RuBackup
Система резервного копирования и восстановления данных

Резервное копирование

и восстановление СУБД

Postgres Pro

RuBackup

Версия 2.1

20.05.2024 г.

Содержание

Введение

Система резервного копирования (СРК) RuBackup поддерживает резервное копирование кластеров СУБД Postgres Pro всех версий начиная с 10-ой.

Принцип резервного копирования кластеров Postgres Pro с использованием RuBackup состоит в периодическом создании базовых резервных копий экземпляра СУБД по определённому расписанию.

Полное резервное копирование – это создание резервной копии всех данных из исходного набора, независимо от того, изменялись данные или нет с момента выполнения последней полной резервной копии.

Дифференциальное резервное копирование сохраняет только данные, изменённые со времени выполнения предыдущего полного резервного копирования.

Инкрементальное резервное копирование сохраняет только данные, изменённые со времени выполнения предыдущей инкрементальной резервной копии, а если такой нет, то со времени выполнения последней полной резервной копии.

В репозитории RuBackup базовые резервные копии будут храниться как **полные резервные копии** (full), а файлы backup.control и backup_content.control, созданные после базовой резервной копии, – как **snap-файл** с расширением «.snap». На основе файлов backup.control и backup_content.control из snap-файла полной резервной копии создаётся **дифференциальная резервная копия** (differential).

После успешного выполнения резервного копирования следует производить периодическую очистку каталога хранения архивных файлов WAL. Автоматическая очистка не предусмотрена.

После окончания операции резервного копирования будут созданы два файла – архивный и снимок состояния – на медиасервере, которому принадлежит пул, указанный в правиле резервного копирования. Точное расположение файлов указано в записи репозитория системы резервного копирования RuBackup.

При необходимости архивный файл может быть преобразован при помощи алгоритма защитного преобразования на клиенте и сжат. Снимок состояния не преобразовывается, так как в нём располагается только информация о ресурсе, о режиме, в котором была сделана резервная копия и время старта и окончания резервного копирования. В снимке состояния отсутствуют конфиденциальные данные.

Для выполнения резервного копирования кластеров Postgres Pro на хосте клиента должно быть достаточно свободного места для создания резервной копии. Локальное местоположение временного каталога для создания резервных копий определено в файле /opt/rubackup/etc/config.file (параметр use-local-backup-directory). Если на хосте клиента недостаточно места для создания резервной копии, ему может быть предоставлена сетевая файловая система NFS с сервера резервного копирования во временное пользование (см. «Руководство системного администратора RuBackup»).

Для выполнения резервного копирования администратор RuBackup может настраивать правила глобального расписания в оконном Менеджере Администратора RuBackup (RBM).

Клиенты RuBackup могут осуществлять восстановление данных резервных копий и создание срочных резервных копий при помощи оконного Менеджера Клиента RuBackup (RBC), а также при помощи утилит командной строки RuBackup.

Подготовка хоста с СУБД Postgres Pro

Для возможности резервного копирования данных кластера Postgres Pro при помощи СРК RuBackup на сервер следует установить следующие пакеты:

- `rubackup-client.deb` – клиент резервного копирования;
- `rubackup-common.deb` – общий пакет для СРК RuBackup;
- `rubackup-postgres-pro.deb` – модуль резервного копирования данных Postgres Pro;
- pg-probackup-std-13.deb – консольная утилита для создания резервных копий кластеров баз данных PostgreSQL и их восстановления;
- python3-psycopg2 – драйвер PostgreSQL, совместимый с DB API 2.0.

Установка клиента RuBackup

Для осуществления резервного копирования и восстановления данных кластера Postgres Pro при помощи RuBackup на сервер должен быть установлен клиент RuBackup со всеми необходимыми модулями. Клиент RuBackup представляет собой фоновое системное приложение (демон или сервис), обеспечивающее взаимодействие с серверной группировкой RuBackup. Для выполнения резервного копирования кластеров СУБД Postgres Pro клиент RuBackup должен работать от имени суперпользователя (root в Linux и Unix). Соответственно, для **версий Postgres Pro 10 и ниже** необходимо, чтобы сервер Postgres Pro был запущен от имени root.

