

Лабораторная работа №7

Резервное копирование

1. Теоретическая часть

Резервное копирование (англ. *backup copy*) — процесс создания копии данных на носителе (жёстком диске, дискете и т. д.), предназначенном для восстановления данных в оригинальном или новом месте их расположения в случае их повреждения или разрушения. Резервное копирование может предполагать копирование системы (установленных пакетов, конфигурационных файлов) или отдельных данных, хранимых в файловой системе или в СУБД.

Перечислим основные требования к системе резервного копирования:

1. *Хранение на отдельном носителе или в другом месте.*
2. *Надежность места хранения*
3. *Доступность места хранения*
4. *Простота использования*

Коротко поясним указанные пункты. Очевидно, что если файлы с резервными копиями будут храниться на том же сервере, что и данные, то выход из строя всей системы приведет к потере как основных данных, так и резервной копии. Однако, хранение данных очень далеко от основного сервера, например, в другой стране, может привести к недоступности данных в определенное время. Простота использования также важна при работе с небольшими проектами, когда всё администрирование ведется только малой группой людей.

Перед выбором системы резервного копирования необходимо ответить на следующие вопросы:

- 1) От каких сбоев предполагается проводить защиту?

В зависимости от типов сбоев будут происходить разные подходы к резервированию. Если мы хотим защитить небольшое количество данных от физической поломки сервера, то правильным решением будет создание бэкапа на другой сервер, физически расположенный рядом с ним. Однако, такой подход не поможет, например, при возникновении пожара.

- 2) Какие данные необходимо копировать?

Очевидно, что необходимо выделить те данные, которые необходимо сохранять в резервной копии и оставить те, чем можно пожертвовать. Например, правильным решением будет развертывание всех установленных утилит с помощью системы управления конфигурациями, например Ansible. Таким образом сохранять данные о каждой программе не потребуется. Системные логи полезно сохранять с определенной периодичностью (если это требуется) на отдельные, медленные носители. Однако, их наличие не повлияет на восстановление системы, поэтому сохранять их в резервную копию также необязательно.

- 3) Как быстро нужно восстановиться?

Для определения времени для восстановления служит метрика RTO. Recovery Time Objective (показатель времени восстановления) - определяет количество времени с момента наступления разрушительного события до момента, когда затронутые ресурсы будут полностью работоспособны. Чем меньше значение RTO, тем выше затраты на восстановление. Показатель является требованием к системе, например, «время восстановления должно занимать не больше 5 часов».

4) Где должна находиться точка восстановления?

Для определения используется показатель RPO. Recovery Point Objective (показатель точки восстановления) – определяет максимально допустимый промежуток времени, за который данные могут быть восстановлены. Другими словами, это время с момента последнего резервного копирования данных до возникновения инцидента. Например, «Данные необходимо восстановить на состояние не раньше, чем за 5 часов до сбоя».

Очевидно, что чем меньше значения RPO и RTO, тем сложнее процесс восстановления и дороже организация резервного копирования.

5) Где хранить резервные копии?

В качестве оборудования для хранения резервных копий могут выступать как обычные жесткие диски собранные в RAID массив, так и магнитные ленты. Ленты дешевле публичных облаков, надежнее дисковых накопителей, более энергоэффективны, проще в обслуживании, обладают высокой емкостью, обеспечивают "холодное" хранение данных, предоставляют средства защиты от кибератак и более долговечны, чем жесткие диски и SSD-накопители. Однако, такой подход удобно использовать при хранении больших объемов данных – непосредственно оборудование для работы с лентами дорогое и неудобно для использования в небольших проектах. Ленточные накопители удобны при долгосрочном хранении данных, которые могут потребоваться через много лет.

Уровни резервного копирования:

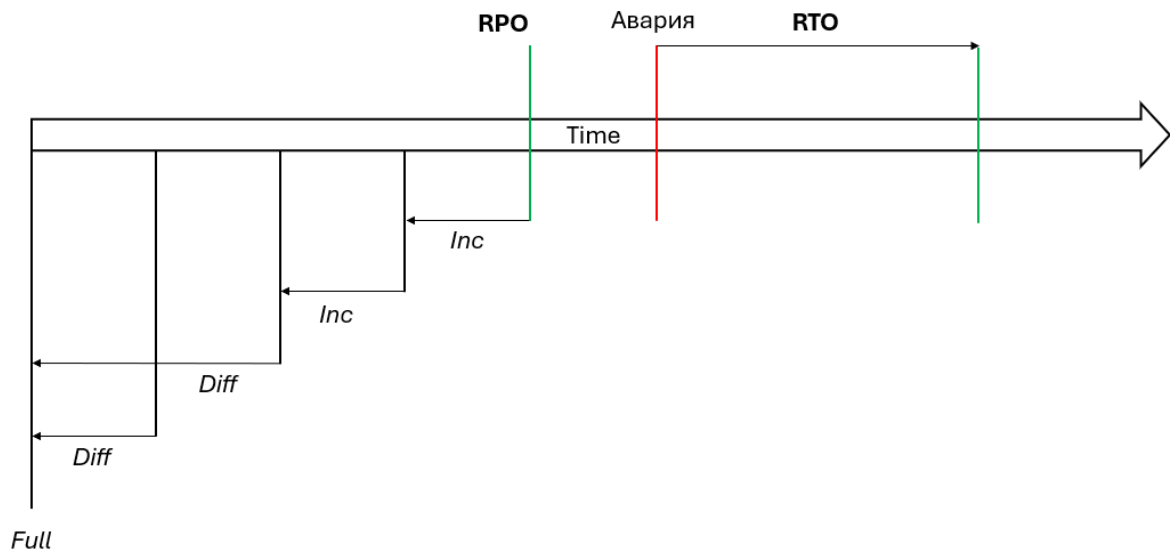
Full - Полное копирование подразумевает копирование всех требуемых файлов вне зависимости от того, были ли произведены их изменения с момента создания. Ежедневное, ежемесячное и ежеквартальное резервное копирование подразумевает создание полной копии всех данных. Обычно оно выполняется тогда, когда копирование большого объема данных не влияет на работу организации. Для восстановления требуется только резервная копия.

