

# day03-linux重定向-查找-压缩-软件

---

## 3.1 linux IO重定向

---

### 3.1.1 重定向基础概述

#### 3.1.1.1 什么是重定向

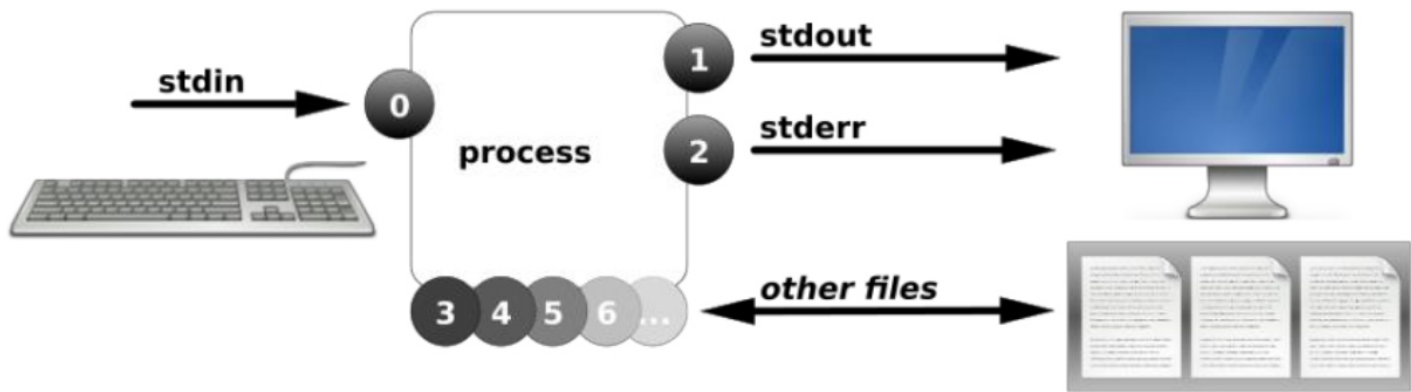
将原本要输出到屏幕中的数据信息，重新定向到某个指定的文件中，或者定向到黑洞中。

#### 3.1.1.2 为什么需要重定向

- 1.当屏幕输出的信息很重要，而且希望保存重要的信息时；
- 2.后台执行中的程序，不希望他干扰屏幕正常的输出结果时；
- 3.系统的例行命令，例如定时任务的执行结果，希望可以存下来时；
- 4.一些执行命令，我们已经知道他可能出现错误信息，想将他直接丢弃时；
- 5.错误日志与正确日志需要分别输出至不同的文件保存时；

#### 3.1.1.3 标准输入与输出

- 当进程操作一个文件时：
  - 1.首先程序是无法直接访问硬件，需要借助内核来访问文件；
  - 2.而内核 `kernel` 需要利用文件描述符 `(file descriptor)` 来访问文件。[文件描述符 百度百科](#)
- 总结：进程使用文件描述符来管理打开的文件对应关系；
- 通常程序访问一个文件至少会打开三个标准文件，分别是标准输入、标准输出、错误输出。
- 进程将从标准输入中得到数据，将正常输出打印至屏幕终端，将错误的输出信息也打印至屏幕终端。



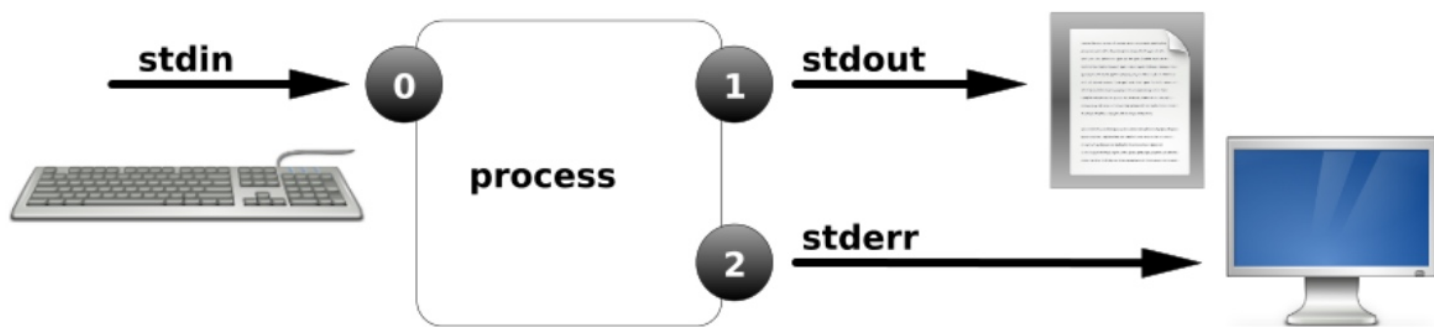
名称	文件描述符	作用
标准输入（STDIN）	0	默认是键盘，也可以是文件或其他命令的输出。
标准输出（STDOUT）	1	默认输出到屏幕。
错误输出（STDERR）	2	默认输出到屏幕。
文件名称（filename）	3+	

### 3.1.2 输出重定向案例

- 输出重定向，改变输出内容的位置。输出重定向有如下几种方式，如表格所示

类型	操作符	用途
标准覆盖输出重定向	>	将程序输出的正确结果输出到指定的文件中,会覆盖文件原有的内容
标准追加输出重定向	>>	将程序输出的正确结果以追加的方式输出到指定文件，不会覆盖原有文件
错误覆盖输出重定向	2>	将程序的错误结果输出到执行的文件中，会覆盖文件原有的内容
错误追加输出重定向	2>>	将程序输出的错误结果以追加的方式输出到指定文件，不会覆盖原有文件

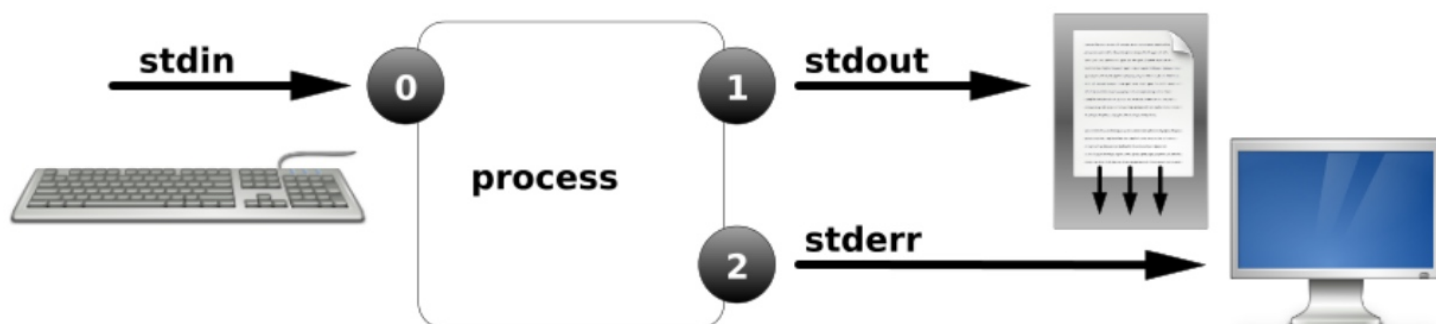
#### 3.1.2.1 案例1-标准输出重定向



- 标准输出重定向示例
  - 1.如果文件不存在则创建
  - 2.如果文件存在则清空内容

```
[root@xuliangwei ~]# > edu.txt
[root@xuliangwei ~]# ifconfig eth0 > edu.txt
```

