LVS Linux Virtual Server

默认写入内核程序默认开启

LVS 原理与 Nginx 完全不同

LVS 为调度功能 及 不修改数据包的IP层 [ SIP(源ip) DIP(目标ip) ] 而是修改数据包包头的MAC地址来达到转发功能 (与交换机原理相同)

Nginx 为代理功能 及 解封数据包后 重新打包封装 发给真实服务器

集群分类

HPC 高性能计算集群

LB 负载均衡

HA 高可用

LVS工作模式

NAT模式 小规模 调度服务器作为一个软路由 负载存在上线

DR模式 大规模 服务器反馈给客户端不通过调度器 通过其他手段实现

TUN模式 不常用

\*\* DR与NAT不同点在于 数据包返回的方式不同 不通过调度服务器

负载均衡常用调度算法

目前有10种

常用

--轮询(Round Robin)

--加权轮询(Weighted Round Robin)

--最少连接(Least Connections)

--加权最少连接(Weighted Least Connections)

最少连接 优于 轮询

\*\* 轮询必定为 1人 1次 若存在客户端持续连接 则会导致 服务器负载不均衡情况

###################################################################################

LVS 也是通过ipvsadm软件实现设置

虚拟集群服务器:

ipvsadm -[A|E|D|C] -[t|u] 192.168.4.5:80(调度器ip及端口) -[s] [rr|wrr|lc|wlc|sh|dh]

-A 添加虚拟集群服务器

-E 编辑虚拟集群服务器

-D 删除虚拟集群服务器(单一)

-C 清空所有虚拟集群服务器

-t TCP协议

-u UDP协议

-s 算法

-rr 轮询 wrr 加权轮询 lc 最少连接次数 wlc 加权最少连接数 sh dh 与ip\_hash作用相同

虚拟集群服务器中真实服务器:

ipvsadm -[a|e|d|] -[t|u] 192.168.4.5:80(虚拟集群服务器ip及端口) -r 192.168.4.10 (真实服务器ip) -[g|m|i] (默认为g) -w (权重)

-a 添加真实服务器

-e 编辑真实服务器

-d 删除真实服务器

-g DR模式

-m NAT模式

-i 隧道模式

ipvsadm -Ln 查看规则列表

####################################################################################

部署LVS-NAT集群

\*\* 由于NAT模式需要将真实服务器地址返回的数据包再次通过调度服务器发送 并且 内部通讯与 客户端与调度通讯 跨网段

\*\* 所以优先要给真是服务器配置网关 及调度ip作为网关 (ip route 查看)

1)部署LVS-NAT模式调度器

确认调度器的路由转发功能

# echo 1 > /proc/sys/net/ipv4/ip\_forward //开启路由转发功能

2)创建集群服务器

# yum -y install ipvsadm

# ipvsadm -A -t 192.168.4.5:80 -s wrr

3)添加真实服务器

# ipvsadm -a -t 192.168.4.5:80 -r 192.168.2.100 -w 1 -m

# ipvsadm -a -t 192.168.4.5:80 -r 192.168.2.200 -w 1 -m

4）查看规则列表，并保存规则

# ipvsadm -Ln

# ipvsadm-save -n > /etc/sysconfig/ipvsadm //存入配置文档 开启服务自动加载. 若不存入 重启及消失

#########################################################################################

部署LVS-DR集群

配置实验网络环境

1）设置Proxy代理服务器的VIP和DIP

注意：为了防止冲突，VIP必须要配置在网卡的虚拟接口！！！

# cp /etc/sysconfig/network-scripts/ifcfg-eth0 /etc/sysconfig/network-scripts/ifcfg-eth0:0 //虚拟网卡文档

TYPE=Ethernet

DEFROUTE=yes

NAME=eth0:0

DEVICE=eth0:0

ONBOOT=yes

IPADDR=192.168.4.15

PREFIX=24

# systemctl restart network //若重启后未生效 则因为网卡是有2个服务共同管理 可关闭 NetworkManager 后再次重启

2）设置Web1 web2服务器网络参数 (假装自己为调度通过vip发送)

cp /etc/sysconfig/network-scripts/ifcfg-lo /etc/sysconfig/network-scripts/ifcfg-lo:0 //虚拟网卡文档

DEVICE=lo:0

IPADDR=192.168.4.15

NETMASK=255.255.255.255 //必须为4个255

NETWORK=192.168.4.15

BROADCAST=192.168.4.15

ONBOOT=yes

NAME=lo:0

3)防止地址冲突的问题

这里因为web1 web2也配置与代理一样的VIP地址，默认肯定会出现地址冲突；

sysctl.conf文件写入这下面四行的主要目的就是访问192.168.4.15的数据包，只有调度器会响应，其他主机都不做任何响应，这样防止地址冲突的问题。

# vim /etc/sysctl.conf

#手动写入如下4行内容

net.ipv4.conf.all.arp\_ignore = 1 //设置所有网卡(包括lo)该回答的回答 不该回答不回答

net.ipv4.conf.lo.arp\_ignore = 1 //单独设置lo 该回答回答 不该回答不回答

net.ipv4.conf.lo.arp\_announce = 2 //设置所有网卡(包括lo)什么都不申明

net.ipv4.conf.all.arp\_announce = 2 //单独设置lo 什么都不申明

#当有arp广播问谁是192.168.4.15时，本机忽略该ARP广播，不做任何回应

#本机不要向外宣告自己的lo回环地址是192.168.4.15

# sysctl -p //刷新配置

4)重启网络服务，关闭防火墙与SELinux

# systemctl restart network

proxy调度器安装软件并部署LVS-DR模式调度器

创建新的集群服务器规则

# ipvsadm -A -t 192.168.4.15:80 -s wrr

添加真实服务器(-g参数设置LVS工作模式为DR模式，-w设置权重)

# ipvsadm -a -t 192.168.4.15:80 -r 192.168.4.100 -g -w 1

# ipvsadm -a -t 192.168.4.15:80 -r 192.168.4.200 -g -w 1

查看规则列表，并保存规则

# ipvsadm -Ln

# ipvsadm-save -n > /etc/sysconfig/ipvsadm //存入配置文档 开启服务自动加载. 若不存入 重启及消失

##########################################################################################################################