Differential - При дифференциальном («разностном») резервном копировании каждый файл, который был изменен с момента последнего полного резервного копирования, копируется каждый раз заново. Дифференциальное копирование ускоряет процесс восстановления. Все копии файлов делаются в определённые моменты времени, что, например, важно при заражении вирусами. Для восстановления требуется последняя полная и последняя дифференциальная копии.

Incremental - При добавочном («инкрементном») резервном копировании происходит копирование только тех файлов, которые были изменены с тех пор, как в последний раз выполнялось полное или добавочное резервное копирование. Последующее инкрементное резервное копирование добавляет только файлы, которые были изменены с момента предыдущего. Инкрементное резервное копирование занимает меньше времени, так как копируется меньшее количество файлов. Однако процесс восстановления данных занимает больше времени, так как должны быть восстановлены данные последнего полного резервного копирования, а также данные всех последующих инкрементных резервных копирований.

На практике полную резервную копию делают один раз в определённый длительный промежуток времени, например, один раз в неделю. Далее в течение времени до следующего полного копирования производится несколько дифференциальных и множество инкрементальных копий.

На схеме представлен пример всех трех уровней резервного копирования. Полная копия была снята всего один раз, далее было произведено два дифференциальных резервных копирования и два инкрементальных. В момент времени, отмеченный красной линией произошла авария. Согласно требованиям к системе, время до её восстановления составит не более, чем RTO, а восстановиться получится на время, равное «времени аварии - RPO».



Отдельно от указанных уровней находится **клонирование**, которое позволяет скопировать целый раздел или носитель (устройство) со всеми файлами и каталогами в другой раздел или на другой носитель. Если раздел является загрузочным, то клонированный раздел тоже будет загрузочным.

Резервное копирование баз данных

Одним из важной области применения резервного копирования является резервное копирование баз данных. Т.к. СУБД являются сложными высоконагруженными системами, то необходимы специальные алгоритмы для обеспечения корректного копирования данных. Далее будут приведены примеры резервного копирования для СУБД Postgres, однако похожие алгоритмы существуют и на других системах управлением баз данных.

Для ускорения работы СУБД использует буферных кэш, хранящийся в оперативной памяти. При нештатной ситуации оперативная память может очиститься и все данные из неё и, в частности, кэша будут потеряны. Для их восстановления используется журнал предзаписи (WAL, write-ahead log). Одновременно с изменениями данных в странице кэша происходит запись в журнале. Она содержит информацию, достаточную для того, чтобы повторить указанную операцию. Такая запись обязательно сохраняется на физическом носителе до того, как на него будет записана эта информация.

Существуют несколько способов резервного копирования баз данных:

1. Логическое резервирование

Логическое резервирование (не путать с логической репликацией!) заключается в выгрузке данных из СУБД на носитель в виде текстового или бинарного файла. Является самым простым способом резервного копирования, однако его основным недостатком является длительное время выполнения и высокая нагрузка на сервер. Также при выгрузке сохраняется только логическая структура

данных – физические объекты, например индексы, не сохраняются. Они восстанавливаются после выгрузки бэкапа и этот процесс может занимать длительное время. Способ удобен для длительного хранения данных или миграции на другую платформу.

2. Физическое резервирование

Физическое резервирование подразумевает копирование всех файлов, относящихся к кластеру БД, то есть создание полной двоичной копии. Основным достоинством подхода является высокая скорость работы, по сравнению с созданием и восстановлением логической копией. Однако, восстановление возможно произвести только на систему с совместимыми характеристиками и невозможность выполнения копирования части базы данных.

Физическое копирование возможно произвести двумя способами: «холодным» и «горячим».

– Холодное резервирование

Предполагает остановку базы данных и создание её копии. Очевидно, что остановка работы является существенным недостатком способа. Холодная копия должна хранить внутри себя кроме данных значения журналов, однако для восстановления они не потребуются (если сервер был выключен штатно).

– Горячее резервирование

Не предполагает остановки базы данных при создании её копии. Очевидной сложностью является то, что в процессе резервирования параллельно данные будут изменяться. Поэтому копирование происходит минимум по двум каналам – по одному передаются сами данные, а по второму – генерируемые с момента старта резервирования журнальные записи.

Репликация

Понятие резервного копирования часто связывают с понятием репликации. Однако следует понимать, что это совершенно другая операция.

Репликация – процесс поддержания двух (или более) наборов данных в согласованном состоянии. Другими словами, существует эталонный набор, каждое изменение которого влечет за собой изменение реплик. Репликация является одной из функций СУБД, когда для базы данных возможно создать её точную реплику, которая позволит восстановить исходную базу данных в случае её потери (отказоустойчивость) и снизит нагрузку за счет переноса части запросов на реплику (масштабируемость).

Существуют следующие подходы к репликации:

Блочная репликация – каждая операция выполняется одновременно на основном носителе и на резервном. Реализуется на уровне системы хранения данных.

Физическая репликация – все изменения на реплике происходят на основе журналов баз данных.

Логическая репликация – все изменения происходят за счет повторения на реплике всех выполненных запросов к БД.