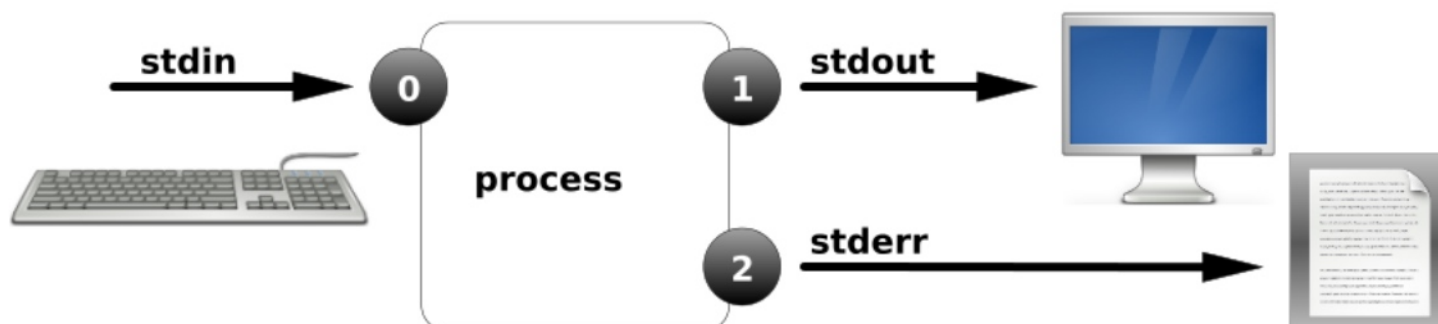
### 3.1.2.2 案例2-标准追加输出重定向



- 标准追加输出重定向示例
  - 如果文件不存在则创建
  - 2.如果文件存在则在文件尾部添加内容

```
[root@xuliangwei ~]# echo "Hello Students" >> if
```

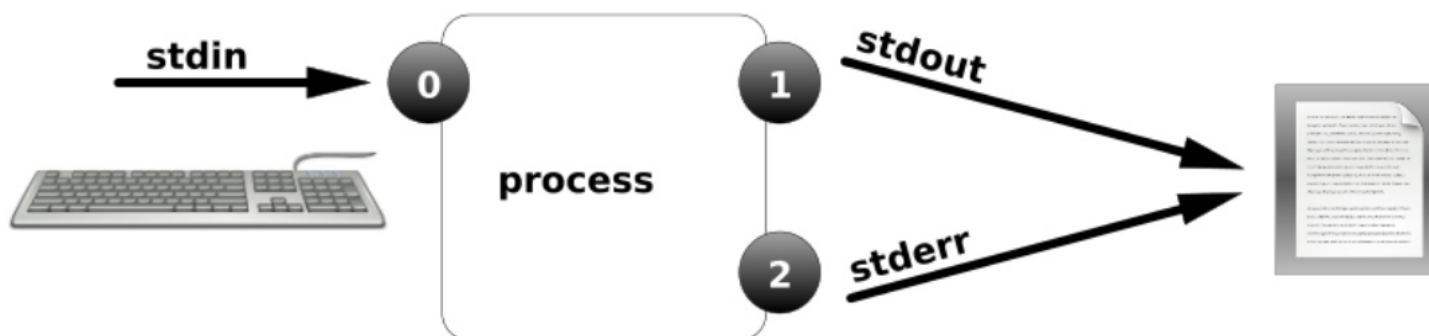
### 3.1.2.3 案例3-错误输出重定向



- 标准错误输出重定向
  - 1.正确输出及错误输出至相同文件
  - 2.正确输出及错误输出至不同的文件

```
[xuliangwei@xuliangwei ~]$ find /etc -name "*.conf" 1>ok 2>ok
[xuliangwei@xuliangwei ~]$ find /etc -name "*.conf" 1>ok 2>err
```

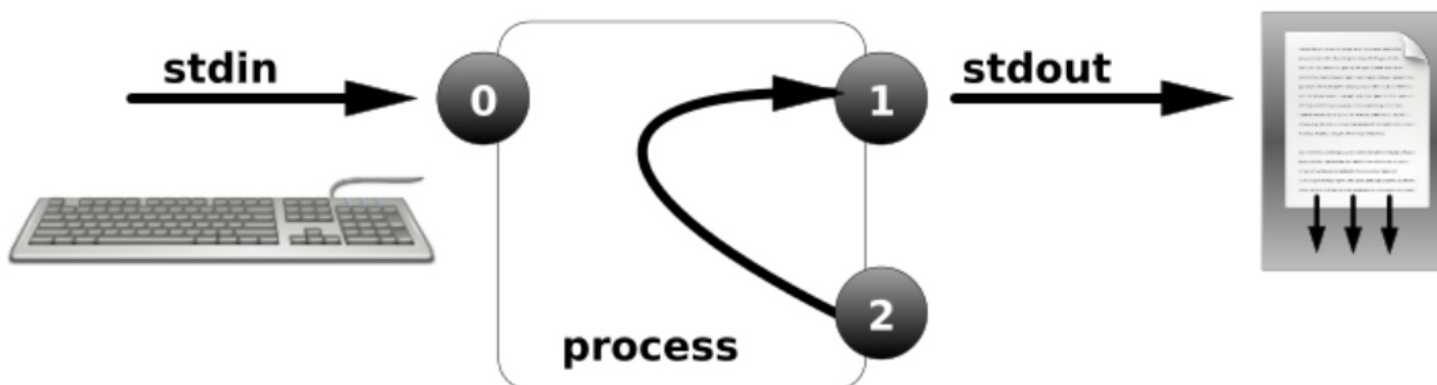
### 3.1.2.4 案例4-混合和输出重定向



- 混合输出重定向
  - 1.将正确输出和错误输出混合至同一文件
  - 2.将两个文件内容组合为一个文件

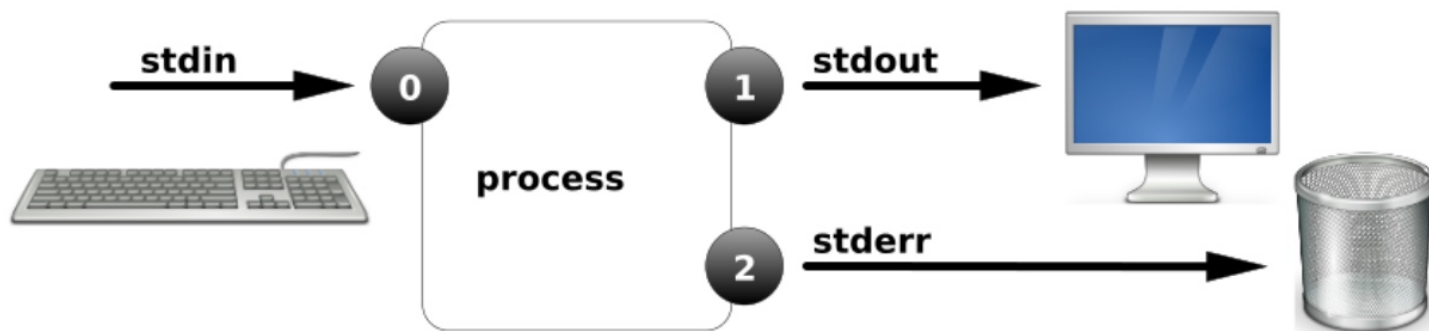
```
[xuliangwei@xuliangwei ~]$ find /etc -name "*.conf" &>ab
[xuliangwei@xuliangwei ~]$ cat a b > c
```

