**创建两台云主机master和slave，在这2台云主机上安装mysql据库系统并配置为主从数据库（master为主节点、slave为从节点）；**

**参考步骤：**

1. **安装对应服务**

**yum install mariadb mariadb-server -y**

1. **修改配置文件**

**主节点 /etc/my.cnf.d/server.cnf**

**[mysqld]字段下添加**

**server-id=1**

**log\_bin = mysql-bin**

**从节点 /etc/my.cnf**

**[mysqld]字段下添加**

**server-id=2**

**log\_bin = mysql-bin**

1. **启动MySQL服务**

**systemctl start mariadb**

1. **进入数据库，创建用于主从数据库同步的用户**

**主节点：**

**grant all privileges on \*.\* to mysql1@'%' identified by '000000';**

**grant replication slave on \*.\* to mysql1@'%' identified by '000000';**

**从节点：**

**change master to master\_host='192.168.200.50',master\_user='mysql1',master\_password='000000';**

**start slave;**

**show slave status \G;**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Slave\_IO\_State: Waiting for master to send event**

**Master\_Host: 192.168.200.50**

**Master\_User: mysql1**

**Master\_Port: 3306**

**Connect\_Retry: 60**

**Master\_Log\_File: mysql-bin.000001**

**Read\_Master\_Log\_Pos: 387**

**Relay\_Log\_File: mariadb-relay-bin.000002**

**Relay\_Log\_Pos: 671**

**Relay\_Master\_Log\_File: mysql-bin.000001**

**Slave\_IO\_Running: Yes**

**Slave\_SQL\_Running: Yes**

**Replicate\_Do\_DB:**

**Replicate\_Ignore\_DB:**

**Replicate\_Do\_Table:**

**Replicate\_Ignore\_Table:**

**Replicate\_Wild\_Do\_Table:**

**Replicate\_Wild\_Ignore\_Table:**

**Last\_Errno: 0**

**Last\_Error:**

**Skip\_Counter: 0**

**Exec\_Master\_Log\_Pos: 387**

**Relay\_Log\_Space: 967**

**Until\_Condition: None**

**Until\_Log\_File:**

**Until\_Log\_Pos: 0**

**Master\_SSL\_Allowed: No**

**Master\_SSL\_CA\_File:**

**Master\_SSL\_CA\_Path:**

**Master\_SSL\_Cert:**

**Master\_SSL\_Cipher:**

**Master\_SSL\_Key:**

**Seconds\_Behind\_Master: 0**

**Master\_SSL\_Verify\_Server\_Cert: No**

**Last\_IO\_Errno: 0**

**Last\_IO\_Error:**

**Last\_SQL\_Errno: 0**

**Last\_SQL\_Error:**

**Replicate\_Ignore\_Server\_Ids:**

**Master\_Server\_Id: 1**

**1 row in set (0.00 sec)**

1. **使用提供的云安全框架组件，将 http://controller/dashboard 中的 keystone安全策略从http优化至 https**

**安装服务**

yum install -y *httpd* mod\_ssl

**修改ssl配置文件**

修改/etc/httpd/conf.d/ssl.conf配置文件，具体修改内容如下：

#SSLProtocol all -SSLv2 -SSLv3 //找到该行，并注释

SSLProtocol all -SSLv2 //添加该行

**配置/etc/openstack-dashboard/local\_settings 配置文件，修改参数如下所示：**

CSRF\_COOKIE\_SECURE = True //将该行的注释取消

SESSION\_COOKIE\_SECURE = True //将该行的注释取消

USE\_SSL = True //添加该行

SESSION\_COOKIE\_HTTPONLY = True //添加该行

**重启服务**

在修改完上述的配置文件后，重启httpd服务和缓存服务，命令如下：

[root@controller ~]# service httpd restart

Redirecting to /bin/systemctl restart httpd.service

[root@controller ~]# service memcached restart

Redirecting to /bin/systemctl restart memcached.service

1. **在openstack私有云平台上，将云主机VM1使用手动迁移的方式，迁移至另一个计算节点。**

**三步走(迁移之前先关机)：**

1. **迁移虚拟机目录**

