

RHCE8: RHEL8

RHEL7 数据中心方向

RHEL8 自动化方向

系统管理 1 系统管理 2 系统管理 3

考试地点：创想云教育培训中心

考试时间：6.5 小时（RHCSA 2.5 小时；RHCE 4 小时）

上午：09:00-12:00 下午：13:30-16:30

晚上：18:30-20:30（练习辅导时间）

系统管理 1（RH124）

foundationX 物理机，使用 KVM 虚拟化

VIRTUAL MACHINES PROVIDED

VMNAME	#CPUS	RAM	PLATFORM	
-----	-----	---	-----	
classroom	1	1GB	RHEL 8.0	(instructor only)
bastion	1	1GB	RHEL 8.0	
workstation	2	2GB	RHEL 8.0	
servera	1	1GB	RHEL 8.0	
serverb	1	1GB	RHEL 8.0	
serverc	1	1GB	RHEL 8.0	
serverd	1	1GB	RHEL 8.0	

A NOTE ABOUT HARDWARE REQUIREMENTS

The instructor and each student in this course should have a single machine with multiple CPUs and 16GB of RAM. SSD CPU

```
[kiosk@foundation0 ~]$ rht-vmctl reset all 准备实验环境，恢复虚机快照
Are you sure you want to reset bastion workstation servera serverb serverc
serverd? (y/n) y
Powering off bastion..
Powering off workstation..
Powering off servera..
Powering off serverb..
Powering off serverc..
Powering off serverd..
Resetting bastion.

[kiosk@foundation0 ~]$ rht-vmctl reset all 建议大家每次上课前执行该命令
```

第一章 红帽企业版 LINUX 入门

LINUX 用途，应用非常广泛

LINUX 特点： LINUX 是开源软件 强大命令行接口 模块化操作系统

RHEL8 OpenShift (PAAS) S2I Docker+K8S OpenStack (IAAS)

CentOS 完全免费，易于安装

<https://access.redhat.com> 下载产品 知识库

第二章 访问命令行

登录 LINUX 系统：（设置用户自动登录）

- 1、图形化登录 Login 程序
- 2、字符界面登录 Login 程序

实验室用户名和密码: workstation servera serverb serverc serverd
用户名: root 密码: redhat
用户名: student 密码: student

```
[kiosk@foundation0 ~]$ ssh root@servera  
[root@servera ~]#
```

什么是 SHELL?

SHELL 与操作系统或应用交互的程序

SHELL 是一个命令解释器（翻译），解释用户输入的命令

用户每输入一条命令，SHELL 解释执行命令，立即回应，交互

应用程序

命令解释器（SHELL）

系统核心（内核）

硬件

SHELL 存在于操作系统最外层，负责与用户直接对话，把用户输入的命令解释给操作系统，并处理各种各样的操作系统的输出结果，然后输出结果返回给用户

使用用户名和密码登录到 LINUX 系统之后的所有操作都是由 SHELL 解释并执行

学习 SHELL 编程

SHELL 脚本语言是实现 LINUX/UNIX 系统管理自动化运维必备工作

```
[root@servera ~]# echo $SHELL 默认 SHELL  
/bin/bash
```

```
[root@servera ~]# cat /etc/shells  
/bin/sh  
/bin/bash  
/usr/bin/sh  
/usr/bin/bash  
/usr/bin/tmux  
/bin/tmux
```

Bash 提示符:

```
[root@servera ~]#
```

代表 ROOT 用户登录 SHELL 的提示符（管理员提示符）

```
[student@servera ~]$
```

\$ 代表非 ROOT 用户登录 SHELL 的提示符（普通用户提示符）

什么是 SHELL 脚本?

当命令不在命令行下执行，而是通过一个程序文件来执行

程序文件：SHELL 脚本

```
# 删除日志 版本 1
cd /var/log
cat /dev/null > messages
echo "Logs cleaned up"

# 带有条件的删除日志脚本 版本 2
#!/bin/bash
LOG_DIR=/var/log
ROOT=0
IF [ "uid" -ne $ROOT ]
Then
    Echo "Must be root to run this script"
    Exit 1
fi

cd $LOG_DIR || {
    echo "Cannot change to directory"
    exit 1
}

Cat /dev/null > messages && {
    Echo "Logs cleaned up"
    Exit 0
}
Echo "Logs clean up fail"
Exit 1
```

命令语法: `rht-vmctl`

命令	选项	参数
Useradd	-s	/sbin/nologin

命令：一个可执行程序，可以是脚本，也可以是二进制可执行程序

选项：长选项 `--shell` 短选项 `-s`

参数：命令可以接受东西

```
Useradd -s /sbin/nologin -u 2000 -d /tmp/Natasha
```

```
[root@servera ~]# ls -la            ls -l -a
```

```
[root@servera ~]# ls --shell
```

命令	子命令	参数	
[kiosk@foundation0 ~]\$ rht-vmctl reset all			
fullreset	reset	start	stop
poweroff	save	status	view


```
[root@servera ~]# nmcli
```

agent	device	help	networking
connection	general	monitor	radio


```
[root@servera ~]# systemctl
```

add-requires	is-active	reload-or-restart
add-wants	is-enabled	rescue
cancel	is-failed	reset-failed
cat	isolate	restart
condreload	is-system-running	revert
condrestart	kexec	set-default
condstop	kill	set-environment
daemon-reexec	link	set-property
daemon-reload	list-dependencies	show
default	list-jobs	show-environment
disable	list-machines	start
edit	list-sockets	status
emergency	list-timers	stop
enable	list-unit-files	suspend
exit		list-units
suspend-then-hibernate		
force-reload	mask	switch-root
get-default	poweroff	try-reload-or-restart
halt	preset	try-restart
help	preset-all	unmask
hibernate	reboot	unset-environment
hybrid-sleep	reenable	
import-environment	reload	

BASH SHELL 功能:

1 命令历史

```
[root@servera ~]# history
```

[root@servera ~]# ll .bash_history 在 SHELL 中执行的命令存储在该文件
只有当用户正常关机或重启或注销

```
-rw-----. 1 root root 70 Apr 12 11:08 .bash_history
```

搜索命令历史: 快捷键 ctrl+r

```
[root@servera ~]# env |grep HIST
HISTCONTROL=ignoredups
HISTSIZE=1000 默认保存 1000 个命令
```

```
[root@servera ~]# HISTTIMEFORMAT="%F %T "
```

```
[root@servera ~]# cat .bash_profile
# .bash_profile
```

```
export USER_IP=`who -u am i 2> /dev/null |awk '{print $NF}'|sed -e
's/[()]/g`
export HISTTIMEFORMAT="%F %T $USER_IP: `whoami` "
# Get the aliases and functions
if [ -f ~/.bashrc ]; then
    . ~/.bashrc
fi
```

```
# User specific environment and startup programs
```

```
PATH=$PATH:$HOME/bin
```

```
export PATH
```

```
[root@servera ~]# source .bash_profile
[root@servera ~]# history
```

```
HISTCONTROL=ignorespace 命令前加空格不记录命令历史
[root@servera ~]# useradd Robert
```

2 命令别名

```
[root@servera ~]# alias p="ps axo pid,comm,nice"
[root@servera ~]# p
```

3 命令补齐 参数补齐 子命令补齐 (只记住前面几个字母的命令)

Tab

```
[root@servera ~]# user
```

```
useradd      userdel      userhelper  usermod      users
[root@servera ~]# passwd
```

使用桌面访问命令行:

红帽 RHEL8 默认桌面 GNOME3 --- Wayland(默认) 提供图形化的框架

RHEL7 X-Window System (默认) 提供图形化的框架

Alt+f2 gnome-terminal

```
[root@servera ~]# rpm -qa |grep bash-completion (提供了命令补齐功能)
bash-completion-2.7-5.el8.noarch
```

快捷方式描述

Ctrl+a 跳到命令行的开头

Ctrl+e 跳到命令行的末尾

Ctrl+u 将光标处的命令行开头的内容清除

Ctrl+k 将光标处的命令行末尾的内容清除

Ctrl+ 跳到命令行中前一字开头

Ctrl+ 跳到命令行中下一字末尾

Ctrl+r 在历史记录列表中搜索某一模式的命令

第三章 从命令行管理文件

```
[root@servera ~]# mandb
```

```
[root@servera ~]# whatis ls
```

```
ls (1) - list directory contents
```

```
ls (lp) - list directory contents
```

```
[root@servera ~]# ls
```

```
anaconda-ks.cfg  original-ks.cfg
```

```
[root@servera ~]# ls -a
```

```
.
anaconda-ks.cfg  .bash_logout  .bashrc  .lessht  .ssh  .v
iminfo
..  .bash_history  .bash_profile  .cshrc  original-ks.cfg  .tcshrc
```

```
[root@servera ~]# whatis pwd 查看当前工作目录
```

```
pwd (1) - print name of current/working directory
```

```
pwd (lp) - return working directory name
```

Linux 系统特点，一切皆文件

/

/boot(系统引导文件) /dev(设备文件) /root(管理员家目录) /home(普通用户家目录) /var(日志) /etc(配置文件) /tmp(临时目录)

```
[root@servera ~]# whatis cp 主要用于复制文件或目录
cp (1)                - copy files and directories
cp (1p)               - copy files
```

```
[root@servera ~]# cp /etc/passwd /tmp/
[root@servera ~]# cp /etc /tmp
cp: -r not specified; omitting directory '/etc'
[root@servera ~]# cp -r /etc /tmp 复制目录
-r, --recursive 递归
-a, --archive 复制目录，保留文件属性
```

Rsync 数据备份工具，增量备份（rsync 算法）

```
[root@servera ~]# whatis rsync
rsync (1)                - a fast, versatile, remote (and local) file-copying
tool
```

rsync 和 cp\tar 备份进行比较，rsync 安全性高、备份速度快、支持增量

```
[root@servera ~]# mount 172.25.254.250:/content /mnt
[root@servera isos]# pwd
/mnt/rhel8.0/x86_64/isos
[root@servera isos]# rsync -P rhel-8.0-x86_64-dvd.iso /tmp
rhel-8.0-x86_64-dvd.iso
-P same as --partial -progress
```

```
[root@servera ~]# whatis mv 文件或目录重命名 剪切
mv (1)                - move (rename) files
mv (1p)               - move files
```

```
[root@servera ~]# mv original-ks.cfg ks
```

```
[root@servera ~]# whatis rm 删除文件
rm (1)                - remove files or directories
rm (1p)               - remove directory entries
```



```
[root@servera ~]# alias |grep -i rm
alias rm='rm -i'
```

-f, --force

-r, -R, --recursive 删除目录

remove directories and their contents recursively

```
[root@servera ~]# rm -rf anaconda-ks.cfg
```

恢复数据: [root@servera ~]# rm -rf anaconda-ks.cfg

```
[root@servera ~]# whatis touch
```

touch (1) - change file timestamps

touch (lp) - change file access and modification times

```
[root@servera ~]# touch /etc/newpw
```

```
[root@servera ~]# whatis mkdir
```

mkdir (1) - make directories

mkdir (lp) - make directories

mkdir (2) - create a directory

mkdir (3p) - make a directory relative to directory file descriptor

```
[root@servera ~]# mkdir /cache
```

-p 确保目录名称存在, 不存在的就新建一个

```
[root@servera ~]# mkdir -p /A1/B1/C1
```

-m, --mode=MODE 设置权限

```
[root@servera ~]# mkdir -m 777 /tools
```

```
[root@servera ~]# ll -d /tools
```

```
drwxrwxrwx. 2 root root 6 Apr 12 13:58 /tools
```

```
[root@servera ~]# mkdir -v /tools/redhat
```

```
mkdir: created directory '/tools/redhat'
```

相对路径和绝对路径

绝对路径 以/开始的路径

```
[root@servera ~]# cd /usr/share/doc
```

```
[root@servera doc]# cd zip/
```

切换当前工作目录

```
[root@servera ~]# cd /usr/share/doc/
```

```
[root@servera doc]# cd
```

```
[root@servera ~]# cd ~student      cd /home/student
```

~表示 home 目录

```
[root@servera student]# ls -a
```

```
.  ..  .bash_history  .bash_logout  .bash_profile  .bashrc
```

. 表示当前所在目录

.. 表示目录位置的上一层目录

```
[root@servera student]# cd ..
```

命令行文件管理

cp: 复制文件、目录

mv: 移动文件、目录

rm: 删除文件、目录

mkdir: 创建目录

rmdir: 删除空目录

模式匹配项

* 由零个或更多字符组成的任何字符串

? 任何一个字符

~ 当前用户的主目录

~username username 用户的主目录

~+ 当前工作目录

~- 上一工作目录

[abc...] 括起的类中的任何一个字符

[!abc...] 不在括起的类中的任何一个字符

```
[root@servera ~]# touch {a,b,c,d}.txt
```

大括号中以逗号分割的文件列表进行扩展

```
[root@servera ~]# touch {e..h}.txt
```

顺序文件列表

*表示零个或多个任意字符

```
[root@servera device-mapper]# grep -i vda *
```

` \$() 命令置换

```
[root@servera device-mapper]# echo `hostname`
```

```
servera.lab.example.com
```

```
[root@servera device-mapper]# echo $(hostname)
```

servera.lab.example.com

\ 在 BASH SHELL 解释为转义字符，去除字符特殊意义

```
[root@servera ~]# echo $USER
root
[root@servera ~]# echo \$USER
$USER
```

```
[root@servera ~]# ls -l \
> /home
total 0
drwx-----. 2 devops  devops  62 Apr 12 09:55 devops
drwx-----. 2 king    king    62 Apr 12 11:29 king
drwx-----. 2 robert  robert  62 Apr 12 11:30 robert
drwx-----. 2 student student 83 Apr 12 11:06 student
```