- 正确和错误都输入到相同位置



```
[root@xuliangwei ~]# ls /root /error >ab 2>&1
```

### 3.1.2.5 案例5-将内容输出至黑洞



- 将内容输出至黑洞设备 `/dev/null`

```
[root@xuliangwei ~]# ls /root /error >ab 2>/dev/null
[root@xuliangwei ~]# ls /root /error >ab &>/dev/null
```

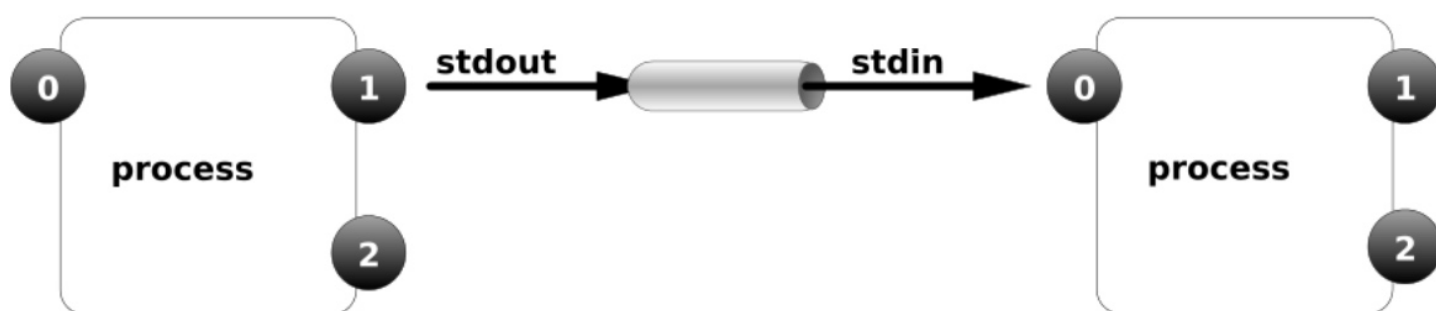
## 3.1.3 进程管道技术

### 3.1.3.1 什么是管道

- 管道操作符号 `"|"`，主要用来连接左右两个命令，将左侧的命令的标准输出，交给右侧命令的标准输入
- 注意事项：无法传递标准错误输出至后者命令

### 3.1.3.2 管道流程示意图

- 格式: `cmd1 | cmd2 [...|cmdn]`



### 3.1.3.3 管道使用案例

#### 3.1.3.3.1 案例1

- 案例1: 将 `/etc/passwd` 中的用户按 `UID` 大小排序

```
[root@xuliangwei ~]# sort -t":" -k3 -n /etc/passwd
[root@xuliangwei ~]# sort -t":" -k3 -n /etc/passwd -r
```

```
[root@xuliangwei ~]# sort -t":" -k3 -n /etc/passwd | head
```

### 3.1.3.3.2 案例2

- 案例2: 统计当前 `/etc/passwd` 中用户使用的 `shell` 类型

# 思路: 取出第七列(shell) | 排序(把相同归类)| 去重

```
[root@xuliangwei ~]# awk -F: '{print $7}' /etc/passwd
[root@xuliangwei ~]# awk -F: '{print $7}' /etc/passwd | sort
[root@xuliangwei ~]# awk -F: '{print $7}' /etc/passwd | sort | uniq
[root@xuliangwei ~]# awk -F: '{print $7}' /etc/passwd | sort | uniq -c
```

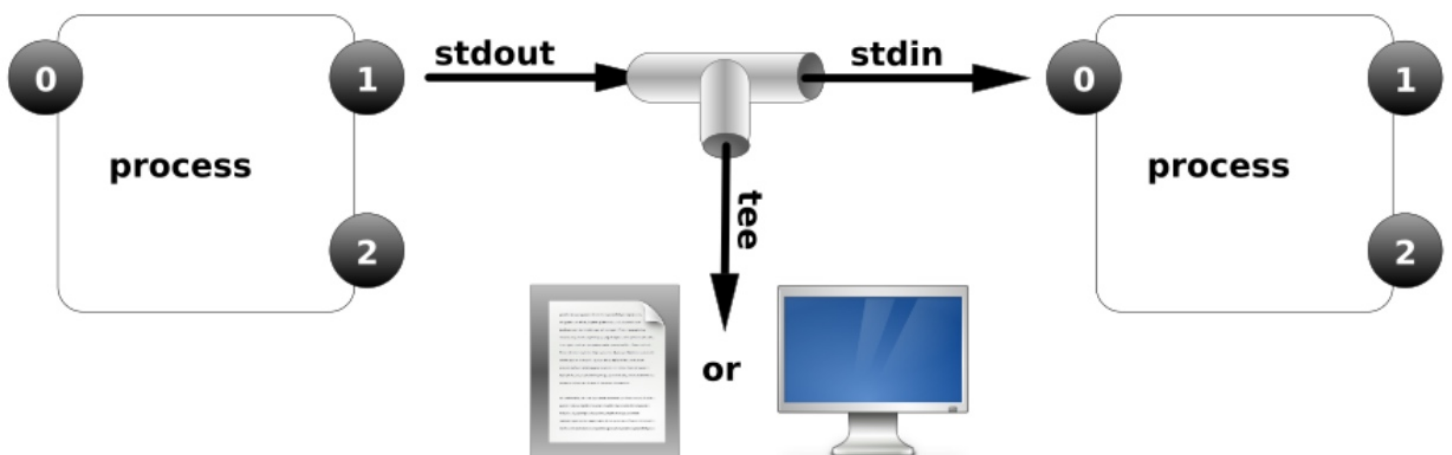
### 3.1.3.3.2 案例3

- 案例3: 打印当前所有 IP

```
[root@xuliangwei ~]# ip addr | grep 'inet ' | awk '{print $2}' | awk -F"/" '{print $1}'
127.0.0.1
192.168.69.112
```

## 3.1.4 tee与xargs

### 3.1.4.1 管道中使用tee



#选项: -a追加

```
[root@xuliangwei ~]# ip addr | grep 'inet ' | tee ip.txt | awk -F"/" '{print $1}' | awk '{print $2}'
127.0.0.1
10.0.0.100
```

```
[root@xuliangwei ~]# cat ip.txt
inet 127.0.0.1/8 scope host lo
inet 10.0.0.100/24 brd 192.168.69.255 scope global ens32
```

重定向与 tee 有他们在使用过程中有什么区别

```
[root@xuliangwei ~]# date > date.txt      #直接将内容写入date.txt文件中
[root@xuliangwei ~]# date |tee date.txt    #命令执行会输出至屏幕，但会同时保存一份至date.txt文件中
```

### 3.1.4.2 管道中使用xargs

- xargs参数传递，主要让一些不支持管道的命令可以使用管道技术

```
# which cat|xargs ls -l
# ls |xargs rm -fv
```

## 3.2 linux文件查找

### 3.2.1 find查找概述

#### 3.2.1.1 为什么需要查找

- 1.很多时候我们可能会忘了文件所在的位置，此时就需要通过 `find` 来查找；
- 2.有时候需要通过内容查找到对应的文件，此时就需要通过 `find` 来查找；

#### 3.2.1.2 为什么是find

- 因为 `find` 命令可以根据不同的条件来进行查找文件
- 比如：
  - 文件名称、
  - 文件大小、
  - 文件时间、
  - 属主属组、
  - 权限等等、
- 可以通过如上几种方式查找文件，从而实现精准定位

#### 3.2.1.3 find命令语法

命令	路径	选项	表达式	动作
find	[path...]	[options]	[expression]	[action]
查找	地区	妹纸	18-25岁	???

