

## Глава 10. Резервное копирование.

### 10.1. Планирование резервного копирования.



#### Планирование резервного копирования

- Что копируем
- Куда копируем
- Как часто
- Частота ротации носителей
- Кто копирует
- Контроль со стороны человека
- В какие моменты времени
- Хранение носителей резервных копий;
- Порядок восстановления утраченных данных
- Время на копирование
- Время на восстановление

Резервное копирование - это единственная операция, которая может гарантировать сохранность данных, находящихся в вычислительной системе.

*Примечание:* Регулярная проверка целостности файловых систем может обеспечить их работоспособность, но она вовсе не гарантирует сохранность данных. Мероприятия по исключению несанкционированного доступа к системе могут снизить вероятность разглашения конфиденциальной информации, но они также не гарантируют ее сохранности (хотя и повышают вероятность ее сохранности).

План мероприятий по резервному копированию данных является одной из важнейших составляющих системной политики.

При планировании резервного копирования следует определить следующие моменты:

1. какие данные должны быть скопированы и где они находятся (в каких каталогах или файловых системах, а также, возможно, на каких компьютерах);
2. какой тип носителей данных будет использован для резервного копирования;
3. с какой периодичностью будет производиться копирование данных;
4. каким образом и с какой частотой будет осуществляться ротация резервных носителей данных;
5. кто будет уполномочен производить резервное копирование;
6. будет ли требоваться человеческий контроль за ходом копирования или же копирование будет производиться полностью автоматически;
7. в какие моменты времени должно осуществляться резервное копирование;

## Глава 10. Резервное копирование.

8. где и как будет производиться хранение носителей резервных копий;

9. каков будет порядок восстановления утраченных данных из резервных копий;

10. максимальная допустимая продолжительность периода времени, необходимого для восстановления данных.

*Примечание:* Одной из наиболее сложных проблем при выборе стратегии резервного копирования является то, что перечисленные выше вопросы являются взаимосвязанными. Так, например, частота ротации копий зависит и от вида используемого носителя и от объема копируемой информации и от частоты ее копирования. И, все-же, в процессе выбора стратегии следует выбрать набор приоритетов, которые и определяют, как эта стратегия будет реализована. Так, например, важнейшим критерием при выборе типа носителя для копирования может явиться суммарный объем копируемых данных.

В процессе принятия решения о том, какие файловые системы и с какой периодичностью следует копировать, необходимо принять во внимание тот факт, что большинство GNU/Linux систем следует стандарту FHS.

*Примечание:* Этот стандарт предписывает сохранять редко изменяемые файлы во вторичной файловой иерархии `/usr`. И, наоборот, часто изменяемые данные обычно находятся в каталогах `/var` и `/home`. Естественно, в зависимости от специфики системы, может быть использована различная тактика резервного копирования, но чаще всего резервному копированию подлежат либо целые файловые системы, либо каталоги с подкаталогами. Гораздо реже встречается необходимость регулярного резервного копирования отдельных файлов.

В большинстве систем содержимое следующих каталогов меняется достаточно редко после установки, обновления или настройки системы, поэтому резервное копирование их содержимого можно провести сразу же после этого.

- `/boot` - файлы, необходимые для загрузки ядра Linux;
- `/etc` - конфигурационные файлы, база данных паролей и скрипты инициализации системы и запуска демонов (учетные записи пользователей в системе могут меняться очень часто);
- `/dev` - файлы устройств (в GNU/Linux системах, ориентированных на использование `udev` создаются автоматически, поэтому архивирование этого каталога в таких системах не имеет большого смысла);
- `/lib*` - основные жизненно важные библиотеки;
- `/opt` - опциональное программное обеспечение, не входящее в поставку операционной системы (в некоторых дистрибутивах в этот каталог устанавливается дополнительное программное обеспечение уже при инсталляции операционной системы);
- `/bin` и `/sbin` - исполняемые файлы, жизненно важные для системы, в некоторых дистрибутивах это символичные ссылки на `/usr/bin` и `/usr/sbin`;
- `/usr` - вторичная иерархия файловой системы, содержащая статические файлы.