Keepalived

Keepalived运行原理

\* 检测每个服务器节点状态

\* 具有安全检测 服务器节点出现异常或故障 keepalived将故障节点从集训中剔除

\* 故障节点恢复后 再将其加入到集群中

\* 所有工作自动完成 无需人工干预

##########################################################################

Keepalived高可用服务器

\*原理与VRRP一致 一台作为主服务器 一台作为备用服务器

\*主 备关系由优先级大小决定(priority) 若一致则通过ip大小比较 ip大优先

安装Keepalived软件

# yum install -y keepalived

修改web1服务器Keepalived配置文件

# vim /etc/keepalived/keepalived.conf

global\_defs {

notification\_email {

admin@tarena.com.cn //设置报警收件人邮箱

}

notification\_email\_from ka@localhost //设置发件人

smtp\_server 127.0.0.1 //定义邮件服务器

smtp\_connect\_timeout 30

router\_id web1 \*\*\* //设置路由ID号（实验需要修改）

}

vrrp\_instance VI\_1 {

state MASTER \*\*\* //主服务器为MASTER（备服务器需要修改为BACKUP）

interface eth0 //定义网络接口

virtual\_router\_id 51 \*\*\*与VRID效果一致 若存在同一id则会影响//主备服务器VRID号必须一致

priority 100 \*\*\*//服务器优先级,优先级高优先获取VIP

advert\_int 1

authentication {

auth\_type pass

auth\_pass 1111 //主备服务器密码必须一致

}

virtual\_ipaddress { \*\*\*\*//谁是主服务器谁获得该VIP（实验需要修改）

192.168.4.80

}

}

修改web2服务器Keepalived配置文件

# vim /etc/keepalived/keepalived.conf

global\_defs {

notification\_email {

admin@tarena.com.cn //设置报警收件人邮箱

}

notification\_email\_from ka@localhost //设置发件人

smtp\_server 127.0.0.1 //定义邮件服务器

smtp\_connect\_timeout 30

router\_id web2 \*\*\* //设置路由ID号（实验需要修改）

}

vrrp\_instance VI\_1 {

state BACKUP //备服务器为BACKUP（实验需要修改）

interface eth0 //定义网络接口

virtual\_router\_id 51 \*\*\* //主辅VRID号必须一致

priority 50 \*\*\* //服务器优先级（实验需要修改）

advert\_int 1

authentication {

auth\_type pass

auth\_pass 1111 \*\*\* //主辅服务器密码必须一致

}

virtual\_ipaddress { \*\*\*//谁是主服务器谁配置VIP（实验需要修改）

192.168.4.80

}

}

启动服务

# systemctl start keepalived

配置防火墙和SELinux

启动keepalived会自动添加一个drop的防火墙规则，需要清空！

# iptables -F //必须要 并且不是马上生效可能存在1s 2s 延迟

# setenforce 0

登录两台Web服务器查看VIP信息

# ip addr show eth0 与 ip a s eth0d

#########################################################################################

Keepalived+LVS服务器

\*\*本次实验目的为设置一台主调度服务器 一台备用调度 达到高可用效果

\*\*本次实验lvs模式为DR模式 需要在真实服务器web1 web2 上设置回环地址与vip一致 及lo:0 (与vip一致)

1）LVS1调度器设置Keepalived，并启动服务

# yum install -y keepalived

# vim /etc/keepalived/keepalived.conf

global\_defs {

notification\_email {

admin@tarena.com.cn //设置报警收件人邮箱

}

notification\_email\_from ka@localhost //设置发件人

smtp\_server 127.0.0.1 //定义邮件服务器

smtp\_connect\_timeout 30

router\_id web1 \*\*\* //设置路由ID号（实验需要修改）

}

vrrp\_instance VI\_1 {

state MASTER \*\*\* //主服务器为MASTER（备服务器需要修改为BACKUP）

interface eth0 //定义网络接口

virtual\_router\_id 51 \*\*\*与VRID效果一致 若存在同一id则会影响//主备服务器VRID号必须一致

priority 100 \*\*\*//服务器优先级,优先级高优先获取VIP

advert\_int 1

authentication {

auth\_type pass

auth\_pass 1111 //主备服务器密码必须一致

}

virtual\_ipaddress { \*\*\*\*//谁是主服务器谁获得该VIP（实验需要修改）

192.168.4.15

}

}

virtual\_server 192.168.4.15 80 { //设置ipvsadm的VIP规则（实验需要修改）

delay\_loop 6

lb\_algo rr //设置LVS调度算法为RR

lb\_kind DR //设置LVS的模式为DR（实验需要修改）

#persistence\_timeout 50 //（实验需要注释）

#注意这样的作用是保持连接，开启后，客户端在一定时间内始终访问相同服务器

protocol TCP

real\_server 192.168.4.100 80 { //设置后端web服务器真实IP（实验需要修改）

weight 1 //设置权重为1

TCP\_CHECK { //对后台real\_server做健康检查（实验需要修改）

connect\_timeout 3

nb\_get\_retry 3

delay\_before\_retry 3

}

}

real\_server 192.168.4.200 80 { //设置后端web服务器真实IP（实验需要修改）

weight 2 //设置权重为1

TCP\_CHECK { //对后台real\_server做健康检查（实验需要修改）

connect\_timeout 3

nb\_get\_retry 3

delay\_before\_retry 3

}

}

}

# systemctl start keepalived

# ipvsadm -Ln #查看LVS规则

# ip a s #查看VIP配置

# iptables -F

##########################################################################################

配置HAProxy负载平衡集群

原理与nginx一致 用户访问haproxy-- haproxy访问真实服务器

功能 lvs<HAProxy<Nginx

性能 lvs>HAProxy>Nginx

\*\* 实用性最高

# yum -y install haproxy

# vim /etc/haproxy/haproxy.cfg

global

log 127.0.0.1 local2 ##[err warning info debug]

chroot /usr/local/haproxy

pidfile /var/run/haproxy.pid ##haproxy的pid存放路径

maxconn 4000 ##最大连接数，默认4000

user haproxy

group haproxy

daemon ##创建1个进程进入deamon模式运行

defaults

mode http ##默认的模式mode { tcp|http|health }

option dontlognull ##不记录健康检查的日志信息

option httpclose ##每次请求完毕后主动关闭http通道

option httplog ##日志类别http日志格式

option forwardfor ##后端服务器可以从Http Header中获得客户端ip

option redispatch ##serverid服务器挂掉后强制定向到其他健康服务器

timeout connect 10000 #如果backend没有指定，默认为10s

timeout client 300000 ##客户端连接超时

timeout server 300000 ##服务器连接超时

maxconn 3000 ##最大连接数

retries 3 ##3次连接失败就认为服务不可用，也可以通过后面设置

listen stats 0.0.0.0:1080 #监听端口

stats refresh 30s #统计页面自动刷新时间

stats uri /stats #统计页面url

stats realm Haproxy Manager #进入管理解面查看状态信息

stats auth admin:admin #统计页面用户名和密码设置

\*\* 以下需要手写

listen websrv-rewrite 0.0.0.0:80 // 与listen XX \*:80 效果相同

balance roundrobin // 计算方式必须全写 lestconn

server web1 192.168.2.100:80 check inter 2000 rise 2 fall 5

server web2 192.168.2.200:80 check inter 2000 rise 2 fall 5

\*\* check 为安全检查参数

\*\* inter 安全检查间隔 默认毫秒为单位

\*\* rise 开启测试2次成功 则加入集群

\*\* fall 失败测试5次 则从集群删除

######################################################################################################

