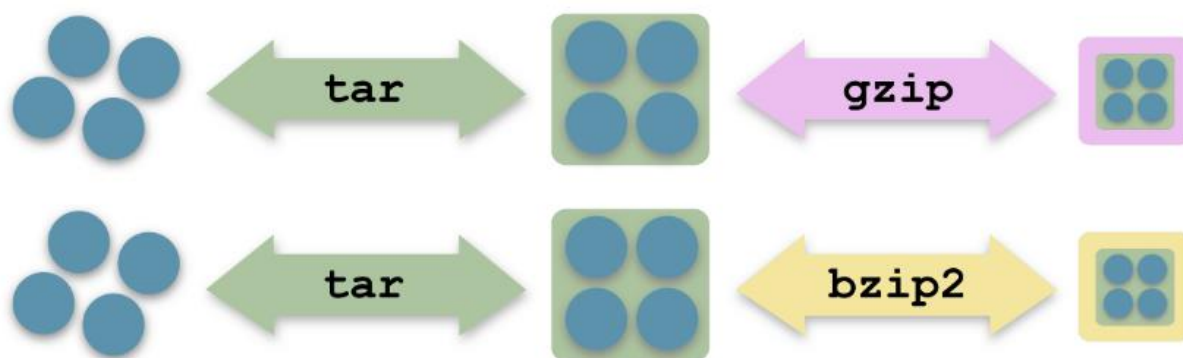


## Архивация

Дистрибутивы Linux предоставляют несколько различных наборов команд для сжатия и архивирования файлов и каталогов. В этой главе будут описаны их преимущества и недостатки, а также их полезность для эффективного создания резервных копий файлов и каталогов.

Более сложные типы копирования с использованием команд `dd` и `cpio` также будут рассмотрены. Эти команды, хотя и более сложные, предлагают мощные функции.



### Команда `tar`

Архив - это отдельный файл, который состоит из множества файлов, хотя и не обязательно сжатых. Команда `tar` обычно используется для создания архивов в Linux. Эти архивные файлы `tar`, иногда называемые **tarballs**, изначально использовались для резервного копирования данных на магнитную ленту. **Tar** происходит от слов "архив ленты".

Хотя основной целью команды **tar** является объединение нескольких файлов в один файл, она способна выполнять множество различных операций и существует множество опций. Функциональность команды **tar** может быть разбита на три основные функции: создание, просмотр и извлечение архивов.

Команда `tar` принимает в качестве параметров все три стиля опций (**x**, **-x**, **--extract**). Не удивляйтесь, увидев, что он используется с опциями, которые не имеют дефисов (стиль BSD), одного дефиса (стиль UNIX) или двух дефисов (стиль GNU).

Команда `tar` изначально не поддерживала сжатие, но более новая версия, разработанная проектом GNU, поддерживает сжатие как **gzip**, так и **bzip2**.

Чтобы использовать сжатие `gzip` с командой `tar`, используйте опцию **-z**.

Чтобы использовать сжатие `bzip2`, используйте параметр **-j**.

Если версия команды `tar` не поддерживает сжатие `gzip` или `bzip2`, команды `gzip` или `bzip2` можно использовать отдельно в файле `tar`.

Чтобы создать архив `tar` из каталога `/etc/systemd`, используйте параметр **-c** для создания и **-f** для указания нового файла, а затем каталог для архивирования. Обратите внимание, что параметр **-f** должен быть указан последним, поскольку он указывает имя файла:

```
tar -cf newfile.tar file
```

```
tar -cf systemd-config.tar /etc/systemd
```

```
[root@R-FW ~]# tar -cf systemd-config.tar /etc/systemd/
tar: Removing leading '/' from member names
[root@R-FW ~]# ls
1.txt          anaconda-ks.cfg  file.txt        ipbase          systemd-config.tar
allowrouting   err.txt          instIPT         passwd.txt      tmp
```