**将对应节点下**/var/lib/nova/instances/中的虚拟机目录移动到另一节点

1. **修改数据库文件**

将nova库中instances表中对应虚拟机的host和node字段都改成另一节点主机名字

1. **重启nova-compute服务**

systemctl restart openstack-nova-compute

**2）创建三台云主机来搭建rabbitmq集群。使用普通集群模式，其中一台做磁盘节点，另外两台做内存节点，配置完毕后启动rabbitmq服务**

1. **安装rabbitmq服务**

**yum install rabbitmq-server -y**

**2.配置域名解析文件**

192.168.200.50 rabbitmq1

192.168.200.51 rabbitmq2

192.168.200.52 rabbitmq3

**3.对应节点更改为对应主机名**

**hostnamectl set-hostname rabbitmq1**

**hostnamectl set-hostname rabbitmq2**

**hostnamectl set-hostname rabbitmq3**

1. **三节点全部启动rabbitmq服务**

**[root@rabbitmq1 ~]# systemctl start rabbitmq-server**

**[root@rabbitmq2 ~]# systemctl start rabbitmq-server**

**[root@rabbitmq3 ~]# systemctl start rabbitmq-server**

1. **复制rabbitmq1节点的/var/lib/rabbitmq/.erlang.cookie 文件中的值到另外两个节点**

**[root@rabbitmq1 rabbitmq]# scp -rp /var/lib/rabbitmq/.erlang.cookie root@rabbitmq2:/var/lib/rabbitmq/**

**The authenticity of host 'rabbitmq2 (192.168.200.51)' can't be established.**

**ECDSA key fingerprint is SHA256:/tj64sznjoZt6JHvFe822K2yl+5Rw5ClPxwt8iau/6U.**

**ECDSA key fingerprint is MD5:03:12:f3:0f:80:c0:3c:7a:5f:bb:4a:9d:2f:bd:77:20.**

**Are you sure you want to continue connecting (yes/no)? yes**

**Warning: Permanently added 'rabbitmq2,192.168.200.51' (ECDSA) to the list of known hosts.**

**root@rabbitmq2's password:**

**.erlang.cookie 100% 20 14.3KB/s 00:00**

**[root@rabbitmq1 rabbitmq]# scp -rp /var/lib/rabbitmq/.erlang.cookie root@rabbitmq3:/var/lib/rabbitmq/**

**The authenticity of host 'rabbitmq3 (192.168.200.52)' can't be established.**

**ECDSA key fingerprint is SHA256:jNA1yKe9YjwjDWE0RARtpAEKGrQuv/y/xyHuEWCoLbo.**

**ECDSA key fingerprint is MD5:29:d1:74:5f:89:a7:5a:96:5b:bc:5b:8b:c0:a1:69:f6.**

**Are you sure you want to continue connecting (yes/no)? yes**

**Warning: Permanently added 'rabbitmq3,192.168.200.52' (ECDSA) to the list of known hosts.**

**root@rabbitmq3's password:**

**.erlang.cookie**

1. **各个节点重启rabbitmq服务**

**更改用户和用户组**

**chown rabbitmq:rabbitmq /var/lib/rabbitmq/.erlang.cookie**

**systemctl restart rabbitmq-server**

1. **加入集群**

**[root@rabbitmq2 rabbitmq]# rabbitmqctl stop\_app**

**[root@rabbitmq2 rabbitmq]# rabbitmqctl join\_cluster --ram rabbit@rabbitmq1**

**Clustering node rabbit@rabbitmq2 with rabbit@rabbitmq1 ...**

**...done.**

**[root@rabbitmq3 rabbitmq]# rabbitmqctl stop\_app**

**Stopping node rabbit@rabbitmq3 ...**

**...done.**

**[root@rabbitmq3 rabbitmq]# rabbitmqctl join\_cluster --ram rabbit@rabbitmq1**

**Clustering node rabbit@rabbitmq3 with rabbit@rabbitmq1 ...**

**...done.**

1. **查看状态进行验证**

**[root@rabbitmq1 rabbitmq]# rabbitmqctl cluster\_status**

**Cluster status of node rabbit@rabbitmq1 ...**

**[{nodes,[{disc,[rabbit@rabbitmq1]},{ram,[rabbit@rabbitmq3,rabbit@rabbitmq2]}]},**

**{running\_nodes,[rabbit@rabbitmq1]},**

**{cluster\_name,<<"rabbit@rabbitmq1">>},**

**{partitions,[]}]**

**...done.**

1. **创建两台云主机redis1和redis2,在这两台云主机上安装Redis服务 并配置redis主从服务。**

