# 命令

## \*系统命令

ctrl +alt+f1~f8

切换控制台

echo 文本

打印文本

reboot

重启

cd 路径e

切换所在目录

pwd

查看当前绝对路径

hostname [主机名]

查看/临时修改 主机名

hostnamectl [选项]

查看主机信息

date [选项] [参数] [+格式符]

查看或修改时间

格式符：

%S 秒

%M 分

%H 时

%Y 年

%m 月

%d 日

%F 年-月-日

%T 时:分:秒

(eg：date +'%Y-%m-%d %H:%M:%S')

选项：

-s 修改时间(date -s '2025-3-20 18:25:00')

man+命令

查看命令帮助

按 '/' 查找关键字

按 'q' 退出

info/pinfo +命令

查看更加详细的命令帮助

history [选项]

查看历史命令

选项：

-c 清处历史

-w 将历史记录保存到$HOME/.bash\_history中

read

从标准输入流中读取信息，存入变量中

格式:read [选项] [变量]

选项:

-t :用于表示等待输入的时间，单位为秒，等待时间超过，将继续执行后面的脚本，注意不作为null输入，参数将保留原有的值

-p 输出提示信息

## \*用户与组管理

useradd [选项] [参数] 用户名

添加用户

选项：

-u 指定uid

-g 指定私有组(主组)

-G 指定公共组(从组)

-d 指定家目录

-s 指定登录shell（/bin/login,/sbin/nologin,/bin/false）

passwd [选项] [用户名]

修改用户密码

选项：

-l 锁定用户，无法登录（在/etc/shadow文件该用户密码开头加’!!’）

-u 解锁用户

-S 查看用户状态

echo 密码 | passwd --stdin [用户名]

修改用户密码（接受标准输入）

userdel [选项] 用户名

删除用户

选项:

-r 同时删除文件信息

usermod [选项] [参数] 用户名

修改用户信息

选项:

-c 修改登录页面的描述信息

-u 修改uid

-g 修改gid

-d 修改家目录

-m 移动家目录(若不存在则会创建目录)

-s 修改登录shell

-l 修改用户名

-aG 将用户添加到附加组中

-G 修改附加组

-L/-U 锁定/解锁用户

ls /home 或 cat /etc/passwd

查看存在的用户

who 或 w

查看登录到本地的用户

su - 用户名 [选项] [参数]

切换用户 （登录shell，su 用户名 为非登录shell）

选项:

-s 为用户临时分配一个shell

id [用户名]

查看用户信息 用户id(uid)、组id(gid)等

uid范围:

0 超级管理员

1-999 系统用户

1000-60000 普通用户

(修改/etc/login defs 中的uid改变普通用户数量的上限)

chage 用户名 [选项] [参数]

修改密码期限(单位:天)

选项:

-l 查看用户密码到期信息

-m 最短有效期限

-M 最长有效期限

-W 提前多少天警告

-E 账号失效日期(格式：YYYY-MM-DD)

-d 上次修改密码的时间(为0时该用户下次登录必须强制修改密码)

groupadd [选项] [参数] 组名

添加组

选项：

-g 指定gid

del [选项] [参数] 组名

删除组

newgrp 组名

登录到该组(exit退出)

gpasswd [选项] [参数] 组名

修改组密码/管理组的成员(普通用户也可以使用)

选项:

-d 删除组成员

-a 增加组成员

-A 设置组管理员

groupmems [选项1] [参数1] [选项2] [参数2]

查看/修改组成员(root用户专用)

选项:

-g 指定要操作的组名

-l 列出组成员

-a 添加组成员

-d 删除组成员

-p 清空所有组成员

groupmod [选项] [参数] 组名

修改组属性

选项：

-g 修改gid

-n 修改组名

-p 修改组密码

## \*vim文本工具

i

进入输入视图(esc退出)

:

进入命令视图(可使用操作定界符)

w|q|!

退出|保存|强制(命令视图下)

G+G

跳到文件尾

g+g

跳到文件头

数字

指定行数(配合删除、复制等)

d+d

剪切

y+y

复制

p

粘贴

/

查找

## \*sed流编辑器

按行读取文件中的内容，放入临时缓冲区(模式空间)，在进行处理，然后打印在屏幕上

sed默认处理的是模式空间中的内容，不会修改原文件。

**sed [选项] '地址定界 操作符' 文件名**

选项：

-n 不输出模式空间的内容

-i 对源文件进行操作

-i.bak 先备份源文件再操作

-e 接多个操作

-r 使用拓展正则表达式

地址定界+操作符：

**地址定界：**

数字 ：指定要处理的行

数字1,数字2 ：指定要处理的行区域

$ ：最后一行

.$ ：行的末尾

/关键字/ ：表示关键字所在行，支持正则表达式

数字,/关键字/ ：指定第几行到关键字所在行的区域

/关键字1/,/关键字2/ ：指定关键字1到关键字2所在的区域

\%关键字% ：表示关键字所在行，转义特殊符号，支持正则表达式

数字1~数字2 ：数字1代表起始位置，数字2代表步长

**操作符：**

p 打印

I 忽略大小写

d 删除

a 在后一行追加( a\字段 )

i 在前一行追加

c 修改所在行（ c\字段 ）

s 替换每一行第一个匹配到的关键字

g 匹配所有

示例：

s/字段1/字段2/N 将每一行第N个匹配到的字段1替换为字段2

Ms/字段1/字段2 将第M行第一个匹配到的字段1替换为字段2

s/字段1/字段2/g 将所有的字段1都替换为字段2

s/字段1/字段2/i 将所有的字段1都替换为字段2，忽略大小写

/字段/d 删除字段

/字段/p 打印字段

/字段/a\字段2 在‘字段’后面一行追加‘字段2’

/字段/c\内容\ 将字段所在行替换为‘内容’

s/^/# /g 添加注释

## \*文件操作

ls [选项] [路径]

列出文件

选项:

-l 列出权限，所有人等信息(即ll命令)

-a 包括隐藏目录

-i 查看inode号

-Z 查看标签

nano / vim / vi

文本编辑器

touch [选项] [参数] 文件名

新建空文件,更新时间戳

选项:

-a 改变访问时间

-m 改变内容修改时间

-d 接参数修改为指定的时间(格式:2004-02-29 16:21:42)

stat 文件名

查看文件的时间戳

时间戳:

Access:最后一次被访问的时间

Modify:最后一次内容被修改的时间

Change:最后一次文件属性被修改的时间

Birth:文件创建时间

rm 文件名

删除文件

选项:

-f 强制删除

-r 删除目录

mv 文件 目标路径

移动文件(若路劲指定为文件名则移动的同时修改文件名)

cp [选项] 文件 目标路径

复制文件

选项:

-r 复制目录但是修改时间戳

-a 复制目录且不改变时间戳

mkdir [选项] 目录名

创建目录

选项:

-p 递归创建目录

rmdir 目录名

删除空目录

file 文件名

查看文件类型

硬链接

inode相同，文件名不同，删除所有硬链接才会删除文件，不可跨文件系统创建

软连接

inode不同，文件名也不同，删除原文件会影响软连接，可跨文件系统创建

ln [选项] 文件名 链接名

创建硬连接

选项：

-s 创建软连接

link 文件名 链接名

创建硬链接（无法创建软连接）

chattr [选项] 文件

修改文件特殊属性

选项：

+i 不允许对文件做任何操作

-I 解除文件i属性

+a 只允许对文件追加

-a 解除文件a属性

lsattr 文件

查看文件特殊属性

## \*文本处理

cat [选项] 文件名

查看文件内容

选项：

-A 显示时文本最后添加'$'符号

-n 显示行数

more 文件名

分页显示文件

less 文件名

分页显示文件(类似vim编辑器)

head [选项] [参数] 文件名

显示文件前10行

选项：

-n 显示文件前X行，X等于参数值(参数取-负数时打印除文件尾以外的内容)

tail [选项] [参数] 文件名

显示文件尾10行

选项：

-n 显示文件尾X行，X等于参数值(参数取+正数时打印除文件头以外的内容)

-f 实时监控文件内容(ctrl+c退出)

grep [选项] 关键字 文件名

输出关键字所在行（关键字可为正则表达式 “\<关键字 以关键字开头的单词；关键字\> 以关键字结尾的单词 ”）

选项：

-i 忽略大小写

-o 只输出关键字

-c 输出匹配行的数量

-q 静默执行

-v 反选，输出没有匹配到的行

-AN 输出匹配行以及下N行

-BN 输出匹配行以及上N行

-CN 输出匹配行以及上下N行

-r 查找哪些文件内包含了关键字('文件名'应该为'目录')

-E 使用拓展正则表达式

cut [选项] [参数] 文件名

匹配列

选项：

-d 指定分隔符

-f 指定列数

-c 指定字符(-c 1,3 表示取第一列和第三列的字符)

wc [选项] [参数] 文件名

统计文件行数，单词数，字节数。

选项：

-l 只统计行数

sort [选项] [参数] 文件名

按照字符表排序（升序）

选项：

-n 以数字的方式排序

-t 指定分隔符

-k 指定列数

-r 降序(通常和-n一起使用)

-f 忽略大小写

-u 去重

uniq [选项] [参数] 文件名

去重相邻的重复行

选项：

-c 统计重复的数目

diff 文件名1 文件名2

比较两个文件的不同

vimdiff 文件名1 文件名2

在vim编辑器中比较两个文件的不同

tr [选项] 字符范围1 [字符范围2]

将键盘输入的文本字符进行转换（通常用于大小写转换）

选项：

-d 删除文字当中的字符

字符范围的格式：

'a-z' 'A-Z'

## \*变量

set

查看所有变量

env [选项]

查看系统环境变量

选项:

-i 开始一个新的空的环境

-u 从当前环境中删除指定的变量

export [选项] [变量名]=[值]

定义环境变量

export 变量名：将系统变量转化为环境变量

unset 变量名

取消变量

alias 别名='命令'

为命令取别名(宏)

alias 别名： 查看当前定义的别名

alias ： 查看所有定义的别名

unalias

取消命令的别名

## \*进程管理

ps [选项] [参数]

查看当前终端运行的进程

选项：

-a 查看所有终端的进程

-x 查看不属于终端的进程

-u 查看执行进程的用户，cpu、内存资源占用

-o 指定要查看的列 （格式）

格式：-eo user,pid,nice,comm,stat,%cpu,%mem

-ef 可查看父进程

--sort 对输出排序

格式： --sort=-%cpu降序 --sort=%cpu 升序

pstree [选项]

按树形结构列出父进程和子进程的关系

选项

-p：显示端口

prtstat PID

显示进程的详细信息

top [选项] [参数]

实时监控系统进程

选项：

-d 刷新时间

-p 指定查看 pid

快捷键：

1 查看每个处理器的信息

M 按照内存的百分比排序

P 按照CPU的百分比排序

m 改变内存显示方式

f 进入指定列的显示选择视图(空格为开关)

pgrep [选项] [参数]

查看用户运行的进程（pgrep 字段 ：过滤ps -ef 中进程名中包含字段的一行）

选项：

-U 以用户名查看运行的进程ID

-u 以uid查看运行的进程id

-G 以组名查看运行的进程ID

-g 以gid查看运行的进程ID

-l 查看进程名

-P 以子进程id查看父进程id

pidof 进程名

精确查询进程名对于的pid

kill [选项] [参数] [PID]

