```
[user@centos8 ~]$ du -h /backup/*.gz
178M    /backup/centos8_13.03.2021_20.51.tar.gz
178M    /backup/centos8_13.03.2021_20.52.tar.gz
178M    /backup/centos8_13.03.2021_20.53.tar.gz
```

В прошлый раз мы с вами настроили простой скрипт для бэкапа. Он будет делать полную копию нашей домашней директории и хранить её в сжатом виде. Такой тип бэкапа, когда копируется всё что нужно, называется полным бэкапом. Например, мы бэкапили домашнюю директорию - du -h /backup/*.gz - и каждый раз копировалось всё - поэтому размеры файлов совпадают. Но, если подумать - за день никакие файлы не изменились, зачем нам ещё одна копия? Она занимает столько же места, а смысла от неё нет. Да и представьте, что у нас изменился один файл из 5 тысяч, а мы всё равно будем делать полный бэкап - копировать все 5 тысяч файлов. Мы просто теряем место.

```
[user@centos8 ~]$ du -h Pictures/
18M     Pictures/
[user@centos8 ~]$ tar -czf pictures.tar.gz -g pictures.snar -C Pictures/ .
[user@centos8 ~]$ du -h pictures.*
4.0K    pictures.snar
18M     pictures.tar.gz
[user@centos8 ~]$
```

Почему бы не бэкапить только изменённые файлы? Такой тип бэкапа называется инкрементальным. У нас есть один полный бэкап, а при следующем будут сохраняться только изменённые файлы. Давайте мы это протестируем. Возьмём какую-нибудь директорию, например, Pictures - du -h Pictures. Для начала нужно создать полный бэкап. Но на этот раз нам понадобится дополнительный ключ - g - который указывает на файл с метаданными. По ним tar решает, какие файлы были изменены и в дальнейшем делает инкрементальный бэкап. Если файла с метаданными нету, то он делает полный бэкап и создаёт этот файл  - tar -czf pictures.tar.gz -g pictures.snar -C Pictures/ .; du -h pictures.*. Как видите, создался сжатый архив и файл snar - именно в нём хранятся метаданные.

```
[user@centos8 ~]$ tar -czf pictures2.tar.gz -g pictures.snar -C Pictures/ .
[user@centos8 ~]$ du -h pictures2.tar.gz
4.0K    pictures2.tar.gz
[user@centos8 ~]$ tar -tf pictures2.tar.gz
./
```

Теперь попытаемся создать ещё один бэкап - tar -czf pictures2.tar.gz -g pictures.snar -C Pictures/ .; du -h pictures2.tar.gz; tar -tf pictures2.tar.gz. Архив создался, но он весит всего 4 килобайта и пустой. То есть, если у нас за день никаких изменений не будет, мы не потратим место на диске на новый бэкап.

```
[user@centos8 ~]$ tar -tf pictures2.tar.gz -g /dev/null -vv
drwxr-xr-x user/user      2666 2021-03-23 10:45 ./
N Screenshot from 2020-05-30 15-48-27.png
N Screenshot from 2020-05-30 15-48-31.png
N Screenshot from 2020-05-30 17-05-15.png
N Screenshot from 2020-05-31 08-37-14.png
N Screenshot from 2020-06-14 18-20-24.png
N Screenshot from 2020-11-04 00-06-06.png
N Screenshot from 2020-11-04 00-06-09.png
```