Опция **-v** verbose заставит команду **tar** отобразить файлы, включенные в архив. Если будет использоваться сжатие, например, **gzip**, то необходимо добавить **-z**. Расширения файлов не имеют отношения к Linux, но обычно добавляется **.tar.gz** к имени сжатого архива:

```
tar -cvzf systemd-config.tar.gz /etc/systemd
```

```
/etc/systemd/system/local-fs.target.wants/  
/etc/systemd/system/local-fs.target.wants/rhel-readonly.service  
/etc/systemd/system.conf  
/etc/systemd/user/  
/etc/systemd/user.conf  
[root@R-FW ~]# ls  
1.txt          anaconda-ks.cfg  file.txt  ipbase      systemd-config.tar  tmp  
allowrouting  err.txt          instIPT   passwd.txt  systemd-config.tar.gz
```

Выберите опцию **-t** команды **tar** для просмотра списка (оглавления) файла **tar**. Даже если файл архива сжат, правильная опция сжатия не требуется указывать для просмотра файла архива **tar**. Для просмотра оглавления требуются только файл **-f** и опция **list -t**. Еще раз обратите внимание, что опция **-f** используется последней, так что имя файла может быть указано в качестве аргумента этой опции:

```
tar -tf systemd-config.tar.gz
```

Чтобы просмотреть оглавление архива в формате, похожем на длинный листинг (например, команда **ls -l**), добавьте параметр verbose **-v**:

```
tar -vtf systemd-config.tar.gz
```

Чтобы извлечь файлы из файла **tar**, используйте опцию **-x**. Обычно команда **tar** пытается извлечь архив в текущий каталог, как показано в следующем примере:

```
tar -xf systemd-config.tar.gz
```

```
ls -R etc
```

Используйте параметр **-C**, чтобы указать альтернативный каталог для извлечения содержимого. Эта опция чувствительна к регистру и не должна быть перепутана с опцией **create** или **-c**. Процесс извлечения также можно сделать подробным, добавив опцию **-v**:

```
tar -vxf systemd-config.tar.gz -C /tmp
```

### Команды **gzip** и **gunzip**

Команда **gzip** используется для создания сжатого файла архива. Аналогично, команда **gunzip** используется для просмотра содержимого архивного файла, а также для извлечения этого содержимого.

Команду **gzip** следует использовать с осторожностью, поскольку по умолчанию она заменяет исходный файл, указанный сжатой версией. В следующем примере файл **words** заменяется сжатым файлом **words.gz** после использования **gzip**:

```
cp words words.backup
```

```
gzip words
```

```
root@L-FW:~# cp words words.backup  
root@L-FW:~# gzip words  
root@L-FW:~# ls  
delzone  ipbase      messages  passwd      words.backup  
delzone1 lpicfile.txt my.sh     personal_backup words.gz
```

Чтобы избежать замены исходной версии файла при использовании **gzip**, используйте параметр **-c**. Это приводит к тому, что команда **gzip** отправляет данные **gzip** на стандартный вывод, и, учитывая, что выходные данные команды **gzip** являются двоичными данными, ее необходимо перенаправить в файл. Не забудьте захватить вывод команды **gzip** и перенаправить его в файл, используя символ **>**:

```
cp /var/log/messages messages
```

```
gzip -c messages > messages.gz
```

Использование команды **gzip** с параметром **-c** и перенаправлением позволило создать файл **gzip**, оставив исходный файл без изменений. Это может быть полезно, так

как сжатый файл может быть перемещен в каталог архива с сохранением исходного файла в его исходном местоположении.