存储的分类

DAS直连存储 ide sata usb 家用

NAS网络附加存储 nfs samba ftp 分享类型为[文件系统]

SAN存储区域网络 iscsi 分享类型为 [块设备]

SDS软件定义存储(分布式存储) ceph

ceph组件(默认3副本) 及一份文件默认存3个副本 若坏了其中一个会自动拷贝到另外一台 保证副本一直有3份

由于过半原则 如果是2台 坏一台也是整个集群坏了 所以mon至少三台

OSDs 存储设备 共享磁盘 ceph-osd

Monitors 集群监控组件 ceph-mon

RadosGateway 对象存储网关 ceph-radosgw

MDSs 存放文件系统的元数据(若不装则是块设备 不存在文件系统) ceph-mds

########################################################################################################

准备四台KVM虚拟机，其三台作为存储集群节点，一台安装为客户端，实现如下功能：

三台作即作为mon 也作为 osd

1)配置无密码连接(包括自己远程自己也不需要密码)，在node1操作 //为后期统一操作做准备

# ssh-keygen -f /root/.ssh/id\_rsa -N ''

# for i in 10 11 12 13

do

ssh-copy-id 192.168.4.$i

done

2）修改/etc/hosts并同步到所有主机。

警告：/etc/hosts解析的域名必须与本机主机名一致！！！！

# vim /etc/hosts

192.168.4.10 client

192.168.4.11 node1

192.168.4.12 node2

192.168.4.13 node3

3)修改所有节点都需要配置YUM源，并同步到所有主机

# cat /etc/yum.repos.d/ceph.repo

[mon]

name=mon

baseurl=ftp://192.168.4.254/ceph/MON

gpgcheck=0

[osd]

name=osd

baseurl=ftp://192.168.4.254/ceph/OSD

gpgcheck=0

[tools]

name=tools

baseurl=ftp://192.168.4.254/ceph/Tools

gpgcheck=0

# for i in client node1 node2 node3

do

scp /etc/yum.repos.d/ceph.repo $i:/etc/yum.repos.d/

done

4）所有节点主机与真实主机的NTP服务器同步时间 //Ceph要求所有主机时差不能超过0.05s，否则就会提示WARN

# vim /etc/chrony.conf

… …

server 192.168.4.254 iburst

# for i in client node1 node2 node3

do

scp /etc/chrony.conf $i:/etc/

ssh $i "systemctl restart chronyd"

done

5)准备存储磁盘

物理机上为每个虚拟机准备3块20G磁盘 使用图形直接添加

5-1）在node1安装部署工具，学习工具的语法格式

# yum -y install ceph-deploy

5-2）创建目录 //必须创建 ceph-XXX 目录 后续ceph-deploy 等命令只能在该目录内生效 与git原理一致

# mkdir ceph-cluster

# cd ceph-cluster/

6)部署Ceph集群

6-1)创建Ceph集群配置,在ceph-cluster目录下生成Ceph配置文件

# ceph-deploy new node1 node2 node3 // 此步创建为 new

6-2)给所有节点安装ceph相关软件包

# for i in node1 node2 node3

do

ssh $i "yum -y install ceph-mon ceph-osd ceph-mds ceph-radosgw"

done

6-3)初始化所有节点的mon服务，也就是启动mon服务（主机名解析必须对 /etc/hosts 目录内）。

# ceph-deploy mon create-initial //启动ceph-mon@XX.service 服务

7)创建OSD

7-1)vdb1和vdb2这两个分区用来做存储服务器的journal缓存盘 //创建2个固态盘 即先将需要存储的数据高效的缓存到缓存盘 之后慢慢同步到osd

# for i in node1 node2 node3

do

ssh $i "parted /dev/vdb mklabel gpt"

ssh $i "parted /dev/vdb mkpart primary 1 50%"

ssh $i "parted /dev/vdb mkpart primary 50% 100%"

done

7-2)磁盘分区后的默认权限无法让ceph软件对其进行读写操作，需要修改权限。

node1、node2、node3都需要操作，这里以node1为例。

# chown ceph.ceph /dev/vdb1

# chown ceph.ceph /dev/vdb2

#上面的权限修改为临时操作，重启计算机后，权限会再次被重置。

#我们还需要将规则写到配置文件实现永久有效。

# vim /etc/udev/rules.d/70-vdb.rules

ENV{DEVNAME}=="/dev/vdb1",OWNER="ceph",GROUP="ceph"

ENV{DEVNAME}=="/dev/vdb2",OWNER="ceph",GROUP="ceph"

7-3）初始化清空磁盘数据

# ceph-deploy disk zap node1:vdc node1:vdd

# ceph-deploy disk zap node2:vdc node2:vdd

# ceph-deploy disk zap node3:vdc node3:vdd

7-4）创建OSD存储空间（仅node1操作即可)