*Примечание:* В каталоге `/etc` содержится конфигурационная информация и база данных учетных записей, следовательно его содержимое меняется гораздо чаще, чем содержимое других каталогов, перечисленных выше. Поэтому его содержимое следует копировать после всех значимых изменений файлов, содержащихся в нем. Более того, следует обеспечивать сохранение и нескольких предыдущих копий, так как в текущих настройках системы может быть обнаружен изъян, который не проявился сразу после изменения конфигурации.

Большинство каталогов с программным обеспечением и библиотеками устанавливается с дистрибутивных носителей, поэтому, при условии их сохранности, резервное копирование этих каталогов

## Глава 10. Резервное копирование.

*совсем не обязательно за редкими исключениями. Такими исключениями, например, могут явиться каталоги /usr/local и /opt, в которые может быть установлено программное обеспечение не из дистрибутивного комплекта.*

Применение того или иного вида аппаратуры для резервного копирования определяется такими факторами, как:

1. стоимость аппаратуры и носителей;
2. количество информации, которое может быть скопировано на носитель;
3. возможность повторной записи на носитель и допустимое количество циклов записи-чтения;
4. скорость копирования данных на носитель и считывания с него;
5. возможность автоматической смены носителей в случае заполнения носителя.

Резервное копирование данных должно производиться тогда, когда эти данные не изменяются. В противном случае нельзя гарантировать сохранности этих данных.

Желательно регулярно выполнять резервное копирование в часы минимальной нагрузки на систему.

*Примечание:* Обычно минимальная нагрузка имеет место в диапазоне от 2 часов ночи до 5 утра (в зависимости от специфики работы системы).

Резервное копирование может производиться в интерактивном режиме, требующем присутствия человека (attended backup).

*Примечание:* Преимуществом такого режима является возможность принятия человеком каких-либо решений в процессе копирования и выполнения им каких-либо действий (например, смены ленты в ленточном накопителе). Недостаток же очевиден - с человека не снимается рутинный труд, который, возможно, требуется выполнять в нерабочее время. Чаще всего такой режим копирования применяется при каких-либо незапланированных ситуациях и он не связан с копированием очень большого объема данных.

Другой вид резервного копирования - не интерактивный (unattended backup). В этом режиме обеспечивается полностью автоматическое копирование данных без участия человека.

*Примечание:* Для обеспечения такого режима объем носителя должен превышать объем данных. При большом объеме копируемых данных следует использовать аппаратуру для автоматической смены заполненных носителей. Не интерактивное резервное копирование идеально подходит для заранее запланированных регулярных процедур копирования, осуществляемых в нерабочее время.

Один из параметров плана резервного копирования время требуемое для восстановления данных, поскольку обычно резервное копирование есть процедура регулярная, а восстановление данных требуется в случае сбоя системы, который заранее предсказать невозможно.

Суммарное время на восстановление данных состоит как минимум из:

1. времени, необходимого для поиска и доставки носителя резервной копии требуемых данных;
2. времени на поиск требуемой информации на носителе (если поиск допускается);
3. времени копирования информации с резервного носителя.

Если в архивной копии находится полная копия файловой системы, то время восстановления с нее может быть недопустимо большим. Поэтому различают два типа резервного копирования:

## Глава 10. Резервное копирование.

1. полное - при котором в архив помещается полная копия файловой системы;
2. инкрементальное - при котором архив разбивается на несколько частей, в которых хранятся копии файлов, изменившихся с момента последнего полного или инкрементального резервного копирования.

В GNU/Linux для инкрементального резервного копирования файловых систем EXT применяется утилита `dump`.

Резервные копии, которые создаются этой утилитой разделяются в соответствии с так называемым уровнем резервного копирования, который представляет собой число от нуля до девяти.

Копия нулевого уровня является полной копией файловой системы.

В резервные копии уровней выше нулевого помещаются файлы, созданные или измененные с момента последнего резервного копирования этого же или меньшего уровня.