Команда **gunzip** переворачивает действия **gzip**, поэтому файлы будут распакованы, а файл **gzip** будет заменен несжатым файлом:

**gunzip words.gz**

```
root@L-FW:~# gunzip words.gz
root@L-FW:~# ls
delzone  ipbase      messages  passwd      words
delzone1 lpicfile.txt my.sh     personal_backup words.backup
root@L-FW:~#
```

Чтобы просмотреть степень сжатия существующего архивного файла, используйте параметр **-l** с **gunzip**:

**gunzip -l messages.gz**

```
root@L-FW:~# gunzip -l messages.gz
      compressed      uncompressed  ratio uncompressed_name
      197737          1204635  83.6% messages
root@L-FW:~# ls -al
total 3304
drwx----- 4 root root 36864 May  8 10:50 .
drwxr-xr-x 22 root root 4096 Jan 24 11:59 ..
-rw----- 1 root root 4821 May  4 07:41 .bash_history
-rw-r--r-- 1 root root 601 May  4 07:41 .bashrc
-rwxr-xr-x 1 root root 221 Jan 24 11:20 delzone
-rwxr-xr-x 1 root root 350 Jan 24 10:47 delzone1
-rwxr-xr-x 1 root root 724 Jan 20 11:13 ipbase
-rw-r--r-- 1 root root 0 May  1 09:38 lpicfile.txt
-rw-r----- 1 root root 1204635 May  1 10:01 messages
-rw-r--r-- 1 root root 197737 May  8 10:50 messages.gz
```

Хотя он поддерживает рекурсию с параметром **-r**, по умолчанию **gzip** пытается заменить исходный файл файлом **gzip**. Это приводит к ошибкам, когда архивируемые файлы не принадлежат пользователю, который пытается сжать их.

Чтобы иметь возможность рекурсивно сжимать файлы с помощью команды **gzip**, пользователь должен иметь правильные разрешения для каталогов, в которых находятся файлы. Обычно это ограничивается каталогами в собственном домашнем каталоге пользователя.

Например, для рекурсивного использования команды **gzip** в каталоге **~/example** было бы успешно заменить обычные файлы архивными файлами **gzip**:

```
mkdir ./example
touch ./example/one ./example/two ./example/three
ls ./example/
one three two
gzip -r ./example
ls ./example/
one.gz three.gz two.gz
```

Команда **gunzip** также может работать рекурсивно, предполагая, что у пользователя есть правильные разрешения. Как это работает, он удаляет расширение **.gz** из каждого файла:

```
gunzip -r ./example/
ls ./example/
one three two
```

Права доступа могут влиять на команды управления файлами, такие как команды **gzip** и **gunzip**. Чтобы сжать или заархивировать файл в каталоге, пользователь должен иметь разрешение на запись и выполнение в каталоге, а также разрешение на чтение в файле. Обычные пользователи обычно имеют такие разрешения только в своем домашнем каталоге и его подкаталогах.

## Команды bzip2 и bunzip2

Команды bzip2 и bunzip2 работают почти так же, как команды gzip и gunzip. Различия заключаются в типе алгоритма (как файлы сжимаются), используемого для сжатия файлов, и расширение .bz2 добавляется или удаляется из имени файла (а не расширения .gz).

Когда новый сжатый файл создается из существующего файла с помощью команды **bzip2**, к имени файла добавляется расширение **.bz2**. Использование опции **-v** заставит **bzip2** сообщать о степени сжатия после его завершения. Команда **gzip** также поддерживает параметр **-v**, поэтому файл можно сжать с использованием обеих команд и коэффициента сжатия по сравнению, чтобы определить, какая команда использует лучший метод сжатия для этого конкретного файла.

```
ls ./example/  
one three two  
bunzip2 -v ./example/*
```

```
root@L-FW:~# bzip2 -v ./example/*  
./example/one: no data compressed.  
./example/three: no data compressed.  
./example/two: no data compressed.  
root@L-FW:~# ls ./example/  
one.bz2 three.bz2 two.bz2
```

Обратите внимание, что данных для сжатия в пустых файлах не было. Как и **gunzip**, команда **bunzip2** распаковывает файл и удаляет расширение **.bz2**.

