第九章 文件和文件系统的压缩和打包

1、压缩原理：

目前计算机系统都是用 bytes 为单位计算的，数字 1 以 0000 0001 记录。压缩就是丢弃那 7 个空位，使文件占用的空间变小。或者是对文件里重复的数据进行统计记录。

但是压缩过的文件无法直接被操作系统使用，因此要使用它们还需要解压缩。

2、Linux 中，压缩文件会有 tar、gz 等扩展名，有什么用？

Linux 支持的压缩指令有很多，互相之间技术不同，因此无法互通压缩/解压缩。

```
*.Z       compress 程序压缩的档案；
*.gz      gzip 程序压缩的档案；
*.bz2     bzip2 程序压缩的档案；
*.tar     tar 程序打包的数据，并没有压缩过；
*.tar.gz  tar 程序打包的档案，其中并且经过 gzip 的压缩
*.tar.bz2 tar 程序打包的档案，其中并且经过 bzip2 的压缩
```

tar 用于打包，zip 用来压缩

bzip2 比 gzip 压缩比更好，而 compress 已经 out 了

用 gz 压缩：

```
[root@localhost tmp]# gzip -v autofs.conf
autofs.conf:     62.9% -- replaced with autofs.conf.gz
```

参数-v 能显示压缩比，将当前目录下的 autofs.conf 压缩成 autofs.conf.gz，且替换掉源文件。

```
[root@localhost tmp]# ll auto*
-rw-r--r--. 1 root root 5368 3月  22 19:47 autofs.conf.gz
```

可见，源文件不见了

zcat 可以读取压缩文本文件的内容

```
[root@localhost tmp]# zcat autofs.conf.gz
#
# Define default options for autofs.
#
[ autofs ]
#
# master_map_name - default map name for the master map.
#
#master_map_name = auto.master
#
# timeout - set the default mount timeout in secons. The internal
#           program default is 10 minutes, but the default installed
#           configuration overrides this and sets the timeout to 5
```

解压缩：

也是用 gzip 指令，但参数为-d，会将原本的压缩文件删除，生成原本的文件

```
[root@localhost tmp]# gzip -d autofs.conf.gz
[root@localhost tmp]# ll auto*
-rw-r--r--. 1 root root 14392 3月  22 19:47 autofs.conf
```

将文件压缩成别的名字且保留源文件：

```
[root@localhost tmp]# gzip -c autofs.conf > auto_NewName.gz
[root@localhost tmp]# ll auto*
-rw-r--r--. 1 root root 14392 3月  22 19:47 autofs.conf
-rw-r--r--. 1 root root  5368 3月  22 20:19 auto_NewName.gz
```

如果没有-c 参数，就不会保留源文件

bzip2 用法与 gzip 基本一样

压缩

```
[root@localhost tmp]# ls auto*
autofs.conf  auto_NewName1.gz  auto_NewName.gz
[root@localhost tmp]# bzip2 -z autofs.conf
[root@localhost tmp]# ls auto*
autofs.conf.bz2  auto_NewName1.gz  auto_NewName.gz
```

查看压缩文本文件

```
[root@localhost tmp]# bzcat autofs.conf.bz2
#
# Define default options for autofs.
#
[ autofs ]
#
# master_map_name - default map name for the master map.
#
#master_map_name = auto.master
```

解压缩：

bzip2 -d autofs.conf.bz2

```
[root@localhost tmp]# ls auto*
autofs.conf  auto_NewName1.gz  auto_NewName.gz
```