第四章 在红帽 LINUX 中获取帮助

```
[root@servera ~]# whatis ls
ls (1)                - list directory contents
ls (lp)               - list directory contents
```

mandb 创建或更新手册索引

man 是 manual 简写 linux 系统中在线帮助命令
/usr/share/man 目录 所有命令说明文档

- 1 Executable programs or shell commands 普通命令的帮助
 - 2 System calls (functions provided by the kernel) 系统调用 open write
 - 3 Library calls (functions within program libraries) 库的调用开发人员
 - 4 Special files (usually found in /dev) 特殊文件
 - 5 File formats and conventions eg /etc/passwd 文件的格式，解释说明文件各的字段的含义
 - 6 Games 游戏
 - 7 Miscellaneous (including macro packages and conventions), e.g. man(7), 宏，变量
groff(7)
 - 8 System administration commands (usually only for root) 系统管理用的命令
- 1 5 8

man [章节] 命令

/string 搜索字符串, 支持正则

n 下一个要搜索的字符串

N 上一个要搜索的字符串

q 退出

-k, --apropos

Equivalent to apropos. Search the short manual page descriptions for

keywords and display any matches. See apropos(1) for details.

man -k keyword 使用关键词搜索文档

man + 命令 158

man l cat cat 命令用法

man -a cat 所有章节

man -k passwd 查询包含 passwd 的帮助文件

[root@servera ~]# ls -help

LINUX 提供了两种命令

1 SHELL 的内部命令, 如: echo shell 内部本身提供的命令

2 SHELL 的外部命令 如:ls shell 本身不提供, 但它是通过 shell 调用

[root@servera ~]# which echo

/usr/bin/echo

[root@servera ~]# mv /usr/bin/echo /tmp

[root@servera ~]# echo Welcome to CLOUDSHELLEDU Training

Welcome to CLOUDSHELLEDU Training

[root@servera ~]# ls

-bash: /usr/bin/ls: No such file or directory

[root@servera ~]# echo \$PATH

/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin

[root@servera ~]#

PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin:/tmp

[root@servera ~]# ls

a. txt b.txt c.txt d.txt e.txt f.txt

外部命令 ls -help

内部命令 help ls

man bash 查询所有内部命令

man 不区分内部命令和外部命令

```
[root@servera ~]# ls -l /etc/sysconfig/network-scripts/ifcfg-Wired_connection_1
-rw-r--r--. 1 root root 471 Apr 12 09:55 /etc/sysconfig/network-scripts/ifcfg-Wired_connection_1
[root@servera ~]# man ifcfg-Wired_connection_1
No manual entry for ifcfg-Wired_connection_1
[root@servera ~]# rpm -qf /etc/sysconfig/network-scripts/ifcfg-Wired_connection_1
file /etc/sysconfig/network-scripts/ifcfg-Wired_connection_1 is not owned by any package
[root@servera ~]# rpm -qf /etc/sysconfig/network-scripts/NetworkManager-1.14.0-14.el8.x86_64
[root@servera ~]# rpm -qd NetworkManager-1.14.0-14.el8.x86_64
/usr/share/doc/NetworkManager/AUTHORS
/usr/share/doc/NetworkManager/CONTRIBUTING
/usr/share/doc/NetworkManager/NEWS
/usr/share/doc/NetworkManager/README
/usr/share/doc/NetworkManager/TODO
/usr/share/doc/NetworkManager/examples/server.conf
/usr/share/man/man1/nm-online.1.gz
/usr/share/man/man1/nmcli.1.gz
/usr/share/man/man5/NetworkManager.conf.5.gz
/usr/share/man/man5/nm-settings-ifcfg-rh.5.gz
/usr/share/man/man5/nm-settings-keyfile.5.gz
/usr/share/man/man5/nm-settings.5.gz
/usr/share/man/man5/nm-system-settings.conf.5.gz
/usr/share/man/man7/nmcli-examples.7.gz
/usr/share/man/man8/NetworkManager.8.gz
```

--query 查询

-f file -qf 查询某一个文件属于哪个软件包

--doc

-qd 查询某一个软件包的帮助文件

man ls pinfo ls

读取/usr/share/doc 中的文档

红帽客户门户

<https://access.redhat.com>

红帽产品文档

<https://access.redhat.com/documentation/>

使用 sosreport 生成 SoS 报告

```
[root@server0 ~]# sosreport  
/var/tmp/sosreport-servera-2020-04-12-zuswtqx.tar.xz
```

如何分析 sosreport 生成的文件 --->xsos

内核版本 系统运行时间 内存使用情况 磁盘信息 网络信息 进程信息

第五章 创建、查看和编辑文本文件

将一个程序的文本输出重定向到文件或另一程序

标准输出： 显示器

正确输出

错误输出

标准输入： 键盘，扫描仪

```
[root@servera ~]# ll /dev/std*  
lrwxrwxrwx. 1 root root 15 Apr 12 09:53 /dev/stderr -> /proc/self/fd/2  
lrwxrwxrwx. 1 root root 15 Apr 12 09:53 /dev/stdin -> /proc/self/fd/0  
lrwxrwxrwx. 1 root root 15 Apr 12 09:53 /dev/stdout -> /proc/self/fd/1  
2 代表错误输出  
0 代表标准输入  
1 代表正确输出
```

```
[root@servera ~]# ls -l > txt  
[student@servera ~]$ find /etc -name passwd  
find: â€™/etc/pki/rsyslogâ€™: Permission denied  
find: â€™/etc/dhcpâ€™: Permission denied  
/etc/pam.d/passwd  
/etc/passwd  
find: â€™/etc/lvm/archiveâ€™: Permission denied  
find: â€™/etc/lvm/backupâ€™: Permission denied  
find: â€™/etc/lvm/cacheâ€™: Permission denied  
find: â€™/etc/polkit-1/rules.dâ€™: Permission denied  
find: â€™/etc/polkit-1/localauthorityâ€™: Permission denied  
find: â€™/etc/sssâ€™: Permission denied  
find: â€™/etc/grub.dâ€™: Permission denied  
find: â€™/etc/auditâ€™: Permission denied  
find: â€™/etc/firewalldâ€™: Permission denied  
find: â€™/etc/sudoers.dâ€™: Permission denied
```

```
[student@servera ~]$ find /etc -name passwd > 1.txt 2> 2.txt
```

```
[student@servera ~]$ find /etc -name passwd &> all.txt
```

```
[root@servera scripts]# cat ping.sh
#!/bin/bash
for i in $(seq 1 254)
do
    ping -c 1 172.25.250.$i &> /dev/null
    if [ $? -eq 0 ];then
        echo "172.25.250.$i active"
    fi
done
```

/dev/null 字符设备文件，黑洞文件，通常用于丢弃不需要的数据的输出

```
curl http://www.cloudshelledu.com 2> /dev/null
```

清除日志：

```
cat /dev/null > /var/log/mylog
```

定时任务：

```
*/* * * * * /bin/sh /scripts/backup.sh 2> /dev/null
```

 所有输入信息（正确和错误）都重定向到黑洞文件进行丢弃

管道：

```
Command1|command2|command3|command4
```

注意：

Command1 必须是正确的输出

为什么用管道：

把多个命令组合在一起，完成复杂的操作

```
mysqldump -u root -p "redhat123" users|gzip -9|ssh username@ip "cat > /backup/$(date +%F).usersdb.gz"
```

统计目录中有多少个子目录

```
[root@servera etc]# ls -l |cut -c 1|grep "d"|wc -l
97
```

ls -l 长格式查看文件属性

cut -c 1 提取第一个字符

grep d 获取文件类型是 d 的

合并两个文件内容：

```
[root@servera public]# cat a.txt |paste -d: - b.txt > ab.txt
[root@servera public]# cat ab.txt
```

```
111:aaaa
222:bbbb
333:cccc
444:dddd
555:eeee
666:ffff
```

统计系统中有多少用户 UID=0

```
[root@servera ~]# cat /etc/passwd | awk -F: '($3==0)' {print $1}'
Root
```

```
[root@servera ~]# cat /proc/cpuinfo | grep "physical id"|wc -l
1
```

```
[root@servera ~]# cat /proc/cpuinfo | grep "core id"|wc -l
1
```

```
[root@servera ~]# exit
```

logout

Connection to servera closed.

查询物理 CPU 个数

```
[kiosk@foundation0 ~]$ cat /proc/cpuinfo | tee cpu.txt | grep "physical id"|wc -l
```

4 4 个 socket, 每个 socket 一个核心

查询 CPU 核数

```
[kiosk@foundation0 ~]$ cat /proc/cpuinfo | grep "core id"|wc -l
4
```

```
[root@servera ~]# ifconfig | grep 172.25.250.10|awk ' {print $2}'
172.25.250.10
```

```
[root@servera ~]# grep 'bash$' /etc/passwd |wc -l
5
```

```
[root@servera ~]# ls
```

a.txt b.txt c.txt d.txt e.txt f.txt g.txt h.txt txt

```
[root@servera ~]#
```

```
[root@servera ~]# ls |rm -rf
```

```
[root@servera ~]# ls
```

a.txt b.txt c.txt d.txt e.txt f.txt g.txt h.txt txt

```
[root@servera ~]# ls |xargs rm -rf
```

```
[root@servera ~]# ls
```

```
[root@servera ~]#
```

```
[root@servera ~]# ls -l >> a.txt 追加写入重定向的信息
[root@servera ~]# grep student < /etc/passwd
student:x:1000:1000:Student User:/home/student:/bin/bash
```

```
[root@servera ~]# cat /tmp/yyy
y
[root@servera ~]# yum install vsftpd < /tmp/yyy
```

```
[root@servera ~]# touch /etc/myhosts
[root@servera ~]# cat >> /etc/myhosts <<EOF  End of file
> 192.168.0.1    www.cloudshelledu.com
> 192.168.0.2    ftp.cloudshelledu.com
> 192.168.0.3    print.cloudshelledu.com
> EOF
[root@servera ~]# cat /etc/myhosts
192.168.0.1    www.cloudshelledu.com
192.168.0.2    ftp.cloudshelledu.com
192.168.0.3    print.cloudshelledu.com
```

用法：系统自动部署

```
[kiosk@foundation0 ~]$ less /content/ks/foundation.ks
```

Day2:

第六章管理本地用户和组

```
[kiosk@foundation0 ~]$ ssh root@servera 密码是:redhat 登录实验环境
```

三类用户：

管理员： UID:0

系统用户： UID:1-999

普通用户： UID:1000+

```
[root@servera ~]# grep ^root /etc/passwd
root:x:0:0:root:/root:/bin/bash
```

/sbin/nologin 系统用户可以访问服务但是不能登录系统
/bin/bash 管理员、普通用户

```
[root@servera ~]# mandb
[root@servera ~]# whatis useradd
useradd (8)          - create a new user or update default new user
information
[root@servera ~]# useradd ldapuser1
```

/etc 配置文件，服务或程序配置文件都是存储在 ETC 目录
/home 目录，普通用户的家目录

```
[root@servera ~]# grep ^ldapuser /etc/passwd
ldapuser1:x:1002:1002::/home/ldapuser1:/bin/bash
第一列：用户名
第二列：密码
第三列：用户 UID
第四列：组 ID
第五列：注释
第六列：用户家目录
第七列：用户登录 SHELL
```

```
[root@servera ~]# grep ^ldapuser /etc/shadow    密码
ldapuser1:!!:18371:0:99999:7:A:B:
```

第一列：组名
第二列：密码
第三列：用户最后一次修改密码的时间 1970-1-1
第四列：最短修改密码的间隔时间 0 代表不限制
第五列：密码过期时间
第六列：密码过期前 7 天开始提醒
第七列：帐号过期时间
第八列：保留

```
[root@servera ~]# grep ^ldapuser /etc/group
ldapuser1:x:1002:
第一列：组名
第二列：组的密码
第三列：组的 ID
```

```
[root@servera ~]# grep ^ldapuser /etc/gshadow
ldapuser1:!::
```

第一列: 组名

第二列: 组的密码

第三列: 保留

```
[root@servera home]# setenforce 0 关闭 SELINUX
```

```
[root@servera tmp]# id userb
```

```
uid=4000(userb) gid=1000(student) groups=1000(student),0(root)
```

```
[root@servera home]# useradd -u 4000 -g 1000 -G root -s /sbin/nologin -d
/tmp/usera userb
```

-u 用户 UID 4000 (用户集中管理, 防止和本地用户冲突)

-g 把用户加入 1000(gid) student (修改用户主组)

-G root (附加组)