## 3.2.2 find查找示例

### 3.2.2.1 find基于名称查找

#1. 创建文件

```
touch /etc/sysconfig/network-scripts/{ifcfg-eth1,IFCFG-ETH1}
```

#2. 查找/etc目录下包含ifcfg-eth0名称的文件

```
[root@xuliangwei ~]# find /etc -name "ifcfg-eth1"
```

#3. -i 忽略大小写

```
[root@xuliangwei ~]# find /etc -iname "ifcfg-eth1"
```

#查找/etc目录下包含ifcfg-eth名称所有文件

```
[root@xuliangwei ~]# find /etc/ -name "ifcfg-eth*"
```

```
[root@xuliangwei ~]# find /etc -iname "ifcfg-eth*"
```

### 3.2.2.2 find基于大小查找

#1. 查找大于5M的文件

```
[root@xuliangwei ~]# find /etc -size +5M
```

#2. 查找等于5M的文件

```
[root@xuliangwei ~]# find /etc -size 5M
```

#3. 查找小于5M的文件

```
[root@xuliangwei ~]# find /etc -size -5M
```

### 3.2.2.3 find基于类型查找

# f 文件

```
[root@xuliangwei ~]# find /dev -type f
```

# d 目录

```
[root@xuliangwei ~]# find /dev -type d
```

# l 链接

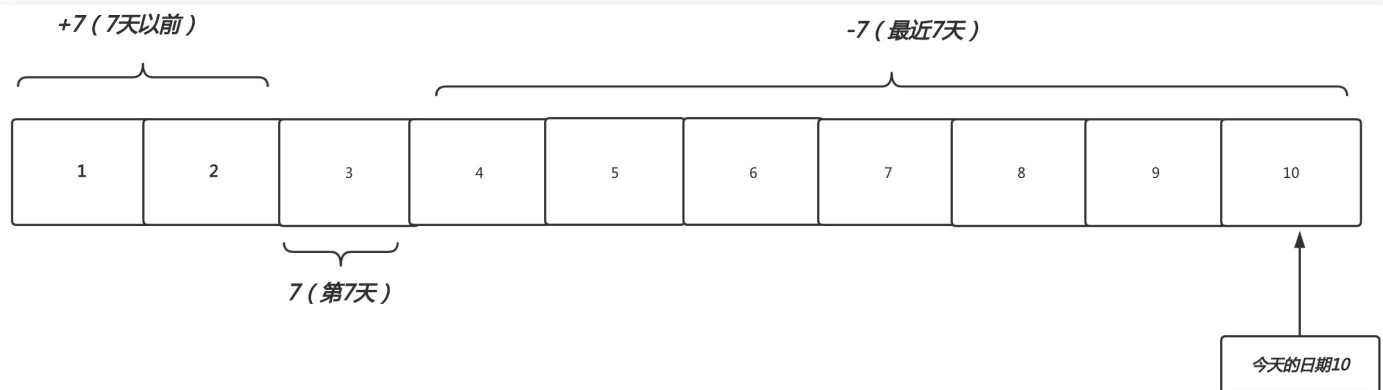
```
[root@xuliangwei ~]# find /dev -type l
```

# b 块设备



```
[root@xuliangwei ~]# find /dev -type b
# c 字符设备
[root@xuliangwei ~]# find /dev -type c
# s 套接字
[root@xuliangwei ~]# find /dev -type s
# p 管道文件
[root@xuliangwei ~]# find /dev -type p
```

### 3.2.2.4 find基于时间查找



#1. 创建测试文件(后期shell会讲)

```
[root@xuliangwei ~]# for i in {01..10};do date -s 201904$i && touch file-$i;done
```

#2. 查找7天以前的文件(不会打印当天的文件)

```
[root@xuliangwei ~]# find ./ -iname "file-*" -mtime +7
```

#3. 查找最近7天的文件, 不建议使用(会打印当天的文件)

```
[root@xuliangwei ~]# find ./ -iname "file-*" -mtime -7
```

#4. 查找第7天文件(不会打印当天的文件)

```
[root@xuliangwei ~]# find ./ -iname "file-*" -mtime 7
```

#面试题: 查找/var/log下所有以.log结尾的文件, 并保留最近7天的Log文件。

```
[root@xuliangwei ~]# find /var/log -type f -name "*.log" -mtime +7 -delete
```

### 3.2.2.5 find基于用户查找

#查找属主是jack

```
[root@xuliangwei ~]# find /home -user jack
```

#查找属组是admin

```
[root@xuliangwei ~]# find /home -group admin
```

#查找属主是jack, 属组是admin

```
[root@xuliangwei ~]# find /home -user jack -group admin
```

```
#查找属主是jack, 并且属组是admin
[root@xuliangwei ~]# find /home -user jack -a -group admin
#查找属主是jack, 或者属组是admin
[root@xuliangwei ~]# find /home -user jack -o -group admin
#查找没有属主
[root@xuliangwei ~]# find /home -nouser
#查找没有属组
[root@xuliangwei ~]# find /home -nogroup
#查找没有属主或属组
[root@xuliangwei ~]# find /home -nouser -o -nogroup
```

### 3.2.2.6 find基于权限查找

```
# 精确
[root@web ~]# find /root -type f -perm 644 -ls

#包含
[root@web ~]# find /root -type f -perm -644 -ls

# 特殊权限
[root@web ~]# find /usr/bin/ /usr/sbin/ -type f -perm -4000 -ls
[root@web ~]# find /usr/bin/ /usr/sbin/ -type f -perm -2000 -ls
[root@web ~]# find /usr/bin/ /usr/sbin/ -type f -perm -1000 -ls
```

### 3.2.2.7 find逻辑运算符

符号	作用
-a	与
-o	或
-not !	非

```
#1. 查找当前目录下, 属主不是root的所有文件
[root@xuliangwei ~]# find . -not -user root
[root@xuliangwei ~]# find . ! -user root

#2. 查找当前目录下, 属主属于hdfs, 并且大小大于1k的文件
[root@xuliangwei ~]# find . -type f -a -user hdfs -a -size +1k

#3. 查找当前目录下的属主为root或者以xml结尾的普通文件
[root@xuliangwei ~]# find . -type f -a \( -user hdfs -o -name '*.xml' \)
```

### 3.2.3 find动作处理

- 查找到一个文件后，需要对文件进行如何处理 `find` 的默认动作是 `-print`

动作	含义
-print	打印查找到的内容(默认)
-ls	以长格式显示的方式打印查找到的内容
-delete	删除查找到的文件(仅能删除空目录)
-ok	后面跟自定义 shell 命令(会提示是否操作)
-exec	后面跟自定义 shell 命令(标准写法 <code>-exec \;</code> )

#### 3.2.3.1 find结合exec

```
#5. 使用-exec实现文件拷贝和文件删除。  
[root@xuliangwei ~]# find /etc -name "ifcfg*" -exec cp -rvf {} /tmp \  
[root@xuliangwei ~]# find /etc -name "ifcfg*" -exec rm -f {} \;
```

#### 3.2.3.2 find结合xargs

```
# xargs将前者命令查找到的文件作为一个整体传递后者命令的输入  
[root@xuliangwei ~]# touch file.txt  
[root@xuliangwei ~]# find . -name "file.txt" |xargs rm -f
```

#### 3.2.3.3 find结合grep

```
[root@xuliangwei ~]# find ./ -type f | xargs grep -R "oldxu.com"
```

## 3.3 linux文件压缩

### 3.3.1 什么是文件压缩

- 将多个文件或目录合并成为一个特殊的文件。比如: 搬家...脑补画面 img.