用最佳压缩比压缩且保留源文件

```
[root@localhost tmp]# bzip2 -9 -c autofs.conf > autoHaHa.bz2
[root@localhost tmp]# ls auto*
autofs.conf  autoHaHa.bz2  auto_NewName1.gz  auto_NewName.gz
```

gzip、bzip2 也能够对目录进行压缩，但只会对目录内所有文件分别进行压缩（骗鬼，我试了一下，根本不能对目录进行压缩操作），

tar 可以将多个文件或目录打包成一个大文件，同时还可以透过 gzip/bzip2 的支持同时对此文件进行压缩

压缩：tar -jcv -f filename.tar.bz2 要被压缩的源文件或目录名

查询：tar -jtv -f filename.tar.bz2

解压：tar -jxv -f filename.tar.bz2 -C 解压后的目录

filename.tar.bz2 是自己取的压缩文件文件名，tar 不会主动生成文件名。

-j 代表有 bzip2 支持，因此最好文件名最好加上扩展名*.tar.bz2

-z 代表有 gzip 支持，最好有扩展名*.tar.gz

tar -j 备份/etc

tar -jpcv -f /root/etc.tar.bz2 /etc

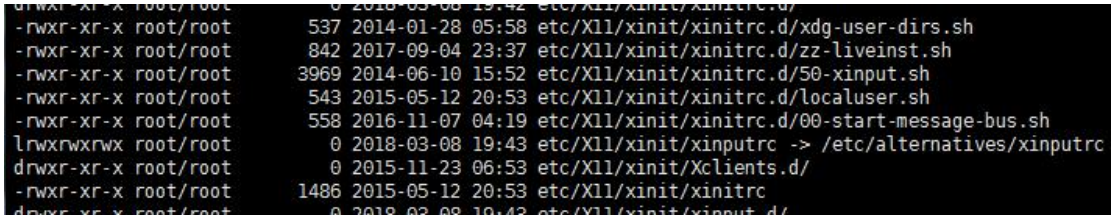
```
[root@localhost tmp]# ll /root/etc.tar.bz2
-rw-r--r--. 1 root root 9398803 3月  22 20:56 /root/etc.tar.bz2
```

加上 p 这个参数的原因是为了保留源文件的权限和属性

查阅 tar 文件的数据内容（可查看文件名）

tar -jtv -f /root/etc.tar.bz2

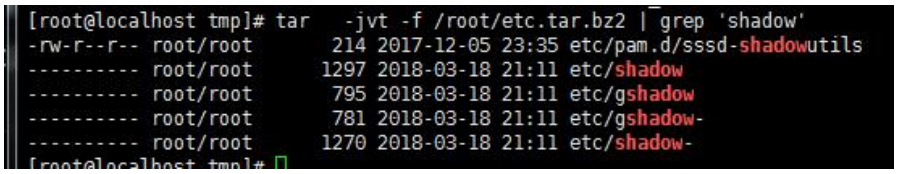
看到的就是以下这些文件名了



每个文件名都没有根目录了，因为之前压缩时出现了那个警告讯息『tar: Removing leading `/' from member names(去除了文件名开头的 `/')』
这是为了安全，解压的时候，没有根目录，文件名就是一个相对路径，会在当前目录解压出来，而如果有根目录就是绝对路径，解压后直接覆盖了原本的/etc 里的文件

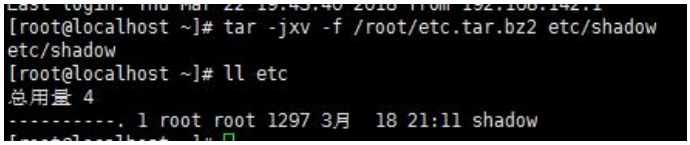
解压缩
tar -jxv -f /root/etc.tar.bz2
以上指令会直接在本目录解压缩，解压完毕后本目录下会多出一个 etc 文件夹
tar -jxv -f /root/etc.tar.bz2 -C /tmp
以上指令指定了解压的目录

以上的解压是把整个打包文件给解压开的，如果想只打开打包文件中的一个文件，可以这样：
1、先找到这个文件



第二个文件是我们要的
grep 是 撷取 关键词 的意思

2、解压这个文件



打包 tar：
打包某目录，但不包含该目录下的某些文件：
tar -jcv -f /root/system.tar.bz2 --exclude=/root/etc*--exclude=/root/system.tar.bz2 /etc /root
透过这个 --exclude="file" 的动作，我们可以将几个特殊的文件或目录排除出去，这句命令排除了以 etc 开头的文件和 system.tar.bz2 这个文件