选项:

-l 列出可用的信号

-s 发送特定的信号给目标

信号:

2 :通常是Ctrl+C产生的信号，可以被进程捕获或忽略。

9 :立即结束进程，不能被捕获或忽略。

10:检查日志文件是否存在，否则生成新的。

15:正常结束进程，可以被捕获或忽略。

19:暂停进程，不能被捕获、忽略或结束。

18:继续执行被暂停的进程。

killall 进程名

杀死所有进程名对应的进程

w 或 who

查看登录到当前系统的用户

pkill [选项] [参数]

杀死终端对应的进程

选项：

-t 指定终端（TTY格式）

-kill 或 -9 踢出用户(配合-t使用)

-u 指定用户名，踢出该用户

renice [选项] [参数] pid

调整已启动的进程的优先级

选项：

-n 指定优先级(-20~19 以小为优 )

nice [选项] [参数] 命令

以指定的优先级开始命令

选项：

-n 指定优先级

jobs

显示当前终端运行的作业

[1]+ Stopped ping 10.0.0.1

作业号 运行状态 作业名

命令 &

将命令暂停，放入后台

nohup 命令 &

将命令放入后台执行

bg %作业号

将命令放入后台执行

fg %作业号

将命令放入前台运行

Ctrl+z

暂停作业

## \*网络管理

ifconfig [网卡名]

查看已开启的网卡信息

ifconfig 网卡名 up/down

开启/关闭网卡

ifconfig 网卡名 IP地址

修改网卡的ip

ifconfig 网卡名 IP地址 netmask 子网掩码 broadcast 广播地址 up

修改网卡信息

ip route

查看网关

ip routing table / ip route show

查看路由信息

ip addr show / ip a

查看网卡信息

ip route add/del 目标地址/网段 via 网关IP地址 dev 网卡名

添加/删除一条路由(目标地址为default时为默认路由)

cat /etc/resolv.conf

查看DNS服务器

route -n

查看路由表

route add/del -host 目标地址 gw 网关IP地址 dev 网卡名

添加/删除一条主机路由

route add/del -net 目标网段 gw 网关IP地址 dev 网卡名

添加/删除一条网段路由（网段格式：192.168.0.0/16）

nmtui

图形化网络配置

### nmcli

**配置文件**：/etc/NetworkManager/system-connections #rhel9版本

/etc/sysconfig/network-scripts/ifcfg-网卡名 #rhel6、7、8版本

nmcli device connect|disconnected 网卡名

激活/禁用用网卡设备和配置文件的连接

nmcli connection add type ethernet ifname 网卡名 con-name 网卡配置文件名 ipv4.method (auto)/(manual ipv4.address IP地址/网络位 ipv4.dns DNS服务器IP地址 ipv4.gateway 网关地址) [autoconnect yes]

设置网卡的配置文件(autoconnet yes代表开机自动激活)

eg：nmcli connection add type ethernet ifname ens33 con-name ens33 ipv4.method manual ipv4.address 192.168.36.78/24 ipv4.dns 192.168.36.254 ipv4.gateway 192.168.36.254 autoconnect yes

nmcli device connect 网卡名

自动激活网卡并添加配置文件(在网卡没有配置文件时)

nmcli connection modify 配置文件名 connection.interface-name 网卡名

将网卡和已存在的配置文件绑定

nmcli connection modify 配置文件名 +ipv4.address IP地址/掩码

添加网卡配置文件的ip地址

nmcli connection modify 配置文件名 ipv4.address IP地址/掩码

修改网卡配置文件的ip地址

nmcli connection modify 配置文件名 ipv4.method (manual ipv4.addresses IP地址/掩码)/(auto) autoconnect yes

修改IP获取的方式（手动或dhcp）

nmcli connection reload

重新加载网络配置文件

nmcli connection up 配置文件名

激活网卡配置文件

nmcli connection delete 配置文件名

删除配置文件(也可在/etc/NetworkManager/system-connections下删除)

nmcli device status

查看系统存在的网卡类型和状态

nmcli device show [网卡名]

查看所有网卡的完整信息

nmcli connection show

显示网卡配置文件的连接信息

nmcli connection add type bond ifname bond设备名 con-name 配置文件名 mode BOND模式

添加虚拟bond网卡配置文件

nmcli connection add type bond-slave ifname 网卡名 con-name 网卡配置文件名 master bond网卡配置文件名

将网卡绑定到虚拟bond网卡

nmcli connection bond网卡配置文件名 ipv4.method manual ipv4.addresses IP地址/掩码 autoconnect yes

更改bond网卡的IP地址

cat /proc/net/bonding/bond1

查看当前使用的主网卡

netstat -tunlp

查看当前系统正在使用的端口

选项：

-t tcp端口

-u udp端口

-n 以数字的方式列出

-l 列出已经监听的端口

-p 显示pid以及进程名

ss -an

高级版的netstat，还能显示收发数据包的数量

export http\_proxy=http://IP地址:端口

export https\_proxy=http://IP地址:端口

设置网络代理

# 特殊目录或文件

/

根目录

~

家目录

/root/

root用户的家目录

/home/用户名/

普通用户的家目录

/usr/

自定义软件、共享库保存目录

/usr/bin/

保存可执行的命令文件(普通用户)

/usr/sbin/

保存可执行的命令文件(root用户)

/usr/local/

自定义软件目录

/etc/

保存系统配置文件

/etc/NetworkManager/system-connections/

网络管理（networkmanager的配置文件 rhcl9版本）

/etc/sysconfig/network-scrips/

网络管理（network-scrtip的配置文件 rhcl9版本以前，修改配置时一般关闭networkmanager）

/etc/hostname

保存主机名

/etc/hosts

保存IP地址与主机名的对应关系(DNS)

内容：

IP地址 域名1 域名2 ...

IP地址 域名1 域名2 ...

/etc/services

保存常见服务与端口的对应信息

/etc/login.defs

密码默认策略设置

/var/

日志文件，网站、协议的数据文件目录

/temp/

临时文件目录，重启时清除

/boot/

系统内核，引导程序

/dev/

磁盘、U盘、CPU等设备文件目录

/proc/

存放进程运行数据，关机时删除

/dev/null

回收站/黑洞(常在shell脚本使用)

/dev/zero

零字符设备

/etc/passwd

存放用户信息,每一行表示一个用户，信息间以冒号分隔

内容:

root:x:0:0:root:/root:/bin/bash

1 root 用户名

2 x 存在'x'表示登录时需要密码验证(与该用户是否存在密码无关)

3 0 用户uid

4 0 用户组id

5 root 登录页面的描述信息

6 \root 家目录

7 /bin/bash 登录shell(若为/sbin/nologin,则该用户无法登录，只能运行程序)

/etc/shadow

存放用户密码文件

内容:

user:$5$luO50wMTME4:19700:0:99999:7:1::

1 user 用户名

2 $5$luO50wMTME4 密码(若为'!!'或'\*'代表该用户无密码)

$6$ SHA512加密

$5$ SHA216加密

$1$ MD5加密

3 19700 上一次修改密码的时间(1970-01-01到现在的天数)

4 0 密码的最短期限(为0则无期限)

5 99999 密码的最长期限(单位:天)

6 7 密码过期前多少天发送警告

7 1 账号过期多少天被锁定

/etc/group

存放组信息,每一行表示一个组，信息间以冒号分隔

内容:

root:x:0:user1

1 组名

2 x 存在'x'表示登录组时需要密码验证

3 0 gid

4 user1 组附加成员(私有组的私有成员不会显示)

/etc/gshadow

存放组密码文件

内容:

group:$1$yvcZJG:user1:user2

1 group 组名

2 $1$yvcZJG 密码(加密方式与用户密码相同)

3 user1 组管理员

4 user2 组附加成员(以','间隔)

登录shell时会执行的文件(按先后顺序)：

/etc/bashrc

/etc/profile

$HOME/.bashrc

$HOME/.bash\_profile

非登录shell时会触发的文件

/etc/bashrc

$HOME/.bash\_profile

退出shell时会触发的文件

$HOME/.bash\_logout

/etc/resolv.conf

DNS服务器保存文件

/etc/at.deny

at命令权限的黑名单

/etc/at.allow

at命令权限的白名单（需手动创建，权限大于黑名单）

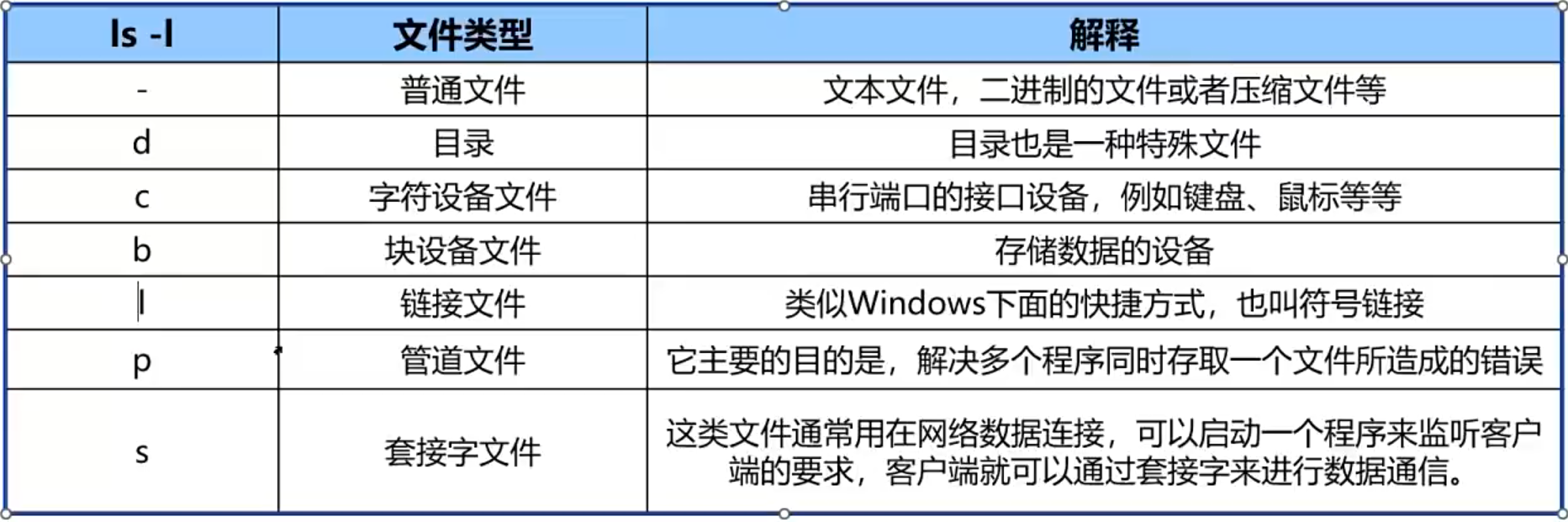
# 权限

## 文件目录权限

ll命令显示信息：

drwxr-xr--. 21 root root 4096 Mar 6 13:20 var

第一栏 'd':文件类型



第二栏 'rwx':拥有人的权限(对于文件：r读 w写 x执行，文件拥有人即使没有权限也可以强制保存)