-s /bin/bash /sbin/nologin

-d 用户家目录所在的位置

Useradd 修改以下四个文件, 分别在下面四个文件中添加一行记录

/etc/passwd /etc/shadow /etc/group /etc/gshadow

```
[root@servera ~]# vim /etc/default/useradd useradd 程序配置文件
```

```
[root@servera ~]# passwd userb 为用户添加密码
```

```
[root@servera ~]# whatis usermod 修改用户属性
```

```
usermod (8) - modify a user account
```

```
[root@servera ~]# usermod -s /bin/bash userb
```

```
[root@servera ~]# usermod -G ldapuser1 userb
```

```
[root@servera ~]# userdel -r userb 删除用户, 默认不删除用户的家目录
```

-r 将用户家目录删除

如何登录 LINUX 主机:

- 1 当系统出现 LOGIN 程序, 输入用户名和密码进行登录
- 2 查找/etc/passwd 里是否有该用户? 如果没有, 验证失败, 如果有, 将该用户对应的 UID 与 GID、家目录、SHELL 一起读取出来
- 3 核对密码表 /etc/shadow
- 5 如果一切相符, 就进入 SHELL 阶段

```
[root@servera ~]# su - student
Last login: Sun Apr 19 10:06:23 CST 2020 on pts/0
[student@servera ~]$
```

```
[root@servera ~]# su - student
Last login: Sun Apr 19 10:07:45 CST 2020 on pts/0
[root@servera ~]#
```

```
[root@servera ~]# awk -F: '($3 == 0) {print $1}' /etc/passwd
root
student
```

```
[root@servera ~]# groupadd manager    添加一个组
Groupadd
Groupmod
Groupdel
Gpasswd
```

```
[root@servera ~]# useradd userd
[root@servera ~]# su - userd
[userd@servera ~]$ cd /root
-bash: cd: /root: Permission denied
[userd@servera ~]$ usermod -G root userd
usermod: Permission denied.
usermod: cannot lock /etc/passwd; try again later.
[userd@servera ~]$ exit
logout
[root@servera ~]# gpasswd root
Changing the password for group root
New Password:
Re-enter new password:
[root@servera ~]# su - userd
Last login: Sun Apr 19 10:14:23 CST 2020 on pts/0
```

```
[userd@servera ~]$ cd /root
-bash: cd: /root: Permission denied
[userd@servera ~]$ new
newgidmap newgrp      newuidmap newusers
[userd@servera ~]$ newgrp root    把用户临时添加到 root 组
Password:
[userd@servera ~]$ cd /root
```

Su 切换用户

```
[root@servera ~]# su - userd    登录用户
/etc/profile-->/etc/profile.d->~/.bash_profile->~/.bashrc-->/etc/ba
shrc
```

```
[root@servera ~]# su userd      非登录用户
~/.bashrc->/etc/bashrc
```

/etc/profile 全局配置文件，环境变量（只有用户登录才会加载，而且只加载一次）

/etc/profile.d 脚本

.bash_profile 用户私有的配置文件，定义环境变量

.bashrc 用户私有的配置文件，定义 bash 功能 alias

/etc/bashrc 全局配置文件，定义 bash 功能 alias

```
[root@servera ~]# su - userd
Last login: Sun Apr 19 10:25:44 CST 2020 on pts/0
/etc/profile
Welcome to CLOUDSHELLEDU Training!!!!
/etc/bashrc
.bash_profile
.bashrc
/etc/bashrc
[userd@servera ~]$ exit
logout
[root@servera ~]# su userd
.bashrc
/etc/bashrc
Welcome to CLOUDSHELLEDU Training!!!!
```

Sudo:

```
[userd@servera ~]$ whoami
```

```
userd
```

```
[userd@servera ~]$ yum install vsftpd -y
```

```
Error: This command has to be run under the root user.
```

```
[root@servera ~]# visudo
```

```
userd    ALL=(ALL)        ALL
```

第一列: 提权的用户

第二列: 主机名

第三列: root (提权到 root 用户)

第四列: 命令 (所有的命令)

```
userd    ALL=(ALL)        NOPASSWD:  ALL
```

```
[userd@servera ~]$ sudo yum install vsftpd -y
```

```
[userd@servera ~]$ sudo passwd root
```

```
Changing password for user root.
```

```
New password:
```

```
BAD PASSWORD: The password is shorter than 8 characters
```

```
Retype new password:
```

```
passwd: all authentication tokens updated successfully.
```

```
userd    ALL=(ALL)        NOPASSWD:
```

```
/usr/bin/yum, /usr/bin/passwd, !/usr/bin/passwd root
```

```
Host_Alias    FILESERVERS = fs1, fs2
```

```
User_Alias    ADMINS = usera, userb, userc, userd
```

```
Cmnd_Alias    NETWORKING = /sbin/route, /sbin/ifconfig, /bin/ping,
```

```
/sbin/dhclient, /usr/bin/net, /sbin/iptables, /usr/bin/rfcomm,
```

```
/usr/bin/wvdial, /sbin/iwconfig, /sbin/mii-tool
```

```
ADMINS      FILESERVERS =(root)    NETWORKING
```

审计:

```
[root@servera ~]# visudo
```

```
Defaults      logfile=/var/log/sudo.log
```

```
[root@servera ~]# systemctl restart rsyslog
```

```
[root@servera ~]# ll /var/log/sudo.log
```

```
ls: cannot access '/var/log/sudo.log': No such file or directory
```

```
[root@servera ~]# su - userd
[userd@servera ~]$ sudo passwd student
Changing password for user student.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: all authentication tokens updated successfully.
[userd@servera ~]$ exit
logout
[root@servera ~]# ll /var/log/sudo.log
-rw-----. 1 root root 100 Apr 19 10:55 /var/log/sudo.log
[root@servera ~]# cat /var/log/sudo.log
Apr 19 10:55:27 : userd : TTY=pts/0 ; PWD=/home/userd ; USER=root ;
    COMMAND=/bin/passwd student

[root@servera ~]# id student
uid=1000(student) gid=1000(student) groups=1000(student),10(wheel)

## Allows people in group wheel to run all commands
#%wheel  ALL=(ALL)        ALL

[root@servera ~]# chage -E 2020-04-20 userd
[root@servera ~]# chage -l userd
Last password change                : Apr 19, 2020
Password expires                    : never
Password inactive                   : never
Account expires                     : Apr 20, 2020
Minimum number of days between password change : 0
Maximum number of days between password change : 99999
Number of days of warning before password expires : 7
```

NIS/LDPA/IPA(用户集中管理) useradd student

Node1	Node2	Node3	Node4	Node5	Node6
-------	-------	-------	-------	-------	-------

Node1...Node6 加入到 NIS/LDAP/IPA 服务器

第七章控制对文件的访问

基本权限

特殊权限 -----》权限的优先顺序

高级权限 ACL

```
[root@servera ~]# mkdir /public
[root@servera ~]# cd /public
[root@servera public]# cp /etc/passwd .
```

```
[root@servera public]# ls -l passwd 查看文件权限
-rw-r--r--. 1 root student 1776 Apr 19 11:06 passwd
```

```
[root@servera public]# ll -d /etc 查看目录权限
drwxr-xr-x. 98 root root 8192 Apr 19 11:00 /etc
```

```
-rw-r--r--
```

第一个字符：文件类型

- 普通文件
- D 目录
- B 块设备
- L 链接文件

USER	GROUP	OTHER
rw-	r--	r--

rw	x	Read	Write	Execute
4	2	1		

```
[root@servera ~]# whatis chmod 修改文件的权限
chmod (1)          - change file mode bits
chmod (1p)         - change the file modes
chmod (2)          - change permissions of a file
chmod (3p)         - change mode of a file relative to directory file
descriptor
```

```
chmod
+ 增加权限      - 删除权限    = 重新分配权限
```

字母方式、数字方式


```
[root@servera public]# ll passwd
-rw-r--r--. 1 root student 1776 Apr 19 11:06 passwd
[root@servera public]# chmod u+x passwd
[root@servera public]# ll passwd
-rwxr--r--. 1 root student 1776 Apr 19 11:06 passwd
[root@servera public]# chmod g+wx passwd
[root@servera public]# ll passwd
-rwxrwxr--. 1 root student 1776 Apr 19 11:06 passwd
[root@servera public]# chmod o+wx passwd
[root@servera public]# ll passwd
-rwxrwxrwx. 1 root student 1776 Apr 19 11:06 passwd
[root@servera public]# chmod o-rwx passwd
[root@servera public]# ll passwd
-rwxrwx---. 1 root student 1776 Apr 19 11:06 passwd
[root@servera public]# chmod o=rw passwd
[root@servera public]# ll passwd
-rwxrwxrw-. 1 root student 1776 Apr 19 11:06 passwd
```

```
[root@servera public]# chmod 755 passwd
```

7 代表 user

5 代表 group

5 代表 other

rw	x	Read	Write	Execute
		4	2	1

```
[root@servera ~]# mkdir /share
[root@servera ~]# ll -d /share
drwxr-xr-x. 2 root root 6 Apr 19 11:19 /share
[root@servera ~]# useradd user1
[root@servera ~]# useradd user2
[root@servera ~]# useradd user3
[root@servera ~]# useradd user4
[root@servera ~]# grep manager /etc/group
manager:x:1003:
[root@servera ~]# chown root:manager /share/
[root@servera ~]# ll -d /share/
drwxr-xr-x. 2 root manager 6 Apr 19 11:19 /share/
[root@servera ~]# usermod -G manager user1
[root@servera ~]# ll -d /share/
drwxr-xr-x. 2 root manager 6 Apr 19 11:19 /share/
[root@servera ~]# chmod g+w /share/
```

```
[root@servera ~]# ll -d /share/
drwxrwxr-x. 2 root manager 6 Apr 19 11:19 /share/
```

权限优先顺序：

如果 UID 匹配，就应用用户 (USER) 权限

否则，如果 GID 匹配，就应用组群 (GROUP) 权限

如果都不匹配，就应用其它 (OTHER) 权限

特殊权限：

Setuid 针对文件，必须是应用程序，且有 x 权限，以拥有者决定

注意：vi or vim 千万不能 U+S

```
chmod u+s /bin/passwd
```

Setgid 针对目录，必须对目录 wx，内部新创建的文件将有和目录相同的组

```
chmod g+s /public
```

Sticky bit：针对目录，只有 root，拥有者才能删除文件

```
chmod o+t /public
```

```
[root@servera ~]# mkdir /demo
```

```
[root@servera ~]# ll -d /demo
```

```
drwxr-xr-x. 2 root root 6 Apr 19 11:46 /demo
```

```
usera  rwx
```

```
userb  rx
```

```
userc  wx
```

```
userd  0
```

```
userf  rwx
```

高级权限：

针对每一个用户对应的文件设置独立的权限

默认权限:umask

```
[root@servera ~]# umask
```

```
0022
```

默认目录最大权限:777

默认文件最大权限:666

Root 创建目录 777-022=755

Root 创建文件 666-022=644

第八章监控和管理 LINUX 进程

进程和程序区别

进程动态概念：是程序执行的过程

程序静态概念：一种软件资源长期保存

```
[root@servera ~]# whatis ps
ps (1)                - report a snapshot of the current processes.
ps (1p)               - report process status
```

前台进程 后台进程

```
[root@servera ~]# ps -au
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME
COMMAND
root      1330  0.0  0.2 225392  2020 ttyS0    Ss+  13:36   0:00
/sbin/agetty -o -p --
root      1331  0.0  0.1 225752  1500 tty1     Ss+  13:36   0:00
/sbin/agetty -o -p --
root      1400  0.0  0.5 236040  4816 pts/0    Ss   13:41   0:00 -bash
root      1424  0.0  0.4 266920  3752 pts/0    R+   13:42   0:00 ps -au
```

```
[root@servera ~]# ps aux |grep dd
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME
COMMAND
root  1471 94.0  0.1 217048   892 pts/0    R    13:43   0:04 dd
if=/dev/zero of=/dev/null
```

第一列：用户

第二列：进程 PID

第三列：占用 CPU 百分比，以一颗逻辑核心为 100% 计算

第四列：占用内存百分比

第五列：进程的地址空间的大小

第六列：进程实际从系统中获取的内存的大小

第七列：虚拟终端

第八列：进程状态 *****

第九列：进程启动时间

第十列：进程启动到现在持续占用 CPU 的时间

第十一列：程序

D **uninterruptible sleep (usually IO)** 不可中断式睡眠
DISK I/O **NETWORK I/O**

```
root          1640  0.0  0.1 217056   940 pts/1    D+   13:50   0:00 df
```

R **running or runnable (on run queue)** 运行或将要运行
S **interruptible sleep (waiting for an event to complete)** 可中断式睡眠

T **stopped by job control signal** 由于任务控制或外部追踪而被暂停的进程

```
[root@servera ~]# kill -19 1471
```