仅备份某个时候之后改动的文件：
1、找出比 某一个文件 还要新的文件

```
[root@localhost ~]# find /etc -newer /etc/passwd
/etc
/etc/tuned/active_profile
/etc/shadow
/etc/gshadow
/etc/resolv.conf
/etc/group
/etc/fstab
/etc/resolv.conf.save
[root@localhost ~]# ll /etc/passwd
-rw-r--r--. 1 root root 2289 3月  18 21:11 /etc/passwd
```

那么怎么打包比 3 月 18 号 21:11 更新的文件呢（就是上面那几个）：

```
[root@localhost ~]# tar -jcv -f /root/etc.newerthanpasswd.tar.bz2 --newer-mtime="2018/03/18" /etc/*
```

显示里面结尾非 / 的文件名

```
[root@localhost ~]# tar -jtv -f /root/etc.newerthanpasswd.tar.bz2 | grep -v '/$'
-rw-r--r-- root/root       589 2018-03-21 23:51 etc/fstab
-rw-r--r-- root/root       994 2018-03-18 21:11 etc/group
-rw-r--r-- root/root       976 2018-03-18 21:11 etc/group-
---------- root/root       795 2018-03-18 21:11 etc/gshadow
---------- root/root       781 2018-03-18 21:11 etc/gshadow-
-rw-r--r-- root/root      2289 2018-03-18 21:11 etc/passwd
```

此外，一些打包但没有压缩的扩展名为.tar，叫做 tarfile。

而 tar 还能降资料打包到某些特别的装置上，比如磁带机（便宜，多用在企业大量存储），磁带机是一次性读取/写入装置，因此不能使用 cp 等指令来复制。如果想把/home,/root,/etc 这三个目录备份到磁带机（/dev/st0）上，可以这样：

tar -cv -f dev/st0 /home /root /etc

系统备份案例：备份数据不应该放在/root 目录下

系统中比较重要的数据：

- /etc/ (配置文件)
- /home/ (用户的家目录)
- /var/spool/mail/ (系统中，所有账号的邮件信箱)
- /var/spool/cron/ (所有账号的工作排成配置文件)
- /root (系统管理员的家目录)

准备将备份数据放在/backups，且此目录仅 root 有权限进入

```
[root@localhost ~]# mkdir /backups
[root@localhost ~]# chmod 700 /backups/
[root@localhost ~]# ll -d /backups/
drwx------. 2 root root 4096 3月  23 12:39 /backups/
[root@localhost ~]# tar -jcv -f /backups/backup-system-20180323.tar.bz2 --exclude=/root/*.gz --exclude=/
home/loop* /etc /home /var/spool/mail /var/spool/cron /root
```

备份结果：

```
[root@localhost ~]# ll -h /backups/
总用量 23M
-rw-r--r--. 1 root root 23M 3月  23 12:42 backup-system-20180323.tar.bz2
```

完整备份工具：dump

它除了能 针对整个文件系统备份外，也能仅针对目录进行备份

尝试备份最小的文件系统：

1、找出最小的文件系统

```
[root@localhost ~]# df -h
文件系统                 容量  已用  可用 已用% 挂载点
/dev/mapper/centos-root  9.8G  3.7G  5.6G   40% /
devtmpfs                 901M     0  901M    0% /dev
tmpfs                    912M     0  912M    0% /dev/shm
tmpfs                    912M  9.0M  903M    1% /run
tmpfs                    912M     0  912M    0% /sys/fs/cgroup
/dev/sda1                477M  113M  340M   25% /boot
/dev/mapper/centos-home  4.7G   57M  4.4G    2% /home
tmpfs                    183M     0  183M    0% /run/user/0
```

/boot 应该是最小的了

发现没有 dump 程序，先安装了

```
[root@localhost ~]# yum install dump
已加载插件：fastestmirror, langpacks
base
extras
```