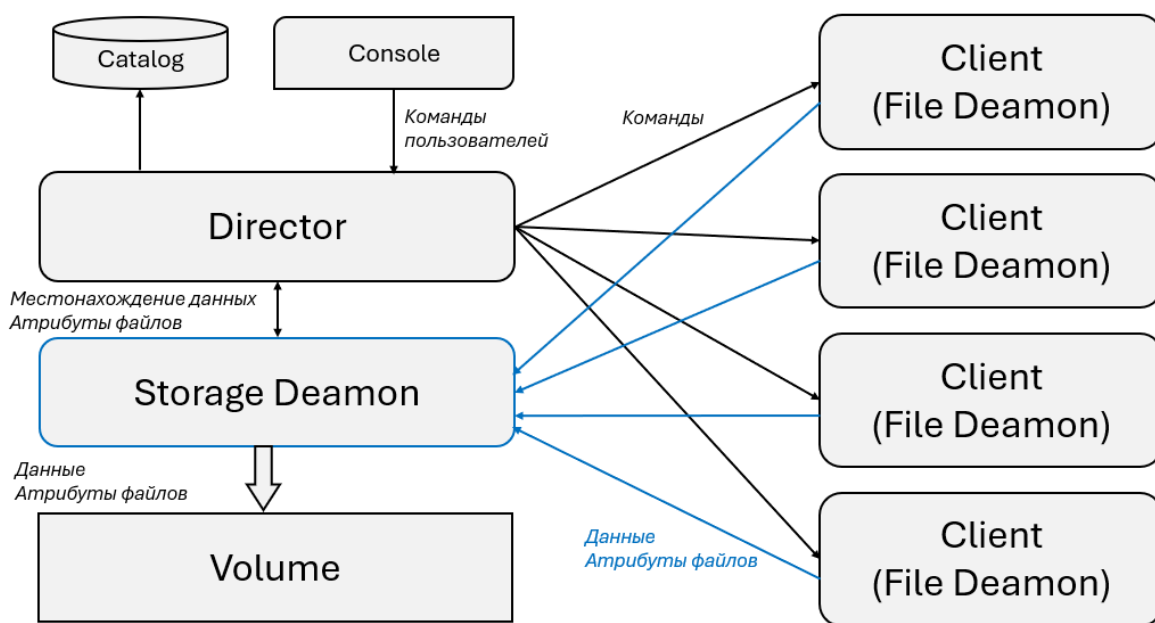
Однако, такой подход не поможет восстановить данные, в случае появления логической ошибки. Например, случайно удалив вручную важную таблицу, эта операция повторится и на реплике. Аналогично, при повреждении внутренних структур баз данных ошибка может перенестись на реплику.

Для решения перечисленных выше проблем применяется именно резервное копирование баз данных.

Bacula

Bacula – это система резервного копирования корпоративного уровня. Она имеет клиент-серверную архитектуру и состоит из следующих компонентов:

- Bacula Director (сервис bacula-dir) – основной сервис, который управляет всеми другими процессами по резервному копированию и восстановлению;
- Bacula Storage (сервис bacula-sd) – хранилище, предназначенное для сохранения резервных копий на диске;
- Bacula File Daemon (сервис bacula-fd) – клиентская часть сервиса, которая нужна для доступа к файлам на сервере, с которого будет выполняться резервное копирование.



Все компоненты могут быть установлены как на одном сервере, так и на разных, но каждый из них должен иметь возможность обратиться к другому по сети. Для управления всем этим используется утилита командной строки `bconsole`. Для неё существует как консольный, так и веб-интерфейс, но основной способ управления – командная строка.

Создание таблицы базы данных для Bacula

Bacula поддерживает большинство СУБД (mysql, postgresql). Подключение к базе данных необходимо для корректной работы сервиса. При запуске сервиса `bacula-director` подключается к базе данных (сведения о имени пользователя пароле и адресе таблицы находится в графе `Catalog` в конфигурационном файле). В ней хранятся сведения обо всех зарезервированных файлах и их местонахождении в резервных копиях. Каталог необходим для обеспечения эффективной адресации к требуемым файлам.

Важно в примере использовалось две виртуальные машины. На первой с ip 192.168.122.2 разворачивались Director и Storage, на второй с ip 192.168.122.3 – file daemon.

Настройка bacula

На сервере установите пакет с программой bacula.

```
sudo apt install bacula
```

Настройка Director

В файле /etc/postgresql/*/main/postgresql.conf указать параметр:

```
listen_addresses='*'
```

В файле /etc/postgresql/*/main/pg_hba.conf указать метод trust для всех. Обязательно добавить host с ip адресом, где будет работать bacula-dir.

local	all	postgres		trust
local	all	all		trust
host	all	all	127.0.0.1/32	trust
host	all	all	192.168.122.2/24	trust

Выполнить перезапуск кластера с БД:

```
sudo pg_ctlcluster 15 main restart
```

Создать пользователя БД для работы с bacula, выполнять не из-под root. Для этого выполните подключение к БД из-под командной строки psql:

```
psql template1 -U postgres -h 192.168.122.2 -p 5432
```

```
template1=# CREATE ROLE bacula;  
template1=# ALTER USER bacula PASSWORD 'bacula';  
template1=# ALTER USER bacula LOGIN SUPERUSER CREATEDB CREATEROLE;
```

Заходим в интерфейс управления psql командой:

```
psql postgres -p 5432 -U postgres
```

Создаем БД командой и устанавливаем её владельца - bacula:

```
postgres=# CREATE DATABASE bacula WITH ENCODING = 'SQL_ASCII' LC_COLLATE =  
'C' LC_CTYPE = 'C' TEMPLATE = 'template0';  
postgres=# ALTER DATABASE bacula OWNER TO bacula;
```

Для корректного функционирования авторизации через PAM, пользователю postgres необходимо выдать права на чтение информации из БД пользователей и сведений о метках безопасности и привилегиях:

```
usermod -a -G shadow postgres  
setfacl -d -m u:postgres:r /etc/parsec/macdb  
setfacl -R -m u:postgres:r /etc/parsec/macdb  
setfacl -m u:postgres:rx /etc/parsec/macdb  
setfacl -d -m u:postgres:r /etc/parsec/capdb  
setfacl -R -m u:postgres:r /etc/parsec/capdb  
setfacl -m u:postgres:rx /etc/parsec/capdb
```

Пользователю bacula задаем минимальный и максимальный уровень:

```
pdpl-user bacula -l 0:0
```