# ceph-deploy osd create \

node1:vdc:/dev/vdb1 node1:vdd:/dev/vdb2

//创建osd存储设备，vdc为集群提供存储空间，vdb1提供JOURNAL缓存，

//每个存储设备对应一个缓存设备，缓存需要SSD，不需要很大

# ceph-deploy osd create \

node2:vdc:/dev/vdb1 node2:vdd:/dev/vdb2

# ceph-deploy osd create \

node3:vdc:/dev/vdb1 node3:vdd:/dev/vdb2

8)验证测试

8-1) 查看集群状态

# ceph -s //修改完之后不是马上生效

---常见错误

health: HEALTH\_WARN

clock skew detected on node2, node3…

clock skew表示时间不同步，解决办法：请先将所有主机的时间都使用NTP时间同步！！！

Ceph要求所有主机时差不能超过0.05s，否则就会提示WARN。

如果状态还是失败，可以尝试执行如下命令，重启ceph服务：

# systemctl restart ceph\\*.service ceph\\*.target

9)创建Ceph块存储

9-1）查看存储池。

# ceph osd lspools

9-2）创建镜像、查看镜像

# rbd create XXX --image-feature layering --size 10G

#--image-feature参数指定我们创建的镜像有哪些功能，layering是开启COW功能。

#提示：ceph镜像支持很多功能，但很多是操作系统不支持的，我们只开启layering。

# rbd list //查看镜像列表

# rbd info XXX //查看镜像详情

--动态调整

缩小容量

# rbd resize --size 7G image --allow-shrink

扩容容量

# rbd resize --size 15G image

10)通过KRBD访问 //之前全在node1操作 下面为client操作

10-1）客户端通过KRBD访问

#客户端需要安装ceph-common软件包

#拷贝配置文件（否则不知道集群在哪）

#拷贝连接密钥（否则无连接权限）

[root@client ~]# yum -y install ceph-common

[root@client ~]# scp 192.168.4.11:/etc/ceph/ceph.conf /etc/ceph/

[root@client ~]# scp 192.168.4.11:/etc/ceph/ceph.client.admin.keyring /etc/ceph/

[root@client ~]# rbd map XXX //镜像文件

[root@client ~]# lsblk

[root@client ~]# rbd showmapped //查询磁盘对应的镜像文件

10-2) 客户端格式化、挂载分区

[root@client ~]# mkfs.xfs /dev/rbd0

[root@client ~]# mount /dev/rbd0 /mnt/

[root@client ~]# echo "test" > /mnt/test.txt

11)创建镜像快照 //为node1操作

11-1) 查看镜像快照（默认所有镜像都没有快照）。

[root@node1 ~]# rbd snap ls image

11-2) 给镜像创建快照。

[root@node1 ~]# rbd snap create XXX --snap image-snap1

#为image镜像创建快照，快照名称为image-snap1

11-3) 删除客户端写入的测试文件

[root@client ~]# rm -rf /mnt/test.txt

[root@client ~]# umount /mnt

11-4) 还原快照 //不能在挂载状态下还原

[root@node1 ~]# rbd snap rollback XXX --snap image-snap1

#客户端重新挂载分区

[root@client ~]# mount /dev/rbd0 /mnt/

[root@client ~]# ls /mnt

12)创建快照克隆

\*\* 原理 镜像---快照-----克隆镜像

\*\*由于镜像内的数据是可以实时改变的 若直接克隆镜像则会造成数据不对应 所以必须通过快照来克隆新的镜像

12-1）克隆快照

[root@node1 ~]# rbd snap protect image --snap image-snap1 //保护快照 避免在克隆期间被删除

[root@node1 ~]# rbd snap rm image --snap image-snap1 //由于被保护 删除会失败

[root@node1 ~]# rbd clone \

image(原镜像) --snap image-snap1(原镜像生成的快照) XXX(克隆镜像名) --image-feature layering

//使用image的快照image-snap1克隆一个新的名称为XXX镜像

12-2）查看克隆镜像与父镜像快照的关系

[root@node1 ~]# rbd info image-clone

rbd image 'image-clone':

size 15360 MB in 3840 objects

order 22 (4096 kB objects)

block\_name\_prefix: rbd\_data.d3f53d1b58ba

format: 2

features: layering

flags:

parent: rbd/image@image-snap1 //存在此行由于是克隆快照无法直接使用 必须通过 rbd flatten XX 命令将数据全部拷贝过来

#克隆镜像很多数据都来自于快照链

#如果希望克隆镜像可以独立工作，就需要将父快照中的数据，全部拷贝一份，但比较耗时！！！

[root@node1 ~]# rbd flatten image-clone //将数据从原盘拷贝

[root@node1 ~]# rbd list //可查询到新生成的快照

块存储应用案例

KVM虚拟机调用Ceph镜像作为虚拟机的磁盘

1)配置libvirt secret

编写账户信息文件，让KVM知道ceph的账户名称

# vim secret.xml //新建临时文件，内容如下

<secret ephemeral='no' private='no'>

<usage type='ceph'>

<name>client.admin secret</name>

</usage>

</secret>

#使用XML配置文件创建secret

[root@room9pc01 ~]# virsh secret-define secret.xml

733f0fd1-e3d6-4c25-a69f-6681fc19802b

//随机的UUID，这个UUID对应的有账户信息

2)给secret绑定admin账户的密码，密码参考ceph.client.admin.keyring文件

virsh secret-set-value \

--secret 733f0fd1-e3d6-4c25-a69f-6681fc19802b \

--base64 AQBTsdRapUxBKRAANXtteNUyoEmQHveb75bISg

//这里secret后面是之前创建的secret的UUID

//base64后面是client.admin账户的密码

//现在secret中既有账户信息又有密钥信息

3)虚拟机的XML配置文件

不推荐直接使用vim修改配置文件，推荐使用virsh edit(退出保存生效)修改配置文件，效果如下：

virsh edit tedu\_node01 //tedu\_node01为虚拟机名称

<disk type='network' device='disk'>

<driver name='qemu' type='raw'/>

<auth username='admin'>

<secret type='ceph' uuid='733f0fd1-e3d6-4c25-a69f-6681fc19802b'/>

</auth>

<source protocol='rbd' name='rbd/vm1-image'>  
 <host name='192.168.4.11' port='6789'/>  
 </source>

<target dev='vda' bus='virtio'/>

</disk>

\*\*备注：修改secret的UUID，修改source中的共享名name，修改dev设备名称。

Ceph文件系统

1）部署元数据服务器

登陆node4，安装ceph-mds软件包

# yum -y install ceph-mds

2)登陆node1部署节点操作

[root@node1 ~]# cd /root/ceph-cluster

//该目录，是最早部署ceph集群时，创建的目录

[root@node1 ceph-cluster]# ceph-deploy mds create node4

//远程nod4，拷贝配置文件，启动mds服务

如果没有配置文件则可以通过admin命令重新发送配置和密钥（备选操作）

[root@node1 ceph-cluster]# ceph-deploy admin node4

//同步配置文件和key

3)创建存储池

[root@node4 ~]# ceph osd pool create cephfs\_data 128

//创建存储池，对应128个PG

[root@node4 ~]# ceph osd pool create cephfs\_metadata 128

//创建存储池，对应128个PG

4）创建Ceph文件系统

# ceph fs new myfs1 cephfs\_metadata cephfs\_data

new fs with metadata pool 2 and data pool 1

//注意，先写medadata池，再写data池

//默认，只能创建1个文件系统，多余的会报错

# ceph fs ls

name: myfs1, metadata pool: cephfs\_metadata, data pools: [cephfs\_data ]

]# ceph mds stat

e4: 1/1/1 up {0=node4=up:creating}

5）客户端挂载

# mount -t ceph 192.168.4.11:6789:/ /mnt/cephfs/ \

-o name=admin,secret=AQBTsdRapUxBKRAANXtteNUyoEmQHveb75bISg==

//注意:文件系统类型为ceph

//192.168.4.11为MON节点的IP（不是MDS节点）

//admin是用户名,secret是密钥

//密钥可以在/etc/ceph/ceph.client.admin.keyring中找到

LVS Linux Virtual Server

默认写入内核程序默认开启

LVS 原理与 Nginx 完全不同

LVS 为调度功能 及 不修改数据包的IP层 [ SIP(源ip) DIP(目标ip) ] 而是修改数据包包头的MAC地址来达到转发功能 (与交换机原理相同)