2、测试一下备份该系统需要多少容量：

```
[root@localhost ~]# dump -S /boot
115187712
```

单位是 bytes

3、开始备份，备份文件名为 boot.dump

```
[root@localhost ~]# dump -0u -f /root/boot.dump /boot
  DUMP: Date of this level 0 dump: Fri Mar 23 13:35:11 20
  DUMP: Dumping /dev/sda1 (/boot) to /root/boot.dump
  DUMP: Label: none
  DUMP: Writing 10 Kilobyte records
```

```
[root@localhost ~]# ll /root/boot.dump /etc/dumpdates
-rw-rw-r--. 1 root disk        43 3月  23 13:35 /etc/dumpdates
-rw-r--r--. 1 root root 115998720 3月  23 13:35 /root/boot.dump
```

由于有参数-u，因此？/etc/dumpdates 这个文件的内容会更新

且这个文件仅在 dump 完整的文件系统时才会有支持主动更新的功能

```
[root@localhost ~]# cat /etc/dumpdates
/dev/sda1 0 Fri Mar 23 13:35:11 2018 +0800
```

可见该文件里记录了 /boot 文件系统的备份时间，备份是 level0

查看有哪些 filesystem 被 dump 备份过：

```
[root@localhost ~]# dump -W
Last dump(s) done (Dump '>' file systems):
> /dev/mapper/centos-root       (     /) Last dump: never
  /dev/sda1     ( /boot) Last dump: Level 0, Date Fri Mar 23 13:35:11 2018
> /dev/mapper/centos-home       ( /home) Last dump: never
```

尝试建立 level1 的备份

可见当前共有三个 filesystem，只有/boot 的文件系统被备份过

1、先在/boot 内新建一个 10MB 的文件

```
[root@localhost ~]# dd -if=/dev/zero of=/boot/testing.img bs=1M count=10
dd：无效选项 -- i
Try 'dd --help' for more information.
[root@localhost ~]# dd if=/dev/zero of=/boot/testing.img bs=1M count=10
记录了10+0 的读入
记录了10+0 的写出
10485760字节(10 MB)已复制，0.315585 秒，33.2 MB/秒
```

2、开始使用 level1 备份

```
10485760字节(10 MB)已复制，0.315585 秒，33.2 MB/秒
[root@localhost ~]# dump -1u -f /root/boot.dump.1 /boot
  DUMP: Date of this level 1 dump: Fri Mar 23 13:57:38 2018
```

```
[root@localhost ~]#  ll /root/boot*
-rw-r--r--. 1 root root 115998720 3月  23 13:35 /root/boot.dump
-rw-r--r--. 1 root root  10577920 3月  23 13:57 /root/boot.dump.1
```

可见，这个新建的备份文件，只有那个步骤 1 里新增的文件的大小那么大，约 10MB

再一看，已经更新了备份时间

```
[root@localhost ~]# dump -W
Last dump(s) done (Dump '>' file systems):
> /dev/mapper/centos-root       (     /) Last dump: never
  /dev/sda1     ( /boot) Last dump: Level 1, Date Fri Mar 23 13:57:38 2018
> /dev/mapper/centos-home       ( /home) Last dump: never
```

使用 dump 备份非文件系统，即单一目录，只能使用 level0，下面备份/etc 目录

```
[root@localhost ~]# dump -0j -f /root/etc.dump.bz2 /etc
  DUMP: Date of this level 0 dump: Fri Mar 23 14:10:05 2018
```

其实也可以不用压缩（即去掉 j 参数）

```
  DUMP: Wrote 48770kB uncompressed, 13843kB compressed, 3.524:1
  DUMP: DUMP IS DONE
```