第三栏 'r-x'；拥有组的权限(对于目录：r读取内容 w创建删除 x进入)

第四栏 'r--':其他人的权限(root用户拥有最高权限)

第五栏 '.'：表示该文件受Selinux的保护 ，若为’+’则表示文件有acl权限列表

第六栏 '21'：表示文件的硬链接数目

第七栏 'root'：表示文件拥有人

第八栏 'root'：表示文件拥有组

第九栏 '4096'：文件大小（单位：字节）

第十栏 'Mar 6 13:20':文件最后一次被修改的时间

第十一栏 'var': 文件名

## 权限修改命令

chmod [选项] 操作 文件名

修改文件或目录的权限

选项：

-R 递归修改目录内的所有文件

操作：

1.对象+/-/=权限

对象：

u 文件拥有者

g 文件组

o 其他人

a 所有人

权限：

r

对于文件：读取内容

对于目录：读取子目录和子文件

w

对于文件：编辑文件内容

对于目录：增加删除目录或文件

x

对于文件：执行

对于目录：可进入

s

二进制文件的可执行权限,others执行时会以文件拥有人的权限执行

eg: chmod u+s file1 或 chmod 4xxx file1 ; chmod g+s file1 或 chmod 2xxx file1

t

目录若拥有该权限，则子下的目录和文件只能由拥有人修改。

eg: chmod o+t file1 或 chmod 1xxx file

2.以八进制表示user，group，other的权限

如765表示rwxrw-r-x

chown 修改信息 文件名

修改拥有人和拥有组（非root用户只能修改文件的拥有组为本身的，不能修改拥有人）

修改信息：

'user' 将拥有人修改为user

':group' 将拥有组修改为group

'user:group' 将用有人和拥有组一次性修改为user和group

umask [权限]

查看[设置] 文件/目录 权限反掩码

文件默认权限=777-文件权限反掩码，文件无法默认拥有执行权限

root用户/etc/bashrc中设置默认权限反掩码

普通用户在家目录~/.bashrc下设置默认权限反掩码

setfacl [选项] [参数] 文件名

修改acl权限列表

选项：

-m 修改acl权限

参数:

u:用户名:权限

修改用户权限

eg：setfacl -m u:user1:rw- f1

g:组名:权限

修改组权限

eg: setfacl -m g:group1:rw- f1

m::权限

修改有效权限（优先级最大）

eg: setfacl -m m::rw- f1

d:u:用户名:权限

修改目录权限继承

eg：setfacl -m d:u:user1:rw- tmp1

-x删除指定用户或组的acl权限

eg：setfacl -x u:user: f1

-b 删除文件所有acl权限

eg：setfacl -b file1

getfacl 文件

查看文件acl权限

## 特殊权限

suid

其他人执行可执行文件时会以文件拥有人的身份执行

sgid

其他人执行可执行文件时会以文件拥有组的身份执行、创建文件或目录会继承该目录的拥有组

stick

目录权限。所有人可在目录下创建文件，但是只有拥有人或拥有组可删除文件

## sudo提权

sudo [选项] [参数] 命令

提权到管理员身份

-u 指定用户

-l 测试本用户有没有sudo权限

-k 清除之前的提权

文件配置：

sudoedit /etc/sudoers 或 visudo /etc/sudoers

编辑提权文件

/etc/sudoers

用户名 主机名=(提权目的用户) 可使用的命令（\*使用绝对路径表示）

指定可使用提权的用户。多个用户名、主机名、命令之间用逗号隔开

用户名 主机名=(提权目的用户) NOPASSWD:可使用的命令

设置提权时不许要输入密码

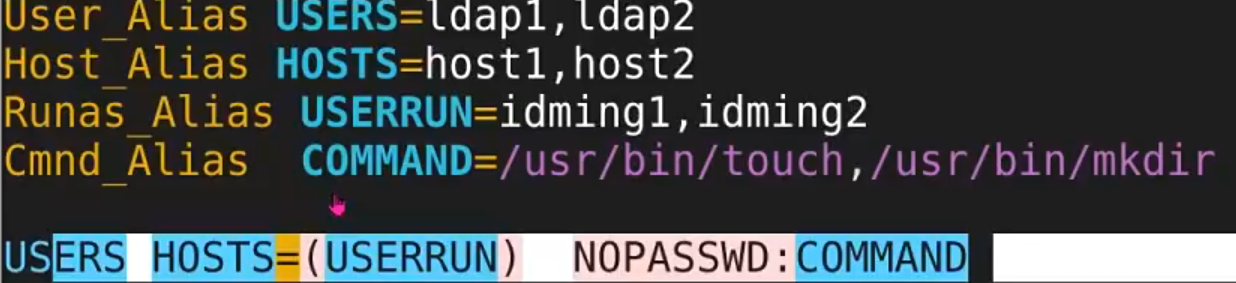
eg：user1 localhost.com=(root) NOPASSWD:/usr/bin

%组名 主机名=(提权目的用户) NOPASSWD:可使用的命令

指定可使用提权的用户

别名 用户名1,用户名2,用户名3,...

为用户列表起别名

eg: 

/etc/sudoers.d/

提权目录，其子文件都可仿照/etc/sudoers格式配置

# I/O输入，输出

>

重定向符号

[命令 参数...] > 文件名

将正常命令的结果覆盖到文件中( >> 为追加)

2>

错误输出符号

[命令 参数...] 2> 文件名

将ERROR显示覆盖到文件中( 2>> 为追加)

&>

并输出

[命令 参数...] &> 文件名

将正常命令显示和ERROR显示都覆盖到文件中( &>> 为追加)

<

输入重定向

[命令 参数...] < 文件名

以文件内容代替原本需要的键盘输入

|

管道符

[命令1 参数1...] | [命令2 参数2...] | [命令3 参数3...] ...

将前一个命令的结果作为后一个命令的输入

三通管道

[命令1 参数1...] | tee 文件名 | [命令2 参数2...]

将命令1的输出保存到文件中，同时作为命令2的输入

# Shell编程

`命令` == $(命令)

获取命令的执行结果

{起始..结束..步长}

连续列表 表示一个范围 ed: {1..10} {a..z..2}

{元素1，元素2,}

离散列表

命令解释器：#!/bin/bash

注释：#

多行注释：

<<EOF

内容

EOF

脚本的执行

1. 相对或绝对路劲执行（需要有执行权限，变量在子shell中定义）
2. Bash 文件名 （无需执行执行权限，变量在子shell中定义）
3. . 文件名 或 source 文件名（脚本的变量在本shell定义）

$数字 脚本或参数的位置参数

$0 文件 名

$1 传入的第一个参数

${10} 传入的第十个参数

$$ 执行脚本的PID

命令1 && 命令2

逻辑与

命令1执行成功才会执行命令2

命令1 || 命令2

逻辑或

命令1执行失败才会执行命令2

！命令

逻辑非

对命令的结果取反

{ 命令1;命令2;}

命令列表

在当前shell执行

(命令1;命令2)

命令列表

在子shell中执行

条件测试语句（0为true，1为false）

test 条件表达式

命令行中测试，若条件条件表达式成立，返回0，否则返回1

test [选项] [参数]

-f 测试文件是否存在

-d 测试目录是否存在

-s 测试文件是否为空文件

[ 条件表达式 ]

脚本中，若条件条件表达式成立，返回0，否则返回1

[ 1 -lt 9 ]返回0 ,[ a = b ]返回1

[[ 条件表达式 ]]

若条件表达式包含||、&&，则使用。

数值运算符

-gt 大于

-lt 小于

…

表达式连接

-a 与&&相似，在[ ]和test中使用

-o 与|| 相似，在[ ]和test中使用

-not 与!相似

-z 判断字符串是否为NULL，为NULL返回0

条件判断语句 if

格式:

if condition ; then

要执行的语句

elif Condition ; then

要执行的语句

else

要执行的语句

fi

注意！上面的condition可以是[]，[[]],或者(()):

[]是最基本的条件测试表达式

[[]]是扩展测试命令，提供了比[]更强大的功能

(())是数学表达式，支持常见的+-\*/等数学运算

两个整数的比较

-eq equal 判断两个整数是否相等

-ne not equal 判断两个整数是否不等

-gt greater than判断整数1是否大于整数2

-lt less than 判断整数1是否小于整数2

-ge greater/equal 判断整数1是否大于等于整数2

-le less/equal 判断整数1是否小于等于整数2

选择语句case

格式：

case 变量

值1)

值2)

…

\*)

esac

循环语句 while

格式

while condition

do

要执行的语句

done

while循环读取文件，将每一行存入变量中：

格式1：

exec < 文件;while read 变量

do

语句

done

格式2:

cat 文件 | while read 变量

do

语句

done

格式3：

while read 变量

do

语句

done < File

循环语句util

util 条件

do

语句

done

循环语句 for

格式1

for i in 列表

do

要执行的语句

done

注意！这里的列表可以是数字序列1 2 3 4 5；{1..100}；文件序列

格式2

for 列表

do

要执 行的语句

done

格式3

for((i=0;i<=10;i++))

do

要执行的语句

done

格式4

for((i=0;i<10;i++));do 要执行的语句;done

# 变量

## 变量类型：

### 1 本地变量（用户自定义变量）

定义：在当前shell中生效

1 直接定义

变量名=值

2 通过declare或typeset定义

格式:declare [选项] [变量名]=[值]

选项:

-a :将后面的variable定义为数组

-i :将后面的variavle定义为整数数字

-x :用法与export一样，就是将后面的variable变成环境变量

-r :将一个variable的亦是设置成只读，读变量不可更改内容，也不能unset

另外，单独使用declare会列出当前所有的变量

### 2 环境变量

定义：在当前shell和子shell中生效

通过export定义

export [选项] [变量名]=[值]

选项:

-f 　代表[变量名]中为函数名称。

-n 　删除指定的变量。变量实际上并未删除，只是不会输出到后续指令的执行环境中。

-p 　列出所有的shell赋予程序的环境变量。

注意！export定义的环境变量只会在当前终端生效，可在~/bashrc, ~/bash\_profile, /etc/bashrc， etc/profile等文件末尾定义永久的环境变量(重启终端生效)

特殊环境变量：

PATH：可执行文件（命令的路径）

PS1：命令提示符的格式

HOME：当前用户的家目录

USER：当前用户名

### 3 位置参数变量

类型:

$数字 脚本或参数的位置参数

### 4 Linux预设变量

类型:

$# 传递给脚本或函数的位置参数的个数

$? 上一命令的退出状态码。0 没有错误;非0 有错误

$\* 传递给脚本或函数的位置参数，双引号包围时作为一个整体

$@ 传递给脚本或函数的位置参数

$$ 当前Shell进程的进程ID(PID)

$! 最后一个后台命令的进程ID

$0 当前脚本的名称

!$ 上一条命令的参数

## 调用变量

$变量名

$(变量名)

# 进程

ps命令显示的内容：

PID：进程号

TTY：得到终端的方式

pts：伪终端（远程ssh连接，图形化登录等）

tty：直连登录

？：表示没有终端，为内核进程

TIME：进程的持续时间

CMD:运行的命令

top命令显示的内容

进程之间的通讯方式：管道，信号，消息队列，共享内存，socket套接字

进程优先级：

PR 动态优先级（内核自动调整）

NI 静态优先级（用户手动调整）（-20~19 以小为优）

# 计划任务

at 时间 [< 脚本]