**参考步骤：**

**requirepass和masterauth密码保持一致**

**修改redis1节点的配置文件/etc/redis.conf如下：**

**#第一处修改**

**# bind 127.0.0.1 //找到bind 127.0.0.1这行并注释掉**

**#第二处修改**

**protected-mode yes //修改前**

**protected-mode no //修改后，外部网络可以访问**

**#第三处修改**

**daemonize no //修改前**

**daemonize yes //修改后，开启守护进程**

**#第四处修改**

**# requirepass foobared //找到该行**

**requirepass "123456" //在下方添加设置访问密码**

**#第五处修改，设定主库密码与当前库密码同步，保证从库能够提升为主库**

**masterauth "123456"**

**#第六处修改，打开AOF持久化支持**

**appendonly yes**

**重启服务**

**redis2从节点：**

**修改redis2节点的配置文件/etc/redis.conf如下：**

**#第一处修改**

**# bind 127.0.0.1 //找到bind 127.0.0.1这行并注释掉**

**#第二处修改**

**protected-mode yes //修改前**

**protected-mode no //修改后，外部网络可以访问**

**#第三处修改**

**daemonize no //修改前**

**daemonize yes //修改后，开启守护进程**

**#第四处修改**

**# requirepass foobared //找到该行**

**requirepass "123456" //在下方添加设置访问密码**

**#第五处修改**

**# slaveof <masterip> <masterport> //找到该行**

**slaveof 192.168.200.21 6379 //在下方添加访问的主节点IP与端口**

**#第六处修改**

**# masterauth <master-password> //找到该行**

**masterauth "123456" //在下方添加访问主节点密码**

**#第七出修改，打开AOF持久化支持**

**appendonly yes**

**重启服务**

**先在主节点redis1中登录redis-cli并输入密码，查看主从复制的信息，命令如下：**

**[root@redis1 ~]# redis-cli**

**127.0.0.1:6379> auth 123456**

**OK**

**127.0.0.1:6379> info replication**

**# Replication**

**role:master**

**connected\_slaves:1**

**slave0:ip=192.168.200.22,port=6379,state=online,offset=1,lag=1**

**master\_repl\_offset:1**

**repl\_backlog\_active:1**

**repl\_backlog\_size:1048576**

**repl\_backlog\_first\_byte\_offset:2**

**repl\_backlog\_histlen:0**

**127.0.0.1:6379>**

**可以看到该节点为master节点，并有一个从节点已连接。**

1. **新建一台虚拟机，添加一块50G硬盘，将该硬盘划分出4个10G的分区，创建一个raid 5，其中1个分区作为热备**

**参考步骤：**

1. **安装相应工具**

**yum install mdadm -y**

1. **进行磁盘分区**

**sdb 8:16 0 40G 0 disk**

**├─sdb1 8:17 0 10G 0 part**

**├─sdb2 8:18 0 10G 0 part**

**├─sdb3 8:19 0 10G 0 part**

**├─sdb4 8:20 0 1K 0 part**

**└─sdb5 8:21 0 10G 0 part**

1. **进行磁盘阵列配置**

**mdadm -C /dev/md1 -v -l 5 -n 3 -x 1 /dev/sdb{1,2,3,5}**

1. **验证配置**

**[root@ansible ~]# mdadm -D /dev/md1**

**/dev/md1:**

**Version : 1.2**

**Creation Time : Sun Sep 12 19:23:24 2021**

**Raid Level : raid5**

**Array Size : 20948992 (19.98 GiB 21.45 GB)**

**Used Dev Size : 10474496 (9.99 GiB 10.73 GB)**

**Raid Devices : 3**

**Total Devices : 4**

**Persistence : Superblock is persistent**

**Update Time : Sun Sep 12 19:24:17 2021**

**State : clean**

**Active Devices : 3**

**Working Devices : 4**

**Failed Devices : 0**

**Spare Devices : 1**

**Layout : left-symmetric**

**Chunk Size : 512K**

**Consistency Policy : resync**

**Name : ansible:1 (local to host ansible)**

**UUID : 5d29a37d:aad3485a:8331f5be:7fe11863**

**Events : 18**

**Number Major Minor RaidDevice State**

**0 8 17 0 active sync /dev/sdb1**

**1 8 18 1 active sync /dev/sdb2**

**4 8 19 2 active sync /dev/sdb3**

**3 8 21 - spare /dev/sdb5**

**6）zookeeper集群的部署**

1. **解压tar包，安装jdk环境**