В скриптах /usr/share/bacula-director/make_postgresql_tables и /usr/share/bacula-director/grant_postgresql_privileges вносим изменения:

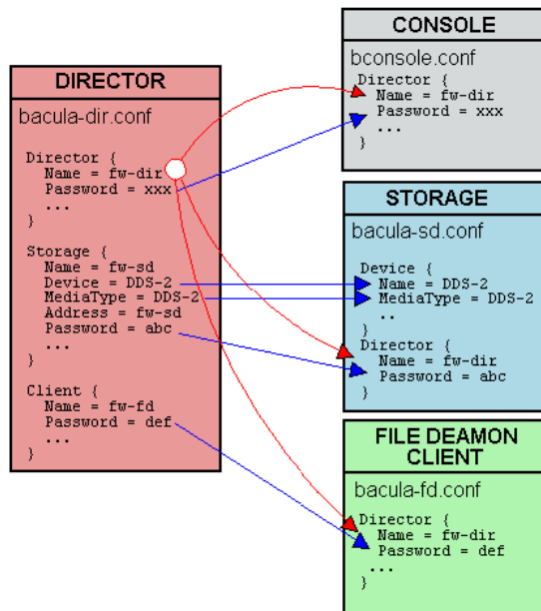
```
db_name=${db_name:-bacula}  
psql -U bacula -h 192.168.122.2 -p 5433 -f - -d ${db_name} $* <<END-OF-DATA
```

Сохраняем изменения.

Для заполнения базы данных воспользуемся скриптами:

```
sudo /usr/share/bacula-director/make_postgresql_tables  
sudo /usr/share/bacula-director/grant_postgresql_privileges
```

Установка и настройка bacula-director



На изображении представлено взаимодействие между элементами утилиты bacula. Обратите внимание, что значения, соединенные стрелочками должны совпадать.

Настройка director производится в файле `/etc/bacula/bacula-dir`. Ниже приведен скрипт настройки, включающий в себя следующие секции:

- Director
- Catalog
- Storage
- Pool
- Messages
- Schedule
- Client
- FileSet

Обратите внимание, что секции `Jobs` расположены в отдельном файле и подгружены с помощью команды, указанной внутри `/etc/bacula/bacula-dir`. Аналогично возможно выделить каждую из перечисленных выше секций в отдельный файл, однако для упрощения работы, в примере они приведены в одном файле.

```
Director {  
  Name = bacula-dir  
  # какой порт слушать (используется значение по умолчанию)
```

```

DIRport = 9101
# путь к сценарию, содержащему sql запросы для работы с Bacula Catalog
QueryFile = "/etc/bacula/scripts/query.sql"
# папка в которой лежат статус-файлы Директора
WorkingDirectory = "/var/lib/bacula"
# pid-файл демона Директора
PidDirectory = "/run/bacula"
# Максимальное количество выполняемых заданий.
# (не рекомендуется одновременно запускать более одного задания)
Maximum Concurrent Jobs = 1
# Пароль Директора
Password = "dirpass"
# Конфигурация параметров уведомлений (описано в секции Messages)
Messages = Standard
# IP-адрес Директора
DirAddress = 192.168.122.2
}

# Подключение к базе данных (Bacula Catalog)
Catalog {
    Name = BaculaCatalog
    # имя базы данных на сервере PostgreSQL
    dbname = "bacula"
    # адрес сервера БД PostgreSQL
    DB Address = "192.168.122.2"
    # порт подключения на сервере
    DB PORT = 5432
    # имя пользователя базы данных на сервере PostgreSQL
    dbuser = "bacula"
    dbpassword = "bacula"
}

# Подключение к Хранилищу (Storage)
Storage {
    Name = stor-sd
    # IP-адрес хранилища.
    Address = 192.168.122.2
    # порт оставляем стандартный
    SDPort = 9103
    Password = "storpass"
    # имя устройства хранения, описанное в файле bacula-sd.conf
    Device = DevStorage
    # Должно соответствовать директиве Media Type ресурса Device настройки
    Media Type = File1
    # Максимальное количество выполняемых заданий.
    # (не рекомендуется одновременно запускать более одного задания)
    Maximum Concurrent Jobs = 1
}

# Настройка групп томов Хранилища (Pool's). Определяет набор носителей информации и параметры
# их обработки
Pool {
    Name = File
    # тип пула
    Pool Type = Backup
    # Bacula может автоматически очищать или перезаписывать тома пула
    Recycle = yes
    # период в течении которого информация о заданиях и файлах хранится в БД
    Volume Retention = 365 days
    # удалять из БД записи о заданиях и файлах , срок хранения которых истёк
    AutoPrune = yes
    # максимальный объем тома в пуле
    Maximum Volume Bytes = 5G