交互式创建计划任务（一次性）

格式：hh:mm如14:24

ctrl + d保存

at -q 优先级 时间 [< 脚本]

创建任务时指定优先级

at -d 任务编号

删除任务

atq 或 at -l

列出所有at任务

2 Wed May 6 11:59:00 2026 a root

2：任务编号

Wed May 6 11:59:00 2026：时间

a：任务优先级（a-z，z最次）

root：创建者

/etc/at.deny

at命令权限的黑名单

/etc/at.allow

at命令权限的白名单（需手动创建，权限大于黑名单）

crontab -e

修改周期性任务（只有root才能指定user）

格式：

\* \* \* \* \* [user] ls

分 时 日 月 周 [用户] 命令

eg：

0 12 \* \* \* echo “It's noon.” #每天12点发送“It' noon”

\*/5 \* \* \* \* date +"%F %T" #每5分钟显示当前时间

crontab -l [-u 用户名]

列出 [某个用户创建的] 周期性任务

/etc/cron.deny

crontab命令黑名单

/etc/cron.allow

crontab命令白名单

/var/spool/cron

使用crontab命令创建的计划任务

/etc/crontab

/etc/cron.d

系统的计划任务

/etc/cron.hourly

/etc/cron.monthly

/etc/cron.daily

/etc/cron.weekly

适合各种时间的计划任务

# 文件查找

locate [选项] [参数] 关键字

根据关键字查找文件

选项：

-i 忽略大小写

-n 显示前..行

updatedb

更新查询数据库

## find 路径 选项 参数 [连接符 选项2 参数2] [动作]

查找文件位置（无需数据库）

选项：

-name 根据文件名查找（支持正则表达式，需要单引号）

-user 根据文件所有人查找

-group 根据文件所有组查找

-uid根据uid查找

-gid 根据gid查找

-size 根据文件大小查找

find /etc -size +5M 查找大于5M的文件

find /etc -size -5M 查找小于5M的文件

find /etc -size 5M 查找等于5M的文件

-type 根据文件类型查找（f文件 d目录 b块设备）

-perm 根据文件权限查找

find /home -perm 766 查找权限为766的文件

find /home -perm \766 查找user group other三者之一权限大于等于766的文件

find /home -perm -766 查找user group other三者之一权限等于766的文件

find /home -perm /g=s 查找有修改组id权限的文件

-atime 根据访问时间查找

-mtime 根据修改文件内容的时间查找

-ctime 根据改变文件的属性时间查找 （单位：距今第..天，+7 ：7天之外，-7：7天之内：7：第七天）

-cmin -mmin -amin 以分钟为单位

连接符：

-a 逻辑与

find /home -user root -a -name bashrc （默认参数）

-o 逻辑或

find /home -user root -o -gid 1001

! 逻辑非 等价于 -not

eg: find . ! -name 'scl' -a -type d

动作：

-print 将查找到的文件路径显示出来（默认动作）

-ls 列出详细信息 类似ll

-fls 另存为

-delete 删除文件

-ok 执行后面的命令，交互式

-exec 执行后面的命令 非交互式

find / -name bin -exec rm {} \; （{} \; 为固定格式，{} 代表find搜索到的项，\；代表结束）

find 命令后接管道传递值时，管道都要接xargs。(等同与-exec参数。)

find . -name file1 | xargs chmod 777

# 下载

curl [选项] [参数] 网址URL

查看网站源代码

选项：

-i查看服务器信息

-O 下载网站资源到当前目录

-o 下载资源到指定目录

wget [选项] [参数] 网址URL

下载文件到当前目录

选项：

-O 指定另存为的文件名

-d 放入后台下载

--spider 测试能否下载

# 远程连接与文件传输

ssh 用户名@IP地址或域名 [命令]

远程连接

RHCL 9 中默认拒绝root用户远程登录。

解决方法：

1. 登录到普通用户然后切换到root用户
2. 修改/etc/ssh/sshd\_conf配置文件，添加 PermitRootLogin yes即可

ssh 用户名@IP地址或域名 命令

在远程的主机上执行命令（不会连接到远程shell）

ssh -X 用户名@IP地址或域名 命令

支持转发图形协议（运行图形化应用）

scp [选项] 要传输的文件 用户名@IP地址:目标路径

远程传输文件（属于ssh）

选项：

-p 保持时间戳等文件属性信息不变

-r 拷贝目录

-P 指定端口

scp [选项] 用户名@IP地址:文件 本地路径

远程接收文件

选项类似。

systemctl restart sshd

重启ssh服务

SCP的BUG：在文件/etc/bashrc或/etc/profile、 $HOME/.bashrc、 $HOME/.bash\_profile内有输出类型命令时会传输失败

配置文件：

客户端：/etc/ssh/ssh\_config

服务端：/etc/ssh/sshd\_config

常用字段：

Port 22 监听端口

ListenAddress 0.0.0.0 监听的IP地址（0.0.0.0表示全网段）

PermitRootLogin prohibit-password root用户远程登录

MaxSessions 10 最大连接会话数量

PasswordAuthentication yes 是否允许使用密码登录

PermitEmptyPasswords no 是否允许空密码登录

## 免密登录

ssh-keygen [选项]

生成公私密钥对（交互式）（位于~/.ssh/id\_rsa.pub）

ssh-copy-id 用户名@IP地址

发送公钥给服务端（位于~/.ssh/ authorized\_keys，每一行为一把公钥）

sshpass -p ‘密码' ssh-copy-id -o StrictHostKeyChecking=no 用户名@IP地址

发送公钥给服务端(非交互式，依赖包：epel-release)

## 查看系统历史登录/登录情况

uptime

查看运行负载，用户在线数量，系统运行时间

who 或 w

查看登录到本地的用户

last

查看登录和重启的记录

lastb

查看登录失败记录

lastlog

查看登录日志

# 文件同步

rsync [选项] 源路径 目标路径

同步文件/目录

选项：

-a 同步文件及目录，不改变属性

-v 列出详细信息

-n 测试同步

-z 同步到目标路径时压缩文件

--delete 修改目标路径的内容使得和源路径一致

--existing 只更新目标端已经存在的文件

--ignore-exisiting 只更新目标端不存在的文件

--remove-sourse 同步完成后删除源路径的文件。类似mv命令

eg：

rsync -a /etc/ /root/ : 将etc目录下的内容同步到家目录

rsync -a /etc /root/ : 将etc目录同步到家目录

# 块设备

## 文件的组成

文件由 元数据+数据 组成

元数据：保存文件属性信息

数据：保存文件内容，即文件系统中的块(block)

rm进行文件的删除时，系统会删除inode与block的对应关系

## 块设备种类：

IDE设备：

/dev/hda\*

/dev/hdb\*

eg: /dev/ha1 第一个IDE设备的第1个分区

SCISI/SATA/SSD

/dev/sda\*

/dev/sdb\*

eg: /dev/sdc2 第三个SCSI设备的第2个分区

virtlO

/dev/vda\*

/dev/vdb\*

eg：/dev/vdc2 第三个virtlO设备的第2个分区 Xen: /dev/xvda1 : xen虚拟化第一个硬盘的第一个分区

Nvme 固态卡

/dev/nvme0n1p\*

/dev/nvme0n2p\*

eg: /dev/nvme0n2p2 第二个Nvme设备的第二个分区

光驱

/dev/sr0

逻辑设备

/dev/mapper/rhel-root

/dev/rhel/root

## 文件系统：

本地文件系统：一般在本地使用

NTFS： windows主流文件系统

fat系列 U盘主流文件系统，无法存储大文件

exfat 移动介质存储

ext系列 linux主流文件系统（旧版）

xfs系列 linux主流文件系统（RHEL7以上版本）

网络文件系统：网络共享

NFS Linux与Linux间共享的文件系统

CIFS Windows与Linuc

集群文件系统：多个主机做一个存储池

cephfs ceph分布式存储

GFSfs google的集群文件系统

VIMS 华为的集群文件系统

光盘文件系统

iso9660 光盘文件系统

du

查看文件/文件夹的磁盘占用情况

格式:du [选项] [地址]

选项:

-s 只列举总的内存占用

-h 以较清晰的格式列出

df

查看磁盘空间使用情况

格式:df [选项]

选项:

-h 以较清晰的格式列出

-T 显示挂载的文件的类型

## 设备的挂载与卸载

要对设备进行文件操作，必须将设备挂载到文件系统的某个路径下

VMware的共享文件夹挂载点在/mnt/hgfs下。

mknod 设备名 设备类型 主版本号 次版本号

创建设备文件

eg：mknod /dev/loop b 260 0

mount

查看挂载信息

mount [选项] [参数] 设备路径 挂载点

临时挂载设备(umount 卸载设备)

eg：mount /dev/sr0 /mnt/sr0

选项：

-r 只读挂载，无法向设备写入数据

-w 读写挂载（默认选项）

-o 添加额外选项

async:异步模式 先保存文件到内存，再到硬盘

sync:同步模式 直接保存到硬盘

atime/noatime：是否更新atime，包含目录和文件

auto/noauto：是否支持自动挂载

exec/noexec：是否支持文件系统上的可执行文件运行

dev/nodev：是否支持在此文件系统上使用设备文件

suid/nosuid：是否支持suid的权限

remount:重新挂载

ro/rw:只读或者读写挂载 与-w -o类似

user/nouser：是否允许普通用户挂载此设备

defaults:默认挂载选项，是rw, suid,dev,exec,auto, nouser,and async的组合

PG：当将要挂载的点已经存在文件时，挂载设备后该文件会隐藏。

永久挂载设备：

1. 修改配置文件/etc/fstab

内容：

/dev/mapper/centos-root / xfs defaults 0 0

/dev/mapper/centos-root ：设备名、UUID或卷标

/ ：挂载路径

xfs：文件系统类型

defualt：挂载选项

0：是否开启备份（0不开启，1开启）

0：是否检查文件系统错误（0不检查，1检查(仅对ext系列文件系统生效) ）

1. 执行mount -a命令，刷新挂载

[挂载故障时](#_故障三：fstab文件系统故障)

自动挂载autofs（自动挂载程序 会阻止用户在/mnt/目录下创建文件）

配置文件/etc/auto.master

内容：/mnt /etc/autofs/conf.autofs

/mnt : 挂载点的父级目录

/etc/autofs/conf.autofs : 执行的触发配置文件（路径和名称自定义，文件后缀必须为.autofs）

触发配置文件（如/etc/autofs/conf.autofs）：

内容：sr0 -rw :/dev/sda

sr0 挂载点

-rw 挂载时附加的参数

:/dev/sda 挂载设备路径 （‘:‘ 前带有IP地址时可挂载网络文件系统）

lsblk [选项]

查看设备挂载情况

选项:

-f 查看详细的设备挂载情况，显示文件信息

参数信息

NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINTS

NAME 设备名称

MAJ 主版本号，标明了设备类型

MIN 次版本号 即设备序号

RM 标明设备是(1)否(0)为移动设备如U盘，移动硬盘等

SIZE 设备的总大小

RO 设备是否为只读设备（0：可读写；1：只读）

TYPE 设备类型（/proc/devices保存了设备类型与主版本号的对应关系）

MOUNTPOINTS 设备的挂载点。

fuser -v 设备名

查看使用该设备的进程

fuser -km 设备名

终止该设备所有进程

## 块设备的分区

MBR（msdos）分区方案

最多支持4个分区（前三个主分区和扩展分区）

每个分区最大2TB

分区表信息保存在硬盘前512字节，前446字节为引导程序

GPT分区方案

最多支持128个分区

每个分区最大支持8ZB

分区表信息保存在头部和尾部(备份)

fdisk [设备名]

磁盘分区MBR型（交互式）

选项:

-l 显示所有硬盘的分区列表

gdisk

磁盘分区GPT型（交互式）

## 格式化/创建文件系统

设备创建文件系统后会生产UUID，文件系统即文件存放的方式

mkfs [选项] [参数] 设备名

格式化设备

-t 指定格式化文件系统类型

eg: mkfs -t xfs /dev/sr0

blkid [设备名]

查看设备的uuid、文件系统类型

mkswap 设备名

将设备格式化为swap分区

swapon [选项] [参数] 设备名

挂载swap分区(临时)

选项：

-s 查看所有swap分区

-p 指定优先级

swapoff 设备名

卸载swap分区z

## 卷标

卷标，即 分区/设备 的别名

xfs文件系统类型

创建卷标

xfs\_admin -L 卷标名 设备名

删除卷标

1. 取消挂载
2. xfs\_admin “--” 设备名

ext文件系统类型

创建卷标

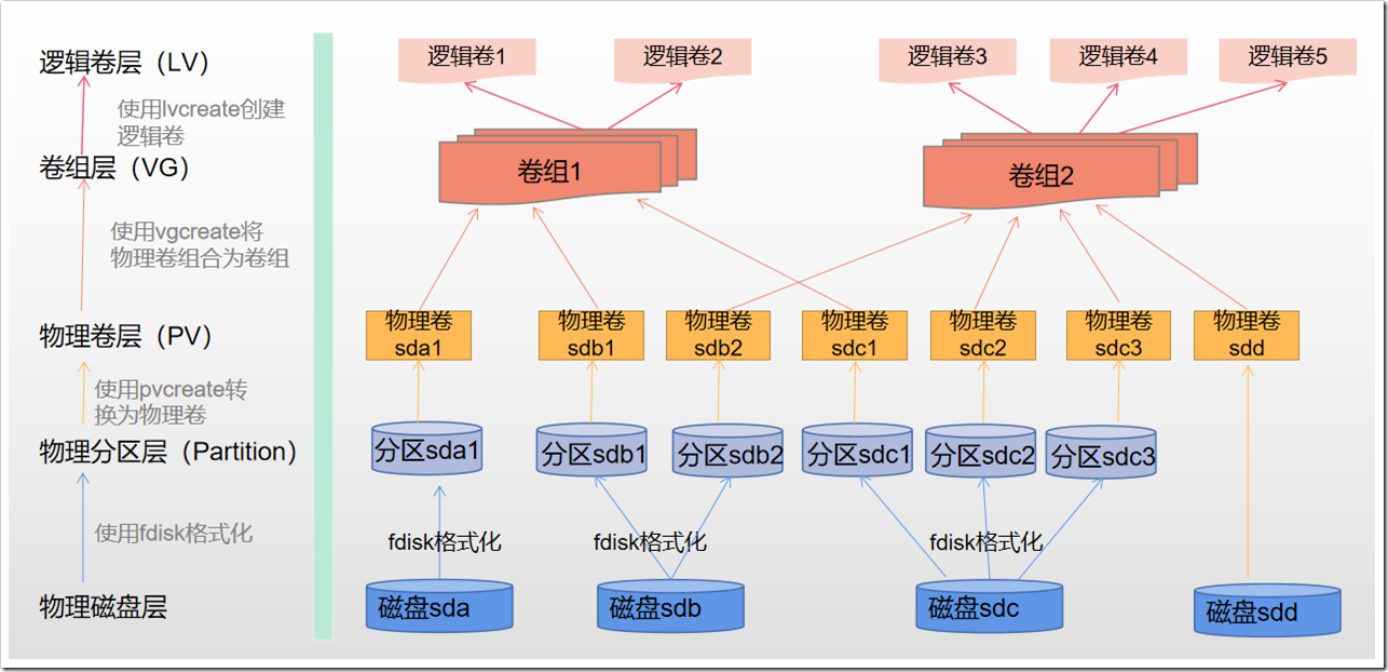
e2label 设备名 卷标名

删除卷标

e2label 设备名 “”

# 逻辑卷

相较于物理块设备，具有动态扩容的能力



## 逻辑卷操作

1. 创建物理卷

pvcreate 块设备名1 块设备名2 …

eg：pvcreate /dev/sda1 /dev/sda2 /dev/sdb1

1. 创建卷组

vgcreate 卷组名 块设备名1 块设备名2 …

选项：

-s 指定PE大小

eg：vgcreate vg0 /dev/sda1 /dev/sda2

1. 创建逻辑卷

lvcreate -n 逻辑卷名 -L 逻辑卷大小 卷组名

-l 指定PE数量

-n 指定逻辑卷名

-L 指定逻辑卷大小

eg：lvcreate -n lv1 -L 10G vg0

1. 格式化文件系统

mkfs -t 文件系统类型 逻辑卷路径

eg：mkfs -t xfs /dev/vg0/lv1

显示物理卷pvs pvdisplay pvscan

显示卷组vgs vgdisplay vgscan

显示逻辑卷lvs lvdisplay lvscan

lvremove 逻辑卷路径

删除逻辑卷

vgremove 卷组名

删除卷组

pvremove 物理卷路径

删除物理卷

lvextend -L 空间大小 逻辑卷路径 -r

扩容逻辑卷

eg：

lvextend -L 10G /dev/vg0/lv0 -r 或 lvextend -L +5G /dev/vg0/lv0 -r

resize2fs 逻辑卷路径 #ext4文件系统

xfs\_growfs 挂载点 #xfs文件系统

vgextend 卷组名 物理卷路径

xfs\_growfs 挂载点

卷组扩容

eg：vgextend vg0 /dev/sda1

vgremove 物理卷路径

卷组缩容

修改PV属性

pvchange -xn device 不可分配

pvchange -xy device 可分配

修改VG属性

vgchange -ay vg0 激活卷组

vgchange -an vg0 禁用卷组

修改LV属性

Ivchange-ay Ivname 激活逻辑卷

Ivchange-an Ivname 禁用逻辑卷

Ivchange-pr Ivname 设置只读

Ivchange-prw Ivname 设置读写

## 逻辑卷日志

日志文件目录：/etc/lvm/archive/

记录了卷组和逻辑卷的操作

vgcfgrestore -f 日志文件名 卷组名

操作回滚

eg：vgcfgrestore -f vg0\_00000-1267727612.vg vg0s

## 快照

快照是一种特殊的逻辑卷，可以保存某个逻辑卷的状态。

COW快照

将数据写入差分卷,还原快照时直接读取原卷即可

ROW快照

将数据写入原卷，差分卷保存原卷被修改的原本的信息 还原快照时用差分卷覆盖原卷即可

lvcreate -n 快照名 -L 快照大小 逻辑卷路径

为某个逻辑卷创建快照(COW快照)

eg：lvcreate -n lv-snap -L 1G /dev/vg0/lv0

lvconvert --merge 快照路径

还原快照（快照和对应的逻辑卷此时都不能处于挂载状态）

## Stratis池

与逻辑卷架构类似，由分区构成存储池，存储池拆分为Stratis池(又叫文件系统)（文件系统只能为xfs，大小默认1G，系统自动扩容）

stratis pool create 存储池名 分区路径1 ...

创建存储池

stratis filesystem create stratis池名 存储池名

创建stratis池

stratis pool list

列出所有存储池

stratis filesystem snapshot 存储池名 Stratis池名 快照名

创建快照

stratis filesystem snapshot 存储池名 快照名 Stratis池名

恢复快照（快照和源文件系统都不能处于挂载状态）

# 网络文件系统

存储类别：

DAS：直连式存储（硬盘，U盘）

NAS：网络附加存储

SAN：存储区域网络

NFS（属于NAS网络附加存储）

软件包：nfs-utils

进程名：nfs-server

主配置文件：/etc/exports

内容：

/share 192.168.36.0/24(rw)

/share ：共享目录

192.168.36.0/24 允许访问的IP网段（‘\*’代表允许所有）

(rw) 挂载参数(权限)

参数：

ro/rw

只读或者读写共享

sync/async

同步或者异步

sec=sys

基于标准 Linux 文件权限访问

root\_squash

打压root用户，客户端的root用户映射到服务端为nobody用户

norootsquash

不打压root用户，客户端是root，服务端也是root

all\_squash

打压普通用户，客户端的普通用户映射到服务端是nobody用户

noallsquash

不打压普通用户，客户端的普通用户映射到服务端是UID相同的用户

showmount -e IP地址

显示IP地址下的共享目录（客户端命令）

mount IP地址:/共享目录名 挂载点

挂载共享目录（可使用[autofs](#_设备的挂载与卸载)自动挂载）

# 打包&压缩

打包：

将整个目录变为一个打包文件（保留了文件属性）

压缩：

压缩文件大小

tar -cf 包名 目录

将目录打包

tar -xf 包名

解包

tar -tf 包名

列出包的内容

gzip [选项] 文件名

压缩文件

选项：

-k 保留源文件

gunzip 文件名

解压文件

bzip2/bunzip2 xc/unxc等工具用法相同

tar -czf 包名 目录

打包并压缩

tar -xf 包名

解包并解压缩

# 软件包管理

软件包（安装包）需要联网下载

iso光盘文件中常自带软件包，位于光盘挂载点BaseOS和AppStream目录下

RPM包（RedHat,CentOS,Openeuler系统软件包支持格式）

DEB包（ubuntu，debian系统软件包支持格式）

RPM包名：

vsftpd-3.0.5-6.el9.x86\_64.rpm

vsftpd： 包名

3.0.5： 软件版本

6 修订次数

el9 发行平台，这里指rhel9

x86\_64 cpu架构和64位操作系统

.rpm 文件后缀

## rpm [选项] [参数]

软件包管理工具

选项：

-i 安装软件（不能解决软件包的依赖问题）

eg: rpm -i vsftpd-3.0.5-6.el9.x86\_64.rpm

-e 卸载软件

eg: rpm -e vsftpd

-vh 查看安装进度

-U 升级软件(软件不存在则直接安装)

eg: rpm -U vsftpd-3.0.5-6.el9.x86\_64.rpm

-q 查看该软件是否安装

eg: rpm -q vsftpd

-aq 查看所有安装的软件

-ql 查询已安装的软件所有文件的路径

eg: rpm -ql vsftpd-3.0.5-6.el9.x86\_64.rpm

-qc 查询已安装的软件的配置文件路径

--import 导入密钥文件（用于安装时的安全校验）

eg：rpm --import RPM-GPG-KEY-redhat-release

验证密钥导入：

rpm -aq | grep pubkey

-K 检验包的安全性(根据系统导入的密钥校验)

eg：rpm -K vsftpd-3.0.5-6.el9.x86\_64.rpm

-Va 检查系统所有软件的安全性

## yum和dnf

解决了包与包之间的依赖问题

### 区别：

1. yum命令性能较慢
2. yum无法多进程安装
3. yum更新时不会检查依赖性

### 配置yum仓库：

路径：/etc/yum.repos.d

创建 .repo 结尾文件

文件内容：

[BaseOS]

name=任意

baseurl=镜像源地址（本地镜像源地址下应该有repodata目录用于存放依赖关系。eg: baseurl=file:///mnt/cdrom/AppStream）

enable=1（1则仓库开启，0关闭）

gpgcheck=1（0则安装时不必校验软件完整性，1则要校验）

gpgkey=密钥文件地址（gpgcheck=1时启用）

[备用仓库]

name=任意

baseurl=镜像源地址2

enable=1

gpgcheck=1

gpgkey=密钥文件地址

**配置阿里镜像源：**

curl -o /etc/yum.repos.d/aliyun.repo http://mirrors.aliyun.com/repo/Centos-7.repo