**tar -xf zookeeper-3.4.14.tar.gz**

**yum install java-openjdk\* -y**

1. **进入对应目录**

**cd zookeeper-3.4.14/conf/**

1. **生成对应配置文件**

**mv zoo\_sample.cfg zoo.cfg**

1. **修改配置文件**

**在zoo.cfg文件添加一下内容**

**server.1=192.168.200.50:2888:3888**

**server.2=192.168.200.51:2888:3888**

**server.3=192.168.200.52:2888:3888**

1. **创建对应ID文件**

**Zookeeper1节点**

**echo "1" >> /tmp/zookeeper/myid**

**Zookeeper2节点**

**echo "2" >> /tmp/zookeeper/myid**

**Zookeeper3节点**

**echo "3" >> /tmp/zookeeper/myid**

1. **运行启动脚本**

**sh zookeeper-3.4.14/bin/zkServer.sh start**

**7.验证服务**

**查看端口号 yum -y install net-tools netstat -tunlp**

**领导节点有2888端口号及2181,3888**

**其余两个节点有2181,3888端口号**

1. **kafka集群的部署**

**注： kafka集群的前提是搭建并启动了zookeeper集群**

1. **解压tar包**

**tar -xf kafka\_2.11-1.1.1.tgz**

1. **修改配置文件**

**Zookeeper1节点：**

**在kafka\_2.11-1.1.1/config/server.properties文件中添加配置**

**broker.id=1**

**listeners=PLAINTEXT://192.168.200.50:9092**

**Zookeeper2节点：**

**在kafka\_2.11-1.1.1/config/server.properties文件中添加配置**

**broker.id=2**

**listeners=PLAINTEXT://192.168.200.51:9092**

**Zookeeper3节点：**

**在kafka\_2.11-1.1.1/config/server.properties文件中添加配置**

**broker.id=3**

**listeners=PLAINTEXT://192.168.200.52:9092**

1. **启动服务**

**sh kafka\_2.11-1.1.1/bin/kafka-server-start.sh -daemon kafka\_2.11-1.1.1/config/server.properties**

1. **验证服务**

**查看端口是否有9092端口**

**5）HAProxy+MariaDB Galera Cluster集群的部署**

节点规划：

192.168.200.50 HAProxy

192.168.200.51 master

192.168.200.52 slave

**操作步骤：**

1. **配置YUM源**
2. **安装对应软件**

**HAProxy节点**

**yum install haproxy -y**

**Master及slave节点**

**yum install mariadb-server mariadb -y**

1. **修改配置文件**

**Haproxy节点**

**编辑/etc/haproxy/haproxy.cfg 文件**

**global**

**log 127.0.0.1 local2**

**chroot /var/lib/haproxy**

**pidfile /var/run/haproxy.pid**

**maxconn 4000**

**user haproxy**

**group haproxy**

**daemon**

**stats socket /var/lib/haproxy/stats**

**defaults**

**log global**

**option httplog**

**option dontlognull**

**option http-server-close**

**option forwardfor except 127.0.0.0/8**

**option redispatch**

**retries 3**

**timeout http-request 10s**

**timeout queue 1m**

**timeout connect 10s**

**timeout client 1m**

**timeout server 1m**

**timeout http-keep-alive 10s**

**timeout check 10s**

**maxconn 3000**

**listen stats**

**bind 192.168.200.50:9000**

**mode http**

**stats enable**

**stats uri /stats**

**listen mariadb**

**balance roundrobin**

**mode tcp**

**bind 192.168.200.50:3307**

**server node1 192.168.200.51:3306 check weight 1**

**server node2 192.168.200.52:3306 check weight 2**

**编辑master及slave节点/etc/my.cnf.d/server.cnf**

**[galera]**

**wsrep\_on=ON**

**wsrep\_provider=/usr/lib64/galera/libgalera\_smm.so**

**wsrep\_cluster\_address="gcomm://192.168.200.51,192.168.200.52"**

**binlog\_format=row**

**default\_storage\_engine=InnoDB**

**innodb\_autoinc\_lock\_mode=2**

**bind-address=0.0.0.0**

1. **启动服务**