```



```

# максимальное количество томов в пуле
Maximum Volumes = 100
# с каких символов начинаются имена томов пула
Label Format = "Vol-"
}

Messages {
  Name = Standard
  # команда отправки письма
  mailcommand = "/usr/sbin/bsmtp -h localhost -f\"(Bacula) \<%r\>" -s \"Bacula: %t %e of %c %l\" %r"
  # команда для передачи сообщений оператору
  operatorcommand = "/usr/sbin/bsmtp -h localhost -f\"(Bacula) \<%r\>" -s \"Bacula: Intervention needed for
%j\" %r"
  # куда = кому = и какие уведомления отправлять
  mail = root = all, !skipped
  # требование к оператору смонтировать том по необходимости
  operator = root = mount
  # какие сообщения выводить в консоль
  console = all, !skipped, !saved
  # путь к логу = какие сообщения записывать в лог
  append = "/var/log/bacula/bacula.log" = all, !skipped
  # записывать в СУБД в таблицу Log; очищается одновременно с удалением записи о задании
  catalog = all
}

# эта строка подгружает все конфигурационные файлы из папки «job.d»
@| "sh -c 'for f in /etc/bacula/job.d/*.conf ; do echo @{$f} ; done'"

Schedule {
  Name = "WeeklyCycle"
  # Тип бекапа, периодичность и время запуска
  Run = Full 1st sun at 23:05
  Run = Differential 2nd-5th sun at 23:05
  Run = Incremental mon-sat at 23:05
}

Client {
  Name = dir-fd
  #IP-адрес Клиента
  Address = 192.168.122.3
  # порт, который клиент слушает
  FDPort = 9102
  # имя PostgreSQL базы данных Bacula
  Catalog = BaculaCatalog
  # пароль Клиента
  Password = "clientpass"
  # период, в течении которого информация о ФАЙЛАХ хранится в базе данных
  File Retention = 60 days
  # период, в течении которого информация о ЗАДАНИЯХ хранится в базе данных
  Job Retention = 6 months
  # удалять записи из Bacula Catalog старше вышеуказанных значений
  AutoPrune = yes
}

FileSet {
  Name = "Full Set"
  # Секция содержит пути к резервируемым файлам/каталогам
  Include {
    # Секция определяющая параметры резервирования файлов/каталогов
    Options {
      # Параметр указывает алгоритм вычисления контрольных сумм файлов
      signature = MD5
      # Параметр указывает алгоритм сжатия файлов

```

```

Compression = GZIP
# Параметр указывает на необходимость рекурсивного резервирования,
recurse = yes
# Параметр указывающий на необходимость сохранения ACL информации
aclsupport = yes
# указывает на возможность включения поддержки расширенных атрибутов,
xattrsupport = yes
}
# копируемый каталог
File = /home
}
# пути к файлам/каталогам, которые необходимо исключить
Exclude {
  File = /tmp
}
}

```

Для настройки заданий используются секции Job (для каждой задачи отдельная секция). Рекомендуется располагать каждое задание в отдельном файле в отдельной директории.

```
sudo mkdir /etc/bacula/job.d/
```

```
/etc/bacula/job.d/backup-dir-fd.conf
```

```

Job {
  Name = "BackupClient1"
  # Тип задания (backup, restore и т.д.)
  Type = Backup
  # Уровень бэкапа (Full, Incremental, Differential и т.п)
  Level = Full
  # Имя клиента на котором выполняется задание
  Client = dir-fd
  # Набор файлов для выполнения задания
  FileSet = "Full Set"
  # Расписание выполнения задания
  Schedule = "WeeklyCycle"
  # Файловое хранилище
  Storage = stor-sd
  # Поведение уведомлений
  Messages = Standard
  # Пул, куда будем писать бэкапы. Если мы хотим сделать отдельный пул для каждого клиента,
  # или использовать префиксы, тогда пул указывается в задании для каждого клиента
  # переопределяя тем самым эту настройку
  Pool = File
  # Буферизация атрибутов файлов
  SpoolAttributes = yes
  # Приоритет. Давая заданиям приоритеты от 1 (max) до 10 (min), можно регулировать послед>
  Priority = 10
  # Файл хранит информацию откуда извлекать данные при восстановлении
  Write Bootstrap = "/var/lib/bacula/%c.bsr"
}

```

```
/etc/bacula/job.d/restore-dir-fd.conf
```

```

Job {
  Name = "RestoreFiles"
  Type = Restore
  # Клиент на который нужно восстановить файлы
  Client=dir-fd
  # Набор восстанавливаемых файлов
  FileSet="Full Set"
}

```

```
# Хранилище где лежит бекап клиента
Storage = stor-sd
# Пул томов где лежит бекап клиента
Pool = File
# Поведение уведомлений
Messages = Standard
# Куда на клиенте восстанавливать файлы
Where = /home2
}
```

Настройка Storage Daemon

Отредактируем права конфигурационного файла Storage Daemon

```
sudo chmod 644 /etc/bacula/bacula-sd.conf
sudo chown root:bacula /etc/bacula/bacula-sd.conf
sudo systemctl restart bacula-sd.service
```

/etc/bacula/bacula-sd.conf

```
Storage {
  Name = stor-sd
  SDPort = 9103
  # папка в которой лежат статус-файлы Хранилища
  WorkingDirectory = "/var/lib/bacula"
  # pid-файл демона Хранилища
  Pid Directory = "/run/bacula"
  Maximum Concurrent Jobs = 1
  # IP-адрес Хранилища
  SDAddress = 192.168.122.2
}
Director {
  # Имя Директора который может подключаться к этому Хранилищу
  Name = bacula-dir
  # Пароль подключения к этому Хранилищу
  Password = "storpass"
}
Device {
  Name = DevStorage
  Media Type = File1
  Archive Device = /backups/files1
  LabelMedia = yes
  Random Access = Yes
  AutomaticMount = yes
  RemovableMedia = no;
  AlwaysOpen = no;
  Maximum Concurrent Jobs = 1
}
Messages {
  Name = Standard
  director = dir-dir = all
}
```

Для хранения бэкапов создадим отдельную директорию и предоставим на неё необходимые права:

```
sudo mkdir -p /backups/files1/
sudo chmod 755 /backups/files1/
sudo chown bacula:bacula /backups/files1/
```

Для проверки правильности синтаксиса воспользуемся командой:

```
sudo /usr/sbin/bacula-dir -t -c /etc/bacula/bacula-dir.conf
```

Перезапуск:

```
sudo systemctl restart bacula-dir  
sudo systemctl restart bacula-sd
```

Настройка File Daemon

На машине клиента установите программу bacula-fd

```
sudo apt install bacula-fd
```