**Пример:** если в ночь с субботы на воскресенье было произведено резервное копирование второго уровня, в понедельник и вторник - четвертого, а в среду - третьего; то в архиве, созданном во вторник окажутся файлы, созданные или измененные, по сравнению с понедельником и воскресеньем, а в архиве среды - все файлы, созданные или измененные с воскресенья. Таким образом, чем выше уровень резервной копии, тем меньше файлов в нем оказывается.

Чаще всего `dump` запускается автоматически на регулярной основе в моменты наименьшей загрузки системы. Для этого разрабатывается календарь резервного копирования.

Кроме `dump` имеются специализированные программы для инкрементального резервного копирования в том числе и коммерческие.

Подробное обсуждение утилиты `dump` выходит за рамки данного курса.

## 10.2. Команда dd.



### Команда dd

- Команда `dd` осуществляет блочное копирование из файла в файл
- Часто применяется для создания образов дисков
- Основные опции
  - `if` — источник
  - `of` — местоназначение
  - `bs` — размер блока
  - `count` — количество блоков

Команда `dd` осуществляет низкоуровневое поблочное копирование из файла, указанного в командной строке после префикса `if=`, в файл, указанный после префикса `of=`.

По умолчанию используются блоки размером 512 байт.

Наиболее часто используются следующие опции команды `dd`:

1. `bs` — размер блока данных;
2. `count` - количество блоков, которое должно быть скопировано;
3. `skip` - количество блоков во входном файле, которое должно быть пропущено при копировании;
4. `seek` - количество блоков в выходном файле, после которого должны быть записаны входные блоки.

Обычно команда `dd` не применяется для резервного копирования файловых систем, однако она с успехом может быть использована для создания образов файловых систем и для восстановления файловых систем из образов.

**Пример:** для поблочного копирования содержимого дискеты в файл можно использовать команду:

```
dd if=/dev/fd0 of=image.img bs=1k
```

*Примечание:* В этом примере все содержимое дискеты блоками размером 1 Кб (аргумент `1k`) копируется в файл `image.img`.

Файловые системы можно копировать с помощью `dd` в размонтированном состоянии, если только специально не требуется обратное.

## Глава 10. Резервное копирование.

Если аргументы с префиксами `if=` и `of=` отсутствуют, то команда `dd` копирует поток ввода в поток вывода.

Интересной возможностью команды `dd` является способность преобразовывать байты в потоке. Для этого необходимо использовать опцию `conv`.

**Пример:** для вывода списка пользователей, вошедших в сеанс, в верхнем регистре можно использовать следующую команду:

```
$ who | dd conv=ucase 2> /dev/null
USER1      :0                JAN 10 15:17
USER1 PTS/0                JAN 10 15:17 (:0.0)
```

*Примечание:* В этом примере вывод команды `who` преобразуется с помощью команды `dd` в верхний регистр. Поток вывода ошибок перенаправляется в нуль-устройство для подавления вывода информации о количестве скопированных блоков.

### 10.3. Утилиты для сжатия файлов.



#### Утилиты для сжатия файлов

- Утилиты сжатия сжимают каждый файл в отдельности
- Разные утилиты имеют схожий синтаксис команд, но разные алгоритмы работы
- Чем лучше сжимает, тем больше процессорных ресурсов требуется для сжатия
- Основные утилиты:
- `gzip` — самая быстрая, но не сильное сжатие
- `bzip2` — сжимает лучше чем `gzip`, но чуть медленнее
- `xz` — сжимает сильно, но требуется больше ресурсов CPU

В UNIX и GNU/Linux системах команды архивирования отделены от утилит сжатия файлов.

Все утилиты сжатия осуществляют компрессию файлов, указанных в качестве аргументов. При этом к исходным названиям файлов добавляются стандартные суффиксы, перечисленные ниже, а права доступа и владения сохраняются.