Nginx 为代理功能 及 解封数据包后 重新打包封装 发给真实服务器

集群分类

HPC 高性能计算集群

LB 负载均衡

HA 高可用

LVS工作模式

NAT模式 小规模 调度服务器作为一个软路由 负载存在上线

DR模式 大规模 服务器反馈给客户端不通过调度器 通过其他手段实现

TUN模式 不常用

\*\* DR与NAT不同点在于 数据包返回的方式不同 不通过调度服务器

负载均衡常用调度算法

目前有10种

常用

--轮询(Round Robin)

--加权轮询(Weighted Round Robin)

--最少连接(Least Connections)

--加权最少连接(Weighted Least Connections)

最少连接 优于 轮询

\*\* 轮询必定为 1人 1次 若存在客户端持续连接 则会导致 服务器负载不均衡情况

###################################################################################

LVS 也是通过ipvsadm软件实现设置

虚拟集群服务器:

ipvsadm -[A|E|D|C] -[t|u] 192.168.4.5:80(调度器ip及端口) -[s] [rr|wrr|lc|wlc|sh|dh]

-A 添加虚拟集群服务器

-E 编辑虚拟集群服务器

-D 删除虚拟集群服务器(单一)

-C 清空所有虚拟集群服务器

-t TCP协议

-u UDP协议

-s 算法

-rr 轮询 wrr 加权轮询 lc 最少连接次数 wlc 加权最少连接数 sh dh 与ip\_hash作用相同

虚拟集群服务器中真实服务器:

ipvsadm -[a|e|d|] -[t|u] 192.168.4.5:80(虚拟集群服务器ip及端口) -r 192.168.4.10 (真实服务器ip) -[g|m|i] (默认为g) -w (权重)

-a 添加真实服务器

-e 编辑真实服务器

-d 删除真实服务器

-g DR模式

-m NAT模式

-i 隧道模式

ipvsadm -Ln 查看规则列表

####################################################################################

部署LVS-NAT集群

\*\* 由于NAT模式需要将真实服务器地址返回的数据包再次通过调度服务器发送 并且 内部通讯与 客户端与调度通讯 跨网段

\*\* 所以优先要给真是服务器配置网关 及调度ip作为网关 (ip route 查看)

1)部署LVS-NAT模式调度器

确认调度器的路由转发功能

# echo 1 > /proc/sys/net/ipv4/ip\_forward //开启路由转发功能

2)创建集群服务器

# yum -y install ipvsadm

# ipvsadm -A -t 192.168.4.5:80 -s wrr

3)添加真实服务器

# ipvsadm -a -t 192.168.4.5:80 -r 192.168.2.100 -w 1 -m

# ipvsadm -a -t 192.168.4.5:80 -r 192.168.2.200 -w 1 -m

4）查看规则列表，并保存规则

# ipvsadm -Ln

# ipvsadm-save -n > /etc/sysconfig/ipvsadm //存入配置文档 开启服务自动加载. 若不存入 重启及消失

#########################################################################################

部署LVS-DR集群

配置实验网络环境

1）设置Proxy代理服务器的VIP和DIP

注意：为了防止冲突，VIP必须要配置在网卡的虚拟接口！！！

# cp /etc/sysconfig/network-scripts/ifcfg-eth0 /etc/sysconfig/network-scripts/ifcfg-eth0:0 //虚拟网卡文档

TYPE=Ethernet

DEFROUTE=yes

NAME=eth0:0

DEVICE=eth0:0

ONBOOT=yes

IPADDR=192.168.4.15

PREFIX=24

# systemctl restart network //若重启后未生效 则因为网卡是有2个服务共同管理 可关闭 NetworkManager 后再次重启

2）设置Web1 web2服务器网络参数 (假装自己为调度通过vip发送)

cp /etc/sysconfig/network-scripts/ifcfg-lo /etc/sysconfig/network-scripts/ifcfg-lo:0 //虚拟网卡文档

DEVICE=lo:0

IPADDR=192.168.4.15

NETMASK=255.255.255.255 //必须为4个255

NETWORK=192.168.4.15

BROADCAST=192.168.4.15

ONBOOT=yes

NAME=lo:0

3)防止地址冲突的问题

这里因为web1 web2也配置与代理一样的VIP地址，默认肯定会出现地址冲突；

sysctl.conf文件写入这下面四行的主要目的就是访问192.168.4.15的数据包，只有调度器会响应，其他主机都不做任何响应，这样防止地址冲突的问题。

# vim /etc/sysctl.conf

#手动写入如下4行内容

net.ipv4.conf.all.arp\_ignore = 1 //设置所有网卡(包括lo)该回答的回答 不该回答不回答

net.ipv4.conf.lo.arp\_ignore = 1 //单独设置lo 该回答回答 不该回答不回答

net.ipv4.conf.lo.arp\_announce = 2 //设置所有网卡(包括lo)什么都不申明

net.ipv4.conf.all.arp\_announce = 2 //单独设置lo 什么都不申明

#当有arp广播问谁是192.168.4.15时，本机忽略该ARP广播，不做任何回应

#本机不要向外宣告自己的lo回环地址是192.168.4.15

# sysctl -p //刷新配置

4)重启网络服务，关闭防火墙与SELinux

# systemctl restart network

proxy调度器安装软件并部署LVS-DR模式调度器

创建新的集群服务器规则

# ipvsadm -A -t 192.168.4.15:80 -s wrr

添加真实服务器(-g参数设置LVS工作模式为DR模式，-w设置权重)

# ipvsadm -a -t 192.168.4.15:80 -r 192.168.4.100 -g -w 1

# ipvsadm -a -t 192.168.4.15:80 -r 192.168.4.200 -g -w 1

查看规则列表，并保存规则

# ipvsadm -Ln

# ipvsadm-save -n > /etc/sysconfig/ipvsadm //存入配置文档 开启服务自动加载. 若不存入 重启及消失

##########################################################################################################################