Когда речь идёт об инкрементальных архивах, нужна опция -g, но для извлечения и просмотра нет нужды указывать файл snar, так как в самом архиве есть эта информация - tar -tf pictures2.tar.gz -g /dev/null -vv. Поэтому вместо snar принято указывать /dev/null, а два -vv позволят увидеть добавленные или не включённые в архив файлы.

```
[user@centos8 ~]$ cp /etc/passwd Pictures/
[user@centos8 ~]$ rm Pictures/Screenshot\ from\ 2020-0
Screenshot from 2020-05-30 15-48-27.png
Screenshot from 2020-05-30 15-48-31.png
Screenshot from 2020-05-30 17-05-15.png
Screenshot from 2020-05-31 08-37-14.png
Screenshot from 2020-06-14 18-20-24.png
[user@centos8 ~]$ rm Pictures/Screenshot\ from\ 2020-05-31\ 08-37-14.png
[user@centos8 ~]$ tar -czf pictures3.tar.gz -g pictures.snar -C Pictures/ .
[user@centos8 ~]$ du -h pictures3.tar.gz
4.0K    pictures3.tar.gz
[user@centos8 ~]$ tar -tf pictures3.tar.gz
./
./passwd
```

Давайте добавим в директорию Pictures новый файл и удалим какой-то из существующих - cp /etc/passwd Pictures/; rm Pictures/Screenshot.... Потом создадим ещё один бэкап - tar -czf pictures3.tar.gz -g pictures.snar -C Pictures/ . ; du -h pictures3.tar.gz; tar -tf pictures3.tar.gz. Как видите, в новом архиве появился файл passwd.

```
[user@centos8 ~]$ tar -xf pictures.tar.gz -g /dev/null -C Pictures/ -v
./
tar: Deleting ‘./passwd’
./Screenshot from 2020-05-30 15-48-27.png
./Screenshot from 2020-05-30 15-48-31.png
./Screenshot from 2020-05-30 17-05-15.png
./Screenshot from 2020-05-31 08-37-14.png
./Screenshot from 2020-06-14 18-20-24.png
./Screenshot from 2020-11-04 00-06-06.png
```

Теперь рассмотрим процесс восстановления. Для этого нужно поочерёдно восстанавливать бэкапы до требуемого дня. Сперва восстановим исходный архив - tar -xf pictures.tar.gz -g /dev/null -C Pictures -v. Как видите, он удалил файл passwd, так как его изначально не было.

```
[user@centos8 ~]$ tar -xf pictures3.tar.gz -g /dev/null -C Pictures/ -v
./
tar: Deleting ‘./Screenshot from 2020-05-31 08-37-14.png’
./passwd
[user@centos8 ~]$ ls Pictures/passwd
Pictures/passwd
[user@centos8 ~]$
```

Смысла восстанавливать второй архив нет, так как там не было никаких изменений, сразу восстановим третий - tar -xf pictures3.tar.gz -g /dev/null -C Pictures -v. Таким образом, у нас удалился тот самый скриншот и восстановился файл passwd.

```
[user@centos8 ~]$ du -h pictures*
4.0K    pictures2.tar.gz
4.0K    pictures3.tar.gz
4.0K    pictures.snar
18M     pictures.tar.gz
[user@centos8 ~]$
```

И при том, что я создал 3 бэкапа, инкрементальные копии практически ничего не весят - du -h pictures*. Естественно, в реальных условиях они бы занимали больше места, и тем не менее, инкрементальные бэкапы позволяют во много раз сэкономить пространство.

```
TARGET=/home/user
BCUR=/backup/current
BOLD=/backup/old
BDATE=$(date +'%d.%m.%Y_%H.%M')
FILENAME=$(hostname)_$BDATE
EXPIRE=13
```

Теперь попытаемся научить наш скрипт делать инкрементальные бэкапы. Это немного усложнит его, потому что мы не можем просто удалять файлы недельной давности - инкрементальные копии зависят от полного бэкапа, без него они теряют смысл. Но так как мы сэкономили пространство, можем хранить бэкап с прошлой недели. Для начала, чтобы превратить наш скрипт в более универсальный, заменим /home/user на переменную - TARGET=/home/user, разделим логи на ошибки и стандартный вывод - > $FILENAME.log 2> FILENAME.err, заменим директорию BDIR на две - /backup/current и /backup/old, немного подкорректируем FILENAME - FILENAME=$(hostname)_$BDATE, уберём отсюда директорию, а также добавим переменную EXPIRE, по которой и будем удалять старые бэкапы - EXPIRE=13.

```
backup() {
tar -g $BCUR/snar -czvf $BCUR/$FILENAME.tar.gz -C $TARGET . > $BCUR/$FILENAME.log
2> $BCUR/$FILENAME.err
}
delete() {
}
full() {
}
```