В GNU/Linux используются следующие утилиты сжатия:

1. `gzip` - используется наиболее часто, сжатые файлы имеют суффиксы `.gz`.
2. `xz (lzma)` — использует алгоритм LZMA, что обеспечивает очень высокую степень сжатия, сжатые файлы имеют суффиксы `.xz`.
3. `bzip2` - обеспечивает лучшую степень сжатия, чем `gzip`, но хуже `xz`, сжатые файлы имеют суффиксы `.bz2`.
4. `compress` - стандартная UNIX утилита сжатия, в GNU/Linux используется реже, чем предыдущие, сжатые файлы имеют суффиксы `.Z`.

**Пример:** Для сжатия файлов достаточно указать их в качестве аргументов:

```
$ ls -l distfiles.*
-rw-r--r--    1 user1   user1      239263  Авг 26 23:10 distfiles.lst
-rw-r--r--    1 user1   user1     1967881  Авг 22 15:01 distfiles.txt

$ gzip distfiles.*

$ ls -l distfiles.*
-rw-r--r--    1 user1   user1      63298  Авг 26 23:10 distfiles.lst.gz
-rw-r--r--    1 user1   user1     187311  Авг 22 15:01 distfiles.txt.gz
```

## Глава 10. Резервное копирование.

*Примечание:* В этом примере командой **gzip** были сжаты два файла. После сжатия к их названиям был добавлен суффикс **.gz**.

Если необходимо получить информацию о сжатых файлах **gzip**, то следует использовать опцию **-l**

### **Пример:**

```
$ gzip -l distfiles.*
      compressed      uncompressed   ratio uncompressed_name
      63298           239263      73.6% distfiles.lst
      187311          1967881     90.5% distfiles.txt
      250609          2207144     88.6% (totals)
```

Ниже приведены наиболее часто используемые опции команды **gzip**:

1. **-d** - произвести декомпрессию сжатых файлов, как **gunzip**;
2. **-c** - копируют сжатое содержимое файлов в стандартный поток вывода, исходные файлы остаются неизменными;
3. **-r** - рекурсивно сжимать содержимое каталога;
4. **-S** - установить иной, чем **.gz**, суффикс;
5. **-t** - тестировать содержимое архива;
6. **-v** - выводить подробную информацию о процессе работы команды.

**Пример:** следующая команда тестирует сжатые файлы:

```
$ gzip -tv distfiles.*
distfiles.lst.gz:      OK
distfiles.txt.gz:      OK
```

Для декомпрессии сжатых **gzip** файлов удобно применять команду **gunzip**.

### **Пример:**

```
$ gunzip -v distfiles.*
distfiles.lst.gz:      73.6% -- replaced with distfiles.lst
distfiles.txt.gz:      90.5% -- replaced with distfiles.txt
```

Команда **zcat** выводит в стандартный поток вывода содержимое сжатых **gzip** файлов.

В командах **xz** и **bzip2** реализован иной алгоритм сжатия, чаще всего обеспечивающий более высокий уровень компрессии.

Чем выше степень сжатия, тем больше потребуется процессорного времени для работы.

Многие опции команды **xz** и **bzip2** функционально идентичны соответствующим опциям **gzip** (но не все!):

### **Пример:**

```
$ time gzip -c big_file.txt > big_file.txt.gz

real 0m2,354s
```



## Глава 10. Резервное копирование.

```
user 0m0,892s
sys 0m0,040s
```

```
$ time bzip2 -c big_file.txt > big_file.txt.bz2
```

```
real 0m7,613s
user 0m7,546s
sys 0m0,041s
```

```
$ time xz -c big_file.txt > big_file.txt.xz
```

```
real 0m9,674s
user 0m9,563s
sys 0m0,078s
```

```
$ stat -c '%s %n' big_file.txt*
49765537 big_file.txt
3313476 big_file.txt.bz2
4282449 big_file.txt.gz
3232704 big_file.txt.xz
```

*Примечание: Если сравнить результаты работы, то заметно, что степень сжатия у xz команды выше, однако xz и работает медленнее остальных.*

```
$ time xz -T4 -c big_file.txt > big_file.txt.xz
```

```
real 0m5,575s
user 0m10,025s
sys 0m0,177s
```

```
$ time xz -T8 -c big_file.txt > big_file.txt.xz
```

```
real 0m5,642s
user 0m10,159s
sys 0m0,174s
```