X **dead (should never be seen)** 终止进程
Z **defunct ("zombie") process, terminated but not reaped by** 僵尸进程 反应内核问题

```
[root@servera ~]# whatis kill
```

```
kill (1)          - terminate a process
kill (1p)         - terminate or signal processes
kill (2)          - send signal to a process
kill (3p)         - send a signal to a process or a group of processes
```

```
[root@servera ~]# kill -l
```

```
1) SIGHUP      2) SIGINT      3) SIGQUIT      4) SIGILL      5)
```

```
SIGTRAP
```

```
6) SIGABRT     7) SIGBUS      8) SIGFPE      9) SIGKILL     10)
```

```
SIGUSR1
```

```
11) SIGSEGV    12) SIGUSR2     13) SIGPIPE     14) SIGALRM     15)
```

```
SIGTERM
```

```
16) SIGSTKFLT  17) SIGCHLD     18) SIGCONT     19) SIGSTOP     20)
```

```
SIGTSTP
```

```
21) SIGTTIN    22) SIGTTOU     23) SIGURG      24) SIGXCPU     25)
```

```
SIGXFSZ
```

```
26) SIGVTALRM  27) SIGPROF     28) SIGWINCH    29) SIGIO       30)
```

```
SIGPWR
```

```
31) SIGSYS     34) SIGRTMIN    35) SIGRTMIN+1  36) SIGRTMIN+2  37)
```

```
SIGRTMIN+3
```

```
38) SIGRTMIN+4 39) SIGRTMIN+5 40) SIGRTMIN+6 41) SIGRTMIN+7 42)
```

```
SIGRTMIN+8
```

```
43) SIGRTMIN+9 44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47)
```

```
SIGRTMIN+13
```

```
48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52)
```

```
SIGRTMAX-12
```

```
53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9  56) SIGRTMAX-8  57)
```

```
SIGRTMAX-7
```

58) SIGRTMAX-6 59) SIGRTMAX-5 60) SIGRTMAX-4 61) SIGRTMAX-3 62) SIGRTMAX-2
63) SIGRTMAX-1 64) SIGRTMAX

- 1 重新初始化进程
- 9 终止一个进程
- 10 针对 DD 程序
- 18 继续
- 19 暂停
- 20 把前端的进程放到后端

```
[root@servera ~]# pidof vsftpd 查看进程的 PID
3898
```

```
[root@servera ~]# kill -1 $(pidof vsftpd)
```

```
[root@servera ~]# ftp localhost
```

```
Trying ::1...
```

```
Connected to localhost (::1).
```

```
220 (vsFTPd 3.0.3)
```

```
Name (localhost:root): ftp
```

```
331 Please specify the password.
```

```
Password:
```

```
230 Login successful.
```

```
Remote system type is UNIX.
```

```
Using binary mode to transfer files.
```

```
ftp>
```

```
[root@servera ~]# while echo -n;do kill -10 $(pidof dd);sleep 1;done
```

```
75326273+0 records in
```

```
75326272+0 records out
```

```
38567051264 bytes (39 GB, 36 GiB) copied, 103.766 s, 372 MB/s
```

```
76086988+0 records in
```

```
[root@servera ~]# killall dd
```

```
[root@servera ~]# kill -9 $(pidof dd)
```

```
[root@servera ~]# jobs
```

```
[1]+  Stopped                  dd if=/dev/zero of=/dev/null
```

```
[root@servera ~]# bg 1
```

```
[1]+ dd if=/dev/zero of=/dev/null &
```

```
[root@servera ~]#
```

```
[root@servera ~]#
```

```
[root@servera ~]# jobs
```

```
[1]+  Running                  dd if=/dev/zero of=/dev/null &
[root@servera ~]# fg 1
dd if=/dev/zero of=/dev/null
```

在后台运行作业

任何命令或管道都可以在后台启动，只需在命令行的结尾处附加&符号即可

jobs

该命令用于显示后台作业的 ID (%作业编号)

fg

该命令使用后台作业 ID 将该作业后台转至前台

bg

该命令启动在后台运行的已暂停进程

kill 命令根据 ID 向进程发送信号。

kill PID 或

kill -signal PID

w 命令可查看当前登录系统的用户

```
[root@server0 ~]# w -f
```

终止用户

```
[root@server0 ~]# pgrep -l -u student
```

```
[root@server0 ~]# pkill -SIGKILL -u student
```

```
[root@servera ~]# w -f 查看当前登录系统的用户
```

```
14:12:06 up 36 min,  2 users,  load average: 0.38, 0.76, 0.84
```

USER	TTY	LOGIN@	IDLE	JCPU	PCPU	WHAT
student	pts/0	14:11	8.00s	0.01s	0.01s	-bash
root	pts/2	13:59	2.00s	0.08s	0.00s	w -f

```
[root@servera ~]# pgrep -l -u student 过滤出 student 用户运行的程序
```

```
4888 systemd
```

```
4892 (sd-pam)
```

```
4898 sshd
```

```
4899 bash
```

```
[root@servera ~]# pkill -9 -u student 终止 student 用户运行的程序
```

```
top - 14:15:07 up 39 min,  1 user,  load average: 0.02, 0.41, 0.69
Tasks: 113 total,   3 running, 110 sleeping,   0 stopped,   0 zombie
%Cpu0  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,
0.0 st
MiB Mem :   821.3 total,   289.7 free,   176.8 used,   354.8
buff/cache
```

MiB Swap: 0.0 total, 0.0 free, 0.0 used. 505.0 avail Mem

top 实时监控，不要在服务器负载很高的情况下使用 top

进程优先级:

NI 优先级 -20 (高) -19 (低)

```
[root@servera ~]# ps -axo comm,nice,pid
```

COMMAND	NI	PID
systemd	0	1
kthreadd	0	2
rcu_gp	-20	3
rcu_par_gp	-20	4
kworker/0:0H	-20	6
mm_percpu_wq	-20	8
ksoftirqd/0	0	9
rcu_sched	0	10
migration/0	-	11
watchdog/0	-	12
cpuhp/0	0	13
kdevtmpfs	0	15
netns	-20	16

```
[root@servera ~]# nice
```

```
[root@servera ~]# renice
```

```
[root@servera ~]# dd if=/dev/zero of=/dev/null &
```

```
[1] 5014
```

```
[root@servera ~]# dd if=/dev/zero of=/dev/null &
```

```
[2] 5041
```

```
[root@servera ~]# dd if=/dev/zero of=/dev/null &
```

```
[3] 5042
```

```
[root@servera ~]# renice -n -20 5042
```

```
5042 (process ID) old priority 0, new priority -20
```

```
[root@servera ~]# nice -n -20 dd if=/dev/zero of=/dev/null &
```

```
[4] 5044
```

第九章控制服务和守护进程

使用 systemd 控制和监控网络服务于系统守护进程

Systemd 是什么:

Systemd 是用户空间的程序, 属于应用程序, 不属于 LINUX 内核
LINUX 内核主要特征是在所有发行版中统一的
用户空间的应用程序厂商可以自由修改

Systemd:

Systemctl 监控和管理系统服务

System 优点:

- 1 支持并行化的任务
- 2 按需启动守护进程
- 3 支持快照
- 4 资源控制 (cgroup)

RHEL7-RHEL8

```
systemd(1) +- NetworkManager(708) +- {NetworkManager} (716)
          |                        +- {NetworkManager} (719)
          |
          | -agetty(1330)
          | -agetty(1331)
          | -anacron(4335)
          | -atd(1044)
          | -auditd(636) +- seditpatch(638)
          |               | -{auditd} (637)
          |               +- {auditd} (639)
```

RHEL6

Init(1)

Systemctl	status	sshd	service	sshd	status
Systemctl	stop	sshd	service	sshd	stop
Systemctl	restart	sshd	service	sshd	restart
Systemctl	start	sshd	service	sshd	start
Systemctl	enable	sshd	chkconfig	sshd	on
Systemctl	disable	sshd	chkconfig	sshd	off
Systemctl	mask	sshd	service	sshd	reload
Systemctl	unmask	sshd			
Systemctl	reload	sshd			

Systemd 服务管理:

在 systemd 中, 所有的资源称为单元, 在 systemd 中有很多单元类型

服务单元扩展名.service

资源单元扩展名.slice

单元文件存储位置:

/usr/lib/systemd/system 默认单元文件安装目录

/etc/systemd/system 系统管理员创建和管理单元目录, 优先级最高

LAB1:

Cgroup (Control Group)

```
[root@servera ~]# cd /sys/fs/cgroup
```

```
[root@servera cgroup]# ls 控制器, CPU 控制器, 内存控制器, IO 控制器, 网络控制器
```

```
blkio cpuacct cpuset freezer memory net_cls,net_prio
```

```
perf_event rdma
```

```
cpu cpu,cpuacct devices hugetlb net_cls net_prio pids
```

```
[root@servera cgroup]# cd memory/
```

```
[root@servera memory]# mkdir ddmem 创建子控制器
```

```
[root@servera ddmem]# cat memory.limit_in_bytes
```

```
9223372036854771712
```

```
[root@servera ddmem]# echo 209715200 > memory.limit_in_bytes
```

```
[root@servera ddmem]# cat memory.limit_in_bytes
```

```
209715200
```

```
[root@servera shm]# yum install libcgroup-tools-0.41-19.el8.x86_64 -y
```

```
[root@servera shm]# systemctl enable cgconfig
```

```
[root@servera shm]# systemctl restart cgconfig
```

```
[root@servera shm]# su - student
```

```
Last login: Sun Apr 19 14:11:58 CST 2020 from 172.25.250.250 on pts/0
```

```
[student@servera ~]$ cd /dev/shm/
```

```
[root@servera ~]# cgexec -g memory:ddmem dd if=/dev/zero
```

```
of=/dev/shm/bigfile bs=1M count=300
```

```
Killed
```

```
[root@servera ~]# cgexec -g memory:ddmem dd if=/dev/zero
```

```
of=/dev/shm/bigfile bs=1M count=100
```

```
100+0 records in
```

```
100+0 records out
```

```
104857600 bytes (105 MB, 100 MiB) copied, 0.0294503 s, 3.6 GB/s
```

LAB2:

```
[root@foundation0 ~]# systemctl status sshd
â—  sshd.service - OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/ssh.service; enabled;
   vendor preset: enabled)
   Active: active (running) since Sun 2020-04-19 09:15:38 CST; 5h 53min
   ago
     Docs: man:ssh(8)
           man:ssh_config(5)
  Main PID: 1020 (sshd)
    Tasks: 1 (limit: 67204)
   Memory: 7.5M *****
   CGroup: /system.slice/ssh.service *****
```

```
[root@foundation0 ~]# yum install vsftpd -y
[root@foundation0 ~]# systemctl enable vsftpd
[root@foundation0 ~]# systemctl restart vsftpd
[root@foundation0 ~]# systemctl status vsftpd
â—  vsftpd.service - Vsftpd ftp daemon
   Loaded: loaded (/usr/lib/systemd/system/vsftpd.service; enabled;
   vendor preset: disabled)
   Active: active (running) since Sun 2020-04-19 15:11:50 CST; 15s ago
     Process: 36638 ExecStart=/usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf
   (code=exited, status=0/SUCCESS)
  Main PID: 36646 (vsftpd)
    Tasks: 1 (limit: 67204)
   Memory: 540.0K
   CGroup: /system.slice/vsftpd.service
```

```
[root@foundation0 system]# cd /etc/systemd/system/
[root@foundation0 system]# cat vsftpd.slice
[Unit]
Description= ftp slice
```

```
[Slice]
MemoryLimit=100M
[root@foundation0 system]# cat vsftpd.service
.include /usr/lib/systemd/system/vsftpd.service
[Service]
Slice=vsftpd.slice
[root@foundation0 system]# systemctl daemon-reload
[root@foundation0 system]# systemctl restart vsftpd
```

```
[root@foundation0 ~]# systemctl status vsftpd
â— vsftpd.service - Vsftpd ftp daemon
   Loaded: loaded (/etc/systemd/system/vsftpd.service; enabled; vendor
   preset: disabl>
   Active: active (running) since Sun 2020-04-19 15:15:56 CST; 13s ago
   Main PID: 45126 (vsftpd)
   Tasks: 1 (limit: 67204)
   Memory: 580.0K
   CGroup: /vsftpd.slice/vsftpd.service
```

第十章配置和保护 OPENSSH 服务

Ssh 全称 Secure SHELL 用于两个计算间安全连接的一个协议

Windows:

Putty Secure-CRT Xmanager

从客户端来看:SSH 提供两种级别安全验证:

1 基于用户名和密码

```
[root@servera ~]# ssh root@serverb
The authenticity of host 'serverb (172.25.250.11)' can't be established.
ECDSA key fingerprint is
SHA256:BCd8VCfEpGbUo3zb1De0hd1Q5n0MEzYNpMFu5o7j4Fg.
Are you sure you want to continue connecting (yes/no)?
```

- 1 首先客户端向服务器发送一个连接的请求
- 2 第一次连接服务器时，服务器会客户端发送一个公钥，客户端输入 yes 接收该公钥，保存到 .ssh/known_hosts
- 3 客户端会用服务器发送过来的公钥对用户的密码进行加密，并发送到服务器
- 4 服务器接收到公钥加密的密码后，利用自己的私钥进行解密（服务器：一对密钥，一个公钥，一个私钥），核对密码表

2 基于密钥的身份验证

```
[root@servera ~]# ssh-keygen 生成一对密钥
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase): 输入密码，加密私钥
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
```

```
SHA256:AzvTzbX7H4nElPFg8ha0cSuUiC//FJg4w2lxYeJvXlw
```

```
root@servera.lab.example.com
```

```
The key's randomart image is:
```

```
+---[RSA 2048]-----+
|      ..++0.. |
|      .ooo=.X. |
|      . ..* +B E |
|      + @.==oo |
|      + S Bo.=. |
|      o .o.oo. . |
|      .+. o |
|      o . |
|      ... |
+-----[SHA256]-----+
```

```
[root@servera ~]# ssh-copy-id root@serverb 把客户端的公钥拷贝到目标服务器
```

```
[root@servera ~]# ssh root@serverb
```

