实验九：系统运维管理

一、实验目的

1、掌握 Linux 系统下常用的监控命令；

2、掌握 Shell 编程的基本语法；

3、掌握使用 Shell 编程实现系统运维的方法。

二、实验学时

2 学时

三、实验类型

创新性

四、实验需求

1、硬件

每人配备计算机 1 台，不低于双核 CPU、8G 内存、500GB 硬盘。

2、软件

Windows 操作系统，安装 VirtualBox 虚拟化软件，安装 Putty 管理终端软件。

3、网络

计算机使用固定 IP 地址接入局域网，并支持对互联网的访问，虚拟主机可通过 NAT 方式访问互联网。

4、工具

提供第三方 FTP 服务，虚拟主机能够访问该 FTP 服务。

五、实验任务

1、完成 Linux 基本运维命令的应用：top、iotop、iftop、sar；

2、编写 Linux Shell 脚本，完成系统日志中用户登录过程分析；

3、编写 Linux Shell 脚本，完成系统指定目录的本地与远程定时数据备份。

4、编写 Linux Shell 脚本，完成批量化网站的创建与发布。

六、实验内容及步骤

1、使用监控命令进行系统监控

（1）top

使用 top 命令查看系统运行状态，并分析系统进程变化。

```
# top
```

```
top - 11:23:37 up  1:57,  2 users,  load average: 0.00, 0.01, 0.05
Tasks:  83 total,   3 running,  80 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem:   1017936 total,   187272 used,   830664 free,      764 buffers
KiB Swap:  2097148 total,        0 used,  2097148 free.    67856 cached Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
 2263 root      20   0       0      0      0 R  0.3  0.0   0:01.38 kworker/0:3
 2292 root      20   0  123660   1584   1096 R  0.3  0.2   0:01.43 top
    1 root      20   0   50768   7160   2484 S  0.0  0.7   0:01.09 systemd
    2 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kthreadd
    3 root      20   0       0      0      0 S  0.0  0.0   0:00.04 ksoftirqd/0
    5 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kworker/0:0H
    6 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kworker/u2:0
    7 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 migration/0
    8 root      20   0       0      0      0 S  0.0  0.0   0:00.00 rcu_bh
    9 root      20   0       0      0      0 S  0.0  0.0   0:00.00 rcuob/0
   10 root      20   0       0      0      0 S  0.0  0.0   0:00.61 rcu_sched
```

图 9-1 系统运行状态

请分析 top 命令下各字段的含义，并将结果写入表 9-1。

表 9-1 top 命令含义

（2）iotop

①使用 yum 安装 iotop

```
# yum install iotop
```

②使用 iotop 查看磁盘 I/O 使用状况

```
# iotop
```

```
Total DISK READ :       0.00 B/s | Total DISK WRITE :      21.78 M/s
Actual DISK READ:       0.00 B/s | Actual DISK WRITE:      61.28 M/s
  TID  PRIO  USER     DISK READ  DISK WRITE  SWAPIN     IO>    COMMAND
 2298 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.01 % [kworker/0:1]
 2738 be/4 root        0.00 B/s   10.77 M/s  0.00 %  0.00 % sftp-server
 2745 be/4 root        0.00 B/s   11.01 M/s  0.00 %  0.00 % sftp-server
    1 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % systemd --switched-em --deserialize 23
    2 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [kthreadd]
    3 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [ksoftirqd/0]
    5 be/0 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [kworker/0:0H]
    6 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [kworker/u2:0]
    7 rt/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [migration/0]
    8 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [rcu_bh]
    9 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [rcuob/0]
   10 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [rcu_sched]
   11 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [rcuos/0]
   12 rt/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [watchdog/0]
   13 be/0 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [khelper]
   14 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [kdevtmpfs]
   15 be/0 root        0.00 B/s    0.00 B/s  0.00 %  0.00 % [netns]
```