- Настройка производится в файле /etc/bacula/bacula-fd.conf

```
FileDaemon {  
    Name = dir-fd  
    FDport = 9102  
    # папка в которой лежат статус-файлы Клиента  
    WorkingDirectory = /var/lib/bacula  
    # pid-файл демона Клиента  
    Pid Directory = /run/bacula  
    Maximum Concurrent Jobs = 1  
    # возможность расширений FD,  
    # расширения оформляются в виде разделяемых библиотек (имя-fd.so) и помещаются в указанный  
    # каталог  
    Plugin Directory = /usr/lib/bacula  
    # fqdn имя или IP-адрес Клиента  
    FDAddress = 192.168.122.14  
}  
Director {  
    # Имя Директора который может подключаться к этому Клиенту  
    Name = bacula-dir  
    # Пароль подключения к этому Клиенту  
    Password = "clientpass"  
}  
Messages {  
    Name = Standard  
    director = dir-dir = all, !skipped, !restored  
}
```

Предоставим права для работы с файлом:

```
sudo chmod 644 /etc/bacula/bacula-fd.conf  
sudo chown root:bacula /etc/bacula/bacula-fd.conf
```

Для проверки синтаксиса возможно воспользоваться командой:

```
sudo /usr/sbin/bacula-fd -t -c /etc/bacula/bacula-fd.conf
```

Перезапуск:

```
sudo systemctl restart bacula-fd.service
```

Настройка подключения к Bacula

Для подключения к утилите возможно использовать консольное приложение или графический интерфейс.

Для настройки консольного приложения необходимо отредактировать конфигурационный файл */etc/bacula/bconsole.conf*

```
Director {  
  Name = dir-dir  
  DIRport = 9101  
  # IP-адрес Директора  
  address = 192.168.122.2  
  Password = "dirpass"  
}
```

Для работы возможно использовать графическое приложение bat. Для его установки воспользуйтесь командой:

```
sudo apt install bacula-console-qt
```

Для его настройки используйте конфигурационный файл */etc/bacula/bat.conf*
Его содержимое должно совпадать с содержимым файла *bconsole.conf*

Работа с программой bacula с использованием консольного приложения

Для подключения к утилите выполните команду:

```
sudo bconsole
```

Откроется окно, представляющее собой строку для ввода команд:

```
adminstd@server:~$ sudo bconsole
Connecting to Director 192.168.122.2:9101
1000 OK: 103 bacula-dir Version: 9.6.7 (10 December 2020)
Enter a period to cancel a command.
*█
```

Список некоторых управляющих команд:

Команда	Описание
run	Команда позволяет вам планировать немедленное выполнение заданий
restore	Команда восстановления бэкапа
message	Чтение системных сообщений
list [параметр]	jobs список выполненных задач pools список томов хранилища clients список клиентов
exit	Выход из программы

Для создания резервной копии воспользуйтесь командой run.

```
*run
A job name must be specified.
The defined Job resources are:
   1: BackupClient1
   2: RestoreFiles
Select Job resource (1-2): █
```

Далее выберите задачу резервного копирования, укажите её номер.

```
Select Job resource (1-2): 1
Run Backup job
JobName: BackupClient1
Level: Incremental
Client: dir-fd
FileSet: Full Set
Pool: File (From Job resource)
Storage: stor-sd (From Job resource)
When: 2024-11-17 18:21:42
Priority: 10
OK to run? (yes/mod/no): █
```

На экране отобразятся параметры создания резервной копии. Для их изменения воспользуйтесь командой mod. Например, чтобы изменить уровень резервного копирования, необходимо после ввода команды mod выбрать пункт – Level -> указать номер требуемого уровня (например, Full).

```

OK to run? (yes/mod/no): mod
Parameters to modify:
    1: Level
    2: Storage
    3: Job
    4: FileSet
    5: Client
    6: When
    7: Priority
    8: Pool
    9: Plugin Options
Select parameter to modify (1-9): 1
Levels:
    1: Full
    2: Incremental
    3: Differential
    4: Since
    5: VirtualFull
Select level (1-5): 1

```

После указания всех необходимых параметров введите команду `yes`. На экране отобразится сообщение о том, что задача принята на исполнение (`Job queued. JobId = N`). Нажмите `Enter` и на экране отобразится информация о том, что пришло сообщение. С помощью команды `message` прочитайте его. Оно должно содержать информацию о выполненном копировании и строчку с содержимым: «Backup OK». На рисунке приведена часть сообщения. Обратите внимание, процесс резервного копирования может занимать некоторое время, поэтому сообщение полностью может отобразиться не сразу.

```

SD termination status: OK
Termination: Backup OK

17-Nov 18:24 bacula-dir JobId 10: Begin pruning Jobs older than 6 months .
17-Nov 18:24 bacula-dir JobId 10: No Jobs found to prune.
17-Nov 18:24 bacula-dir JobId 10: Begin pruning Files.
17-Nov 18:24 bacula-dir JobId 10: No Files found to prune.
17-Nov 18:24 bacula-dir JobId 10: End auto prune.

```

Для восстановления из резервной копии воспользуйтесь командой `restore`. На экране будет представлен список возможных команд для выполнения. Вы можете выбрать команду *List last 20 Jobs run* – с её помощью определите `jobid` задачи, которая создала резервную копию.

```

*restore

First you select one or more JobIds that contain files
to be restored. You will be presented several methods
of specifying the JobIds. Then you will be allowed to
select which files from those JobIds are to be restored.

To select the JobIds, you have the following choices:
    1: List last 20 Jobs run
    2: List Jobs where a given File is saved
    3: Enter list of comma separated JobIds to select
    4: Enter SQL list command
    5: Select the most recent backup for a client
    6: Select backup for a client before a specified time
    7: Enter a list of files to restore
    8: Enter a list of files to restore before a specified time
    9: Find the JobIds of the most recent backup for a client
   10: Find the JobIds for a backup for a client before a specified time
   11: Enter a list of directories to restore for found JobIds
   12: Select full restore to a specified Job date
   13: Cancel
Select item: (1-13): █

```

Далее, выберите команду *Enter list of comma separated JobsIds to select* и укажите номер с задачей вашей резервной копии. В новом приглашении командной строки возможно отметить те файлы и директории, которые необходимо восстановить из копии. Для простоты восстановите всю сохраненную директорию. Для этого выполните команду *mark home/*

```
Select item: (1-13): 3
Enter JobId(s), comma separated, to restore: 10
You have selected the following JobId: 10

Building directory tree for JobId(s) 10 ... ++++++
6,616 files inserted into the tree.

You are now entering file selection mode where you add (mark) and
remove (unmark) files to be restored. No files are initially added, unless
you used the "all" keyword on the command line.
Enter "done" to leave this mode.

cwd is: /
$ ls
home/
$ mark home/
7,249 files marked.
```

Запустите задачу с помощью последовательного ввода команд *done* -> *yes*.