Подробно процедура установки клиента описана в «Руководстве по установке серверов резервного копирования и Linux клиентов RuBackup», для операционной системы Windows — в «Руководстве по установке Windows клиентов RuBackup».

Установка пакетов модулей резервного копирования

Установка пакета модулей резервного копирования RuBackup производится из учётной записи с административными правами на узле с СУБД Postgres Pro после установки на него клиента RuBackup.

Для установки пакета модулей используйте следующий вызов:

```
# dpkg -i rubackup-postgres-pro.deb
Выбор ранее не выбранного пакета rubackup-postgres-pro.
(Чтение базы данных … на данный момент установлено
137334 файла и каталога.)
Подготовка к распаковке rubackup-postgres-pro.deb …
Распаковывается rubackup-postgres-pro (2021-02-20) …
Настраивается пакет rubackup-postgres-pro (2020-02-20) …
```

Конфигурационный файл модуля

Для работы модуля необходимо произвести ручную настройку конфигурационного файла. Формат конфигурационного файла - **YAML**. Конфигурационный файл модуля располагается по пути **/opt/rubackup/etc/rb_module_postgres_pro.conf**.

Содержание конфигурационного файла:

1) restore_target_action: возможные значения - (PAUSE, PROMOTE, SHUTDOWN). Задаёт действие, которое должен выполнить сервер по достижении цели восстановления.

   По умолчанию установлено значение PAUSE, при котором после восстановления кластер находится в состоянии Read Only. Для того, чтобы кластер после восстановления был в режиме Read-Write нужно установить значение PROMOTE.

2) restore_target: возможные значения - (IMMEDIATE, LATEST). LATEST – восстановить последнее возможное состояние, исходя из содержимого архива WAL. IMMEDIATE - восстановить самое раннее из возможных согласованное состояние кластера;

3) restore_mode: возможные значения (NONE, CHECKSUM). Если значение параметра равно CHECKSUM, то pg_probackup при восстановлении будет повторно использовать валидные страницы доступные в каталоге кластера, если они не изменялись.

4) pg_probackup: Абсолютный путь до утилиты pg_probackup;

5) direct_restore: произвести восстановление в ресурс или в директорию.

Внимание! Перед началом работы с модулем обязательно нужно указать актуальный полный путь до утилиты pg_probackup.

Подготовка к использованию pg_probackup

Для выполнения резервного копирования кластера Postgres Pro используется утилита pg_probackup. Перед использованием модуля необходимо выполнить следующие действия:

1) Инициализировать каталог резервных копий;

2) Добавить копируемый экземпляр в каталог копий;

3) Настроить копируемый кластер баз данных для использования pg_probackup.

Инициализация каталога резервных копий

Для инициализации каталога резервных копий используйте следующую команду:

```
# pg_probackup init -B /opt/rubackup/mnt/pg_probackup
```

Внимание! Не изменяйте данную команду. Если директория /opt/rubackup/mnt/pg_probackup уже существует, то она должна быть пустой.

После выполнения данной команды утилита pg_probackup создаст подкаталоги в каталоге резервных копий /opt/rubackup/mnt/pg_probackup:

- `wal/` — каталог для файлов WAL,
- `backups/` — каталог для файлов резервных копий.

Определение копируемого экземпляра

Утилита pg_probackup может сохранять резервные копии разных кластеров баз данных в одном каталоге резервных копий. Для создания необходимых подкаталогов вы должны добавить копируемый экземпляр в каталоге копий для каждого кластера баз данных, копию которого вы будете делать.