图 9-2 磁盘 I/O 使用状况

③使用 iotop -o 查看磁盘正在产生 I/O 的进程或线程

```
# iotop -o
```

```
Total DISK READ :       0.00 B/s | Total DISK WRITE :       8.60 M/s
Actual DISK READ:       0.00 B/s | Actual DISK WRITE:      14.76 M/s
  TID  PRIO  USER     DISK READ  DISK WRITE  SWAPIN     IO>    COMMAND
  234 be/4 root        0.00 B/s    0.00 B/s  0.00 % 99.99 % [kworker/u2:3]
 2745 be/4 root        0.00 B/s    4.30 M/s  0.00 % 97.79 % sftp-server
 2738 be/4 root        0.00 B/s    4.30 M/s  0.00 % 97.58 % sftp-server
 2719 be/4 root        0.00 B/s    0.00 B/s  0.00 %  0.06 % [kworker/0:3]
```

图 9-3 正在产生磁盘 I/O 的进程

请分析 iotop 命令下各字段的含义，并将结果写入表 9-2。

表 9-2 iotop 命令含义

（3）iftop

①使用 yum 安装所需依赖包

```
# yum install flex byacc libpcap ncurses ncurses-devel libpcap-devel
```

②访问 iftop 官网，下载 iftop 安装包

```
http://www.ex-parrot.com/~pdw/iftop/download/iftop-0.17.tar.gz
```

③使用 FileZilla 上传 iftop 安装包至/home 目录

④使用 yum 安装 gcc-c++

```
#yum install gcc-c++
```

⑤解压 iftop-0.17.tar.gz，安装 iftop

```
#tar -xvzf /home/iftop-0.17.tar.gz
#cd iftop-0.17
#./configure
#make && make install
```

⑥使用 iftop 查看接口流量

```
# iftop -i enp0s3
```

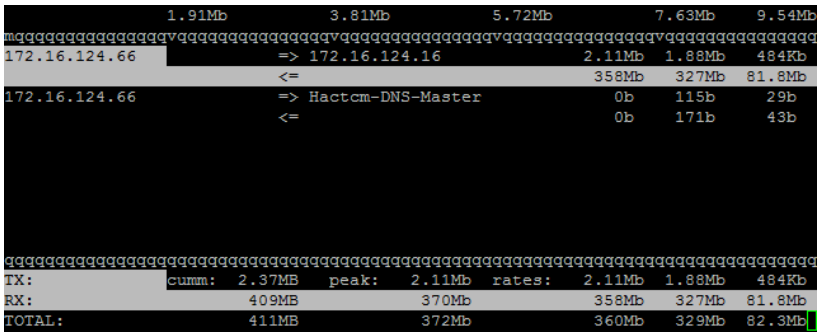

图 9-4 enp0s3 接口流量

（4）sar

①使用 yum 安装 sar

```
#yum install sysstat
```

②使用 sar 查看 CPU 使用信息，每隔 1 秒，共输出 6 次

```
#sar -u 1 6
```

③使用 sar 查看 CPU 所有核心运行数据，每隔 1 秒，共输出 6 次

```
#sar -P ALL 1 6
```

④使用 sar 每隔 1 秒记录 CPU 的使用情况，直到 11 点 02 分，数据将保存到/opt/cpu.log 文件中

```
#sar 1  -u -e 11:02:00 > /opt/cpu.log
```

记录当前 CPU 的使用情况，将结果写入表 9-3，并分析各字段含义。

表 9-3 cpu 使用情况

⑤使用 sar 每隔 1 秒记录内存的使用情况，直到 11 点 02 分，数据将保存到/opt/memory.log 文件中

```
#sar 1  -r -e 11:02:00 > /opt/memory.log
```

记录当前内存的使用情况，将结果写入表 9-4，并分析各字段含义。

表 9-4 内存使用情况

⑥使用 sar 每隔 1 秒记录网络的使用情况，直到 11 点 02 分，数据将保存到/opt/memory.log 文件中。

```
#sar 1 -n DEV -e 11:02:00 > /opt/network.log
```