Как показано в приведенных выше примерах, команда **bzip2** не имеет опции **-r** для выполнения рекурсии, поэтому можно использовать подстановочный знак для сопоставления нескольких файлов. Команда **bzip2** также имеет опцию **-c** для отправки данных в стандартный вывод, чтобы их можно было перенаправить в новый файл:

```
bzip2 -c messages > messages.bz2
```

## Команда xz

Используя опцию **-z**, **xz** можно использовать для сжатия группы файлов по отдельности.

```
xz *
```

```
ls
```

```
root@L-FW:~# xz *  
xz: example: Is a directory, skipping  
xz: personal_backup: Is a directory, skipping  
root@L-FW:~# ls  
delzone1.xz example lpicfile.txt.xz messages.xz passwd.xz words.backup.xz  
delzone.xz ipbase.xz messages.gz.xz my.sh.xz personal_backup words.xz
```

Опция **-d** может быть использована для простого распаковывания файлов.

```
xz -d *
```

```
ls
```

```
root@L-FW:~# xz -d *  
xz: example: Is a directory, skipping  
xz: personal_backup: Is a directory, skipping  
root@L-FW:~# ls  
delzone example lpicfile.txt messages.gz passwd words  
delzone1 ipbase messages my.sh personal_backup words.backup
```

Однако часто предпочтительнее связывать файлы перед их архивированием в любом формате, будь то **bzip**, **gzip** или **xz**. Следующий пример архивирует тестовый каталог с использованием **tar** перед сжатием его с помощью **xz**:

```
tar -cvf example.tar ./example
```

```
xz example.tar
```

```
ls -al
```

```
drwxr-xr-x 2 root root 4096 May 8 22:28 example  
-rw-r--r-- 1 root root 216 May 8 22:29 example.tar.xz
```

Команда **tar** также может сжимать, используя **xz** напрямую, используя опцию **-J**:

```
tar -cJf example.tar.xz ./example
```

Для команды **xz** существует огромное количество параметров, некоторые из которых относятся к степени сжатия. При использовании **xz** помните, что чем агрессивнее сжатие, тем сложнее будет работать процессор.

### Команда *cpio*

Командой архивирования другого типа, которая может объединить множество файлов в один файл, является команда **cpio**. Эта команда получает свое имя из двух режимов: **copy-in mode** и режим **copy-out**.

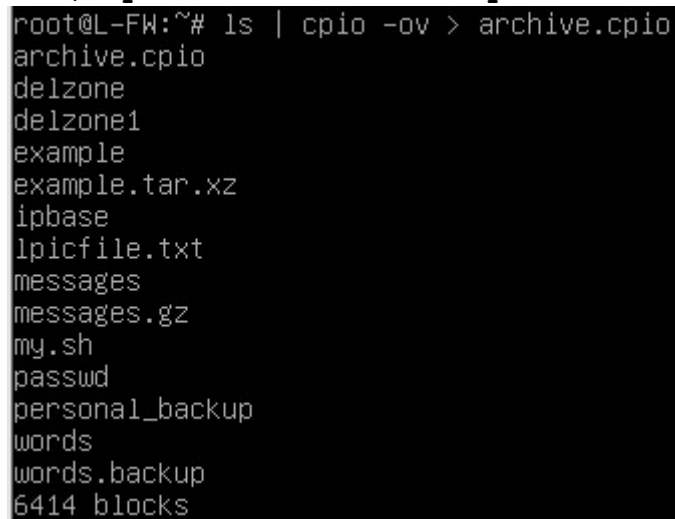
В режиме **copy-out** команда **cpio** скопирует файлы из каталога в архив. В режиме **copy-in** команда **cpio** либо выведет список содержимого файла архива, либо скопирует файлы из архива в каталог. Легко изменить эти утверждения и запутаться. Просто помните, что архив находится за пределами операционной системы.