压缩比为 3.5 : 1 左右

备份文件的作用是恢复系统重要数据，那如何复原数据呢？

dump 备份的数据对应的复原是 restore 指令

查阅 dump 文件内容:

```
[root@localhost ~]# restore -t -f /root/boot.dump
Dump   date: Fri Mar 23 13:35:11 2018
Dumped from: the epoch
Level 0 dump of /boot on localhost.localdomain:/dev/sda1
Label: none
         2      .
        11      ./lost+found
     95505      ./efi
     95506      ./efi/EFI
     95507      ./efi/EFI/centos
     52833      ./grub2
```

为什么 dump 能进行累积备份呢？这是因为它可以查找到 filesystem 与备份文件之间的差异，且将差异数据进行备份

查询有变动过的数据：

1、先变更一下文件系统的内容：修改文件名

```
[root@localhost ~]# cd /boot
[root@localhost boot]# mv config-3.10.0-693.el7.x86_64 config-3.10.0-693.el7.x86_64_back
```

2、查看差异：

```
[root@localhost boot]# restore -C -f /root/boot.dump
Dump   date: Fri Mar 23 13:35:11 2018
Dumped from: the epoch
Level 0 dump of /boot on localhost.localdomain:/dev/sda1
Label: none
filesys = /boot
restore: unable to stat ./config-3.10.0-693.el7.x86_64: No such file or directory
```

发现了差异（最后改回去）

由于 dump 是备份了整个文件系统的，因此还原的时候应该给予一个全新的文件系统。即先建一个新的文件系统，再还原：

先建一个新的 partition，boot 大概占用 100 多 M，因此取容量 150M

```
[root@localhost boot]# df
文件系统                 1K-块     已用     可用 已用% 挂载点
/dev/mapper/centos-root 10190136 3976396 5689452   42% /
devtmpfs                  922584       0  922584    0% /dev
tmpfs                     933512       0  933512    0% /dev/shm
tmpfs                     933512    9120  924392    1% /run
tmpfs                     933512       0  933512    0% /sys/fs/cgroup
/dev/sda1                 487652  124974  337078   28% /boot
/dev/mapper/centos-home  4908544   58196 4594348    2% /home
```

```
   设备 Boot      Start         End      Blocks   Id  System
/dev/sdb1         1312768     1368063       27648   83  Linux
/dev/sdb2         1368064     1777663      204800    5  Extended
/dev/sdb3            2048     1050623      524288   83  Linux
/dev/sdb4         1050624     1312767      131072   83  Linux
/dev/sdb5         1370112     1677311      153600   83  Linux
```

如上图所示 sdb5 就是要用的分区

然后按惯例格式化并挂载

```
[root@localhost boot]# mkfs -t ext3 /dev/sdb5
mke2fs 1.42.9 (28-Dec-2013)
```

```
[root@localhost boot]# mount /dev/sdb5 /mnt
[root@localhost boot]# df
文件系统                 1K-块     已用     可用 已用% 挂载点
/dev/mapper/centos-root 10190136 3976404 5689444   42% /
devtmpfs                  922584       0  922584    0% /dev
tmpfs                     933512       0  933512    0% /dev/shm
tmpfs                     933512    9140  924372    1% /run
tmpfs                     933512       0  933512    0% /sys/fs/cgrou
/dev/sda1                 487652  124974  337078   28% /boot
/dev/mapper/centos-home  4908544   58196 4594348    2% /home
tmpfs                     186704       0  186704    0% /run/user/0
/dev/sdb5                 144646    1567  135399    2% /mnt
```