记录当前网络使用情况，并将结果写入表 9-5，并分析各字段含义。

表 9-5 网络使用情况

2、使用 Shell 编写脚本进行系统维护

（1）编写 Shell 脚本，查看系统用户登录日志

Centos 中记录登录信息的日志文件如下表所示。

表 1-1 登录信息日志一览表

登录日志类型	日志存储位置
当前正在登录系统的用户信息	/var/run/utmp
当前正在登录和历史登录系统的用户信息	/var/log/wtmp
最后一次登录的用户信息	/var/log/btmp

wtmp 和 utmp 文件是二进制文件，可使用 who、w、users、last 和 ac 来查看两个文件包含的信息。

编写 Shell 脚本，将用户登录历史信息导入到文本中。

```
#!/bin/bash
logPath="/var/log/wtmp"
savePath="/home"
\who ${logPath} >> ${savePath}/"userLogin.txt
```

（2）编写 Shell 脚本，实现指定目录的定时备份

①日志获取

将/var/log/下的系统日志备份到临时目录，将该目录进行日志压缩和上传。

```
#!/bin/bash
##定义日志存放路径
path="/var/log"
zipPath="/home"
##创建临时文件夹
time=`date +%Y%m%d`
mkdir ${zipPath}/${time}-"log"
\cp -r ${path}/* ${zipPath}/${time}-"log"/
```

②日志压缩

将临时目录中日志进行压缩，压缩后删除临时目录文件，文件压缩使用“zip”命令。

①使用 yum 安装 zip。

```
#yum install zip
```

②临时目录压缩

```
zip -r ${time}-"log.zip" ${zipPath}/${time}-"log"/
##删除原有的目录文件
rm -rf ${zipPath}/${time}-"log"/
```

③使用 yum 安装 ftp

```
#yum install ftp
```

④上传压缩文件

```
##将当天的日志文件进行上传
ftp -n <<- EOF
open FTP 服务器 IP 地址
user User Password
bin
if ![ -f /${time} ] ;then
mkdir /${time}
fi
cd ${time}
put ${time}-"log.zip"
bye
EOF
```

⑤本地保存

```
##将文件移到本地文件夹下进行保存
mv ${time}-"log.zip" ${path}/log
##判断本地 LOG 日志文件总个数，大于 15 时自动删除之前的文件
```

```
cd ${path}/log/
FileNum=$(ls -l | grep ^- | wc -l)
ReservedNum=15
while(( ${FileNum} > ${ReservedNum} ))
do
    ##取最旧的文件，*.*可以改为指定文件类型
    OldFile=$(ls -rt *.* | head -1)
    rm -f ${path}/log/${OldFile}
    let "FileNum--"
done
```

⑥日志清空

```
##清除本地存放日志的初始日志
rm -rf ${path}/data/query/*
##创建目录，目录名称可自定义设置
mkdir –p ${path}/data/query
##创建日志文件，日志文件名称可自定义设置
touch ${path}/data/query/名称.log
##将新创建的日志文件赋予权限
chown -R named ${path}/data/
chgrp -R named ${path}/data/#取最旧的文件，*.*可以改为指定文件类型
    OldFile=$(ls -rt *.* | head -1)
    rm -f ${path}/log/${OldFile}
    let "FileNum--"
done
```

⑦定时任务

通过 Linux 下定时任务定期执行 shell 脚本，以实现日志每天定时（每天晚上 23.59）保存备份，将定时任务写入/etc/crontab 文件下，其操作命令如下所示。

```
# vi /etc/crontab
59      23      *       *       *       root      /bin/bash 脚本存放路径/file.sh
```

将完整的 Shell 脚本写入表 9-6 中。

表 9-6 网络使用情况

（3）编写 Shell 脚本，实现网站创建

①Apache 安装

使用 yum 命令安装 Apache 服务，并设置服务自动开机启动，其安装命令如下。

```
#yum install httpd
#systemctl start httpd
#systemctl enable httpd
```

②配置虚拟目录并引用

本次实验需发布 10000 个网站，每个网站需创建一个配置文件，将所有网站配置文件放入虚拟目录中，其操作命令如下。

```
# mkdir /etc/httpd/conf.d/vhost
```