### 3.3.2 为什么要对文件进行压缩

- 当我们在传输大量的文件时，通常都会选择将该文件进行压缩，然后在进行传输。
  - 首先：压缩后的文件会比压缩前的文件小。一个28G的文件夹压缩后能达到6G
  - 其次：多个文件传输很慢，但单个文件传输会很快，同时还能节省网络的消耗。
  - （比如：搬家时，单行李往外拿和打包后往外拿？）

### 3.3.3 Linux下常见压缩包类型

格式	压缩工具
.zip	zip压缩工具
.gz	gzip压缩工具，只能压缩文件，会删除原文件(通常配合tar使用)
.bz2	bzip2压缩工具，只能压缩文件，会删除原文件(通常配合tar使用)
.tar.gz	先使用tar命令归档打包，然后使用gzip压缩
.tar.bz2	先使用tar命令归档打包，然后使用bzip压缩

### 3.3.4 文件打包与压缩-gzip

使用gzip方式进行压缩文件

```
[root@xuliangwei ~]# yum install gzip -y
[root@xuliangwei ~]# gzip file          #对文件进行压缩
[root@xuliangwei ~]# zcat file.gz      #查看gz压缩后的文件
[root@xuliangwei ~]# gzip -d file.gz  #解压gzip的压缩包

#使用场景: 当需要让某个文件不生效时
[root@xuliangwei ~]# gzip CentOS-Vault.repo --> CentOS-Vault.repo.gz
[root@xuliangwei ~]# zcat CentOS-Vault.repo.gz --> 查看不想解压的压缩包文件内容
```

### 3.3.5 文件打包与压缩-zip

使用zip命令可以对文件进行压缩打包，解压则需要使用unzip命令

```
# 默认情况下没有zip和unzip工具，需要进行安装
[root@xuliangwei ~]# yum install zip unzip -y

#1. 压缩文件为zip包
```

```
[root@xuliangwei ~]# zip filename.zip filename
```

#2. 压缩目录为zip包

```
[root@xuliangwei ~]# zip -r dir.zip dir/
```

#3. 查看zip压缩包是否是完整的

```
[root@xuliangwei ~]# zip -T filename.zip
test of filename.zip OK
```

#4. 不解压压缩查看压缩包中的内容

```
[root@xuliangwei ~]# unzip -l filename.zip
[root@xuliangwei ~]# unzip -t filename.zip
```

#5. 解压zip文件包, 默认解压至当前目录

```
[root@xuliangwei ~]# unzip filename.zip
```

#6. 解压zip内容至/opt目录

```
[root@xuliangwei ~]# unzip filename.zip -d /opt/
```

## 3.3.6 文件打包与压缩-tar

tar是linux下最常用的压缩与解压缩, 支持文件和目录的压缩归档

#语法: tar [-zjxcvfpP] filename

c #创建新的归档文件

x #对归档文件解包

t #列出归档文件里的文件列表

v #输出命令的归档或解包的过程

f #指定包文件名, 多参数f写最后

z #使用gzip压缩归档后的文件(.tar.gz)

j #使用bzip2压缩归档后的文件(.tar.bz2)

J #使用xz压缩归档后的文件(tar.xz)

C #指定解压目录位置

X #排除多个文件(写入需要排除的文件名称)

h #打包软链接

--hard-dereference #打包硬链接

--exclude #在打包的时候写入需要排除文件或目录

#常用打包与压缩组合

czf #打包tar.gz格式

cjf #打包tar.bz格式

cJf #打包tar.xz格式

zxf #解压tar.gz格式

```
jxf    #解压tar.bz格式
xf     #自动选择解压模式
tf     #查看压缩包内容
```

## 1. 将文件或目录进行打包压缩

```
#1. 以gzip归档方式打包并压缩
tar czf test.tar.gz test/ test2/

#2. 以bz2方式压缩
tar cjf test.tar.bz2 dir.txt dir/

#3. 打包链接文件, 打包链接文件的真实文件
[root@xuliangwei ~]# cd /
[root@xuliangwei /]# tar czfh local.tar.gz etc/rc.local

#4. 打包/tmp下所有文件
[root@xuliangwei ~]# cd /
[root@xuliangwei /]# find tmp/ -type f | xargs tar czf tmp.tar.gz

#5. 打包/tmp下所有文件
[root@xuliangwei /]# tar czf tmp.tar.gz $(find /tmp/ -type f)
```

## 2. 排除文件, 并打包压缩

```
#1. 排除单个文件
[root@xuliangwei /]# tar czf etc.tar.gz --exclude=etc/services etc/

#2. 排除多个文件
[root@xuliangwei /]# tar czf etc.tar.gz --exclude=etc/services --exclude=etc/rc.local etc/

#3. 将需要排除的文件写入文件中
[root@xuliangwei /]# cat paichu.list
etc/services
etc/rc.local
etc/rc.d/rc.local
#指定需要排除的文件列表, 最后进行打包压缩
[root@xuliangwei /]# tar czfX etc.tar.gz paichu.list etc/
```

## 3. 查看压缩文件

```
#查看压缩包内容和解压
[root@xuliangwei /]# tar tf test.tar.gz
```

## 4. 解压缩文件

#1. 解压至当前目录

```
[root@xuliangwei /]# tar xf test.tar.gz
```

#2. 将解压内容存储至指定的/tmp目录

```
[root@student ~]# tar xf /etc/local.tar.gz -C /tmp
```

## 3.4 linux软件包管理

### 3.4.1 RPM基本概述

#### 3.4.1.1 什么是rpm

RPM 全称 RedHat Package Manager 缩写，由红帽开发用于软件包的安装、升级、卸载与查询。

#### 3.4.1.2 rpm包名组成部分

RPM 包命名以-将软件分成了若干部分 bash-4.2.46-28.el7.x86\_64.rpm



#### 3.4.1.3 如何获取rpm包

在我们刚开始学习 rpm 包，建议先从本地镜像中获取 rpm 但实际生产环境中大多数是通过联网方式获取 rpm 包，或者搭建企业私有包管理仓库平台。



## 先学会，你懂的

### 3.4.1.4 其他类型的安装包

在 Linux 中除了 rpm 格式类型的包，还存在一些其他类型的软件包。

分类	安装	版本
rpm包	预先编译打包,安装简单	软件版本偏低
源码包	手动编译打包,安装繁琐	软件版本随意
二进制包	解压即可使用, 安装简单	不能修改源码

## 3.4.2 RPM包管理命令

### 3.4.2.1 rpm安装软件包

- -i: 安装软件包
- -v: 显示安装过程
- -h: 显示安装进度条

1.使用 rpm 命令安装本地路径下软件包

```
[root@xuliangwei ~]# rpm -ivh /mnt/Packages/tree-1.6.0-10.el7.x86_64.rpm
[root@xuliangwei ~]# rpm -ivh /mnt/Packages/vsftpd-3.0.2-22.el7.x86_64.rpm
```