进入/mnt 目录，开始还原

```
[root@localhost mnt]# restore -r -f /root/boot.dump
restore: ./lost+found: File exists
```

```
[root@localhost mnt]# ls
config-3.10.0-693.el7.x86_64
efi
grub
grub2
initramfs-0-rescue-f0fca8a83e7f4ff6862b9778f132d93c.img
initramfs-3.10.0-693.el7.x86_64.img
initramfs-3.10.0-693.el7.x86_64kdump.img
initrd-plymouth.img
lost+found
restoresymtable
symvers-3.10.0-693.el7.x86_64.gz
System.map-3.10.0-693.el7.x86_64
vmlinuz-0-rescue-f0fca8a83e7f4ff6862b9778f132d93c
vmlinuz-3.10.0-693.el7.x86_64
[root@localhost mnt]# ls /boot
config-3.10.0-693.el7.x86_64
efi
grub
grub2
initramfs-0-rescue-f0fca8a83e7f4ff6862b9778f132d93c.img
initramfs-3.10.0-693.el7.x86_64.img
initramfs-3.10.0-693.el7.x86_64kdump.img
initrd-plymouth.img
lost+found
symvers-3.10.0-693.el7.x86_64.gz
System.map-3.10.0-693.el7.x86_64
testing.img
vmlinuz-0-rescue-f0fca8a83e7f4ff6862b9778f132d93c
vmlinuz-3.10.0-693.el7.x86_64
[root@localhost mnt]#
```

上图可见，还原完毕后，/mnt 里的文件与/boot 里的一模一样

只解出备份文件中的某些内容：使用 restore 的互动模式（interactive mode）

下面尝试把 etc.dump 中的 passwd 和 shadow 抓取出来：

```
[root@localhost mnt]# restore -i -f /root/etc.dump.bz2
Dump tape is compressed.
restore > help
restore > cd etc
restore > pwd
/etc
restore > ls passwd shadow group
passwd
shadow
group
restore > add passwd shadow group
restore > delete group
restore > ls passwd shadow group
*passwd
*shadow
group
restore > extract
You have not read any volumes yet.
Unless you know which volume your file(s) are on you should start
with the last volume and work towards the first.
Specify next volume # (none if no more volumes): 1
set owner/mode for '.'? [yn] n
```

进入 dump 文件内后，首先进入/etc 目录，然后查看是否有那三个文件，再通过 add 指令将他们加入解压文件列表，通过 delete 指令将文件移除解压文件列表，再 ls 显示会发现处于解压列表中的文件前面会有*号，最后通过指令 extract 将解压列表中的文件解压，然后通过 quit 指令退出互动模式

当前这个目录下，会出现一个 etc 目录，打开一看：

```
[root@localhost mnt]# cd etc
[root@localhost etc]# ls
passwd  shadow
```

就有刚才解压的那两个文件

将数据备份到光盘上这部分 P320~P324，就不管了

其他常见的压缩和备份工具：
dd 这个指令，可以读取硬盘装置的内容，可以将整个装置备份成一个文件
dd if=input_file of=output_file bs=block_size count=number

将/etc/passwd 备份到 /tmp/passwd.back 中

```
[root@localhost etc]# dd if=/etc/passwd of=/tmp/passwd.back
记录了4+1 的读入
记录了4+1 的写出
2289字节(2.3 kB)已复制，0.0125062 秒，183 kB/秒
[root@localhost etc]# ll /tmp/passwd.back
-rw-r--r--. 1 root root 2289 3月  23 16:12 /tmp/passwd.back
```

事实上，跟 cp 指令的效果差不多
将硬盘的第一个扇区备份下来：

```
[root@localhost etc]# dd if=/dev/sdb of=/tmp/mbr.back bs=512 count=1
记录了1+0 的读入
记录了1+0 的写出
512字节(512 B)已复制，0.308675 秒，1.7 kB/秒
```

备份/boot:
dd if=/dev/sda1 of=/tmp/boot.whole.disk
复原就反向使用 dd：
dd if=/tmp/boot.whole.disk of=/dev/sda1

可以说 tar 用来备份关键数据，dd 可以用来备份整个分区或硬盘
当需要将/dev/sda1 完整的复制到另一个分区 partition 上时 ,由于需要复制 boot sector 区块，所以 cp 或 tar 指令不行，需要用到 dd
如果你想要建置两颗一模一样的磁盘
时， 只要下达指令： dd if=/dev/sda of=/dev/sdb ，就能够让两颗磁盘一模一样，甚至 /dev/sdb 不需要分割和格式化， 因为该指令可以将 /dev/sda 内的所有资料，包括 MBR 和 partition table 也复制到 /dev/sdb