创建虚拟目录完成后，需要在 Apache 的主配置文件中进行引用才能实现对每个网站的配置文件加载，其操作方法是在配置文件/etc/httpd/conf/httpd.conf 末尾添加如下命令。

```
IncludeOptional conf.d/vhost/*.conf
```

③安全配置

关闭系统 SELinux 安全访问控制，本次实验通过 Alias 网站虚拟目录方法实现 10000 个网站的创建，默认使用 80 端口，所以修改防火墙配置运行 TCP/80 通过防火墙，其操作命令如下。

```
# vi /etc/selinux/config
  ##将 SELINUX=enforcing 改为 SELINUX=disabled
  SELINUX=disabled
##配置 SELinux 文件后，需重启操作系统
# reboot
##重启完成后修改防火墙规则
# firewall-cmd --zone=public --add-port=80/tcp --permanent
# firewall-cmd --reload
```

③编写 Shell 脚本

●编写 Shell 脚本，使用循环方法，创建 10000 个网站目录，并将每个网站下创建 1 个 index.html 首页，并将展示出“这是 shell 生成的第 XX 个网页”内容，其操作命令如下。

```
htmlPath="/var/www/html"
\rm -rf ${htmlPath}/*
for i in `seq 1 10000`
do
 websiteName="website"$i
 \cd ${htmlPath}
 \mkdir ${websiteName}
 \chmod -R 777 ${websiteName}
 echo "" > ${htmlPath}/${websiteName}/index.html
 echo "<html><head><title>shell 生成 HTML 文件</title></head><bo
dy>" >> ${htmlPath}/${websiteName}/index.html
 echo "<h1>这是 shell 生成的第$i 个网页</h1>" >> ${htmlPath}/${web
siteName}/index.html
 echo "</body></html>" >> ${htmlPath}/${websiteName}/index.html
done
```

●编写 Shell 脚本，使用循环方法，为 10000 个网站创建虚拟配置文件，每个网站创建虚拟目录，并将配置文件放入在/etc/httpd/conf.d/vhost 虚拟目录中，其操作命令如下。

```
httpdConfdPath="/etc/httpd/conf.d/vhost"
\rm -rf ${httpdConfdPath}/*
for j in `seq 1 10000`
do
  websiteConfName="websiteConf${portNum}".conf
  echo "Alias /website$j ${htmlPath}/website$j" >> ${httpdConfdPath}/${websiteConfName}
  echo "<Directory \"${htmlPath}/website$j\">" >> ${httpdConfdPath}/${websiteConfName}
  echo "AllowOverride All" >> ${httpdConfdPath}/${websiteConfName}
  echo "Require all granted" >> ${httpdConfdPath}/${websiteConfName}
  echo "</Directory>" >> ${httpdConfdPath}/${websiteConfName}
done
```

④服务重启

网站和配置文件生成后，需重启 httpd 服务，其操作命令如下。

```
# systemctl restart httpd
```

⑤测试访问

在本地浏览器中可输入网站地址：http://IP 地址/website9999（访问为第 9999 个网站，虚拟目录名称为 website+第 N 个网站），验证网站是否创建成功，并将结果填写至表 9-7 中。

表 9-7 网站访问测试

七、实验扩展

1、系统监控

（1）除了上述列举的监控命令，还有哪些监控命令？其主要功能是什么？

（2）常用的系统监控软件有哪些？并简要说出其主要特点。

（3）除了编写 Shell 脚本进行系统运维外，还有哪些系统自动化运维工具？并简要说出其主要特点。

2、Shell

（1）Shell 是否有数据类型？能否可将字符型数据转换为整型数据？

（2）Shell 中常见的数据结构有哪些？

（3）Shell 脚本是否可以编译、封装、加密？

（4）Shell 脚本程序如何进行知识产权保护？

3、自动化运维

（1）假如有 1000 台 Linux 服务器，如何进行高效率的系统升级？

（2）假如有 1000 台 Linux 服务器，数万个网站服务，如何进行高效率的网站数据备份？