2.使用 rpm 命令安装互联网上的软件包

```
[root@xuliangwei ~]# rpm -ivh https://mirrors.aliyun.com/zabbix/zabbix/3.0/rhel/7/x86_64/zabbix-agent-3.0.9-1.el7.x86_64.rpm
```



### 3.4.2.2 rpm依赖包安装

包依赖是指 A-->依赖-->B， B-->依赖-->C， 而C-->依赖-->A。当我们需要安装的 rpm 类型包出现了依赖关系应该如何处理，比如安装 samba 软件包。

```
[root@xuliangwei ~]# rpm -ivh /mnt/Packages/samba-4.8.3-4.el7.x86_64.rpm
error: Failed dependencies:
  libxattr-tdb-samba4.so()(64bit) is needed by samba-0:4.8.3-4.el7.x86_64
  libxattr-tdb-samba4.so(SAMBA_4.8.3)(64bit) is needed by samba-0:4.8.3-4.el7.x86_64
  samba-common-tools = 4.8.3-4.el7 is needed by samba-0:4.8.3-4.el7.x86_64
  samba-libs = 4.8.3-4.el7 is needed by samba-0:4.8.3-4.el7.x86_64
```

#### 1.尝试安装依赖包 samba-common-tools

```
[root@xuliangwei ~]# rpm -ivh /mnt/Packages/samba-common-tools-4.8.3-4.el7.x86_64.rpm
```

#### 2.尝试安装依赖包 samba-libs

```
[root@xuliangwei ~]# rpm -ivh /mnt/Packages/
[root@xuliangwei ~]# rpm -ivh /mnt/Packages/samba-libs-4.8.3-4.el7.x86_64.rpm
```

#### 3.尝试安装依赖包 samba-common-tools

```
[root@xuliangwei ~]# rpm -ivh /mnt/Packages/samba-common-tools-4.8.3-4.el7.x86_64.rpm
```

#### 4.最后尝试安装 samba 主程序包

```
[root@xuliangwei ~]# rpm -ivh /mnt/Packages/samba-4.8.3-4.el7.x86_64.rpm
```

PS: 由于rpm工具安装rpm包依赖关系太强，所以通常我们都是使用 yum 来解决

### 3.4.2.3 rpm升级软件包

下载 zabbix-agent 软件包，分别下载 3.0 低版本、然后下载 4.2 高版来进行测试与实验。

```
# wget https://mirrors.aliyun.com/zabbix/zabbix/3.0/rhel/7/x86_64/zabbix-agent-3.0
```

```
.9-1.el7.x86_64.rpm
# wget https://mirrors.aliyun.com/zabbix/zabbix/4.2/rhel/7/x86_64/zabbix-agent-4.2
.0-1.el7.x86_64.rpm
```

### 1.先安装 `zabbix-agent-3.0` 低版本

```
[root@www.xuliangwei.com ~]# rpm -ivh zabbix-agent-3.0.9-1.el7.x86_64.rpm
```

### 2.尝试使用 `rpm -ivh` 安装 `zabbix-agent-4.2` 高版本（会出现报错）

```
[root@www.xuliangwei.com ~]# rpm -ivh zabbix-agent-4.2.0-1.el7.x86_64.rpm
```

### 3.使用 `rpm -Uvh` 升级 `zabbix-agent` 至 `4.2` 版本。（完美解决）

```
[root@www.xuliangwei.com ~]# rpm -Uvh zabbix-agent-4.2.0-1.el7.x86_64.rpm
```

## 3.4.2.4 rpm卸载软件包

如果需要卸载 `rpm` 包，可以先查看该包是否存系统中，然后在进行卸载操作。

### 1.使用 `rpm -q` 查询软件包是否存在系统

```
[root@www.xuliangwei.com ~]# rpm -q zsh
```

### 2.使用 `rpm -e` 卸载软件包

```
[root@www.xuliangwei.com ~]# rpm -e zsh
```

## 3.4.2.5 rpm查询软件包

选项	描述
<code>rpm -q</code>	查看指定软件包是否安装
<code>rpm -qa</code>	查看系统中已安装的所有RPM软件包列表
<code>rpm -qi</code>	查看指定软件的详细信息
<code>rpm -ql</code>	查询指定软件包所安装的目录、文件列表

rpm -qc	查询指定软件包的配置文件
rpm -qf	查询文件或目录属于哪个RPM软件

1.查询 `vsftpd` 这个 `rpm` 包是否安装

```
[root@xuliangwei ~]# rpm -q vsftpd
```

2.模糊查找系统已安装的 `rpm` 包

```
[root@xuliangwei ~]# rpm -qa |grep ftp
```

3.查询 `vsftpd` 软件包相关信息

```
[root@xuliangwei ~]# rpm -qi vsftpd
```

4.查询 `vsftpd` 软件包所安装后在系统中生成的文件路径

```
[root@xuliangwei ~]# rpm -ql vsftpd
```

5.查询 `vsftpd` 软件包的主配置文件

```
[root@xuliangwei ~]# rpm -qc vsftpd
```

6.查询配置文件或系统命令是由哪个 `rpm` 包提供

```
[root@xuliangwei ~]# rpm -qf /etc/vsftpd/vsftpd.conf  
[root@xuliangwei ~]# rpm -qf /usr/sbin/vsftpd
```

7.查询未安装的 `rpm` 包会产生哪些文件

```
[root@xuliangwei ~]# rpm -qlp /mnt/Packages/samba-3.6.23-41.el6.x86_64.rpm
```

### 3.4.2.6 rpm包管理小结

- 1.如何查询 `util-linux` 软件包安装了哪些文件?
- 2.如何查询 `mkdir` 命令是由哪个 `RPM` 软件包安装的?

- 3.安装 .rpm 软件包时, -i、-U、选项有何区别?

## 3.4.3 YUM基本介绍

### 3.4.3.1 什么是YUM

yum/dnf 是 RedHat 及 CentOS 系统中的软件包管理器。它能够通过互联网下载 .rpm 格式包进行安装, 并能自动处理其依赖间关系, 无须繁琐地一次次下载安装。

### 3.4.3.2 什么是YUM源

要使用 yum 命令工具安装更新软件, 需要有一个包含各种 rpm 软件包的仓库, 这个软件仓库我们一般称为 yum 源。当然这个源可以是本地仓库、也可以是网络仓库。如图所示:

client --ftp/http/file-> yum地址 --->yum仓库 (rpm包集合)

### 3.4.3.3 YUM配置文件

CentOS8 的配置文件

```
[root@e84356b681bf etc]# cat /etc/yum.conf
[main]
gpgcheck=1                                # 检查来源是否合法, 需要有制作者的公钥信息
installonly_limit=3                       # 同时可以安装5个软件包、最小为2, 设置为0或者1则不
限制
clean_requirements_on_remove=True         # 删除包时, 是否将不再使用的包删除
best=True                                # 升级时, 自动选择安装最新版, 即使缺少包的依赖
skip_if_unavailable=False
```