Через некоторое время проверьте наличие сообщения об успешности выполнения задачи с помощью команды *message*.

```
Termination:          Restore OK

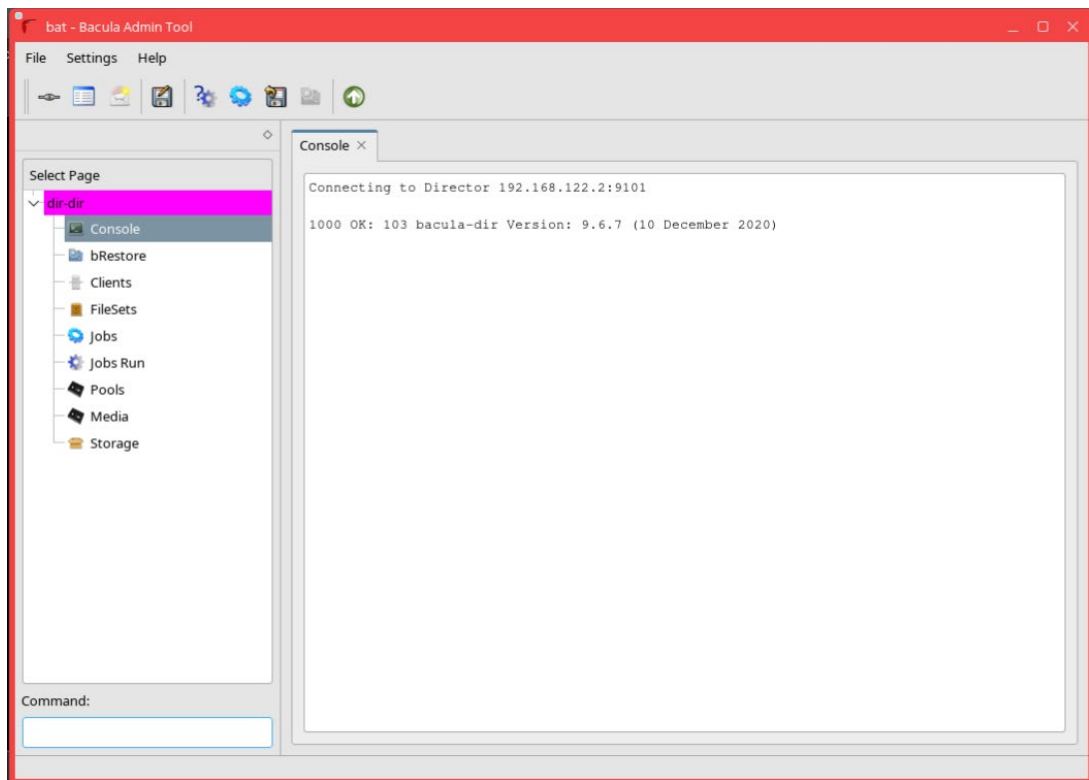
17-Nov 18:36 bacula-dir JobId 11: Begin pruning Jobs older than 6 months .
17-Nov 18:36 bacula-dir JobId 11: No Jobs found to prune.
17-Nov 18:36 bacula-dir JobId 11: Begin pruning Files.
17-Nov 18:36 bacula-dir JobId 11: No Files found to prune.
17-Nov 18:36 bacula-dir JobId 11: End auto prune.
```

Работа с программой bacula с использованием графического приложения


Для запуска графического приложения воспользуйтесь командой:

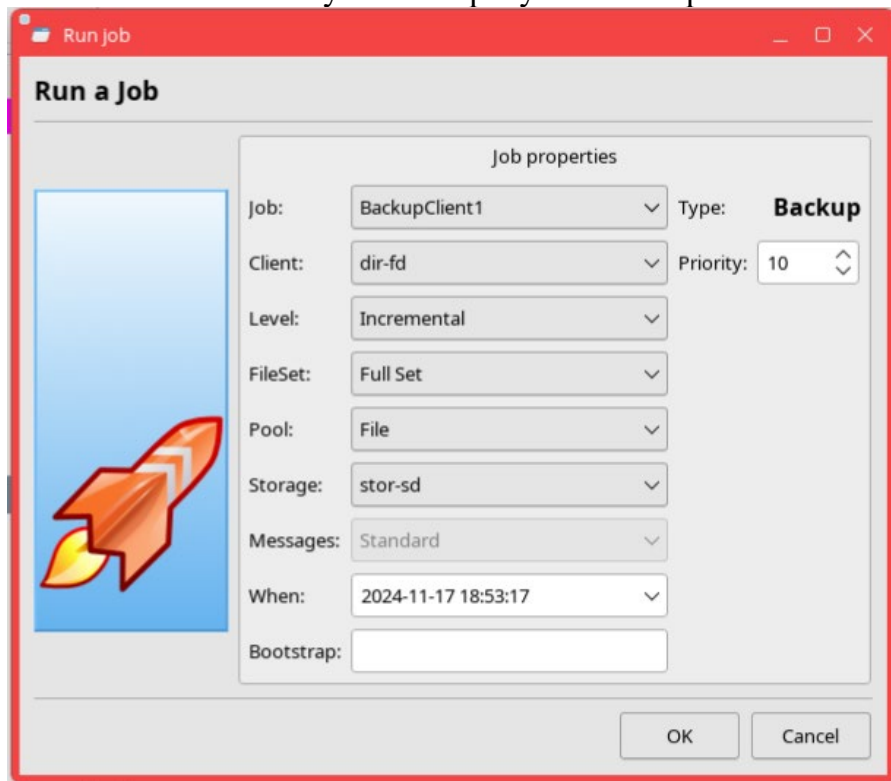
```
sudo /etc/sbin/bat
```

Окно интерфейса выглядит следующим образом:



С помощью приглашения в нижнем левом углу возможно вводить команды для управления, аналогичные консольной утилите.