Для добавления копируемого экземпляра при помощи pg_probackup выполните команду:

 # pg_probackup add-instance -B /opt/rubackup/mnt/pg_probackup -D каталог_данных --instance имя_экземпляра (по названию каталога данных),

где **каталог_данных** – это каталог, в котором хранятся все данные кластера баз данных. Например, для кластера postgres СУБД Postgres Pro 13 каталогом данных будет /var/lib/pgpro/std-13/data/.

Внимание! Имя экземпляра должно совпадать с именем каталога данных. Например, добавляется каталог данных '/var/lib/pgpro/std-13/data/', значит именем экземпляра будет 'data'.

Внимание! Директория '/opt/rubackup/mnt/pg_probackup' и все вложенные в неё папки должны быть доступны для записи и чтения пользователю postgres, а также пользователю, под контролем которого работает клиент RuBackup.

Обеспечить доступ можно следующим образом:

 # sudo chgrp postgres -R /opt/rubackup/mnt/pg_probackup

 # sudo chmod g+rwx -R /opt/rubackup/mnt/pg_probackup

Если планируется использование модуля для резервного копирования кластеров Postgres Pro в составе *patroni*, то необходимо выполнить следующую команду:

 # pg_probackup set-config -B /opt/rubackup/mnt/pg_probackup/ --instance=patroni --pghost ip_кластера,

где **ip_кластера** – ip адрес копируемого инстанса *patroni*.

Настройка копируемого кластера баз данных для использования pg_probackup

Для выполнения резервного копирования в защищённом режиме необходимо создать роль с ограниченными правами и базу данных резервного копирования. Имя роли, пароля и базы данных условны и могут быть изменены по усмотрению пользователя. Последовательность действий следующая:

1. Сначала создайте базу данных резервного копирования. Данная операция производится от имени пользователя postgres:

 # su postgres

 # createdb backupdb

2. Далее описан процесс создания роли для выполнения резервного копирования. В целях обеспечения безопасности копируемых данных, создаваемая роль будет обладать минимальными правами, необходимыми для выполнения резервного копирования экземпляра Postgres Pro. В этом примере такой ролью будет rubackup_backuper.

   2.1 Выполните подключение к базе данных backupdb от имени пользователя postgres:

   # sudo -u postgres psql -d backupdb

   2.2 Далее, в psql создайте роль rubackup_backuper (имя пользователя может быть изменено) и задайте пароль (в качестве пароля укажите желаемый пароль вместо 12345). Затем, при помощи приведённого ниже скрипта, определите следующие разрешения на сервере Postgres Pro (только в базе данных, к которой производится подключение).

   Для Postgres Pro версии 14 и ниже:

```
BEGIN;
CREATE ROLE rubackup_backuper WITH LOGIN;
ALTER USER rubackup_backuper WITH PASSWORD ‘12345’;
GRANT USAGE ON SCHEMA pg_catalog TO rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.current_setting(text) TO
rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.set_config(text, text, boolean) TO
rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.pg_is_in_recovery() TO
rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.pg_start_backup(text, boolean, boolean)
TO rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.pg_stop_backup(boolean, boolean) TO
rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.pg_create_restore_point(text) TO
rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.pg_switch_wal() TO rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.pg_last_wal_replay_lsn() TO
rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.txid_current() TO rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.txid_current_snapshot() TO
rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.txid_snapshot_xmax(txid_snapshot) TO
rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.pg_control_checkpoint() TO
rubackup_backuper;
ALTER ROLE rubackup_backuper WITH REPLICATION;
COMMIT;
```

   Для Postgres Pro 15:

```
BEGIN;
CREATE ROLE rubackup_backuper WITH LOGIN;
ALTER USER rubackup_backuper WITH PASSWORD ‘12345’;
GRANT USAGE ON SCHEMA pg_catalog TO rubackup_backuper;
```

```
GRANT EXECUTE ON FUNCTION pg_catalog.current_setting(text) TO
rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.set_config(text, text, boolean) TO
rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.pg_is_in_recovery() TO
rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.pg_backup_start(text, boolean) TO
rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.pg_backup_stop(boolean) TO
rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.pg_create_restore_point(text) TO
rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.pg_switch_wal() TO rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.pg_last_wal_replay_lsn() TO
rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.txid_current() TO rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.txid_current_snapshot() TO
rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.txid_snapshot_xmax(txid_snapshot) TO
rubackup_backuper;
GRANT EXECUTE ON FUNCTION pg_catalog.pg_control_checkpoint() TO
rubackup_backuper;
ALTER ROLE rubackup_backuper WITH REPLICATION;
COMMIT;
```