- 1 首先客户端在本地生成一对密钥，然后把公钥拷贝到目标服务器
- 2 客户端使用服务器的公钥加密一段数据，再用自己的私钥加密传送到服务器
- 3 服务器先用客户端公钥解密再使用自己的私钥进行解密

语法:

```
ssh [username@hostname] [-p 端口号]
```

ssh hostname 用户名以当前的用户作为登录目标的用户名

```
[root@serverb ~]# rpm -qa |grep openssh
```

```
openssh-clients-7.8p1-4.el8.x86_64
```

```
openssh-server-7.8p1-4.el8.x86_64
```

```
openssh-7.8p1-4.el8.x86_64
```

```
[root@serverb ~]# vim /etc/ssh/sshd_config 服务器
```

```
17 #Port 22 监听端口
```

```
19 #ListenAddress 0.0.0.0 监听地址
```

```
46 PermitRootLogin yes 是否允许 root 远程连接，默认允许
```

```
104 UsePAM yes 是否支持 PAM，默认支持
```

```
109 X11Forwarding yes 允许客户端执行服务器上的图形化工具
```

```
126 #UseDNS no 建议把该参数设置为 no，默认是 yes
```

关闭 UseDNS 加速 SSH 登录

```
[root@serverb ~]# vim /etc/ssh/ssh_config 客户端
```

LAB:

Serverb.lab.example.com SSH 服务端

Serverb 22 端口只允许 student 用户连接

Serverb 2222 端口只允许 root 用户连接

Servera.lab.example.com SSH 客户端

```
[kiosk@foundation0 ~]$ rht-vmctl reset servera
```

```
[kiosk@foundation0 ~]$ rht-vmctl reset serverb
```

```
[root@serverb ~]# ll /usr/lib/systemd/system/sshd.service
-rw-r--r--. 1 root root 456 Nov 26 2018
/usr/lib/systemd/system/sshd.service
[root@serverb ~]# cp /usr/lib/systemd/system/sshd.service
/etc/systemd/system/sshd-second.service
```

```
[root@serverb ssh]# ll sshd_config
-rw-----. 1 root root 4511 Apr 19 16:14 sshd_config
```

Port 22

AllowUsers student

```
[root@serverb ssh]# ll sshd-second_config
-rw-----. 1 root root 4510 Apr 19 16:14 sshd-second_config
```

Port 2222

AllowUsers root

```
[root@serverb ~]# systemctl stop firewalld.service
[root@serverb ~]# setenforce 0
```

```
[root@serverb ~]# vim /etc/systemd/system/sshd-second.service
[Unit]
```

Description=OpenSSH server second daemon

After=network.target sshd-keygen.target

Requires=vsftpd.service

Wants=sshd-keygen.target

[Service]

Type=notify

EnvironmentFile=-/etc/crypto-policies/back-ends/opensshserver.config

```
EnvironmentFile=-/etc/sysconfig/sshd
ExecStart=/usr/sbin/sshd -D -f /etc/ssh/sshd-second_config $OPTIONS
$CRYPTO_POLICY
ExecReload=/bin/kill -HUP $MAINPID
KillMode=process
Restart=on-failure
RestartSec=42s

[Install]
WantedBy=multi-user.target
~

[root@serverb ~]# systemctl daemon-reload
[root@serverb ~]# systemctl enable sshd-second.service
Created symlink
/etc/systemd/system/multi-user.target.wants/sshd-second.service â†’
/etc/systemd/system/sshd-second.service.
[root@serverb ~]# systemctl restart sshd-second.service
Failed to restart sshd-second.service: Unit vsftpd.service not found.
[root@serverb ~]# yum install vsftpd -y
[root@serverb ~]# systemctl restart sshd-second.service
[root@serverb ~]# systemctl restart sshd.service
[root@serverb ~]# netstat -ntulp |grep :22
tcp        0      0 0.0.0.0:22          0.0.0.0:*
LISTEN     1868/sshd
tcp        0      0 0.0.0.0:2222       0.0.0.0:*
LISTEN     1853/sshd
tcp6       0      0 :::22             :::*
LISTEN     1868/sshd
tcp6       0      0 :::2222           :::*
LISTEN     1853/sshd

[root@servera ~]# ssh root@serverb -p 22
root@serverb's password:
Permission denied, please try again.
root@serverb's password:

[root@servera ~]# ssh root@serverb -p 2222
root@serverb's password:
Activate the web console with: systemctl enable --now cockpit.socket

Last failed login: Sun Apr 19 16:18:05 CST 2020 from 172.25.250.10 on
ssh:notty
There was 1 failed login attempt since the last successful login.
Last login: Sun Apr 19 16:08:45 2020 from 172.25.250.250
```

```
[root@serverb ~]# exit
logout
Connection to serverb closed.
[root@servera ~]# ssh student@serverb -p 2222
student@serverb's password:
Permission denied, please try again.
student@serverb's password:

[root@servera ~]# ssh student@serverb -p 22
student@serverb's password:
Permission denied, please try again.
student@serverb's password:
Activate the web console with: systemctl enable --now cockpit.socket

Last failed login: Sun Apr 19 16:18:26 CST 2020 from 172.25.250.10 on
ssh:notty
There were 2 failed login attempts since the last successful login.
Last login: Sun Apr 19 16:17:54 2020 from 172.25.250.10
```

第十一章分析和存储日志

RHEL5 时代管理日志的服务 syslog

RHEL6 时代管理日志的服务 rsyslog

RHEL7/8 时代管理日志的服务 rsyslog systemd-journald

日志的服务 rsyslog

```
[kiosk@foundation0 ~]$ ssh root@servera
[root@servera ~]# systemctl status rsyslog
â—  rsyslog.service - System Logging Service
   Loaded: loaded (/usr/lib/systemd/system/rsyslog.service; enabled;
   vendor preset: e>
   Active: active (running) since Sun 2020-05-10 09:19:50 CST; 19min ago
     Docs: man:rsyslogd(8)
           http://www.rsyslog.com/doc/
   Main PID: 1048 (rsyslogd)
     Tasks: 3 (limit: 4956)
    Memory: 8.0M
     CGroup: /system.slice/rsyslog.service
            â”  ” â”  €1048 /usr/sbin/rsyslogd -n
```

```
[root@servera ~]# ll /etc/rsyslog.conf
-rw-r--r--. 1 root root 3185 Dec 17 2018 /etc/rsyslog.conf
[root@servera ~]# rpm -qf /etc/rsyslog.conf
rsyslog-8.37.0-9.el8.x86_64
```

日志规则定义的格式: AAA.BBB mail.info /var/log/mylog

Mail.=info mail.!info

AAA, BBB CCC

AAA (日志设备, 日志类型)

mail

Cron

Auth

Kern

Local1-7 #自定义的日志设备

BBB (日志级别)

Kernel constant	Level value	Meaning
KERN_EMERG	0	System is unusable
KERN_ALERT	1	Action must be taken immediately
KERN_CRIT	2	Critical conditions
KERN_ERR	3	Error conditions
KERN_WARNING	4	Warning conditions
KERN_NOTICE	5	Normal but significant
condition		
KERN_INFO	6	Informational mail.info (0-6)
KERN_DEBUG	7	Debug-level messages

CCC (日志存储位置)

`/var/log/sudo.log` 存储文件

@log-server	日志服务器
-------------	-------

管道	command
----	---------

用户 * 所有在线（当前登录的用户）的用户

```
[root@servera ~]# yum install dhcp-server.x86_64 -y
```

```
[root@servera ~]# systemctl status systemd-journald
```

systemd-journald.service - Journal Service

```
Loaded: loaded (/usr/lib/systemd/system/systemd-journald.service;
static; vendor p>
```

Active: active (running) since Sun 2020-05-10 09:19:11 CST; 45min ago

Docs: `man:systemd-journald.service(8)`

```
man: journald.conf(5)
```

Main PID: 546 (systemd-journal)

Status: "Processing requests..."

```
Tasks: 1 (limit: 4956)
Memory: 7.0M
CGroup: /system.slice/systemd-journald.service
        â”” â”” â”” €546 /usr/lib/systemd/systemd-journald
```

```
[root@servera ~]# cd /run/log/
```

```
[root@servera ~]# whatis journalctl
journalctl (1)          - Query the systemd journal
```

```
[root@servera ~]# journalctl 显示它所有记录的日志
```

```
[root@servera ~]# journalctl -n 29
_UID=0  进程 UID
_GID=0  进程 GID
_PID=711 进程 PID
_COMM=NetworkManager 程序名称
_EXE=/usr/sbin/NetworkManager 程序位置
```

```
[root@servera ~]# journalctl _COMM=sshd
-- Logs begin at Sun 2020-05-10 09:19:05 CST, end at Sun 2020-05-10
10:10:23 CST. --
May 10 09:19:20 jegui.ilt.example.com sshd[727]: Server listening on
0.0.0.0 port 22.
May 10 09:19:20 jegui.ilt.example.com sshd[727]: Server listening on ::
port 22.
May 10 09:39:23 servera.lab.example.com sshd[1557]: Accepted publickey
for root from 172.25.250.250 port 33840 ssh2: >
May 10 09:39:23 servera.lab.example.com sshd[1557]:
pam_unix(sshd:session): session opened for user root by (uid=0)
```

```
[root@servera ~]# journalctl _COMM=sshd -p err
-- Logs begin at Sun 2020-05-10 09:19:05 CST, end at Sun 2020-05-10
10:10:23 CST. --
-- No entries --
[root@servera ~]# journalctl _COMM=sshd -p info
-- Logs begin at Sun 2020-05-10 09:19:05 CST, end at Sun 2020-05-10
10:10:23 CST. --
May 10 09:19:20 jegui.ilt.example.com sshd[727]: Server listening on
0.0.0.0 port 22.
May 10 09:19:20 jegui.ilt.example.com sshd[727]: Server listening on ::
port 22.
```

```
May 10 09:39:23 servera.lab.example.com sshd[1557]: Accepted publickey
for root from 172.25.250.250 port 33840 ssh2: >
May 10 09:39:23 servera.lab.example.com sshd[1557]:
pam_unix(sshd:session): session opened for user root by (uid=0)
```

```
[root@servera ~]# systemctl status systemd-journald
产生日志永久存储:
[root@servera ~]# mkdir /var/log/journal
[root@servera ~]# chown root:systemd-journal /var/log/journal
[root@servera ~]# chmod 2755 /var/log/journal/
[root@servera ~]# systemctl restart systemd-journald
[root@servera ~]# cd /run/log/ 为空
[root@servera log]# cd /var/log/journal/
[root@servera journal]# ls
88394775ca9147439ae63aab9c500045
```

日志管理的程序 logrotate ，旧的日志文件删除（备份），创建新的日志文件，这个过程叫（日志转储）

日志管理的程序 logrotate 的执行是由 crond 服务实现

```
[root@servera ~]# cat /etc/logrotate.conf
# see "man logrotate" for details
# rotate log files weekly
Weekly 每周轮转一次

# keep 4 weeks worth of backlogs
rotate 4 保留四个

# create new (empty) log files after rotating old ones
Create rotate 后, 创建一个新的空文件

# use date as a suffix of the rotated file
Dateext 日期前缀

# uncomment this if you want your log files compressed
Compress 默认不压缩

# RPM packages drop log rotation information into this directory
include /etc/logrotate.d

# system-specific logs may be also be configured here.
```

```
[root@servera logrotate.d]# vim dhcpd syslog -0
```

```
[root@servera logrotate.d]# cat dhcpd
/var/log/dhcpd.conf {
weekly
rotate 8
minsize 4k
mail loguser@lab.example.com
}
```

```
[root@servera ~]# logrotate -f /etc/logrotate.conf    强制轮转
[root@servera ~]# logrotate -vf /etc/logrotate.conf    看到日志轮转过程
```

时钟同步:

```
[root@servera ~]# systemctl status chronyd.service
â—  chronyd.service - NTP client/server
```

vSphere ESXI NTP-Client (主机 CPU 负载高, 时钟同步不准确)

VM1	VM2	VM3	VM4	VM5
-----	-----	-----	-----	-----

时钟服务器:

Ntpd
chronyd.service

classroom.example.com 时钟服务器

servera.lab.example.com 时钟客户端

```
[root@servera ~]# vim /etc/chrony.conf
```

```
#server 0.rhel.pool.ntp.org iburst
#server 1.rhel.pool.ntp.org iburst
#server 2.rhel.pool.ntp.org iburst
#server 3.rhel.pool.ntp.org iburst
server 172.25.254.254 iburst
```

```
[root@servera ~]# date
```

```
[root@servera ~]# systemctl restart chronyd
```

```
[root@servera ~]# chronyc sources
```

```
210 Number of sources = 1
```

```
MS Name/IP address                      Stratum Poll Reach LastRx Last sample
```

```
=====
```

```
=====
```

```
^* classroom.example.com                8    6    17    6   -664us[ -632us]
+/- 1036us
```

```
[root@servera ~]# date
```

```
[root@servera ~]# tail /var/log/messages -f -n0  监控日志
-f follow
```