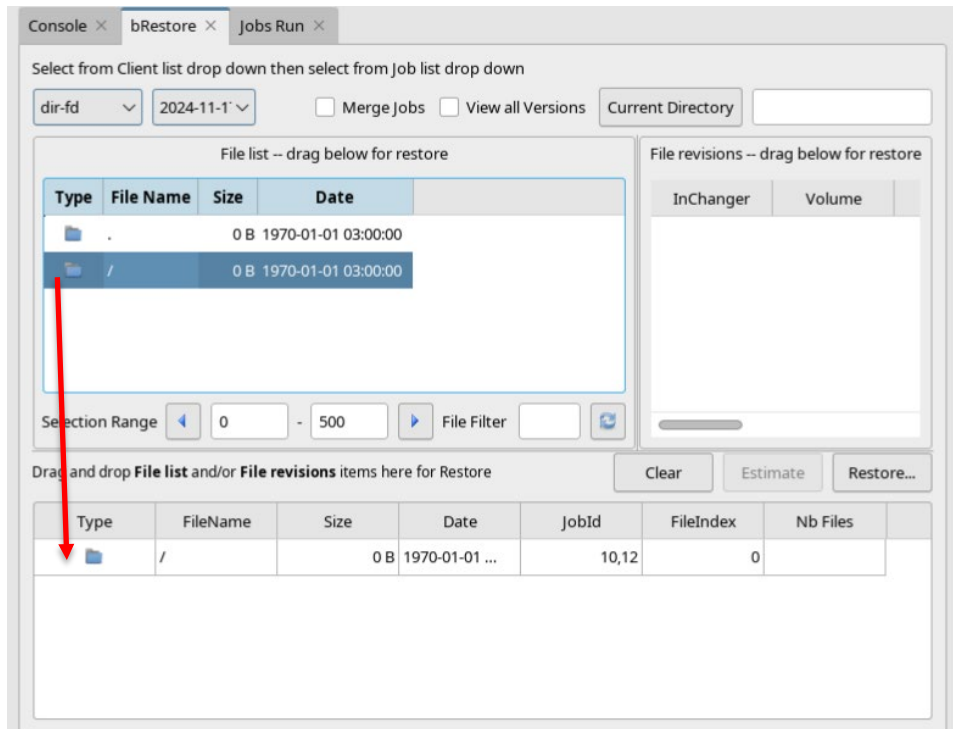
Для создания резервной копии нажмите на кнопку . В появившемся окне укажите требуемые настройки и нажмите «OK»



Состояние выполнения задачи можно посмотреть на вкладке “Jobs Run”

Job Id ▾	Job Name	Client	Job Starttime	Job Type	Job Level	Job Files	Job Bytes	
13	BackupClient1	dir-fd	2024-11-17 18:59:27	Backup	Incremental	2	51 B	Comp

Для восстановления на вкладке bRestore в выпадающих окошках выберите имя клиента и номер резервной копии, с которой необходимо восстановиться. Далее в нижнем окне выберите файлы (директории) для восстановления и перетащите их мышкой в нижнее окно.



Для выполнения команды нажмите команду Restore -> OK.






Практическая работа

1. Установите программу Bacula, следуя указаниям из лабораторной работы. Параметры настройки возьмите из таблицы:

Расположение Director	Машина server
Расположение Storage	Машина server
Расположение клиента	Машина client
Имя Director	dir-ВашиИнициалы (например, dir-sab)
Имя Storage	stor-ВашиИнициалы (например, stor-sab)
Имя клиента	cli-ВашиИнициалы (например, cli-sab)
Пароль от Director	dirВашиИнициалы (например, dirsab)
Пароль от Storage	storВашиИнициалы (например, storsab)
Пароль от клиента	clientВашиИнициалы (например, clientsab)
Директория для восстановления	/homeВашиИнициалы (например, /homesab)

2. Используя консольную утилиту сделайте полное резервное копирование домашней директории клиента.
3. Восстановите резервную копию
4. Создайте несколько файлов на клиенте. С помощью графической утилиты выполните дифференциальное копирование.

5. Отредактируйте созданные файлы. Удалите один из них и создайте несколько новых. С помощью графической утилиты выполните инкрементальное копирование.
6. Повторите пункты 4 и 5, но с использованием консольной утилиты. Таким образом, будет выполнено 5 резервных копирований:
 1. Полное
 2. Дифференциальное
 3. Инкрементальное
 4. Дифференциальное
 5. Инкрементальное
7. С помощью графической утилиты откройте вкладку bRestore и выбирая различные JobId перетягивайте папку для резервного копирования в нижнюю панель. Таким образом в ней будет расположено 5 строк:

Type	FileName	Size	Date	JobId
	/	0 B	1970-01-01 ...	
	/	0 B	1970-01-01 ...	
	/	0 B	1970-01-01 ...	
	/	0 B	1970-01-01 ...	
	/	0 B	1970-01-01 ...	

Проанализируйте значение в поле JobId, включающее в себя номера необходимых копий для восстановления. Объясните полученные значения.

8. Восстановите данные из последней резервной копии.

Список литературы

1. <https://wiki.astralinux.ru/display/doc/BACULA>
2. https://www.bacula.org/7.2.x-manuals/en/main/Brief_Tutorial.html
3. <https://postgrespro.ru/education/courses/DBA3>
4. Рогов Е. В., PostgreSQL изнутри. М: ДМК Пресс, 2023. 662 с.