1 root root 1907 3 月  15 20:39 www.crushlinux.com.crt
-rw-r--r-- 1 root root 3272 3 月  15 20:39 www.crushlinux.com.key
```

确保自己受系统信任

```
[root@slave harbor]#
 cp /etc/ssl/harbor/www.crushlinux.com.crt /etc/pki/ca-trust/source/anchors/
[root@slave harbor]# update-ca-trust extract
[root@slave harbor]# update-ca-trust enable
```

```
[root@slave harbor]# ./install.sh  --with-clair --ha
```

浏览器访问：192.168.200.11

客户端提交新的测试镜像

```
[root@slave harbor]# docker pull busybox
[root@slave harbor]# docker login www.crushlinux.com
[root@slave harbor]# docker tag busybox www.crushlinux.com/library/busybox
```

刷新 web 页面

客户端更换仓库

```
vim /etc/hosts
192.168.200.12 www.crushlinux.com
[root@slave harbor]# docker login www.crushlinux.com
[root@slave harbor]#docker  tag postgres  www.crushlinux.com/library/postgres
[root@slave harbor]# docker push www.crushlinux.com/library/postgres
d300f37cc3a6: Pushed
161fd0c67322: Pushed
aa8207e5bc6c: Pushed
5796a6662853: Pushed
725e23b6689a: Pushed
70e263450d12: Pushing  48.92MB/204.8MB
9cd7c4e12078: Pushed
73cf3adf6112: Pushed
065d45f80eac: Pushed
3aac10e9b066: Pushed
117725f5c702: Pushed
a01778662164: Pushed
883d24bc9ae1: Pushed
f5600c6330da: Pushing  38.48MB/69.24MB
```

到这一步高可用基本已经部署完毕。

7. 部署 DNS 服务（nfs 主机）高可用的配置

安装依赖包

```
[root@nfs anchors]# yum -y install  bind
[root@nfs ~]# vim /etc/named.conf
options {
        directory  "/var/named";
};

zone "crushlinux.com" IN {
        type master;
```

```
    file "crushlinux.zheng";
};
```

配置正文件

```
[root@nfs ~]# vim /var/named/crushlinux.zheng
$TTL    86400
@       SOA     crushlinux.com. admin.crushlinux.com (
            2020030501
            3H
            15M
            1W
            1D
)

        IN      NS      ns.crushlinux.com.
ns      IN      A       192.168.200.12
www     IN      A       192.168.100.10
www     IN      A       192.168.100.11
```

启动测试：

```
[root@nfs ~]# vim /etc/resolv.conf    #修改 DNS
nameserver 192.168.200.12
[root@nfs ~]# systemctl   start  named
[root@nfs ~]# nslookup www.crushlinux.com
Server:         192.168.200.12
Address:  192.168.200.12#53

Name:     www.crushlinux.com
Address: 192.168.100.11
Name:     www.crushlinux.com
```