CentOS7 的配置文件

```
[root@xuliangwei ~]# vim /etc/yum.cnf
cachedir=/var/cache/yum/$basearch/$releasever # 缓存目录
keepcache=0                                    # 缓存软件包, 1启动 0 关闭
debuglevel=2                                  # 调试级别
logfile=/var/log/yum.log                     # 日志记录位置
exactarch=1                                  # 检查平台是否兼容
obsoletes=1                                  # 检查包是否废弃
gpgcheck=1                                    # 检查来源是否合法, 需要有制作者的公钥信息
plugins=1
installonly_limit=5                          # 同时可以安装5个软件包、最小为2, 设置为0或者1则不限制

# metadata_expire=90m    #每小时手动检查元数据
```

```
# in /etc/yum.repos.d #包含repos.d目录中的.repo文件
```

### 3.4.3.4 配置YUM源示例

系统默认的源是国外提供，需要替换为国内的源

#### 1.配置阿里 yum 源

```
[root@www.xuliangwei.com ~]# wget -O /etc/yum.repos.d/CentOS-Base.repo \
http://mirrors.aliyun.com/repo/Centos-7.repo
```

#### 2.配置第三方 yum 源 (EPEL)

```
[root@www.xuliangwei.com ~]# wget -O /etc/yum.repos.d/epel.repo \
http://mirrors.aliyun.com/repo/epel-7.repo
```

#### 3. Nginx 官方源，后期在学习 Nginx 时需要使用官方的 yum 源来安装软件

```
[root@www.xuliangwei.com ~]# vim /etc/yum.repos.d/nginx.repo
[nginx]
name=nginx repo
baseurl=http://nginx.org/packages/centos/7/$basearch/
gpgcheck=0
enabled=1
```

- PS:源查找方式基本一致,Docker、Nginx、openstack、ELKStack

## 3.4.4 YUM日常操作

### 3.4.4.1 yum查询软件包

#### 1.使用 dnf/yum list 查询所有仓库中的所有软件包

```
[root@www.xuliangwei.com ~]# yum list
[root@www.xuliangwei.com ~]# yum list|grep ftp
```

#### 2.使用 dnf/yum list installed 查询所有已安装至系统中的软件包

```
[root@www.xuliangwei.com ~]# dnf list installed
```

### 3.使用 `dnf/yum provides` 查询系统命令来自于哪个软件包

```
[root@www.xuliangwei.com ~]# rpm -qf $(which cd)
bash-4.2.46-31.el7.x86_64
# PS: rpm需要知道命令的绝对路径, 如果不存在该命令是无法查找到该命令所属的软件包

[root@www.xuliangwei.com ~]# yum provides cd
[root@www.xuliangwei.com ~]# yum provides sl
```

## 3.4.4.2 yum安装软件包

### 1.使用 `dnf/yum install` 通过仓库获取软件包进行安装

```
# 交互, 麻烦
[root@www.xuliangwei.com ~]# yum install vsftpd

# 非交互
[root@www.xuliangwei.com ~]# yum install vsftpd -y
```

### 2.使用 `dnf/yum localinstall` 安装本地的rpm包, 如果rpm包存在依赖, 会自动查找当前系统上已有的仓库解决依赖关系

```
[root@www.xuliangwei.com ~]# yum install https://mirrors.aliyun.com/centos/7.6.1810/os/x86_64/Packages/samba-4.8.3-4.el7.x86_64.rpm

#yum localinstall 直接安装本的rpm包, 会自动查找当前系统上已有的仓库解决依赖关系
[root@www.xuliangwei.com ~]# yum localinstall samba-4.8.3-4.el7.x86_64.rpm -y
```

## 3.4.4.3 yum重装软件包

当我们安装好服务后, 如果不小心删除了服务的配置文件, 此时可以通过重装的方式修复。

### 1.首先删除 `vsftpd` 配置主文件

```
[root@www.xuliangwei.com ~]# rpm -qc vsftpd
[root@www.xuliangwei.com ~]# rm -f /etc/vsftpd/vsftpd.conf
```

### 2.使用 `dnf/yum reinstall` 对软件进行重新安装

```
[root@www.xuliangwei.com ~]# yum reinstall vsftpd
```

3.检查 `vsftpd` 服务配置文件是否恢复，以及软件是否能正常使用。

```
[root@www.xuliangwei.com ~]# rpm -qc vsftpd
/etc/logrotate.d/vsftpd
/etc/pam.d/vsftpd
/etc/vsftpd/ftpusers
/etc/vsftpd/user_list
/etc/vsftpd/vsftpd.conf
```

### 3.4.4.4 yum更新软件包

#1. 对比Linux已安装的软件和yum仓库中的软件，有哪些需要升级

```
[root@www.xuliangwei.com ~]# yum check-update
```

#2. 更新[acl](#) 软件

```
[root@www.xuliangwei.com ~]# yum update acl -y
```

#3. 更新整个系统所有的软件，包括内核（通常刚装完系统会进行执行）

```
[root@www.xuliangwei.com ~]# yum update -y
```

### 3.4.4.5 yum删除软件包

```
[root@www.xuliangwei.com ~]# yum install samba -y
```

```
[root@www.xuliangwei.com ~]# yum remove samba -y
```

### 3.4.4.6 yum管理组包

1.使用 `dnf/yum groups install` 安装一整个组的软件

```
[root@www.xuliangwei.com ~]# yum groups list
[root@xuliangwei ~]# yum groups install Development tools \
Compatibility libraries \
Base Debugging Tools
```

2.使用 `dnf/yum groups remove` 删除包组

```
[root@www.xuliangwei.com ~]# yum groups remove -y Base
```

### 3.4.4.7 yum管理仓库

#### 1.列出 `dnf/yum repolist` 源可用的软件仓库

```
[root@www.xuliangwei.com ~]# yum repolist
[root@www.xuliangwei.com ~]# yum repolist all # 查看所有的仓库
```

#### 2.通过 `dnf/yum-config-manager` 启用和禁用仓库

```
# C7
[root@www.xuliangwei.com ~]# yum install https://dev.mysql.com/get/mysql80-community-release-el7-3.noarch.rpm -y

# C8
[root@www.xuliangwei.com ~]# yum install https://dev.mysql.com/get/mysql80-community-release-el8-1.noarch.rpm -y

[root@www.xuliangwei.com ~]# yum repolist all|grep mysql
[root@www.xuliangwei.com ~]# yum-config-manager --disable mysql80-community # 关闭仓库
[root@www.xuliangwei.com ~]# yum-config-manager --enable mysql80-community # 启用仓库
```

- PS: 本质都是在修改 `repo` 文件中的 `enable` 参数值 `0` 不启用 `1` 启用

### 3.4.4.8 yum管理历史记录

当我们删除了某个软件时，希望撤销删除的操作，可以使用 `yum history undo`

#### 1.删除 `httpd` 软件，然后查看操作记录

```
[root@www.xuliangwei.com ~]# yum remove httpd -y
[root@www.xuliangwei.com ~]# yum history
```

#### 2.使用 `dnf/yum history undo Number` 撤销

```
[root@www.xuliangwei.com ~]# yum history info N
[root@www.xuliangwei.com ~]# yum history undo N
```