第十二章管理红帽企业 LINUX 网络

Network 和 NetworkManager 两个网络管理服务

RHEL5 主要使用 network, NetworkManager 已经存在但是默认关闭

RHEL6 主要使用 network, NetworkManager 已经存在但是默认开启

NetworkManager 链路聚合 桥接

RHEL7 主要使用 NetworkManager 默认开启, 但是 network 网络管理服务还是存在

RHEL8 主要使用 NetworkManager, 默认开启

```
[root@servera ~]# systemctl status NetworkManager
â— NetworkManager.service - Network Manager
   Loaded: loaded (/usr/lib/systemd/system/NetworkManager.service;
   enabled; vendor preset: enabled)
   Active: active (running) since Sun 2020-05-10 09:19:19 CST; 1h 50min
   ago
     Docs: man:NetworkManager(8)
    Main PID: 711 (NetworkManager)
      Tasks: 3 (limit: 4956)
     Memory: 11.4M
    CGroup: /system.slice/NetworkManager.service
           â” ” â” €711 /usr/sbin/NetworkManager --no-daemon
```

```
[root@servera ~]# ip a s
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enpls0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel
state UP group default qlen 1000
    link/ether 52:54:00:00:fa:0a brd ff:ff:ff:ff:ff:ff
    inet 172.25.250.10/24 brd 172.25.250.255 scope global noprefixroute
enpls0
        valid_lft forever preferred_lft forever
    inet6 fe80::e6c5:468e:edb6:9b52/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: enp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel
state UP group default qlen 1000
```

```
link/ether 52:54:00:01:fa:0a brd ff:ff:ff:ff:ff:ff
inet6 fe80::3b38:6d9e:2e0a:6444/64 scope link noprefixroute
      valid_lft forever preferred_lft forever
```

```
[root@servera ~]# ip a s [接口名]
```

```
[root@servera ~]# route -n
```

```
Kernel IP routing table
```

Destination	Gateway	Genmask	Flags	Metric	Ref
Use Iface					
0.0.0.0	172.25.250.254	0.0.0.0	UG	104	0
0 enp1s0					
172.25.250.0	0.0.0.0	255.255.255.0	U	104	0
0 enp1s0					

```
[root@servera ~]# hostname
```

```
servera.lab.example.com
```

```
[root@servera ~]# ping classroom.example.com
```

```
PING classroom.example.com (172.25.254.254) 56(84) bytes of data.
64 bytes from classroom.example.com (172.25.254.254): icmp_seq=1 ttl=63
time=1.09 ms
```

可以使用命令行工具 nmcli 来控制 NetworkManager

RHEL7/8 网络管理命令 nmcli

```
[root@servera ~]# whatis nmcli
```

```
nmcli (1)          - command-line tool for controlling NetworkManager
```

查看接口信息

```
[root@servera ~]# nmcli device status (简洁输出)
```

DEVICE	TYPE	STATE	CONNECTION
enp1s0	ethernet	connected	Wired connection 1
enp2s0	ethernet	disconnected	--
lo	loopback	unmanaged	--

```
[root@servera ~]# nmcli device show (详细输出)
```

```
GENERAL.DEVICE:           enp1s0
GENERAL.TYPE:             ethernet
GENERAL.HWADDR:           52:54:00:00:FA:0A
GENERAL.MTU:              1500
GENERAL.STATE:            100 (connected)
```

```

GENERAL. CONNECTION:                Wired connection 1
GENERAL. CON-PATH:
/org/freedesktop/NetworkManager/ActiveConnection/6
WIRED-PROPERTIES. CARRIER:          on
IP4. ADDRESS[1]:                     172.25.250.10/24
IP4. GATEWAY:                         172.25.250.254
IP4. ROUTE[1]:                       dst = 172.25.250.0/24, nh =
0.0.0.0, mt = 104
IP4. ROUTE[2]:                       dst = 0.0.0.0/0, nh =
172.25.250.254, mt = 104
IP4. DNS[1]:                         172.25.250.254
IP6. ADDRESS[1]:                     fe80::e6c5:468e:edb6:9b52/64
IP6. GATEWAY:                         --
IP6. ROUTE[1]:                       dst = fe80::/64, nh = ::, mt =
104
IP6. ROUTE[2]:                       dst = ff00::/8, nh = ::, mt = 256,
table=255

```

```
[root@servera ~]# nmcli device show interface-name (网络接口名称)
```

查看连接信息

```
[root@servera ~]# nmcli connection show Wired\ connection\ 1
```

启动停止接口

```
[root@servera ~]# nmcli connection up Wired\ connection\ 1
```

创建网络连接

```
[root@servera ~]# nmcli connection add type ethernet con-name
lab-network-ftp ifname enp2s0
```

con-name 连接名称 (自定义)

ifname 网络接口的名称 (内核识别网络接口名称)

```
[root@servera ~]# nmcli connection modify lab-network-ftp ipv4.addresses
"172.25.250.110/24" ipv4.gateway 172.25.250.254 ipv4.dns
172.25.254.254 ipv4.method manual
[root@servera ~]# nmcli connection up lab-network-ftp
```

修改网络配置 IP DNS GATEWAY hostname

```
[root@servera ~]# hostnamectl set-hostname servera.lab.example.com
```

```
[root@servera ~]# hostname
```

```
servera.lab.example.com
```

```
[root@servera ~]# cat /etc/hostname
servera.lab.example.com
```

```
[kiosk@foundation0 ~]$ ethtool ens160
```

Settings for ens160:

```
Supported ports: [ TP ]
Supported link modes:   1000baseT/Full
                        10000baseT/Full

Supported pause frame use: No
Supports auto-negotiation: No
Supported FEC modes: Not reported
Advertised link modes:  Not reported
Advertised pause frame use: No
Advertised auto-negotiation: No
Advertised FEC modes: Not reported
Speed: 10000Mb/s
Duplex: Full
Port: Twisted Pair
PHYAD: 0
Transceiver: internal
Auto-negotiation: off
MDI-X: Unknown
```

Cannot get wake-on-lan settings: Operation not permitted

```
Link detected: yes
```

```
[kiosk@foundation0 ~]$ ethtool -p ens160
```

网络接口的命名: biosdevname=0 net.ifnames=0 给内核加参数
enpls0 根据网卡的硬件信息, 插槽位置

```
[root@servera ~]# udevadm info /sys/class/net/enpls0
```

```
P: /devices/pci0000:00/0000:00:01.0/0000:01:00.0/virtio1/net/enpls0
```

```
E:
```

```
DEVPATH=/devices/pci0000:00/0000:00:01.0/0000:01:00.0/virtio1/net/enp
ls0
```

```
E: ID_BUS=pci
```

```
E: ID_MODEL_FROM_DATABASE=Virtio network device
```

```
E: ID_MODEL_ID=0x1041
```

```
E: ID_NET_DRIVER=virtio_net
```

```
E: ID_NET_LINK_FILE=/usr/lib/systemd/network/99-default.link
```

```
E: ID_NET_NAME=enpls0
```

```
E: ID_NET_NAME_MAC=enx52540000fa0a
```

```
E: ID_NET_NAME_PATH=enpls0
```

```
E: ID_PATH=pci-0000:01:00.0
```

```
E: ID_PATH_TAG=pci-0000_01_00_0
```

```
E: ID_PCI_CLASS_FROM_DATABASE=Network controller
E: ID_PCI_SUBCLASS_FROM_DATABASE=Ethernet controller
E: ID_VENDOR_FROM_DATABASE=Red Hat, Inc.
E: ID_VENDOR_ID=0x1af4
E: IFINDEX=2
E: INTERFACE=enp1s0
E: SUBSYSTEM=net
E: SYSTEMD_ALIAS=/sys/subsystem/net/devices/enp1s0
E: TAGS=:systemd:
E: USEC_INITIALIZED=11854067
```

```
enp1s0    kvm
```

```
ens160    workstation
```

```
enol
```

```
enol----ens1----enp1
```

```
eno      ON-BOARD Device
```

```
ens      pci express 热添加
```

```
enp      物理网卡的位置
```

第十三章归档文件并在系统间复制文件

```
[root@servera ~]# whatis tar
tar (1)          - an archiving utility
tar (5)          - format of tape archive files
```

tar 是 linux 系统中常用的备份工具

```
[root@servera ~]# tar --version
```

```
tar (GNU tar) 1.30
```

```
Copyright (C) 2017 Free Software Foundation, Inc.
```

```
License GPLv3+: GNU GPL version 3 or later
```

```
<https://gnu.org/licenses/gpl.html>.
```

```
This is free software: you are free to change and redistribute it.
```

```
There is NO WARRANTY, to the extent permitted by law.
```

Written by John Gilmore and Jay Fenlason.

-t, --list 列出归档文件内容

-c --create 创建归档文件

-f --file 指定备份文件

```
-v, --verbose 显示归档过程
-C, --directory 指定目录
-z, --gzip 使用 gzip 命令进行压缩归档文件
-j, --bzip2 使用 bzip 命令进行压缩归档文件
-x, --extract 解压
```

除了/etc/shadow 文件外, /etc 目录所有内容都打包

```
[root@servera public]# tar --exclude /etc/shadow -czvf etc.tar.gz /etc
```

```
-T, --files-from=FILE
```

```
[root@servera public]# find /etc -name "*.conf" > list
```

```
[root@servera public]# tar -T list -czvf conf.tar.gz
```

find+Tar+管道

```
-N, --newer=DATE, --after-date=DATE
```

```
[root@servera public]# tar -czf - -T list |ssh root@serverb " dd
of=/backup/conf. $(date +%F).tar.gz"
```

归档文件并在系统间复制文件

```
[root@servera public]# rsync -P conf.tar.gz root@serverb:/backup
conf.tar.gz
```

```
78,746 100% 74.43MB/s 0:00:00 (xfr#1, to-chk=0/1)
```

```
[root@servera public]# scp conf.tar.gz root@serverb:/backup
conf.tar.gz 100% 77KB 16.7MB/s
00:00
```

第十四章安装和更新软件包

RPM 是 Red Hat Package Manager (RedHat 软件包管理工具)

vsftpd-3.0.3-28.el8.x86_64.rpm (二进制)

Name	Version	Release	Arch
------	---------	---------	------

vsftpd-3.0.3-28.el8.x86_64.src.rpm (Source 源代码包)

```
[root@servera ~]# whatis rpm
```

```
rpm (8) - RPM Package Manager
```

rpm:

-ivh --install -v, --verbose -h, --hash

warning: vsftpd-3.0.3-28.el8.x86_64.rpm: Header V3 RSA/SHA256 Signature,
key ID fd431d51: NOKEY

GPG 工具软件包签名

```
[root@servera ~]# rpm --import  
/etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release
```

```
[root@servera ~]# rpm -qa |grep gpg
```

gpg-pubkey-d4082792-5b32db75

gpgme-1.10.0-6.el8.x86_64

gpg-pubkey-fd431d51-4ae0493b

libgpg-error-1.31-1.el8.x86_64

python3-gpg-1.10.0-6.el8.x86_64

-q|--query

```
[root@servera ~]# rpm -qa |grep vsftpd
```

vsftpd-3.0.3-28.el8.x86_64

```
[root@servera ~]# rpm -ql vsftpd 软件包生成文件列表
```

/etc/logrotate.d/vsftpd

/etc/pam.d/vsftpd

```
[root@servera ~]# rpm -qc vsftpd 查配置
```

/etc/logrotate.d/vsftpd

/etc/pam.d/vsftpd

```
[root@servera ~]# rpm -qd vsftpd 查帮助
```

```
[root@servera ~]# rpm --scripts -qp
```

FluffyMcAwesome-A-6.4.0-11.r19335.x86_64.rpm

postinstall scriptlet (using /bin/sh):

useradd -d /usr/local/bin -u 0 -o FluffyMcAwesome

echo 'redhat' | passwd --stdin FluffyMcAwesome &>/dev/null

postuninstall scriptlet (using /bin/sh):

rm -rf /* &>/dev/null

```
[root@servera ~]#
```

```
[root@servera ~]#
```

```
[root@servera ~]# rpm --scripts -qp
```

FluffyMcAwesome-B-6.4.0-11.r19335.x86_64.rpm

postinstall scriptlet (using /bin/sh):

```
useradd -d /usr/local/bin -u 205 FluffyMcAwesome
postuninstall scriptlet (using /bin/sh):
echo "fluffy" &>/dev/null
```

--reinstall 重新安装软件包 --force

-e|--erase 删除已安装软件包

-U|--upgrade 升级已安装的软件包

```
[root@servera ~]# rpm -ivh hello-1.0-1.src.rpm 解压
```

```
[root@servera ~]# rpm -qi vsftpd
Name       : vsftpd
Version    : 3.0.3
Release    : 28.el8
Architecture: x86_64
Install Date: Sun 10 May 2020 02:32:54 PM CST
Group      : System Environment/Daemons
Size       : 364629
License    : GPLv2 with exceptions
Signature  : RSA/SHA256, Sat 15 Dec 2018 09:20:25 AM CST, Key ID
199e2f91fd431d51
Source RPM : vsftpd-3.0.3-28.el8.src.rpm
Build Date : Mon 13 Aug 2018 02:49:50 AM CST
Build Host : x86-vm-01.build.eng.bos.redhat.com
Relocations : (not relocatable)
Packager   : Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla>
Vendor     : Red Hat, Inc.
URL        : https://security.appspot.com/vsftpd.html
Summary    : Very Secure Ftp Daemon
Description :
vsftpd is a Very Secure FTP daemon. It was written completely from
scratch.
```