Так как удалять по старости бэкапа мы не можем, иначе сотрём полный бэкап, стоит придумать другую схему. Например, раз в неделю архивировать бэкап прошлой недели и создавать новый полный бэкап. Чтобы не решать это в скрипте, а сделать через планировщик задач, в самом скрипте я добавлю 3 функции - backup() {}, delete() {} и full() {}. В функцию backup добавлю нашу старую команду по бэкапу, подкорректированную по директориям и с опцией -g:

```
backup() {
tar -g $BCUR/snar -czvf $BCUR/$FILENAME.tar.gz -C $TARGET . >
$BCUR/$FILENAME.log 2> $BCUR/$FILENAME.err
}
```

```
full() {
tar -czvf $BOLD/$FILENAME.full.tar.gz -C $BCUR . > $BOLD/$FILENAME.full.log
2> $BOLD/$FILENAME.full.err
rm -f $BCUR/*
backup
}
```

Когда будем делать полный бэкап, скрипт заархивирует содержимое директории /backup/current и сохранит его в директории /backup/old. После чего удалит содержимое current директории и создаст новый бэкап, запуская готовую функцию backup:

```
full() {
tar -czvf $BOLD/$FILENAME.full.tar.gz -C $BCUR . >
$BOLD/$FILENAME.full.log 2> $BOLD/$FILENAME.full.err
rm -f $BCUR/*
backup
}
```

```
delete() {
find $BOLD/ -mtime +$EXPIRE -delete
}
```

А при необходимости старые бэкапы будут удаляться из директории /backup/old, используя переменную EXPIRE:

```
delete() {
find $BOLD/ -mtime +$EXPIRE -delete
}
```

```
case $1 in
        incremental) backup ;;
        full) full ;;
        delete) delete ;;
        *) echo "Options: incremental, full, delete" ;;
esac
```

Но пока мы только задали функции, их нужно ещё запускать. Для этого мы будем передавать скрипту аргумент - $1 - и на основе него запускать ту или иную функцию:

```
case $1 in
        incremental) backup ;;
        full) full ;;
        delete) delete ;;
        *) echo "Options: incremental, full, delete" ;;
esac
```

```
[user@centos8 ~]$ rm -rf /backup/*
[user@centos8 ~]$ mkdir /backup/old /backup/current
[user@centos8 ~]$ ./mybackupscript
Options: incremental, full, delete
[user@centos8 ~]$ ./mybackupscript incremental
[user@centos8 ~]$ ls /backup/current/
centos8_23.03.2021_15.31.err  centos8_23.03.2021_15.31.tar.gz
centos8_23.03.2021_15.31.log  snar
[user@centos8 ~]$
```

Теперь давайте проверим работу скрипта. Для начала очистим содержимое директории /backup - rm -rf /backup/* - и создадим необходимые директории - mkdir /backup/old /backup/current. Попробуем просто запустить скрипт - ./mybackupscript. Как видите, он выдаёт информацию об опциях. В нашем случае в первый раз лучше запускать incremental, чтобы не архивировать пустую директорию - ./mybackupscript incremental. После чего можем проверить содержимое директории /backup/current - ls /backup/current/.

```
[user@centos8 ~]$ head -5 /backup/current/*{err,log}
==> /backup/current/centos8_23.03.2021_15.31.err <==
tar: .: Directory is new
tar: ./.cache: Directory is new
tar: ./.config: Directory is new
tar: ./.local: Directory is new
tar: ./.mozilla: Directory is new

==> /backup/current/centos8_23.03.2021_15.31.log <==
./
./.cache/
./.cache/evolution/
./.cache/evolution/WebKitCache/
./.cache/evolution/WebKitCache/Version 14/
[user@centos8 ~]$
```