Keepalived

Keepalived运行原理

\* 检测每个服务器节点状态

\* 具有安全检测 服务器节点出现异常或故障 keepalived将故障节点从集训中剔除

\* 故障节点恢复后 再将其加入到集群中

\* 所有工作自动完成 无需人工干预

##########################################################################

Keepalived高可用服务器

\*原理与VRRP一致 一台作为主服务器 一台作为备用服务器

\*主 备关系由优先级大小决定(priority) 若一致则通过ip大小比较 ip大优先

安装Keepalived软件

# yum install -y keepalived

修改web1服务器Keepalived配置文件

# vim /etc/keepalived/keepalived.conf

global\_defs {

notification\_email {

admin@tarena.com.cn //设置报警收件人邮箱

}

notification\_email\_from ka@localhost //设置发件人

smtp\_server 127.0.0.1 //定义邮件服务器

smtp\_connect\_timeout 30

router\_id web1 \*\*\* //设置路由ID号（实验需要修改）

}

vrrp\_instance VI\_1 {

state MASTER \*\*\* //主服务器为MASTER（备服务器需要修改为BACKUP）

interface eth0 //定义网络接口

virtual\_router\_id 51 \*\*\*与VRID效果一致 若存在同一id则会影响//主备服务器VRID号必须一致

priority 100 \*\*\*//服务器优先级,优先级高优先获取VIP

advert\_int 1

authentication {

auth\_type pass

auth\_pass 1111 //主备服务器密码必须一致

}

virtual\_ipaddress { \*\*\*\*//谁是主服务器谁获得该VIP（实验需要修改）

192.168.4.80

}

}

修改web2服务器Keepalived配置文件

# vim /etc/keepalived/keepalived.conf

global\_defs {

notification\_email {

admin@tarena.com.cn //设置报警收件人邮箱

}

notification\_email\_from ka@localhost //设置发件人

smtp\_server 127.0.0.1 //定义邮件服务器

smtp\_connect\_timeout 30

router\_id web2 \*\*\* //设置路由ID号（实验需要修改）

}

vrrp\_instance VI\_1 {

state BACKUP //备服务器为BACKUP（实验需要修改）

interface eth0 //定义网络接口

virtual\_router\_id 51 \*\*\* //主辅VRID号必须一致

priority 50 \*\*\* //服务器优先级（实验需要修改）

advert\_int 1

authentication {

auth\_type pass

auth\_pass 1111 \*\*\* //主辅服务器密码必须一致

}

virtual\_ipaddress { \*\*\*//谁是主服务器谁配置VIP（实验需要修改）

192.168.4.80

}

}

启动服务

# systemctl start keepalived

配置防火墙和SELinux

启动keepalived会自动添加一个drop的防火墙规则，需要清空！

# iptables -F //必须要 并且不是马上生效可能存在1s 2s 延迟

# setenforce 0

登录两台Web服务器查看VIP信息

# ip addr show eth0 与 ip a s eth0d

#########################################################################################

Keepalived+LVS服务器

\*\*本次实验目的为设置一台主调度服务器 一台备用调度 达到高可用效果

\*\*本次实验lvs模式为DR模式 需要在真实服务器web1 web2 上设置回环地址与vip一致 及lo:0 (与vip一致)

1）LVS1调度器设置Keepalived，并启动服务

# yum install -y keepalived

# vim /etc/keepalived/keepalived.conf

global\_defs {

notification\_email {

admin@tarena.com.cn //设置报警收件人邮箱

}

notification\_email\_from ka@localhost //设置发件人

smtp\_server 127.0.0.1 //定义邮件服务器

smtp\_connect\_timeout 30

router\_id web1 \*\*\* //设置路由ID号（实验需要修改）

}

vrrp\_instance VI\_1 {

state MASTER \*\*\* //主服务器为MASTER（备服务器需要修改为BACKUP）

interface eth0 //定义网络接口

virtual\_router\_id 51 \*\*\*与VRID效果一致 若存在同一id则会影响//主备服务器VRID号必须一致

priority 100 \*\*\*//服务器优先级,优先级高优先获取VIP

advert\_int 1

authentication {

auth\_type pass

auth\_pass 1111 //主备服务器密码必须一致

}

virtual\_ipaddress { \*\*\*\*//谁是主服务器谁获得该VIP（实验需要修改）

192.168.4.15

}

}

virtual\_server 192.168.4.15 80 { //设置ipvsadm的VIP规则（实验需要修改）

delay\_loop 6

lb\_algo rr //设置LVS调度算法为RR

lb\_kind DR //设置LVS的模式为DR（实验需要修改）

#persistence\_timeout 50 //（实验需要注释）

#注意这样的作用是保持连接，开启后，客户端在一定时间内始终访问相同服务器

protocol TCP

real\_server 192.168.4.100 80 { //设置后端web服务器真实IP（实验需要修改）

weight 1 //设置权重为1

TCP\_CHECK { //对后台real\_server做健康检查（实验需要修改）

connect\_timeout 3

nb\_get\_retry 3

delay\_before\_retry 3

}

}

real\_server 192.168.4.200 80 { //设置后端web服务器真实IP（实验需要修改）

weight 2 //设置权重为1

TCP\_CHECK { //对后台real\_server做健康检查（实验需要修改）

connect\_timeout 3

nb\_get\_retry 3

delay\_before\_retry 3

}

}

}

# systemctl start keepalived

# ipvsadm -Ln #查看LVS规则

# ip a s #查看VIP配置

# iptables -F

##########################################################################################

配置HAProxy负载平衡集群

原理与nginx一致 用户访问haproxy-- haproxy访问真实服务器

功能 lvs<HAProxy<Nginx

性能 lvs>HAProxy>Nginx

\*\* 实用性最高

# yum -y install haproxy

# vim /etc/haproxy/haproxy.cfg

global

log 127.0.0.1 local2 ##[err warning info debug]

chroot /usr/local/haproxy

pidfile /var/run/haproxy.pid ##haproxy的pid存放路径

maxconn 4000 ##最大连接数，默认4000

user haproxy

group haproxy

daemon ##创建1个进程进入deamon模式运行

defaults

mode http ##默认的模式mode { tcp|http|health }

option dontlognull ##不记录健康检查的日志信息

option httpclose ##每次请求完毕后主动关闭http通道

option httplog ##日志类别http日志格式

option forwardfor ##后端服务器可以从Http Header中获得客户端ip

option redispatch ##serverid服务器挂掉后强制定向到其他健康服务器

timeout connect 10000 #如果backend没有指定，默认为10s

timeout client 300000 ##客户端连接超时

timeout server 300000 ##服务器连接超时

maxconn 3000 ##最大连接数

retries 3 ##3次连接失败就认为服务不可用，也可以通过后面设置

listen stats 0.0.0.0:1080 #监听端口

stats refresh 30s #统计页面自动刷新时间

stats uri /stats #统计页面url

stats realm Haproxy Manager #进入管理解面查看状态信息

stats auth admin:admin #统计页面用户名和密码设置

\*\* 以下需要手写

listen websrv-rewrite 0.0.0.0:80 // 与listen XX \*:80 效果相同

balance roundrobin // 计算方式必须全写 lestconn

server web1 192.168.2.100:80 check inter 2000 rise 2 fall 5

server web2 192.168.2.200:80 check inter 2000 rise 2 fall 5

\*\* check 为安全检查参数

\*\* inter 安全检查间隔 默认毫秒为单位

\*\* rise 开启测试2次成功 则加入集群

\*\* fall 失败测试5次 则从集群删除

######################################################################################################