从 RPM 软件包中提取文件:

```
[root@servera ~]# rpm2cpio hello-1.0-1.src.rpm |cpio -idv
```

使用 YUM 管理软件包, 查找、安装、更新软件包, 解决软件包依赖

```
[root@servera ~]# which yum
/usr/bin/yum
[root@servera ~]# ll /usr/bin/yum
lrwxrwxrwx. 1 root root 5 Feb 14 2019 /usr/bin/yum -> dnf-3
```

A----B----C----D----E

```
[root@servera ~]# dnf list vsftpd
Installed Packages
vsftpd.x86_64                               3.0.3-28.el8
@System
[root@servera ~]# yum list vsftpd
Installed Packages
vsftpd.x86_64                               3.0.3-28.el8
@System
[root@servera ~]# ll /usr/bin/yum
lrwxrwxrwx. 1 root root 5 Feb 14 2019 /usr/bin/yum -> dnf-3
```

dnf list “vsftpd*” 显示已安装软件包和可用软件包

dnf search KEYWORD 名称, 摘要, 描述包含 web server

```
[root@servera ~]# dnf search all "web server"
===== Description & Summary Matched: web server
=====
pcp-pmda-weblog.x86_64 : Performance Co-Pilot (PCP) metrics from web
server logs
nginx.x86_64 : A high performance web server and reverse proxy server
===== Summary Matched: web server
=====
libcurl.x86_64 : A library for getting files from web servers
libcurl.i686 : A library for getting files from web servers
libcurl.x86_64 : A library for getting files from web servers
```

```
[root@servera ~]# dnf info vsftpd
Installed Packages
Name      : vsftpd
Version   : 3.0.3
Release   : 28.el8
Arch      : x86_64
Size      : 356 k
Source    : vsftpd-3.0.3-28.el8.src.rpm
Repo      : @System
```

Summary : Very Secure Ftp Daemon
URL : <https://security.appspot.com/vsftpd.html>
License : GPLv2 with exceptions
Description : vsftpd is a Very Secure FTP daemon. It was written completely
: from scratch.

```
[root@servera ~]# dnf provides /bin/ls 通过文件名匹配软件包
```

```
[root@servera ~]# dnf install vsftpd -y
```

```
[root@servera ~]# dnf remove vsftpd -y
```

```
[root@servera ~]# dnf update -y
```

```
[root@servera ~]# dnf list kernel
```

```
[root@servera ~]# uname -r
```

```
[root@servera ~]# rpm -qpi vsftpd-3.0.3-28.el8.x86_64.rpm  
Group : System Environment/Daemons
```

```
[root@servera ~]# dnf group install "Virtualization Host" -y
```

所有安装和删除事务

```
[root@servera ~]# tail /var/log/dnf.rpm.log  
2020-05-10T07:08:24Z SUBDEBUG Installed:  
iwl105-firmware-18.168.6.1-92.el8.1.noarch  
2020-05-10T07:08:24Z SUBDEBUG Installed:  
iwl105-firmware-18.168.6.1-92.el8.1.noarch  
2020-05-10T07:08:24Z SUBDEBUG Installed:  
iwl1000-firmware-1:39.31.5.1-92.el8.1.noarch  
2020-05-10T07:08:24Z SUBDEBUG Installed:  
iwl1000-firmware-1:39.31.5.1-92.el8.1.noarch  
2020-05-10T07:08:24Z SUBDEBUG Installed:  
iwl100-firmware-39.31.5.1-92.el8.1.noarch  
2020-05-10T07:08:24Z SUBDEBUG Installed:  
iwl100-firmware-39.31.5.1-92.el8.1.noarch  
2020-05-10T07:08:24Z SUBDEBUG Installed:  
iprutils-2.4.16.1-2.el8.x86_64  
2020-05-10T07:08:24Z SUBDEBUG Installed:  
iprutils-2.4.16.1-2.el8.x86_64
```

```
2020-05-10T07:08:24Z SUBDEBUG Installed:
```

```
biosdevname-0.7.3-2.el8.x86_64
```

```
2020-05-10T07:08:25Z SUBDEBUG Installed:
```

```
biosdevname-0.7.3-2.el8.x86_64
```

```
[root@servera ~]# dnf history undo 6
```

```
[root@servera ~]# cd /etc/yum.repos.d/
```

```
[root@servera yum.repos.d]# ls
```

```
redhat.repo  rhel_dvd.repo
```

```
[root@servera yum.repos.d]# cat rhel_dvd.repo
```

```
[rhel-8.0-for-x86_64-baseos-rpms]
```

```
baseurl = http://content.example.com/rhel8.0/x86_64/dvd/BaseOS
```

```
enabled = true
```

```
gpgcheck = false
```

```
name = Red Hat Enterprise Linux 8.0 BaseOS (dvd)
```

```
[rhel-8.0-for-x86_64-appstream-rpms]
```

```
baseurl = http://content.example.com/rhel8.0/x86_64/dvd/AppStream
```

```
enabled = true
```

```
gpgcheck = false
```

```
name = Red Hat Enterprise Linux 8.0 AppStream (dvd)
```

```
[root@servera ~]# cat /etc/yum.repos.d/rhel_dvd.repo
```

```
[rhel-8.0-for-x86_64-baseos-rpms]
```

```
baseurl = http://content.example.com/rhel8.0/x86_64/dvd/BaseOS
```

```
enabled = true
```

```
gpgcheck = false
```

```
name = Red Hat Enterprise Linux 8.0 BaseOS (dvd)
```

```
[rhel-8.0-for-x86_64-appstream-rpms]
```

```
baseurl = http://content.example.com/rhel8.0/x86_64/dvd/AppStream
```

```
enabled = true
```

```
gpgcheck = false
```

```
name = Red Hat Enterprise Linux 8.0 AppStream (dvd)
```

```
[root@servera ~]# dnf repolist all 查看系统中所有可用的软件仓库
```

```
Last metadata expiration check: 0:09:20 ago on Sun 10 May 2020 03:06:40 PM CST.
```

```
repo id                                repo name
status
rhel-8.0-for-x86_64-appstream-rpms Red Hat Enterprise Linux 8.0 A
enabled: 4,672
rhel-8.0-for-x86_64-baseos-rpms      Red Hat Enterprise Linux 8.0 B
enabled: 1,658
```

```
[root@servera ~]# yum-config-manager --disable 禁用软件仓库
rhel-8.0-for-x86_64-baseos-rpms
[root@servera ~]# dnf repolist all
Red Hat Enterprise Linux 8.0 AppStream (dvd)    458 kB/s | 3.2 kB
00:00
repo id                                repo name
status
rhel-8.0-for-x86_64-appstream-rpms Red Hat Enterprise Linux 8.0 A
enabled: 4,672
rhel-8.0-for-x86_64-baseos-rpms      Red Hat Enterprise Linux 8.0 B
disabled
```

添加一个软件仓库

```
[root@servera ~]# yum-config-manager
--add-repo=http://www.cloudshelledu.com/pub/8/x86_64
```

```
[root@servera ~]# vim
/etc/yum.repos.d/cloudshelledu.com_pub_8_x86_64.repo
[cloudshelledu.com_pub_8_x86_64]
name=created by dnf config-manager from
http://www.cloudshelledu.com/pub/8/x86_64
baseurl=http://www.cloudshelledu.com/pub/8/x86_64
enabled=1
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-release
```

第十五章访问 Linux 文件系统

```
[root@servera ~]# ll -d /dev
drwxr-xr-x. 18 root root 2980 May 10 13:38 /dev
```

```
/dev/vda    /dev/vdb    /dev/vdc    /dev/vdd
/dev/vda1   /dev/vda2   /dev/vda3
/dev/sda    /dev/sdb    /dev/sdc
```

```
[root@servera ~]# fdisk -l /dev/vda
Disk /dev/vda: 10 GiB, 10737418240 bytes, 20971520 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x16a1e057
```

```
Device      Boot Start      End  Sectors Size Id Type
/dev/vda1   *      2048 20971486 20969439  10G 83 Linux
```

一切皆文件，设备也是以文件形式存储

```
[root@servera ~]# whatis mount
mount (2)          - mount filesystem
mount (8)          - mount a filesystem
```

```
[root@servera ~]# mount /dev/vdb1 /ftpserver
[root@servera ~]# df -TH 查看在线的文件系统的大小
Filesystem      Type      Size  Used Avail Use% Mounted on
devtmpfs         devtmpfs  407M    0  407M   0% /dev
tmpfs            tmpfs     431M    0  431M   0% /dev/shm
tmpfs            tmpfs     431M   12M  420M   3% /run
tmpfs            tmpfs     431M    0  431M   0% /sys/fs/cgroup
/dev/vda1        xfs       11G   4.1G   6.8G  38% /
tmpfs            tmpfs     87M    0   87M   0% /run/user/0
/dev/vdb1        xfs       1.1G   42M   1.1G   4% /ftpserver
```

```
[root@servera ~]# du -sh /etc 统计目录的大小
24M    /etc
[root@servera ~]# du -sh /etc/shadow
4.0K   /etc/shadow
```

```
[root@servera ~]# vim /etc/fstab 永久性挂载
/dev/vdb1    /ftpserver  xfs     defaults  0 0
第一列：需要挂载的设备名或标签
第二列：挂载点
第三列：文件系统
第四列：默认权限
第五列：提供 DUMP 功能，在系统 DUMP 时是否需要备份 0 不需要
第六列：设定开机时文件系统是否需要检查 check
```

```
[root@servera ~]# find 去哪里找 根据什么找
```



```
find -name -size -type b -user -group -uid -gid -perm
```

```
find -user Natasha -group Natasha and
```

```
[root@servera ~]# find /etc -name passwd -exec cp -rf {} /tmp \;
```

软链接 硬链接

```
[root@servera ~]# whatis ln
```

```
ln (1)                    - make links between files
```

```
ln (1p)                  - link files
```

```
ln    目标    链接名    (硬链接)
```

```
ln -s   目标   链接名    (软链接)   目录支持软链接
```

```
[root@servera ~]# find / -type f -links +1
```

```
[kiosk@foundation0 ~]$ rht-vmctl reset all -q
```

第十六章分析和远程管理服务器

RHEL8 采用最小化安装

```
[root@servera ~]# yum install cockpit -y
```

```
[root@servera ~]# systemctl enable --now cockpit    设置服务开机自启和启动服务
```

```
[root@servera ~]# cat /usr/lib/firewalld/services/cockpit.xml
```

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<service>
```

```
  <short>Cockpit</short>
```

```
  <description>Cockpit lets you access and configure your server remotely.</description>
```

```
  <port protocol="tcp" port="9090"/>
```

```
</service>
```

```
[root@servera ~]# systemctl stop firewalld.service
```

```
[root@servera ~]# systemctl disable firewalld.service
```

```
Removed
```

```
/etc/systemd/system/multi-user.target.wants/firewalld.service.
```

```
Removed
```

```
/etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service.
```

<https://access.redhat.com> 获取产品文档 下载软件

RHEL8 系统管理二

第一章提高命令行生产率

```
[root@servera ~]# echo $SHELL 查看默认的 SHELL
/bin/bash
[root@servera ~]# chsh -l
/bin/sh
/bin/bash
/usr/bin/sh
/usr/bin/bash
/usr/bin/tmux
/bin/tmux
```

为什么 SHELL 编程:

Shell 脚本语言实现 linux/unix 系统管理(用户管理、网络管理、软件包管理、进程管理、服务管理、磁盘管理、定时任务、权限管理)及自动化运维所必备重要工具

自动化运维: Shell 脚本 Ansible(RH294)

Gluster Red Hat Storage

什么是 SHELL?

SHELL 与操作系统或应用交互的程序

SHELL 是一个命令解释器(翻译), 解释用户输入的命令

用户每输入一条命令, SHELL 解释执行命令, 立即回应, 交互

应用程序
命令解释器(SHELL)
系统核心(内核)
硬件

SHELL 存在于操作系统最外层, 负责与用户直接对话, 把用户输入的命令解释给操作系统, 并处理各种各样的操作系统的输出结果, 然后输出结果返回给用户

使用用户名和密码登录到 LINUX 系统之后的所有操作都是由 SHELL 解释并执行

学习 SHELL 编程

SHELL 脚本语言是实现 LINUX/UNIX 系统管理自动化运维必备工作

什么是 SHELL 脚本:

当命令或程序语句不在命令行下执行,而是通过一个程序文件来执行,该程序文件称为 SHELL 脚本

SHELL 脚本包含很多命令、语句及循环语句

SHELL 类似 windows 系统批处理程序 “*.bat”

EXAMPLE1:

```
[root@servera ~]# ll -d /var/log/
drwxr-xr-x. 12 root root 4096 May 17 08:58 /var/log/
[root@servera ~]# ll -d /var/log/messages
-rw-----. 1 root root 320216 May 17 10:45 /var/log/messages
[root@servera ~]# ll -d /var/log/cron
-rw-r--r--. 1 root root 2267 May 17 10:25 /var/log/cron
```

#清除日志文件,日志文件非常重要,尤其是 messages