Если посмотреть лог ошибок - head -5 /backup/current/*{err,log} - можно увидеть, что в ошибки попадают записи "Directory is new". Это не совсем ошибки, просто информация о том, что при прошлом бэкапе этих директорий не было. Но это логично, потому что у нас прошлого бэкапа и не было.

```
[user@centos8 ~]$ touch file100 file101 file102
[user@centos8 ~]$ ./mybackupscript incremental
[user@centos8 ~]$ head -5 /backup/current/centos8_23.03.2021_15.
centos8_23.03.2021_15.31.err     centos8_23.03.2021_15.41.err
centos8_23.03.2021_15.31.log     centos8_23.03.2021_15.41.log
centos8_23.03.2021_15.31.tar.gz  centos8_23.03.2021_15.41.tar.gz
[user@centos8 ~]$ head -5 /backup/current/centos8_23.03.2021_15.41.err
[user@centos8 ~]$ head -5 /backup/current/centos8_23.03.2021_15.41.log
./
./.cache/
./.cache/evolution/
```

```
[user@centos8 ~]$ tail -8 /backup/current/centos8_23.03.2021_15.41.log
./this/
./file100
./file101
./file102
./.cache/tracker/meta.db
./.cache/tracker/meta.db-shm
./.cache/tracker/meta.db-wal
./.local/share/tracker/data/tracker-store.journal
```

Создадим пару файлов - touch file100 file101 file102 и запустим скрипт заново - ./mybackupscript incremental. В err файле больше ошибок нет - потому что нет новых директорий. А в логах перечислены все директории, а также новые и изменённые файлы.

```
[user@centos8 ~]$ ./mybackupscript full
[user@centos8 ~]$ ls /backup/current/
centos8_23.03.2021_15.49.err  centos8_23.03.2021_15.49.tar.gz
centos8_23.03.2021_15.49.log  snar
[user@centos8 ~]$ ls /backup/old/
centos8_23.03.2021_15.49.full.err  centos8_23.03.2021_15.49.full.tar.gz
centos8_23.03.2021_15.49.full.log
[user@centos8 ~]$ cat /backup/old/centos8_23.03.2021_15.49.full.log
./
./centos8_23.03.2021_15.31.log
./centos8_23.03.2021_15.31.err
./snar
./centos8_23.03.2021_15.31.tar.gz
./centos8_23.03.2021_15.41.log
./centos8_23.03.2021_15.41.err
./centos8_23.03.2021_15.41.tar.gz
```

Теперь попробуем запустить полный бэкап - ./mybackupscript full. Как видите - ls /backup/current - создались новые файлы, а старые заархивировались - ls /backup/old; cat /backup/old/*full.log.

```
[user@centos8 ~]$ crontab -l
0 0 * * * /home/user/mybackupscript delete
50 23 * * 1-6 /home/user/mybackupscript incremental
50 23 * * 0 /home/user/mybackupscript full
[user@centos8 ~]$
```

Осталось настроить cron - crontab -e; crontab -l. Пусть каждую ночью find ищет и удаляет старые бэкапы старее 14 дней. С понедельника по субботу пусть выполняется инкрементальное резервное копирование, а в воскресенье полное.

Подведём итоги. Сегодня мы с вами разобрали, что такое полные и инкрементальные бэкапы, зачем они нужны, научились ими пользоваться, а также добавили их в наш скрипт. Это позволяет значительно снизить занимаемое пространство, делать больше копий и в целом делать их быстрее. Но при этом бэкапы становятся зависимы друг от друга, недостаточно просто распаковать один архив, иногда нужно это делать по порядку их создания. Но сэкономленное пространство этого стоит.