*Примечание: даже запущенная в несколько потоков xz (опция -T) все равно медленнее gzip.*

## 10.4. Команда tar.



### Команда tar

- Режим работы tar задается первой опцией:
- c — create - создать
- x — extract - извлечь
- t — test - протестировать
- A — concatenate — объединить архивы
- u — update - обновить
- d — diff - сравнить архив и файлы
- r — append - добавить к архиву
- Опция f указывает файл с архивом, без нее будет использована лента

Один из наиболее часто используемых инструментов резервного копирования - команда tar (tape archive).

С помощью этой команды можно создавать архивы файлов и каталогов, заданных ей в качестве аргументов.

Команда tar поддерживает BSD стиль задания опций, поэтому перед опциями можно не указывать знак дефиса - .

В команде tar одна из опций задает режим работы. Такая опция в команде может быть только одна, и обычно она указывается первой.

Наиболее важные из операционных опций:

1. -c, --create — создать архив, существующий файл перезаписывается;
2. -x, --extract, --get — извлечь файлы из архива;
3. -t, --list — просмотреть архив;
4. -r, --append — добавить файлы к архиву.

Опции -f команды tar указывает файл, в который будет помещен или извлечен архив.

Опция -v заставляет команду tar выводить информацию об обрабатываемых файлах.

**Пример:** приведенная ниже команда поместит архив, содержащий всю информацию в каталоге /home , на магнитную ленту.

```
# tar cvf /dev/st0 /home
```

*Примечание:* После выполнения этой команды на магнитной ленте будет создан архив файлов в

## Глава 10. Резервное копирование.

каталоге (со всеми подкаталогами) /home. Для работы с магнитными лентами предназначены символьные файлы устройств, поэтому информация на них записывается в виде потока байт.

В различных UNIX системах поведение команды tar может отличаться в деталях.

**Пример:** используемая в GNU/Linux версия tar по умолчанию не сохраняет в архиве абсолютные имена файлов, удаляя из них лидирующую косую черту - имя корневого каталога. Так, имя файла /home/user1/.bashrc будет преобразовано в home/user1/.bashrc, что позволяет разархивировать содержимое архива в текущем каталоге.

Команда tar позволяет создавать архивы и сразу сжимать их при создании утилитами gzip при использовании опции -z, xz при использовании опции -J или bzip2 при использовании опции -j. Могут быть использованы и другие утилиты сжатия.

**Пример:** содержимое каталогов /bin и /sbin помещено в архив binaries.tar.gz

```
$ tar czvf binaries.tar.gz /{,s}bin
```

*Примечание:* Стандартным расширением для архивов tar является .tar. Если архив сжат, то к имени архива принято добавлять соответствующий суффикс (например, .gz).

Для просмотра содержимого архива следует использовать опцию -t. Если архив сжат, то необходимо установить опцию, соответствующую примененной утилите сжатия.

**Пример:** для просмотра содержимого архива binaries.tar.gz следует использовать команду:

```
$ tar tzvf binaries.tar.gz
```

Для извлечения файлов из архива следует использовать опцию -x. При использовании GNU версии команды tar содержимое архива будет извлечено в текущем каталоге. При этом будут созданы все подкаталоги, которые находятся в архиве.

**Пример:** команда, показанная ниже, развернет содержимое архива binaries.tar.gz в текущем каталоге:

```
$ tar xzvf binaries.tar.gz
```

Если вместо имени файла архива после опции -f в командной строке tar указан знак дефиса -, то архив будет взят из стандартного потока ввода.

**Пример:** приведенная ниже команда, по действию аналогична предыдущей:

```
$ gzip -dc binaries.tar.gz | tar xvf -
```

*Примечание:* В этом примере архив сначала подвергается декомпрессии командой gzip, причем использование опции -c заставляет gzip выводить имя обработанного файла в стандартный поток вывода. Из него через конвейер имя разжатого файла передается на стандартный поток ввода команде tar, которая извлекает файлы из уже не сжатого архива.