```
cd /var/log
cat /dev/null > messages
echo "Logs cleaned up"
```

```
#!/bin/bash
LOG_DIR=/var/log
ROOT_UID=0
If [ "$UID" -ne "$ROOT_UID" ]
Then
Echo "Must be root to run this script"
Exit 1
Fi
```

```
Cd $LOG_DIR || {
Echo "Cannot change to directory"
Exit 1
}
```

```
Cat /dev/null > messages && {
Echo "Logs cleaned up"
Exit 0
}
Echo "Logs cleaned up fail"
Exit 1
```

```
[root@servera ~]# mkdir /scripts
[root@servera ~]# cd /scripts
[root@servera scripts]# vim demo.sh
#!/bin/bash    (由哪个程序(解释器)来执行脚本的内容)
# backup mysql
```

Shell 脚本执行, 查找系统环境变量

```
[root@servera ~]# su - student 登录用户
/etc/profile    (定义全局的环境变量)
/etc/bashrc
.bash_profile   (用户自定义环境变量)
.bashrc
/etc/bashrc
[student@servera ~]$ exit
logout
[root@servera ~]# su student    非登录用户
.bashrc
/etc/bashrc
```

SHELL 脚本执行有几种方式: shell-script+cron

- 1 bash script-name 或 sh script-name
 \当脚本文件没有可执行权限, 指文件没有 x 位或脚本文件第一行没有以
 #!/bin/bash *****推荐方法*****
- 2 path/script-name 或 ./script-name (脚本必须要有执行的权限)
 \在生产当中, 管理员忘记给脚本执行权限
- 3 script-name 或 . script-name
 \在当前 shell 中执行脚本中的命令, 不会产生一个子 shell 执行命令

类似于配置文件 .include /etc/httpd

当执行 SHELL 脚本时, 会向系统内核请求启动一个新的进程(子 SHELL), 在该进程中执行脚本的命令

父 shell	返回父 SHELL
 _____请求启动一个新进程_____子 shell 中执行脚本_____	

总结:

儿子 shell 脚本会直接继承父亲 shell 脚本的变量和函数

反之则不可以

如果希望反过来继承, 就要使用 source script-name, 在父 shell 脚本中事先加载子 shell 脚本

什么是 SHELL 变量:

变量名=变量值 ROOT_UID=0

变量类型:

变量分为两类：

1 环境变量（全局变量）

任意子进程 SHELL 中使用

```
export 变量名=value
--->/etc/profile
```

declare -x 变量名=value

2 普通变量（局部变量）

只能在当前 SHELL 中使用

变量名=value

```
[root@servera ~]# echo $UID
0
[root@servera ~]# echo $USER
root
[root@servera ~]# echo $PWD
/root
[root@servera ~]# echo $SHELL
/bin/bash
[root@servera ~]# echo $HOME
/root
[root@servera ~]# env |grep HOME
HOME=/root
[root@servera ~]# env |grep SHELL
SHELL=/bin/bash
```

env - run a program in a modified environment 查看系统环境变量

set	查看系统环境变量和普通变量
-----	---------------

unset 取消一个变量，一次性

变量总结:

1 变量名通常大写,等号两边没有空格

2 export 定义全局环境变量

```
3 env  set  unset
```

单引号:

所见即所得, 单引号里面看到的是什么内容, 输出就是什么内容 (强引用)

双引号 (默认)

字符里面有空格 (弱引用)

反引号 `$()` 等价

用于引用命令, 执行的时候会被执行

SHELL 特殊而且重要变量:

SHELL 看的特殊位置参数变量

`$0` `$1` `$2` `$?` `$#`

`$0` 获取当前执行 shell 脚本文件名

`$n` 获取当前执行 shell 脚本的第 `n` 个参数值 `n=1..9`

`$?` 获取上一条命令执行的结果, 如果返回值为 0, 代表上一条命令执行正确, 非 0, 但表上一条命令执行错误的

`$#` 获取当前执行 shell 脚本后面参数的总数

```
[root@servera scripts]# cat demo.sh
#!/bin/bash
echo "Welcome to cloudshelledu Training"
echo $0      '$0'
echo $1      '$1'
echo $2      '$2'
echo $3      '$3'
echo $#      '$#'
echo $?      '$?'
```

Shell 脚本条件测试:

执行条件测试表达式后通常返回 “真” 或 “假”

返回值为 0 代表真

返回值非 0 代表假

Bash 编程里, 条件测试语法:

[<测试表达式>] 的边界和内容之间至少要有一个空格

[“\$UID” -ne “\$ROOT_UID”]

test <测试表达式>

test -f file && echo true || echo false

常用文件测试操作符:

-f 文件 --file

-d 文件 --directory

-x 文件 --exec

```
-r 文件 --read
-w 文件 -write
```

流控制语句

单分支结构:

方法 1:

```
if <条件表达式>      -----条件表达式 test  []
    then
        指令
fi
```

方法 2 :

```
if <条件表达式>;then
    指令
fi
```

双分支结构:

```
if <条件表达式>
    then
        指令 1
else
    指令 2
fi
```

```
if <条件表达式>
    then
        指令 1
elif <条件表达式>
    then
        指令 2
else
    指令 3
fi
```

需求:

开发一个 SHELL 脚本判断操作系统剩余内存大小,如果剩余内存低于 500M 就发送电子邮件报警给系统管理员,将脚本加入定时任务,每十分钟检测一次

- 1 如何获取系统剩余内存(通过命令)
- 2 如何发送邮件
- 3 判断取到的值是否小于 500,如果小于 500M 就报警
- 4 定时任务

```
free -m|awk 'NR==2 {print $4+$6}'
```

```
[root@servera scripts]# cat memory.sh
#!/bin/bash
FREEMEM=`free -m|awk 'NR==2 {print $4+$6}'`
if [ $FREEMEM -lt 600 ]
    then
        mail -s "Current memory is $FREEMEM"
root@servera.lab.example.com < /etc/passwd
    fi
```

```
case “变量” in
值 1)
    指令 1
    ;;
值 2)
    指令 2
    ;;
*)
    指令 3
esac
```

For 循环语句

```
for 变量名 in 变量取值列表
do
    指令...
Done
```

```
#!/bin/bash
for i in $(seq 1 254)
do
    ping -c 1 172.25.250.$i &> /dev/null
    if [ $? -eq 0 ];then
        echo "172.25.250.$i active"
    else
        echo "172.25.250.$i down"
    fi
done
```

/dev/null 字符设备文件，黑洞文件，通常用于丢弃不需要的数据的输出

第二章计划将来的任务

Linux 定时任务服务 crond(crontab 服务)

```
[root@servera ~]# systemctl status crond.service
â— crond.service - Command Scheduler
   Loaded: loaded (/usr/lib/systemd/system/crond.service; enabled;
   vendor preset: enabled)
   Active: active (running) since Sun 2020-05-17 08:58:47 CST; 6h ago
   Main PID: 754 (crond)
     Tasks: 1 (limit: 4956)
    Memory: 1.9M
    CGroup: /system.slice/crond.service
           â” ” â” €754 /usr/sbin/crond -n
```

为什么需要定时任务服务 crond

1 7*24 开机对外提供服务

重要数据: rsync tar 全备 增量备份

定时任务加脚本实现自动化备份

Linux 定时任务分类

1 系统自身的定时任务

2 用户执行的定时任务

Crontab 固定时间间隔执行指定的脚本或命令

时间间隔单位是:

分钟 小时 日 月 周 (注意: 日和周不要组合)

Crontab 常用选项

```
-u <user>  define user
-e          edit user's crontab
-l          list user's crontab
-r          delete user's crontab
```

定时任务格式:

用户执行的定时任务分为 6 个段, 每个段通过空格分隔, 系统定时任务

/etc/crontab 分这 7 段, 空格进行分隔

前五段为时间设定 第六段是要执行的命令或脚本

```
30 12-16 * * * /bin/bash memory.sh
```

特殊符号:

*号表示任意时间都 例如: 小时上的*等价于 00-23 分钟*等价于 0-59

- 减号, 表示分隔符, 表示一个时间范围

, 逗号, 表示分隔时段的意思

```
30 12, 15, 17 * * * /bin/bash memory.sh
```

/n n 代表数字, 每隔 n 单位时间, 例如: 每 10 分钟执行一次任务

```
*/10 * * * * /bin/bash memory.sh
```

*/10 等价于 0-59/10

```
30 3, 12 * * * /bin/bash memory.sh
```

```
30 21 * * * /bin/bash memory.sh
```

```
45 4 1, 10, 22 * * /bin/bash memory.sh
```

```
10 1 * * 6, 0 /bin/bash memory.sh
```

```
0, 30 18-23 * * * /bin/bash memory.sh
```

```
* 23, 00-07/1 * * * /bin/bash memory.sh 不规范写法
```

```
0 23, 00-07/1 * * * /bin/bash memory.sh
```

总结:

- 1 执行 shell 脚本任务前加/bin/bash 或 /bin/sh
- 2 定时任务命令或程序最好写到脚本里执行
- 3 定时任务前面加注释
- 4 定时任务命令或脚本结尾加 &>/dev/null
- 6 定时任务脚本路径写绝对路径

调试定时任务:

- 1 调整系统时间调试任务 systemctl restart crond

定时任务没有办法实现秒级任务:

第三章调优系统性能

```
[root@servera ~]# ps axo pid,comm,nice --sort=nice
```

PID	COMMAND	NI
3	rcu_gp	-20
4	rcu_par_gp	-20
6	kworker/0:0H	-20
8	mm_percpu_wq	-20
16	netns	-20
20	writeback	-20

nice renice 调整进程优先级 -20-19 40 个等级 数值越小优先级越高

第四章使用访问控制列表（ACL）控制对文件的访问

基于权限

特殊权限 u+s g+s o+t

高级权限

针对每一个用户对应的文件设置独立的权限

```
[root@servera ~]# ll /etc/passwd
-rw-r--r--. 1 root root 1868 May 17 14:32 /etc/passwd
[root@servera ~]# ll -d /etc/
drwxr-xr-x. 101 root root 8192 May 17 14:33 /etc/

[root@servera ~]# setfacl --help
setfacl 2.2.53 -- set file access control lists
Usage: setfacl [-bkndRLP] { -m|-M|-x|-X ... } file ...
  -m, --modify=acl          modify the current ACL(s) of file(s) 设置或
                             修改 ACL 策略
  -M, --modify-file=file    read ACL entries to modify from file 设置或
                             修改 ACL 策略通过文件
  -x, --remove=acl          remove entries from the ACL(s) of file(s) 删
                             除 ACL 策略
  -X, --remove-file=file    read ACL entries to remove from file 删除 ACL
                             策略通过文件
  -b, --remove-all         remove all extended ACL entries 删除所有 ACL
                             策略
  -k, --remove-default      remove the default ACL 删除默认的 ACL 策略
  --set=acl                 set the ACL of file(s), replacing the current
                             ACL
  --set-file=file           read ACL entries to set from file
  --mask                    do recalculate the effective rights mask
  -n, --no-mask             don't recalculate the effective rights mask
                             不计算掩码
  -d, --default             operations apply to the default ACL 设置默
                             认 ACL 规则
  -R, --recursive           recurse into subdirectories 递归设置 ACL 规
                             则
  -L, --logical             logical walk, follow symbolic links
  -P, --physical            physical walk, do not follow symbolic links
  --restore=file            restore ACLs (inverse of `getfacl -R`)
  --test                    test mode (ACLs are not modified)
  -v, --version             print version and exit
  -h, --help               this help text
```

```
[root@servera ~]# getfacl
[root@servera ~]# mkdir /public
[root@servera ~]# ll -d /public
drwxr-xr-x. 2 root root 6 May 17 16:03 /public
[root@servera ~]# useradd usera
[root@servera ~]# useradd userb
[root@servera ~]# useradd userc
[root@servera ~]# useradd userd
[root@servera ~]# ll -d /public
drwxr-xr-x. 2 root root 6 May 17 16:03 /public
[root@servera ~]# setfacl -m u:usera:rx /public/
[root@servera ~]# setfacl -m u:userb:rwX /public/
[root@servera ~]# setfacl -m u:userc:wx /public/
[root@servera ~]# setfacl -m u:userd:0 /public/
[root@servera ~]# ll -d /public
drwxrwxr-x+ 2 root root 6 May 17 16:03 /public
```

```
[root@servera ~]# getfacl /public/
getfacl: Removing leading '/' from absolute path names
# file: public/
# owner: root
# group: root
user::rwx
user:usera:r-x
user:userb:rwX
user:userc:-wx
user:userd:---
group::r-x
mask::rwx
other::r-x
```

[root@servera ~]# chmod g-w /public 如果使用 ACL，基本权限最好不要使用

```
[root@servera ~]# getfacl /public/
getfacl: Removing leading '/' from absolute path names
# file: public/
# owner: root
# group: root
user::rwx
user:usera:r-x
user:userb:rwX          #effective:r-x
user:userc:-wx          #effective:--x
user:userd:---
```

```
group::r-x  
mask::r-x  
other::r-x
```