### 缓存文件

路径: /var/cache/dnf

dnf clean all

清除缓存

dnf makecache

生产缓存文件

### dnf命令

dnf install 软件名 [选项]

安装软件

选项：

-y 直接同意安装

dnf groups install [包组名]

安装包组

dnf remove 软件名 [选项]

卸载软件

dnf reinstall 软件名

重装软件

dnf repolist all

列出所有软件仓库

dnf list [选项]

列出所有软件包（最后一列带@表示已安装）

选项：

all 查看所有软件包（默认选项）

available 查看未安装的软件包

installed 查看已安装的软件包

update 查看可升级的软件包

dnf info 包名

查看包信息

dnf provides 文件

查看文件来自哪个软件包

eg：dnf provides “\*/semanage“

dnf search 关键字

查找包含关键字的软件包

dnf history [包名]

查看执行过的dnf命令

dnf history undo 历史记录条目

撤回该操作

createrepo [--update]

制作repodata[增量更新]

源码包安装

1. 获取源码包
2. 解压缩源码包
3. 安装编译环境
4. 预配置、编译、编译安装
   1. 在解压缩后的源码包下运行./configure --prefix=要安装的目标路径 （检查是否有缺少的依赖）
   2. make
   3. make install

# 系统启动及服务管理

系统运行级别

0~6 0：关机 3：带有网络的字符页面 5：图形化页面 6：重启

init 运行级别

切换系统运行级别

runlevel

查看当前系统运行级别

## 单元类型



systemctl -t device/service/mount/...

查看系统单元

systemctl -t service：

UNIT 服务名

LOAD 载入配置文件是否成功

ACTIVE 服务高级状态，一系列进程的状态，服务的最终状态。

SUB 服务的低级状态

DESCRIPTION 服务的描述信息

## 服务状态

systemctl list-unit-files --type service

查看服务状态

UNIT FILE 服务名

STATE 状态

enable 开机自启

disenable 开机不自启

static 服务无法自己启动，只能由其他服务调动

mask 服务被锁定，任何情况都无法启动

PRESET 下一次的状态

修改服务状态：

systemctl enable/disable 服务名

设置服务开机自启/不自启

systemctl static 服务名

设置服务无法自己启动

systemctl mask/unmask 服务名

锁定/解锁 服务，使其无法启动

服务操作：

systemctl start 服务

启动服务

systemctl stop 服务

暂停服务

systemctl restart 服务

重启服务

systemctl status 服务

查看服务状态

systemctl reload服务

重新读取配置文件

systemctl list-dependencies 服务名

查看服务的依赖

## 启动级别

决定了系统启动时默认的运行级别。

常用：

graphical.target 图形化

multi-user.target 字符页面

rescue.target 系统修复

emergency.target 系统修复（紧急）

systemctl set-default 启动级别

设置启动级别

systemctl isolate 启动级别

临时设置系统运行级别（等价与init命令）

systemctl get-default

查看系统当前的启动级别

## 系统启动流程：

BIOS:最小化的输入输出系统

1.上电开机

2.BIOS开始自检并且加载硬件

3. BIOS根据启动顺序选择对应的启动设备

4.从硬盘去加载引导代码（引导程序）

a)引导代码-->引导程序-->grub2

b)存放在硬盘的前446字节的位置(MBR分区的最前面）

5. 引导程序加载启动引导分区中的配置文件(引导配置文件)/boot/grub2/grub2.cfg

6.grub2会加载内核和initramfs->要挂载根文件系统，然后执行systemd来初始化(启动服务、根据启动目标获取操作页面）

a)内核能够挂载根文件系统吗？想要挂载文件系统，是需要对应的文件系统驱动xfs系列ext系列

b)initramfs目的就是为了帮助内核挂载根文件系统，提供文件系统的驱动以及硬件驱动

7.initramfs会执行自己的systemd并且将磁盘上的根文件系统挂载到/sysroot(真正的根文件系统挂载到/sysroot)

8.内核就会将initramfs的假根切换为/sysroot(真正的根)，然后执行systemd进行初始化系统

9.systemd通过启动目标来得到操作页面（字符页面、图形化页面）

grubby --set-default 内核路径 或 grubby --set-index 内核index

修改默认启动内核

grubby --default-kernel

查看默认内核

grubby --info=ALL

查看所有内核及其信息

grubby --gefault-title 标题名

修改启动时内核的标题（配置文件位于/boot/loader/entriescd/）

## 启动时添加启动参数：

方法1（永久）

grubby --args=”内核参数” --update-kernel 内核路径（一般位于/boot/下）

添加启动参数（等同与 将命令写在文件/boot/loader/entries/内核名 下）

方法2（永久）

在文件/etc/default/grub中最后一行添加命令

执行grub2-mkconfig -o /boot/grub2/grub.conf 命令

方法3（非永久）

开机内核选择菜单中，按‘e’进入启动菜单，在‘linux……’行末尾添加内核参数。（eg: systemd.unit=multi-user.target）

按ctrl+x继续启动

## 开机自运行

1. rc.local文件

编辑/etc/rc.d/rc.local文件

添加可执行权限

1. 配置systemd单元文件：

cat > /etc/systemd/system/myapp.service <<EOF

[Unit]

Description=MyApp Service

After=network.target

[Service]

Type=simple

ExecStart=/usr/local/bin/myapp --config /etc/myapp.conf

ExecRestart=/usr/local/bin/myapp -s reload

ExecStop=/bin/kill -s QUIT $MAINPID

Restart=on-failure

User=myuser

Group=myuser

WorkingDirectory=/usr/local/myapp

[Install]

WantedBy=multi-user.target

EOF

systemctl daemon-reload #重新加载单元文件

## 常见启动故障

### 故障一：忘记 root用户密码(RHEL9选择救援内核，其他版本选择正常内核)

1. 重启系统，选择救援内核，按e键

2.在Linux这一行后面，（删除后面的参数，一直到ro后面），然后添加参数rd.break

3.把/sysroot重新以读写的方式进行挂载，然后切换到真正的根文件系统

a)mount -o remount,rw /sysroot

b)chroot /sysroot

4. 修改root密码

5.最重要！创建autorelabel文件，告诉selinux放行此次操作，重新给文件打上标签

a) touch /.autorelabel

### 故障二：grub.cfg引导配置文件丢失

方法1：

使用命令 grub2-mkconfig -o /boot/grub2/grub.cfg

方法2:

1.指定系统上的boot分区，如果有多个硬盘，请记住每一个硬盘的第一个分区永远是boot分区

a) set root='hd0,msdos1’选择第一个分区

2.指定内核文件以及真正的根文件系统；根据硬盘的类型来判断真正的根文件系统是什么

a)例如我的系统盘是nvme的，所以的知道/dev/nvmeOn1p2可能是根文件系统

b)例如我的系统盘是scsi的，所以我知道/dev/sd2可能是根文件系统

c)例如我的系统盘是逻辑卷，所以我知道/dev/mapper/rhel-root是根文件系统(如果centos,那么/dev/mapper/centos-root)

i. linux /vmlinux-5.14xxxxxxxxx root=/dev/nvmeOn1p2 ro

3. 指定内核配对的 initramfs

a) initrd /initramfs-5.14xxxxxxxxx.img

4. boot引导启动进入系统

5. 执行命令grub2-mkconfig-o /boot/grub2/grub.cfg生成 grub.cfg 的引导配置文件

### 故障三：fstab文件系统故障

若[/etc/fstab文件配置](#_设备的挂载与卸载)错误:

重启系统会自动进入紧急模式

执行命令mount -a（挂载fstab文件中未挂载的设备），根据报错信息处理。

### 故障四：引导程序故障（需要有与操作系统版本一至的光盘镜像）

模拟故障:dd if=/dev/zero of=/dev/nvme0n1 bs=446 count=1

维修故障：

1.通过光盘启动，进入救援模式

2.扫描硬盘的文件系统，并且挂载到/mnt/sysroot

3.切换到真正的根，执行grub2-install/dev/nvmeOn1安装引导程序

4. exit 退出

### 故障五：内核文件丢失，initramfs文件丢失，引导分区文件丢失

模拟故障:rm-rf /boot/\*

维护思路：boot分区的文件出现丢失，找到文件不就可以了吗？

1.通过光盘启动，进入救援模式

2.切换真正的根文件系统

3.配置软件仓库，执行dnf installkernel\*安装所有的内核软件包达到释放文件的作用

4. 执行grub2-install安装释放 grub2目录，执行grub2-mkconfig生成引|导配置文件grub.cfg

5.exit退出引导进入系统

# 日志

## 日志服务类型

systemd-journald

易失性日志，关机可消除，二进制类型保存

路径：/run/log/journal /

rsyslog

持久性日志，文本类型保存

路径：/var/log/

## rsyslog

### 日志优先级

定义不同的消息警告类别



### 配置文件：/etc/rsyslog.conf

格式：

日志类型.优先级; 日志类型2.优先级2; 日志类型2.优先级2;... 保存路径

匹配该优先级及以上的保存到指定路径。

(类型，优先级通配符：.日志.\* 匹配所有；日志.=优先级 只匹配该优先级；日志.!优先级 取反；日志.none 不匹配该日志)

eg: mail.info /var/log/mail 将mail日志保存到本地mail目录下

eg: mail.info @192.168.36.2 将mail日志保存到rsyslog服务器上，使用tcp转发

eg: mail.info @@192.168.36.2 将mail日志保存到rsyslog服务器上，使用udp转发

logger [选项] [参数] ‘内容’

发送日志（默认发送路径为/var/log/messages/）

选项：

-t日志类型

-p 优先级

eg:logger -t mail -p mail.info -i ‘hello world’

-i 记录该日志的pid

### 查看日志

日志路径：/var/log/

格式：

Apr 10 11:54:59 rhclnamed[1147]: received control channel command 'reload'

Apr 10 11:54:59 日志产生时间

rhcl日志产生的主机名

named[1147] 产生的程序及pid

received control channel command 'reload' 日志内容

### 配置日志服务器

配置/etc/rsyslog.conf文件：

# Provides UDP syslog reception

# for parameters see http://www.rsyslog.com/doc/imudp.html

module(load="imudp") # needs to be done just once

input(type="imudp" port="514")

# Provides TCP syslog reception

# for parameters see http://www.rsyslog.com/doc/imtcp.html

module(load="imtcp") # needs to be done just once

input(type="imtcp" port="514")

