线上nginx 日志切割并做gzip压缩，同步到线下备份服务器

一，线上ngxin日志切割

由于 Nginx 的日志都是写在一个文件当中的，因此，我们需要每天零点将前一天的日志存为另外一个文件，这里我们就将 Nginx 位于 logs 目录中的 access.log 存为 access\_[yyyy-MM-dd].log 的文件。其实 logs 目录中还有个 error.log 的错误日志文件，这个文件也需要每天切割一个，在这里就说 access.log 了，error.log 的切割方法类似。

[root@nginx-2 logs]# vim /data/nginx/sbin/cut-log.sh

#!/bin/bash

## 零点执行该脚本

## Nginx 日志文件所在的目录

LOGS\_PATH=/data/nginx/logs

## 获取昨天的 yyyy-MM-dd

YESTERDAY=$(date -d "yesterday" +%Y-%m-%d)

## 移动文件

mv ${LOGS\_PATH}/access.log ${LOGS\_PATH}/access\_${YESTERDAY}.log

## 向 Nginx 主进程发送 USR1 信号。USR1 信号是重新打开日志文件

kill -USR1 $(cat /data/nginx/logs/nginx.pid)

[root@nginx-2 logs]# crontab -l

0 0 \* \* \* /bin/bash /data/nginx/sbin/cut-log.sh

在零点的时候执行脚本

二，做rsync的安装及配置，

rsync 实例

实施步骤：

在线上服务器端安装rsync服务(多台nginx可以用saltstack安装）

#yum install –y rsync

#yum install –y xinetd

编辑rsync配置文件，用以收集/data/nginx下的nginx日志

#vim /etc/rsyncd.conf (手动配置）

uid=nobody

gid=nobody

use chroot = yes

pid file = /var/run/rsyncd.pid

log file = /var/log/rsyncd.log

[nginxlog]

path = /data/nginx/logs

comment = nginx

auth users = ruser

secrets file = /etc/rsync\_pass

read only = yes

编辑rsync认证文件

   #vim /etc/rsync\_pass

   ruser:[jjjr@2016](mailto:jjjr@2016)

**# chmod 600** /etc/rsync\_pass

修改xinetd启动配置

#sed -i "s/yes/no/g" /etc/xinetd.d/rsync

把disable = yes改成no

启动xinetd服务

#service xinetd start

#chkconfig xinetd on

客户端

  #vim /etc/rsync\_pass 客户端只要密码就行

   j[jjr@2016](mailto:jjjr@2016)

**# chmod 600** /etc/rsync\_pass

三，在客户端上做线下的rsync同步备份（线下的有拨VPN到线上）

rsync同步脚本：

[root@FTP nginx]# vim /usr/local/bin/nginxback.sh

#!/bin/bash

set -e

#info格式为 IP地址:项目名:主机名

info=( '10.10.105.97:jjjr2:nginx-1'

'10.10.102.240:jjjr2:nginx-2'

'10.10.53.33:b52:B52-nginx-2'

'10.10.39.245:xd:XD-Nginx-1'

'10.10.18.226:xd:XD-Nginx-2'

'10.10.85.96:yyjf:YYJF-nginx-1'

'10.10.89.105:yyjf:YYJF-nginx-2'

)

#num为数组长度

num=${#info[@]}

#IP为地址，PN为项目名称，HN为主机名

INFO() {

IP=`echo ${info[$i]}| awk -F ':' '{print $1}' `

PN=`echo ${info[$i]}| awk -F ':' '{print $2}' `

HN=`echo ${info[$i]}| awk -F ':' '{print $3}' `

ssh 10.10.109.47 "salt ${HN} state.sls rsyncd\_nginx\_log saltenv=base" 在同步之前先执行脚本

wait

}

#执行同步操作

RSYNC() {

if [ -f /ndata/nginx/$PN/$HN ]

then

rsync -arvz --bwlimit=1000 --exclude '\*.log' --exclude 'hack' --password-file=/root/rsync\_pass ruser@$IP::nginxlog /ndata/nginx/$PN/$HN/

#rsync -arvz --password-file=/root/rsync\_pass ruser@10.10.105.97::nginxlog /ndata/nginx/jjjr2/nginx-1

else

mkdir -p /ndata/nginx/$PN/$HN

rsync -arvz --bwlimit=1000 --exclude '\*.log' --exclude 'hack' --password-file=/root/rsync\_pass ruser@$IP::nginxlog /ndata/nginx/$PN/$HN/

fi

if [ $? != 0 ]

then

echo "some problems occured about $HN backup" >> /ndata/nginx/logs/`date +%F`.log

else

echo "`date +%F\_%T` $HN rsync done" >> /ndata/nginx/logs/`date +%F`.log

fi

echo "The $HN logs backup done."

}

for ((i=0;i<num;i++))

do

INFO

RSYNC;

done

echo "All of nginx logs backup done." >> /ndata/nginx/logs/`date +%F`.log

~

~

做定时任务 crontable -e

30 00 \* \* \* /bin/bash /usr/local/bin/nginxback.sh 在00：30分同步备份压缩后的日志到线下服务器

附：

[root@monitor base]# vim /data/salt/base/sls/rsyncd\_nginx\_log.sls 做saltstack 执行sls

1 #打开文件优化

2 /tmp/rsyncd\_nginx\_log.sh:

3 file.managed:

4 - source: salt://scripts/rsyncd\_nginx\_log.sh

5 - user: root

6 - mode: 644

7

8

9 #加载sysctl配置文件

10 start\_rsyncd\_nginx\_log:

11 cmd.run:

12 - name: /bin/bash /tmp/rsyncd\_nginx\_log.sh

13 - watch:

14 - file: /tmp/rsyncd\_nginx\_log.sh

[root@monitor base]# vim /data/salt/base/scripts/rsyncd\_nginx\_log.sh

1 #!/bin/bash

2 set -e

3 #DATE=$(/bin/date '+%F');

4 for i in $(/bin/find /data/nginx/logs -name 'access\*.log');

5 do

6 #out\_file=${i}.gz

7 #sed -n "/${DATE} ${HOUR\_BEGIN}:00:00/,/${DATE} ${HOUR\_END}:00:00/p" ${i}|gzip > ${out\_file};

8 gzip -c ${i} > ${i}.gz;

9 done

10

11 find /data/nginx/logs -type f -mtime +15 -name 'access\*.log' -exec rm -rf {} \; 删除15天前的‘access\*.log'日志