Утилита tar по умолчанию сохраняет только стандартные атрибуты файлов, например права доступа, владельца и пр.. Если необходимо также сохранять расширенные атрибуты, то нужно явно это указать в команде. Это опции --acls, --selinux, --xattrs.

## 10.5. Команда `cpio`.



### Команда `cpio`

- Команда работает со стандартными потоками
- Часто для создания списка файлов, которые подлежат архивации используется команда `file`
- Три основных режима `cpio`
  - `-o` (`copy-out`) — в архив
  - `-i` (`copy-in`) — из архива
  - `-p` (`pass-through`) — копирование в каталог

Команда `cpio` является вторым по значимости инструментом архивирования после `tar`.

*Примечание:* Это связано с тем, что команда `cpio` позволяет передавать через стандартный поток ввода имена файлов как для архивирования, так и имена архивов для извлечения из них файлов.

Команда `cpio` способна работать в трех режимах, определяемых опциями:

1. `-o` - для копирования в архив (`copy-out`), в котором архивируемые файлы помещаются в архив, а сам поток байт архива копируется в выходной (`output`) файл;
2. `-i` - для копирования из архива (`copy-in`), в котором файлы извлекаются из архива, который передается команде на вход (`input`);
3. `-p` - проходном (`pass-through`), при использовании которого файлы копируются из одного каталога в другой без реального создания архива.

Чаще всего список файлов, которые должны быть помещены в архив, передается на стандартный поток ввода команды `cpio` с помощью команды `find`. Это и определяет исключительную гибкость выбора файлов, которые необходимо поместить в `cpio` архив.

### Пример:

```
# find ~tania -mtime 1 | cpio -ov > tania.cpio
```

*Примечание:* Эта команда найдет все файлы в домашнем каталоге пользователя `tania`, измененные в течении последних 24 часов, передаст их список в стандартный поток ввода утилиты `cpio`, которая поместит их в файл архива `tania.cpio`, перенаправив поток вывода блоков архива в этот файл. Опция `-v` используется для получения подробной информации о процессе архивирования. В этом случае команда `cpio` работала в режиме `copy-out` (опция `-o`).

## Глава 10. Резервное копирование.

### **Пример:**

```
# cpio -ivt < tania.cpio
```

*Примечание:* В этом примере содержимое архива было считано командой `cpio` из стандартного потока ввода. Команда `cpio` работала в режиме `copy-in` (опция `-i`). Для вывода в стандартный поток вывода списка файлов в архиве и информации о них требуется воспользоваться дополнительной опцией `-t` (*type*).

Извлечь файлы из архива можно с помощью опций `-i` и `-d`, вынуждающих команду `cpio` создавать отсутствующие каталоги при восстановлении файлов, так как по умолчанию этого не делается.

**Пример:** следующая команда разархивирует в текущем каталоге файлы из `tania.cpio`:

```
# cpio -ivd < tania.cpio
```

Ниже приводится список других часто используемых опций `cpio`:

1. `-B` - устанавливает размер блока для архивирования 5120 байт вместо 512 байт по умолчанию;
2. `--block-size` - устанавливает размер блока для архивирования `px` 512 байт вместо 512 байт по умолчанию;
3. `-F` - указывает имя файла для ввода или вывода архива;
4. `-O` - указывает имя файла для записи архива;
5. `-I` - указывает имя файла для считывания архива;
6. `-A` - добавление файлов к архиву, указанному после опции `-F` или `-O`;
7. `-E` - извлекать из архива только те файлы, имена которых удовлетворяют заданному после этой опции шаблону;
8. `-f` - не копировать файлы, имена которых удовлетворяют заданному после этой опции шаблону;
9. `-n` - не переводить UID и GID в имена пользователей и групп;
10. `-r` - интерактивно переименовывать файлы;
11. `-u` - безусловно заменять новыми существующие файлы.