重启服务

systemctl restart rsyslog

### 日志轮询logrotate

作用：

定期保存，删除日志文件

将/var/log下的配置文件加上日期

最多保存7天

配置文件：/etc/logrotate.conf

## systmed-journald

日志保存路径(二进制)：/run/log/

配置文件：/etc/systemd/journald.conf 或 /usr/lib/systemd/journald.conf

查看方式：journalctl命令

### journalctl [选项] [参数]

查看systemd-journal 日志

选项：

-f 实时查看日志

-n 指定查看尾x条目

eg：journalctl -n 10

-xe 查看系统上的错误信息

-u 指定服务名

-p 根据优先级查找

优先级类型：

emerg alert crit err warning notice info debug

--since 查看某个时间段的日志

eg：journalctl --since ‘2023-12-30‘ 查看2023年12月30号到现在的日志

\_PID=x 查看pid为x的进程日志

\_EXE=绝对路径 查看可执行程序的日志

\_COMM=命令

的日志

SYSTEMDUNIT=服务名 查看指定服务的日志

### 日志持久化保存

修改配置文件中Storage的值：

auto 如果存在/var/log/journal/(需手动创建)则持久化保存

persistent 强制持久化保存

volatile 不持久化保存

刷新journald 服务：

journalctl --flush

systemctl restart systemd-journald

# 系统时间/时区

## timedatectl时区控制命令

timedatectl

显示当前的时间及时区

timedatectl list-timezones

列出所有时区

timedatectl set-timezone 时区

设置当前系统时区

## chrony时间同步

**客户端：**

配置文件：/etc/chrony.conf

pool 时间服务器IP地址1 iburst

pool 时间服务器IP地址2 iburst

指定时间服务器

chronyc sources [-v]

查看所有的时间服务器及状态[详细信息]

chronyc -a makestep

立即进行一次时间同步

systemctl restart chronyd

重启chronyd服务

systemctl stop chronyd

暂停自动时间同步

**服务端：**

修改/etc/chrony.conf

添加：

allow IP网段/网络位

local stratum 10

重启chronyd服务

# Linux内核系统调优

## 内核功能：

1. 系统启动和初始化
2. 进程调度
3. 内存管理
4. 提供缓存加速
5. 支持标准网络协议和文件系统

内核的组成

/boot/initramfs-0-rescue-72a9a1d2a00f4a628bc5e84da2b748dd.img 初始化内核

/boot/vmlinuz-6.12.0-65.el10.x86\_64 运行内核

/proc/

保存了硬件信息和进程信息(进程信息目录以PID命令)

cat /proc/cpuinfo 或lscpu

查看cpu信息

cat /proc/meminfo

查看内存信息

cat /proc/partitions

uname [选项]

显示内核信息

选项：

-r：显示内核版本

-a：显示完整信息

sysctl [选项]

查看内核参数

选项：

-a 查看所有的内核参数

-p 重新加载/etc/sysctl.conf文件

配置文件/etc/sysctl.conf

修改内核参数：在配置文件末尾添加。

常用内核参数：

net.ipv4.ip\_forward = 0

配置转发报文的能力（0关闭，1开启）

vm.swappiness = 60

配置交换分区的权重（0~100，为100则使用物理内存，为0则使用swap分区）

## tuned工具 内核一键调优

配置目录：/usr/lib/tuned （保存调优配置集）

tuned-adm list

列出所有配置级别和当前的配置级别

tuned-adm recommend

列出系统推荐的配置级别

tuned-adm profile 配置级别

将系统调整到该配置级别

## 内核模块

内核功能被划分为一个个模块，只有需要该功能时才会加载模块，节省设备性能

lsmod

显示已加载的内核模块

Module Size Used by

Module: 模块名

Size：模块占用空间

Used by 依赖的模块名

modprobe [选项] 模块名

加载内核模块（会自动处理依赖关系）

选项：

-r 移除该模块

modinfo 模块名

显示模块的信息

# SELINUX

## 基本

允许或拒绝资源的访问权限

配置文件：/etc/selinux/config

模式:

强制模式(enforcing)：拒绝对加以限制的资源的访问

许可模式(permissive)：允许访问限制资源，但会记录

关闭模式(disabled)：SELINUX处于关闭状态

setenforce 0/1

切换为 关闭/强制 模式（临时）

getenforce

显示安全级别

## selinux标签

控制文件访问的具体标识，当进程与文件标签不一致时selinux阻止访问

所有目录与文件的安全上下文保存位置：/etc/selinux/targeted/contexts/files/file\_contexts

### 文件管控：

ls -Z 文件名

查看文件标签

格式：finedu:objectr:usr\_t:s0 f1

usr\_t 标签

chcon -t 标签类型 文件名

修改文件标签

选项：

-t 指定标签类型

-R 修改目录的所有子文件

restorecon 文件名

重置为默认标签

选项：

-R 重置目录的所有子文件

-z 查看详情信息

semanage fcontext -a -t 安全标签 文件绝对路径

修改文件或目录的安全标签的默认值（命令所在软件包：policycoreutils）

配置文件：（/etc/selinux/targeted/contexts/files/file\_contexts.local）

semanage fcontext -a -t 安全标签 “目录绝对路径(/.\*)?“

修改目录的标签默认值，且标签会继承其子文件

eg：semanage fcontext -a -t httpdsyscontent\_t “/etc/httpd(/.\*)?“

### 应用程序管控：

selinux布尔值：

SELinux 是 Linux 内核的一个强制访问控制（MAC）系统，它通过一系列的策略规则来限制进程和用户对系统资源的访问。布尔值是 SELinux 策略里的可动态调整参数，借助更改这些布尔值，你能够在不重新加载整个 SELinux 策略的情况下，对系统的安全策略进行微调。

semanage boolean -l 或 getsebool -a

列出当前系统中所有的 SELinux 布尔值（Boolean）及其状态

sesetbool SELINUX布尔值名 on/off [-P]

semanage boolean -m 布尔值名 --on/--off

设置应用程序管控[设置为默认]

### 端口标签

semanage port -a -t 端口标签类型 -p 协议 端口号

添加端口标签类型

选项：

-d删除标签类型

-a添加标签类型

-t 指定标签类型

-p 指定协议

-m 修改标签类型

# 防火墙

linux内核模块的nfs\_tables (老版本为netfilter)（网络流量过滤子系统）

实现防火墙功能

管理工具：firewalld 和 iptables

管控分区域（zone）概念，处于不同区域的服务有不同的权限。

firewall-cmd --get-zones

查看所有的区域

firewall-cmd --zone=区域名 --add-service=服务名 (--permanent)

允许一条服务（永久）

eg：firewall-cmd --zone=public --add-service=http

firewall-cmd --zone=区域名 --remove-service=服务名 (--permanent)

删除允许的服务（永久）

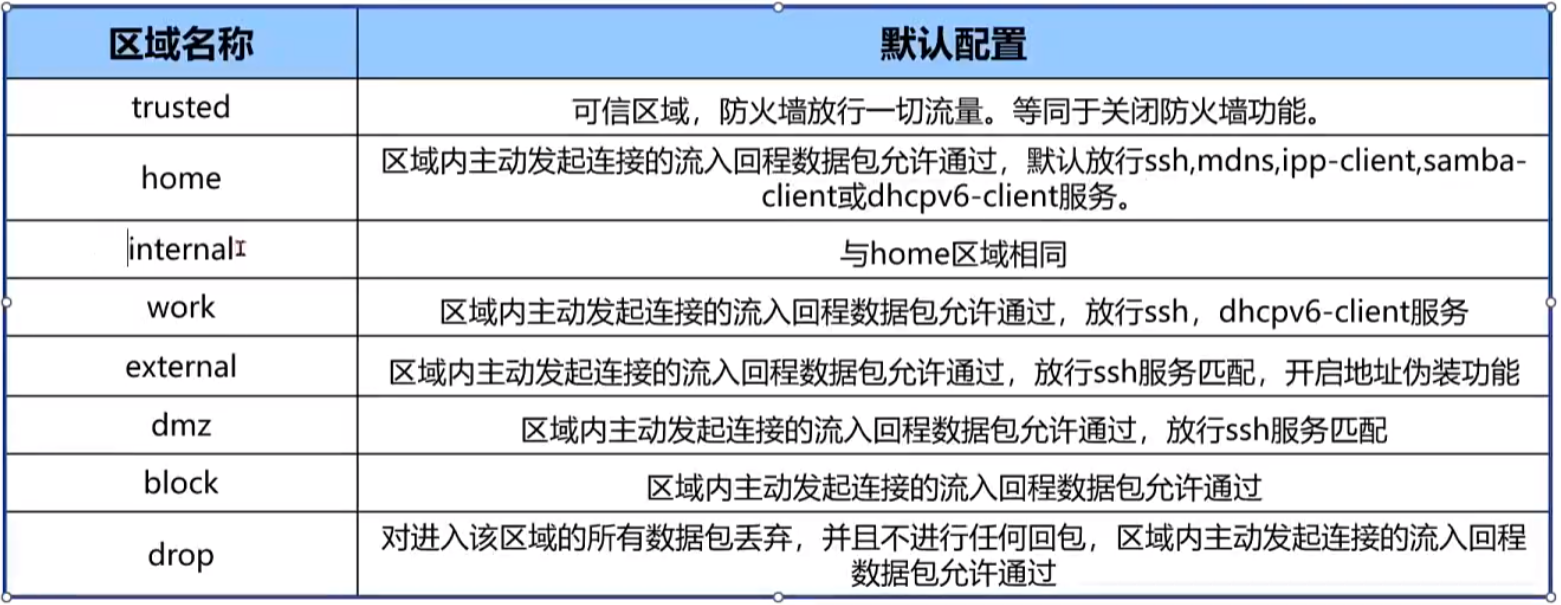
firewall-cmd –reload

重新加载配置文件

firewall-cmd --set-default-zone=区域名

将区域设置为默认区域（未指定的服务将会使用默认区域的配置）

所有的区域：



firewall-cmd --list-all --zone=区域名

查看该区域的规则等详情信息

firewall-cmd --list-all-zones

查看所有区域的信息

firewall-cmd --add-service=服务名 --zone=区域名 [--permanent]

添加过滤的服务 [永久]

firewall-cmd --add-port=端口号 --zone=区域名 [--permanent]

添加过滤的端口 [永久]

firewall-cmd --add-source=IP地址 --zone=区域名 [--permanent]

添加过滤的IP地址 [永久]

firewall-cmd --set-target=TARGAT --zone=区域名 [--permanent]

修改区域的动作 [永久]

TARGAT：

DROP 拒绝流量，且不会返回消息

ACCEPT 接受流量

REJECT 拒绝流量，会返回消息

firewall-cmd --add-forward-port=port=端口号:proto=tcp:toport=端口号

设置端口转发（default区域）

firewall-cmd --add-forward-port=port=端口号:proto=tcp:toport=端口号:toaddr=IP地址 && firewall-cmd --add-masquerade