存储的分类

DAS直连存储 ide sata usb 家用

NAS网络附加存储 nfs samba ftp 分享类型为[文件系统]

SAN存储区域网络 iscsi 分享类型为 [块设备]

SDS软件定义存储(分布式存储) ceph

ceph组件(默认3副本) 及一份文件默认存3个副本 若坏了其中一个会自动拷贝到另外一台 保证副本一直有3份

由于过半原则 如果是2台 坏一台也是整个集群坏了 所以mon至少三台

OSDs 存储设备 共享磁盘 ceph-osd

Monitors 集群监控组件 ceph-mon

RadosGateway 对象存储网关 ceph-radosgw

MDSs 存放文件系统的元数据(若不装则是块设备 不存在文件系统) ceph-mds

########################################################################################################

准备四台KVM虚拟机，其三台作为存储集群节点，一台安装为客户端，实现如下功能：

三台作即作为mon 也作为 osd

1)配置无密码连接(包括自己远程自己也不需要密码)，在node1操作 //为后期统一操作做准备

# ssh-keygen -f /root/.ssh/id\_rsa -N ''

# for i in 10 11 12 13

do

ssh-copy-id 192.168.4.$i

done

2）修改/etc/hosts并同步到所有主机。

警告：/etc/hosts解析的域名必须与本机主机名一致！！！！

# vim /etc/hosts

192.168.4.10 client

192.168.4.11 node1

192.168.4.12 node2

192.168.4.13 node3

3)修改所有节点都需要配置YUM源，并同步到所有主机

# cat /etc/yum.repos.d/ceph.repo

[mon]

name=mon

baseurl=ftp://192.168.4.254/ceph/MON

gpgcheck=0

[osd]

name=osd

baseurl=ftp://192.168.4.254/ceph/OSD

gpgcheck=0

[tools]

name=tools

baseurl=ftp://192.168.4.254/ceph/Tools

gpgcheck=0

# for i in client node1 node2 node3

do

scp /etc/yum.repos.d/ceph.repo $i:/etc/yum.repos.d/

done

4）所有节点主机与真实主机的NTP服务器同步时间 //Ceph要求所有主机时差不能超过0.05s，否则就会提示WARN

# vim /etc/chrony.conf

… …

server 192.168.4.254 iburst

# for i in client node1 node2 node3

do

scp /etc/chrony.conf $i:/etc/

ssh $i "systemctl restart chronyd"

done

5)准备存储磁盘

物理机上为每个虚拟机准备3块20G磁盘 使用图形直接添加

5-1）在node1安装部署工具，学习工具的语法格式

# yum -y install ceph-deploy

5-2）创建目录 //必须创建 ceph-XXX 目录 后续ceph-deploy 等命令只能在该目录内生效 与git原理一致

# mkdir ceph-cluster

# cd ceph-cluster/

6)部署Ceph集群

6-1)创建Ceph集群配置,在ceph-cluster目录下生成Ceph配置文件

# ceph-deploy new node1 node2 node3 // 此步创建为 new

6-2)给所有节点安装ceph相关软件包

# for i in node1 node2 node3

do

ssh $i "yum -y install ceph-mon ceph-osd ceph-mds ceph-radosgw"

done

6-3)初始化所有节点的mon服务，也就是启动mon服务（主机名解析必须对 /etc/hosts 目录内）。

# ceph-deploy mon create-initial //启动ceph-mon@XX.service 服务

7)创建OSD

7-1)vdb1和vdb2这两个分区用来做存储服务器的journal缓存盘 //创建2个固态盘 即先将需要存储的数据高效的缓存到缓存盘 之后慢慢同步到osd

# for i in node1 node2 node3

do

ssh $i "parted /dev/vdb mklabel gpt"

ssh $i "parted /dev/vdb mkpart primary 1 50%"

ssh $i "parted /dev/vdb mkpart primary 50% 100%"

done

7-2)磁盘分区后的默认权限无法让ceph软件对其进行读写操作，需要修改权限。

node1、node2、node3都需要操作，这里以node1为例。

# chown ceph.ceph /dev/vdb1

# chown ceph.ceph /dev/vdb2

#上面的权限修改为临时操作，重启计算机后，权限会再次被重置。

#我们还需要将规则写到配置文件实现永久有效。

# vim /etc/udev/rules.d/70-vdb.rules

ENV{DEVNAME}=="/dev/vdb1",OWNER="ceph",GROUP="ceph"

ENV{DEVNAME}=="/dev/vdb2",OWNER="ceph",GROUP="ceph"

7-3）初始化清空磁盘数据

# ceph-deploy disk zap node1:vdc node1:vdd

# ceph-deploy disk zap node2:vdc node2:vdd

# ceph-deploy disk zap node3:vdc node3:vdd

7-4）创建OSD存储空间（仅node1操作即可)