Существует третий режим, называемый режимом **copy-pass**. В этом режиме команда **cpio** копирует файлы из одного каталога в другой, что объединяет режимы **copy-out** и **copy-in** без создания файла архива.

Чтобы создать новый архивный файл, команда **cpio** будет работать в режиме **copy-out**, беря список файлов из стандартного ввода и создавая поток файлов, который можно перенаправить в новый архивный файл. Стандартный ввод в этом случае относится к вводу с клавиатуры по умолчанию, но этот ввод также может исходить от вывода других команд.

Опция **-o** переводит команду **cpio** в режим **copy-out**. При использовании опции **-v** команда **cpio** выведет список файлов, которые она обрабатывает. Итак, чтобы заархивировать текущий каталог, выполните команду **ls**, а затем отправьте список файлов в команду **cpio** в качестве входных данных, используя символ **pipe** «|» (напомним, что символ **>** захватит вывод команды и поместит его в файл):

```
ls | cpio -ov > archive.cpio
```



```
root@L-FW:~# ls | cpio -ov > archive.cpio
archive.cpio
delzone
delzone1
example
example.tar.xz
ipbase
lpicfile.txt
messages
messages.gz
my.sh
passwd
personal_backup
words
words.backup
6414 blocks
```

Команда **find** - это хороший способ создать список файлов для отправки в **cpio**. Команда **find** автоматически рекурсивна, поэтому ее можно использовать для создания списка всех файлов, начиная с определенного каталога. Например, чтобы заархивировать домашний каталог и все его подкаталоги и файлы, выполните следующую команду:

```
find / -name "*passwd*"
```



```
/usr/share/man/zh_CN/man1/passwd.1.gz
/usr/share/man/zh_CN/man1/gpasswd.1.gz
/usr/share/man/zh_CN/man8/chpasswd.8.gz
/usr/share/man/man8/chgpasswd.8.gz
/usr/share/man/man8/update-passwd.8.gz
/usr/share/man/man8/chpasswd.8.gz
/usr/share/man/cs/man5/passwd.5.gz
```

```
find / -name "*passwd*" > passwd.cpio
ls
```

```
-rw-r--r-- 1 root root 3170 May 8 22:46 passwd.cpio
```

Опция **-v verbose** используется для отображения активности команды **cpio** в терминале.

Чтобы извлечь файлы из архива **cpio**, используйте параметр **-i** с командой **cpio**, чтобы указать режим копирования. По умолчанию **cpio** не будет перезаписывать существующие файлы, если не используется параметр **-u**. Команда **cpio** не будет создавать каталоги, если не используется опция **-d**.

Команда **cpio** также использует стандартный ввод для определения имени файла, который будет извлечен из архива. Для извлечения файлов из архива предназначен параметр **-i**, причем на вход утилиты должны передаваться данные из файла архива. Извлеченные файлы будут сохраняться в текущей директории. Поэтому для извлечения файлов и каталогов, а также перезаписи существующих файлов выполните следующее:

```
cpio -i < passwd.cpio
```

Для копирования всех скрытых файлов из текущей директории в директорию **/tmp** может использоваться следующая команда:

```
find .* | cpio -pmud /tmp
```

В данном случае команда **find** предназначена для формирования списка временных файлов, который отправляется на вход утилиты **cpio**. Если вы не хотите копировать какие-либо файлы, вы можете создать фильтр на основе утилиты **grep**. Параметр **-p** предназначен для копирования файлов, параметр **-m** — для сохранения меток времени их модификации, а параметр **-d** — для воссоздания дерева директорий при необходимости.