设置远程端口转发（default区域）

firewall-cmd --reload

重置防火墙配置

# 容器

容器，可看作精简板的虚拟机，与物理机共享一个内核、cpu、硬盘等。

镜像，即一系列软件和依赖文件的集合,容器的启动需要依赖镜像

镜像仓库，即存放一系列镜像的网站

关键要素：

namespace 资源隔离

cgroup 资源控制

chroot 切换根文件系统

容器管理工具：podman（软件包：container-tools.noarch）

全局配置文件/etc/container/registries.conf

用户配置文件：~/config/container/registries.conf （优先级更高）

内容：

unqualified-search-registries = ["registry.access.redhat.com", "registry.redhat.io", "docker.io"]

配置拉取镜像的默认镜像仓库地址(IP地址或域名)

无根容器

配置文件位于用户家目录(可自行创建) ~/.config/containers/register.config

## 镜像操作

podman pull docker.io/redhat/ubi9:latest

**拉取**镜像（普通用户不能使用su – 切换然后拉取）

docker.io 镜像仓库网址（可省略）

redhat 镜像仓库名（可省略）

ubi9 镜像名

latest 标签，表示最新版本（可省略）

podman images

**查看**本地所有镜像

podman inspect 镜像名

查看本地镜像的**详情信息**

skopeo inspect docker://镜像仓库地址/镜像仓库名/镜像名:标签

查看远端镜像**详情信息**

podman search [仓库地址/]镜像名

**搜索**远端的镜像

podman rmi 镜像名

**删除**镜像

podman history 镜像名

查看制作历史

podman tag 镜像名:标签 新镜像名:新标签

创建新的镜像名和标签

podman save -o 包名 镜像名

**导出**镜像

podman load < 包名

**导入**镜像

## 容器操作

podman run [选项] [参数] 镜像名 [命令]

通过镜像创建并运行容器[创建时在容器中使用命令]

选项：

--name 设置容器名

-h 设置主机名

-it 运行容器时分配一个终端

eg：podman run -it ubi9 /bin/bash

-d 放入后台运行

-v 持久化存储 指定一个存储卷映射到容器的目录下

eg：podman run -it --name my\_container -v pv0:/opt/data ubi9 /bin/sh

--network=网络类型 指定网络类型

-p 映射端口

podman attach 容器名

进入容器（退出会影响容器状态）

podman exec -it 容器名 /bin/sh

进入容器（退出不会影响容器状态）

podman create/rm [-f] 容器名/ID

创建/删除容器[强制]

eg：podman rm -f ubi9

eg：podman rm -f `podman ps -aq`

podman ps [-a]

查看所有运行的容器[所有状态]

--filter 过滤器

eg: podman ps --filter status=exited

-q 只显示容器ID

podman stop/start/restart/pause/unpause 容器名

停止/启动/重启/挂起/恢复 容器

podman export 容器名 > 包名

将容器打包

podman import 包名

解包并运行

podman commit 容器名 镜像名:标签

将容器制作成镜像

podman cp 文件 容器名:路径

复制文件到容器内

## 容器数据持久化存储

1. 将容器数据保存到存储卷中

podman volume create 存储卷名

创建存储卷

podman volume ls

列出所有存储卷

podman volume inspect 存储卷名

显示存储卷的详细信息

podman run -it --name 别名 -v 存储卷名:容器内目录 镜像名 /bin/sh

创建并运行容器时将存储卷映射到容器的某个目录

1. 将容器数据直接保存到本地目录中

mkdir 目录名

创建目录

podman run -it --name 容器名 -v 本机目录路径:容器内目录路径:Z 镜像名 /bin/sh

创建并运行容器时将本机目录映射到容器的某个目录

**或**：

mkdir 目录名

创建目录

chcon -t containerfilet 目录名

修改目录标签

podman run -it --name 别名 -v 目录绝对路径:容器内目录 镜像名 /bin/sh

创建并运行容器时将目录映射到容器的某个目录

## 容器网络

默认的容器网卡：

桥接模式 bridge

主机网络 host

无网络 none

创建容器时默认使用桥接模式。

podman network ls

查看所有容器网卡

podman network inspect 容器网卡名

查看容器网卡详细信息

podman network create --submit IP网段/网络位 --gateway 网关IP地址 网卡名

创建容器网卡

podman network rm 网卡名

删除容器网卡

podman run -it --name 别名 --network=网络类型 镜像名 /bin/sh

创建并运行容器时指定容器网卡

podman run -it --name 别名 -p 主机端口:容器端口 镜像名 /bin/sh

创建并运行容器时将主机端口映射到容器端口

## 容器服务集成

podman generate systemd 容器名 --name --files [--new]

生成服务单元文件（需将文件移动到/usr/lib/systemd/system/目录下）

--files 生成单元文件

--name 单元文件以容器名命名

--new 每次启动服务都会运行一个新的容器

systemctl daemon-reload

重新加载配置文件

systemctl start container-容器名.service

启动服务

systemctl enable container-容器名.service

服务开机自启

**普通用户：**

生成服务单元文件（需将文件移动到/home/$USER/.config/systemd/user/目录下）

systemctl --user daemon-reload

重新加载配置文件

systemctl start container-容器名.service

启动服务

sudo loginctl enable-linger $USER

确保本用户可以开机自启服务

systemctl enable --user container-容器名.service

服务开机自启

# ansible集群自动化

主控机安装ansible即可集群管理被控机，自动使用ssh方式登录操作被控

配置文件优先级：1.由变量ANSIBLE\_CONFIG定义 2.当前目录下的ansible.cfg 3.用户家目录~./ansible.cfg 4./etc/ansible/ansible.cfg

**默认配置文件**：/etc/ansible/ansible.cfg

inventory=/etc/ansible/host #定义清单文件路径

remote\_user=用户名 #远程时使用的用户

become=True #是否要切换用户

become\_method=sudo #以sudo的方式切换

become\_user=root #切换到root

becomeaskpass=False #不使用密码

ansiblesshport=22 #定义ssh的端口

roles\_path= #角色的路径

collections\_paths= #集合的路径(ansible版本>=2.9)

**清单文件**：/etc/ansible/hosts

IP地址1 变量名=值 #设置变量(对IP地址1有效)

IP地址2 #定义管理的主机(单独)

[组名1] #定义管理的主机组

变量名=值 #设置变量(对整个组有效)

IP地址3

IP地址4

[组名2:children] #定义嵌套组

组名3

组名4

##ansible自带组‘all’，即清单文件内的所有主机

## ad-hoc模式：

**ansible-doc 模块名**

查询模块帮助文档

**ansible 节点 [选项] [参数] -m 模块 -a ‘模块参数’**

对单一节点操作

选项：

-u 接用户名，使用指定用户操作

-k 使用密码登录，默认为密钥登录(ssh-keygen)

模块：

shell 通用命令模块，使用所有命令

commend 默认命令模块，不支持’>|&’，不加’-m 模块名’ 时默认使用此模块

script 远程执行本地的脚本文件

##########前三者不支持幂等性##########

file 文件模块，文件创建删除操作

ping 测试联通性

lineinfile 修改文件的某一行

blockinffile 修改文件的多行

replace 替换文件中的某个关键词

fetch 文件拉取

copy 文件推送

tempelet 带有变量的文件推送

user 用户管理

yum和yum\_repository yum模块

service 类systemctl命令

lvg和lvol 逻辑卷

debug 显示变量的值

eg：

ansible 192.168.36.17 -u guest -k -m shell -a 'id'

ansible node1 -a ‘date‘

ansible all -m shell -a 'nmcli device show | sed -n "/IP4.ADDRESS/p"'

**setup模块**:获取被控节点的事实变量（ansible剧本默认会执行setup模块收集facts变量）



eg：

ansible all -m setup

ansible all -m setup -a "filter=ansible\_fqdn"

**魔法变量：**

主机与组

inventory\_hostname 当前执行任务的主机名（来自 inventory）

inventory\_hostname\_short 主机名的简写（去掉域名部分）

groups 所有组的字典，可取 groups['web']

group\_names 当前主机所属的组列表

hostvars 所有主机的变量字典，可跨主机访问

Play 执行

play\_hosts 当前 play 运行的所有主机

ansible\_play\_hosts\_all play 中的所有主机（不论是否执行过）

ansible\_play\_hosts 已执行过任务的主机

ansible\_limit 受 --limit 限制的主机集合

环境信息

inventory\_dir inventory 文件所在目录

inventory\_file inventory 文件的路径

playbook\_dir 当前 playbook 文件所在目录

role\_path 当前角色所在路径（仅在 role 内可用）

系统与 facts

ansible\_hostname 主机名（短名）

ansible\_fqdn 主机的完整域名

ansible\_default\_ipv4.address 默认网卡 IP

ansible\_all\_ipv4\_addresses 所有 IPv4 地址列表

ansible\_distribution 系统发行版（如 CentOS、Ubuntu）

ansible\_distribution\_major\_version 系统主版本号

执行模式

ansible\_check\_mode 是否在 --check 模式

ansible\_diff\_mode 是否在 --diff 模式

ansible\_version 当前 ansible 版本信息（字典）

## playbook模式：

**ansible-playbook 脚本文件 [syntax-check]**

运行脚本文件 [检查语法]

**脚本文件格式： #{{ 变量名 }} 引用变量**

- name: 脚本名

hosts: 主机

vars\_file: 文件 #引用变量文件

gather\_facts: false #关闭setup模块

vars:

变量名: 变量值 #设置全局变量

tasks:

- name: 任务名 #基本任务模块

模块名:

选项: 参数

选项:2 参数2

notify: 任务名3 #若该任务执行，则执行handlers任务

when: 条件 #添加条件

- block： #任务组

基本任务模块1

基本任务模块2

rescue: #若block内有任务模块失败，则执行rescue

基本任务模块

always: #无论block内是否有任务模块失败，都执行always

基本任务模块

handlers:

基本任务模块

……

## Ansible-Role角色

ansible-galaxy init 角色名

创建角色目录

目录内容：

defaults：存放默认变量

files：存放普通文件

handlers：存放handler模块

meta：作者信息

README.md：描述使用方法

tasks：存放任务模块

templates：存放jinja2模板

test：测试用

vars：存放定义的变量

调用角色：

1. 编写yaml文件

- name: roles test

hosts: all

pre\_tasks:

任务模块

roles:

- 角色1

- 角色2

......

post\_tasks:

任务模块

1. ansible-playbook xxx.yml

公共Roles仓库：<https://galaxy.ansible.com>

ansible-galaxy list

列出本机的角色

ansible-galaxy search 关键词

从公共仓库中搜索角色

ansible-galaxy role install 链接 [-p 目录]

下载公共仓库的角色[到指定的目录]

ansible-galaxy info 角色名

获取角色的信息

ansible-vault { encrypt | decrypt | rekey } 文件 [--vault-id=密码文件]

加密 | 解密 | 重置密码 [使用密码文件]

## ansible导航器

即容器化的ansible

ansible-navigator run yaml文件 [-m stdout]

使用容器来运行playbook[输出到控制台]

ansible-navigator doc 模块名

查询模块帮助（容器内）

ansible-navigator doc -l

列出所有模块（容器内）

配置导航器停止检查更新：

cat << EOF > ~/.ansible-navigator.yml

ansible-navigator:

excution-environment:

image: 镜像名

pull:

policy: mission

EOF

## ansible集合

新版本将多个功能类似的模块整合为一个集合

ansible-galaxy collection install 链接

下载集合