# ceph-deploy osd create \

node1:vdc:/dev/vdb1 node1:vdd:/dev/vdb2

//创建osd存储设备，vdc为集群提供存储空间，vdb1提供JOURNAL缓存，

//每个存储设备对应一个缓存设备，缓存需要SSD，不需要很大

# ceph-deploy osd create \

node2:vdc:/dev/vdb1 node2:vdd:/dev/vdb2

# ceph-deploy osd create \

node3:vdc:/dev/vdb1 node3:vdd:/dev/vdb2

8)验证测试

8-1) 查看集群状态

# ceph -s //修改完之后不是马上生效

---常见错误

health: HEALTH\_WARN

clock skew detected on node2, node3…

clock skew表示时间不同步，解决办法：请先将所有主机的时间都使用NTP时间同步！！！

Ceph要求所有主机时差不能超过0.05s，否则就会提示WARN。

如果状态还是失败，可以尝试执行如下命令，重启ceph服务：

# systemctl restart ceph\\*.service ceph\\*.target

9)创建Ceph块存储

9-1）查看存储池。

# ceph osd lspools

9-2）创建镜像、查看镜像

# rbd create XXX --image-feature layering --size 10G

#--image-feature参数指定我们创建的镜像有哪些功能，layering是开启COW功能。

#提示：ceph镜像支持很多功能，但很多是操作系统不支持的，我们只开启layering。

# rbd list //查看镜像列表

# rbd info XXX //查看镜像详情

--动态调整

缩小容量

# rbd resize --size 7G image --allow-shrink

扩容容量

# rbd resize --size 15G image

10)通过KRBD访问 //之前全在node1操作 下面为client操作

10-1）客户端通过KRBD访问

#客户端需要安装ceph-common软件包

#拷贝配置文件（否则不知道集群在哪）

#拷贝连接密钥（否则无连接权限）

[root@client ~]# yum -y install ceph-common

[root@client ~]# scp 192.168.4.11:/etc/ceph/ceph.conf /etc/ceph/

[root@client ~]# scp 192.168.4.11:/etc/ceph/ceph.client.admin.keyring /etc/ceph/

[root@client ~]# rbd map XXX //镜像文件

[root@client ~]# lsblk

[root@client ~]# rbd showmapped //查询磁盘对应的镜像文件

10-2) 客户端格式化、挂载分区

[root@client ~]# mkfs.xfs /dev/rbd0

[root@client ~]# mount /dev/rbd0 /mnt/

[root@client ~]# echo "test" > /mnt/test.txt

11)创建镜像快照 //为node1操作

11-1) 查看镜像快照（默认所有镜像都没有快照）。

[root@node1 ~]# rbd snap ls image

11-2) 给镜像创建快照。

[root@node1 ~]# rbd snap create XXX --snap image-snap1

#为image镜像创建快照，快照名称为image-snap1

11-3) 删除客户端写入的测试文件

[root@client ~]# rm -rf /mnt/test.txt

[root@client ~]# umount /mnt

11-4) 还原快照 //不能在挂载状态下还原

[root@node1 ~]# rbd snap rollback XXX --snap image-snap1

#客户端重新挂载分区

[root@client ~]# mount /dev/rbd0 /mnt/

[root@client ~]# ls /mnt

12)创建快照克隆

\*\* 原理 镜像---快照-----克隆镜像

\*\*由于镜像内的数据是可以实时改变的 若直接克隆镜像则会造成数据不对应 所以必须通过快照来克隆新的镜像

12-1）克隆快照

[root@node1 ~]# rbd snap protect image --snap image-snap1 //保护快照 避免在克隆期间被删除

[root@node1 ~]# rbd snap rm image --snap image-snap1 //由于被保护 删除会失败

[root@node1 ~]# rbd clone \

image(原镜像) --snap image-snap1(原镜像生成的快照) XXX(克隆镜像名) --image-feature layering

//使用image的快照image-snap1克隆一个新的名称为XXX镜像

12-2）查看克隆镜像与父镜像快照的关系

[root@node1 ~]# rbd info image-clone

rbd image 'image-clone':

size 15360 MB in 3840 objects

order 22 (4096 kB objects)

block\_name\_prefix: rbd\_data.d3f53d1b58ba

format: 2

features: layering

flags:

parent: rbd/image@image-snap1 //存在此行由于是克隆快照无法直接使用 必须通过 rbd flatten XX 命令将数据全部拷贝过来

#克隆镜像很多数据都来自于快照链

#如果希望克隆镜像可以独立工作，就需要将父快照中的数据，全部拷贝一份，但比较耗时！！！

[root@node1 ~]# rbd flatten image-clone //将数据从原盘拷贝

[root@node1 ~]# rbd list //可查询到新生成的快照

块存储应用案例

KVM虚拟机调用Ceph镜像作为虚拟机的磁盘

1)配置libvirt secret

编写账户信息文件，让KVM知道ceph的账户名称

# vim secret.xml //新建临时文件，内容如下

<secret ephemeral='no' private='no'>

<usage type='ceph'>

<name>client.admin secret</name>

</usage>

</secret>

#使用XML配置文件创建secret

[root@room9pc01 ~]# virsh secret-define secret.xml

733f0fd1-e3d6-4c25-a69f-6681fc19802b

//随机的UUID，这个UUID对应的有账户信息

2)给secret绑定admin账户的密码，密码参考ceph.client.admin.keyring文件

virsh secret-set-value \

--secret 733f0fd1-e3d6-4c25-a69f-6681fc19802b \

--base64 AQBTsdRapUxBKRAANXtteNUyoEmQHveb75bISg

//这里secret后面是之前创建的secret的UUID

//base64后面是client.admin账户的密码

//现在secret中既有账户信息又有密钥信息

3)虚拟机的XML配置文件

不推荐直接使用vim修改配置文件，推荐使用virsh edit(退出保存生效)修改配置文件，效果如下：

virsh edit tedu\_node01 //tedu\_node01为虚拟机名称

<disk type='network' device='disk'>

<driver name='qemu' type='raw'/>

<auth username='admin'>

<secret type='ceph' uuid='733f0fd1-e3d6-4c25-a69f-6681fc19802b'/>

</auth>

<source protocol='rbd' name='rbd/vm1-image'>  
 <host name='192.168.4.11' port='6789'/>  
 </source>

<target dev='vda' bus='virtio'/>

</disk>

\*\*备注：修改secret的UUID，修改source中的共享名name，修改dev设备名称。

Ceph文件系统

1）部署元数据服务器

登陆node4，安装ceph-mds软件包

# yum -y install ceph-mds

2)登陆node1部署节点操作

[root@node1 ~]# cd /root/ceph-cluster

//该目录，是最早部署ceph集群时，创建的目录

[root@node1 ceph-cluster]# ceph-deploy mds create node4

//远程nod4，拷贝配置文件，启动mds服务

如果没有配置文件则可以通过admin命令重新发送配置和密钥（备选操作）

[root@node1 ceph-cluster]# ceph-deploy admin node4

//同步配置文件和key

3)创建存储池

[root@node4 ~]# ceph osd pool create cephfs\_data 128

//创建存储池，对应128个PG

[root@node4 ~]# ceph osd pool create cephfs\_metadata 128

//创建存储池，对应128个PG

4）创建Ceph文件系统

# ceph fs new myfs1 cephfs\_metadata cephfs\_data

new fs with metadata pool 2 and data pool 1

//注意，先写medadata池，再写data池

//默认，只能创建1个文件系统，多余的会报错

# ceph fs ls

name: myfs1, metadata pool: cephfs\_metadata, data pools: [cephfs\_data ]

]# ceph mds stat

e4: 1/1/1 up {0=node4=up:creating}

5）客户端挂载

# mount -t ceph 192.168.4.11:6789:/ /mnt/cephfs/ \

-o name=admin,secret=AQBTsdRapUxBKRAANXtteNUyoEmQHveb75bISg==

//注意:文件系统类型为ceph

//192.168.4.11为MON节点的IP（不是MDS节点）

//admin是用户名,secret是密钥

//密钥可以在/etc/ceph/ceph.client.admin.keyring中找到