### 3.4.4.9 yum缓存软件包



## 1.缓存rpm包方式一、通过修改 `dnf/yum` 全局配置文件

```
[root@www.xuliangwei.com ~]# vim /etc/yum.conf
[main]
cachedir=/var/cache/yum/$basearch/$releasever
keepcache=1      # 启动缓存
[root@www.xuliangwei.com ~]# yum install lrzsz -y
[root@www.xuliangwei.com ~]# find /var/cache/yum/ -type f -name "*.rpm"
```

## 2.缓存rpm包方式二、通过 `dnf/yum` 下载该软件包至本地，不进行安装

```
[root@www.xuliangwei.com ~]# yum install httpd -y \
--downloadonly \
--downloadaddir=/opt
```

## 3.如果缓存的数据包太多，可以使用 `dnf/yum` 清除缓存

```
# 清理所有yum缓存信息，包括缓存的软件包
[root@www.xuliangwei.com ~]# yum clean all

#仅清理所有缓存的软件包
[root@www.xuliangwei.com ~]# yum clean packages
```

# 3.4.5 构建YUM仓库实践

## 3.4.5.1 搭建本地yum仓库

很多时候刚安装的linux系统不能联网，但需要安装相应环境的软件包。这时候我们就可以利用光盘制作一个本地yum仓库。

### 1.挂载镜像

```
[root@xuliangwei ~]# mount /dev/cdrom /mnt
```

### 2.备份原有仓库

```
[root@xuliangwei ~]# gzip /etc/yum.repos.d/*
```

### 3.使用 `yum-config-manager` 命令可快速添加一个本地仓库

```
[root@xuliangwei ~]# yum install yum-utils -y
[root@xuliangwei ~]# yum-config-manager --add-repo="file:///mnt"
```

4.当然我们也可以直接去编辑一个 `.repo` 文件，将仓库信息存储至该文件

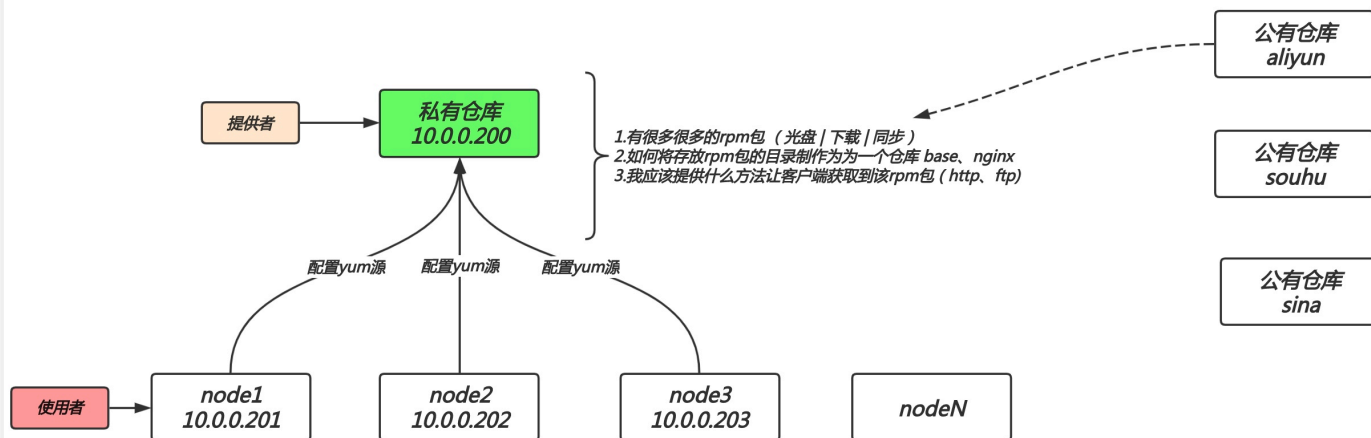
```
[root@xuliangwei ~]# vim /etc/yum.repos.d/cdrom.repo
# 仓库名称
[cdrom]
# 仓库描述信息
name=This is local cdrom
# 仓库url地址 ,可以是file:// ftp:// http:// 等协议
baseurl=file:///mnt
# 是否使用该YUM源(0代表禁用, 1代表激活)
enabled=1
# 是否验证软件签名(0代表禁用, 1代表激活)
gpgcheck=0
```

5.生成缓存信息、然后使用 `dnf/yum` 安装软件测试

```
[root@xuliangwei ~]# yum makecache
[root@xuliangwei ~]# yum install lrzsz -y
```

### 3.4.5.2 搭建企业yum仓库

很多时候不仅仅是一台机器无法上网，而是很多机器都无法上网，但都有联网下载软件的需求，这个时候难道每台机器都挂在光盘吗，当然可以，但如果软件出现了更新又该怎么办。所以我们需要构建一个企业级的 `yum` 仓库，为多台客户端提供服务。



- 本地光盘提供基础软件包: Base

- yum缓存提供常用软件包: nginx, zabbix, docker

### 3.4.5.2.1 环境准备

系统	IP	角色
centos7	10.0.0.99	yum仓库服务端
centos7	10.0.0.98	yum仓库客户端

### 3.4.5.2.2 服务端操作

#### 1.关闭 iptables 防火墙、与 selinux

```
[root@yum_server ~]# systemctl stop firewalld
[root@yum_server ~]# setenforce 0
```

#### 2.安装 ftp 服务，启动并加入开机启动

```
[root@yum_server ~]# yum -y install vsftpd
[root@yum_server ~]# systemctl start vsftpd
[root@yum_server ~]# systemctl enable vsftpd
```

#### 3.首先提供基础 base 软件包

```
[root@yum_server ~]# mkdir /var/ftp/centos7
[root@yum_server ~]# mount /dev/cdrom /mnt
[root@yum_server ~]# cp -rp /mnt/Packages/*.rpm /var/ftp/centos7/
```

#### 4.其次提供第三方源的 rpm 软件包

```
[root@yum_server ~]# cat wget_rpm_scripts.sh
#!/usr/bin/bash

get_zabbix_rpm_url=https://mirrors.aliyun.com/zabbix/zabbix/5.0/rhel/8/x86_64/
rpm_name=$(curl -s ${get_zabbix_rpm_url} | grep "<a" | awk -F '"' '{print $2}')
rpm_dir=/var/ftp/ops

for name in ${rpm_name}
do
    if [ ! -d ${rpm_dir} ];then
        mkdir -p ${rpm_dir}
    fi
```

```
wget -O ${rpm_dir}/${name} ${get_zabbix_rpm_url}${name}
done
```

*#安装 createrepo 并创建生成仓库*

```
[root@yum_server ~]# yum -y install createrepo
```

```
[root@yum_server ~]# createrepo /var/ftp/ops
```

*#PS：如果此仓库每次新增软件则需要重新生成一次*

### 3.4.5.2.3 客户端操作

所有客户端仅需将 `yum` 源指向本地服务端，即可使用本地服务器提供的软件包。

#### 1.客户端配置并使用 `base` 基础源

```
[root@yum_client ~]# gzip /etc/yum.repos.d/*
[root@yum_client ~]# vim /etc/yum.repos.d/centos7.repo
[centos7]
name=centos7_base
baseurl=ftp://10.0.0.99/centos7
gpgcheck=0
```

#### 2.客户端配置并使用 `ops` 源

```
[root@yum_client ~]# vim /etc/yum.repos.d/ops.repo
[ops]
name=local ftpserver
baseurl=ftp://10.0.0.99/ops
gpgcheck=0
```