**Master：**

**galera\_new\_cluster**

**Slave：**

**systemctl start mariadb**

**Haproxy：**

**systemctl start haproxy**

1. **验证**

**在node1节点远程登录查看**

**Master节点登录数据库授予root远程登录权限**

**MariaDB [(none)]> grant all privileges on \*.\* to root@'%' identified by '000000';**

**Query OK, 0 rows affected (0.004 sec)**

**MariaDB [(none)]> flush privileges;**

**Query OK, 0 rows affected (0.005 sec)**

**登录haproxy负载均衡节点查看效果**

**mysql -uroot -p000000 -h 192.168.200.50 -P 3307**

**网页访问查看连接状态：**

**http://192.168.200.50:9000/stats**

1. **分别对文件进行解压**

**tar -zxvf /usr/local/apache-tomcat-8.5.69.tar.gz -C /usr/local**

**tar -zxvf /usr/local/jdk-8u301-linux-x64.tar.gz -C /usr/local**

1. **修改目录名**

**mv /usr/local/apache-tomcat-8.5.69 /usr/local/tomcat8**

**mv /usr/local/jdk1.8.0\_301 /usr/local/jdk8**

1. **赋予权限**

**chmod +x /usr/local/tomcat8/bin/\*.sh**

1. **编写脚本/etc/profile，（添加至文件末尾）**

**export JAVA\_HOME=/usr/local/jdk8**

**export CLASSPATH=.:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar**

**export PATH=$JAVA\_HOME/bin:$PATH**

**sh /usr/local/tomcat8/bin/startup.sh**

**tail -f /usr/local/tomcat8/logs/catalina.out**

1. **导入环境变量并启动服务**

**source /etc/profile**

1. **可以通过 curl http://127.0.0.1:8080 访问默认网页**
2. **在openstack私有云平台上，创建一个名为“lvm”的卷类型，创建1块卷类型为lvm的40G云盘，并附加到虚拟机VM1上。**

**Openstack volume type create lvm**

**Openstack volume create --type lvm --size 40 test**

**Openstack server add volume VM1 test**

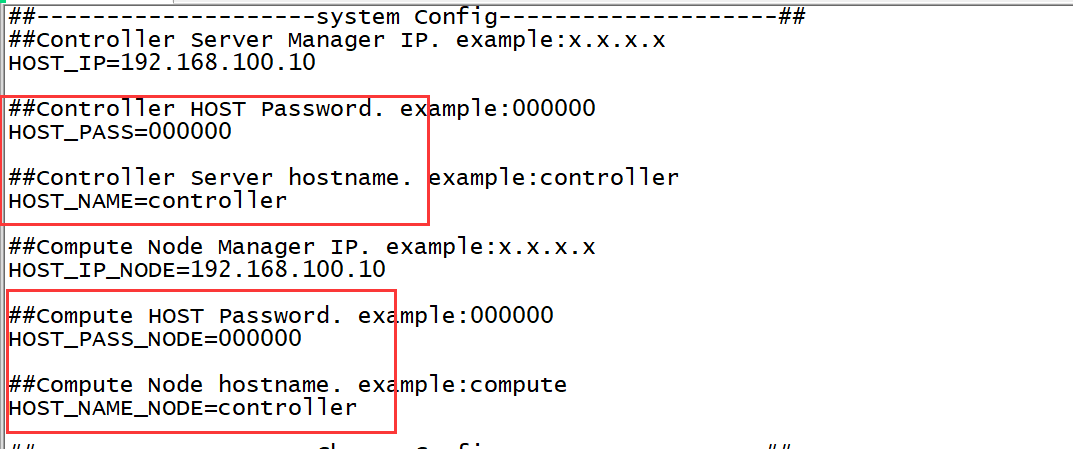
1. **在openstack私有云平台上，将云主机VM1保存为qcow2格式的快照并保存到controller节点/root/imagesave目录下**

openstack image save --file /root/imagesave/VM1.qcow2 xxx

1. **在controller节点上新建名为laiwu的容器，并获取该容器的存放路径；将 centos7\_5.qcow2 镜像上传到laiwu容器中，并设置分段存放， 每一段大小为 10M**

swift upload laiwu -S 10M centos7\_5.qcow2

1. **在openstack私有云平台上，基于提供的脚本添加controller节点资源到云平台（即controller节点也作为compute节点使用）。**



1. **在openstack私有云平台上，编写模板flavor.yml，创建名为“m1.flavor”、 ID 为 1234、内存为 1024MB、硬盘为 20GB、 vcpu数量为 2的云主机类型，堆栈命名为test。**

**相关参考文档：**

[**https://docs.openstack.org/heat/latest/**](https://docs.openstack.org/heat/latest/)

heat\_template\_version: 2015-04-30  
resources:  
  server:  
    type: OS::Nova::Flavor  
    properties:  
      name: m1.flavor  
      flavorid: 1234  
      ram: 1024  
      disk: 20   
      vcpus: 2

编写完之后，执行 openstack stack create -t flavor.yml test 进行创建

openstack stack list 和 openstack flavor list 查看是否创建成功