Чтобы указать режим передачи команды **cpio**, укажите параметр **-p**. Опять же, если какие-либо каталоги включены, необходимо указать параметр **-d**. Чтобы скопировать все файлы из домашнего каталога в каталог с именем **/tmp/destination**, используйте следующую командную строку:

```
find ~ | cpio -pd /tmp/destination
```

Чтобы избежать проблем с файлами, в которые встроены символы пробела (например, символ пробела), укажите параметр **-print0** в команде **find**. Это приводит к тому, что список файлов разделяется нулевым символом, а не символом новой строки, что позволяет использовать имена файлов с пробелами в них как одно имя файла (в противном случае файл с именем **hello there** будет рассматриваться как два файла, один названный **hello** и другой названный **there**).

Чтобы команда **cpio** обработала список файлов, разделенных нулями, добавьте параметр **--null**. Это приводит к более надежной версии предыдущей сквозной команды, которая выглядит следующим образом:

```
find . -print0 | cpio --null -vd /tmp/destination
```

Чтобы понять, почему команда **cpio** может использоваться для копирования файлов из одного каталога в другой вместо рекурсивного использования команды **cp**, рассмотрите следующие преимущества:

Команда **cpio** автоматически сохраняет атрибуты файла (метаданные), такие как ссылки, разрешения, метки времени и владельцы. Эти атрибуты не сохраняются при использовании команды **cp**.

Команда **cpio** также работает со специальными файлами лучше, чем команда **cp**.

<https://www.computerhope.com/unix/ucpio.htm>

<https://rtfm.co.ua/linux-cpio-управление-архивами-cpio/>

## Команда dd

Команда **dd** - это утилита для копирования файлов или целых разделов на битовом уровне. Эта команда имеет несколько полезных функций, в том числе:

- Его можно использовать для клонирования или удаления (стирания) целых дисков или разделов.
- Его можно использовать для копирования необработанных данных на съемные устройства, такие как USB-накопители и компакт-диски.
- Он может создавать резервные копии и восстанавливать MBR (Master Boot Record), критический программный компонент, который используется для загрузки системы.
- Его можно использовать для создания файла определенного размера, заполненного двоичными нулями, который затем можно использовать в качестве файла подкачки (виртуальной памяти).

Команда dd использует специальные аргументы, чтобы указать, как она будет работать. Следующее иллюстрирует некоторые из наиболее часто используемых аргументов:

**if** - входной файл для чтения.

**of** - выходной файл для записи.

**bs** - размер блока, который будет использоваться. По умолчанию значение считается в байтах. Используйте следующие суффиксы, чтобы указать другие единицы измерения: K, M, G и T для килобайт, мегабайт, гигабайт и терабайт.

**count** - количество блоков для чтения из входного файла.

В следующем примере файл с именем **/tmp/swapex** создается с 500 блоками нулей размером «один мегабайт»:

```
dd if=/dev/zero of=/tmp/swapex bs=1M count=500
```

```
root@L-FW:~# dd if=/dev/zero of=/tmp/swapex bs=1M count=500
500+0 records in
500+0 records out
524288000 bytes (524 MB, 500 MiB) copied, 4.31503 s, 122 MB/s
```

При копировании на целые устройства не нужно указывать размер или количество блоков. Например, для клонирования с одного жесткого диска (**/dev/sda**) на другой (**/dev/sdb**) выполните следующую команду:

```
dd if=/dev/sda of=/dev/sdb
```

Команда dd может даже использоваться для создания резервной копии образа **.iso** вашего CDROM или устройства DVD. Далее будут взяты все данные с DVD (**/dev/dvd**) и сохранены данные в локальном файле с именем **dvd.iso**:

```
dd if=/dev/dvd of=dvd.iso
```

Файлы устройств - это файлы, используемые для обозначения устройств в системе, таких как жесткие диски, компакт-диски и разделы. Следующая информация предоставлена, чтобы внести ясность в примеры, показанные с помощью команды dd:

**/dev/sda** - файл устройства, который обычно ссылается на первый жесткий диск в системе.

**/dev/sdb** - файл устройства, который обычно относится ко второму жесткому диску в системе.

**/dev/dvd** - файл устройства, который обычно ссылается на первый привод DVD в системе.