3. Создайте файл .pgpass в каталоге ‘/root’. В файле /root/.pgpass необходимо указать данные для подключения к ранее созданной базе данных и для репликации.

Это распространённый способ хранения информации о соединении в PostgreSQL вместо ввода пароля при каждой совершённой операцией с pg_probackup. Этот файл должен содержать строки в таком формате:

сервер:порт:база_данных:имя_пользователя:пароль

Пример:

```
localhost:5432:backupdb:rubackup_backuper:12345
localhost:5432:replication:rubackup_backuper:12345
```

Внимание! Файл ‘.pgpass’ обязательно должен находиться в домашнем каталоге суперпользователя ‘/root’. Не меняйте местами информацию в строке, она должна быть указана как в примере. Также, маска разрешений файла должна соответствовать маске 0600. Если одно из этих условий будет нарушено, то выполнение резервной копии будет прервано ошибкой.

4. Далее необходимо произвести изменения в файле pg_hba.conf. Найти конфигурационный файл, относящийся к настраиваемому кластеру, можно так:

 # sudo -u postgres psql
 # psql –c 'show hba_file'

Вызовите psql при помощи команды:

 # sudo -u postgres psql

Если для пользователя postgres не установлен пароль, установите его изменив ‘12345’ на подходящий:

 # alter user postgres with password ‘12345’;

В конец файла pg_hba.conf добавьте следующие строки:

host backupdb rubackup_backuper 127.0.0.1/32 md5

host replication rubackup_backuper 127.0.0.1/32 md5

Вместо **peer** везде установите **md5**. Пример итогового файла:

```
# TYPE  DATABASE        USER            ADDRESS                 METHOD

# "local" is for Unix domain socket connections only
local   all             all                                     md5
# IPv4 local connections:
host    all             all             127.0.0.1/32            md5
# IPv6 local connections:
host    all             all             ::1/128                 md5
# Allow replication connections from localhost, by a user with the
# replication privilege.
local   replication     all                                     md5
host    replication     all             127.0.0.1/32            md5
host    replication     all             ::1/128                 md5
host    backupdb        rubackup_backuper       127.0.0.1/32    md5
host    replication     rubackup_backuper       127.0.0.1/32    md5
```

Проверяем не было ли опечаток и перечитываем конфигурацию:

 # psql -c 'select * from pg_hba_file_rules'

 # psql -c 'select pg_reload_conf()'

```
postgres@postgresPro-client:~$ psql -c 'select * from pg_hba_file_rules'
Пароль пользователя postgres:
 line_number | type  |   database    |     user_name      |  address  |                netmask                 | auth_method | options | error
-------------+-------+---------------+--------------------+-----------+----------------------------------------+-------------+---------+-------
          80 | local | {all}         | {all}              |           |                                        | md5         |         |
          82 | host  | {all}         | {all}              | 127.0.0.1 | 255.255.255.255                        | md5         |         |
          84 | host  | {all}         | {all}              | ::1       | ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff | md5         |         |
          87 | local | {replication} | {all}              |           |                                        | md5         |         |
          88 | host  | {replication} | {all}              | 127.0.0.1 | 255.255.255.255                        | md5         |         |
          89 | host  | {replication} | {all}              | ::1       | ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff | md5         |         |
          90 | host  | {backupdb}    | {rubackup_backuper} | 127.0.0.1 | 255.255.255.255                        | md5         |         |
          91 | host  | {replication} | {rubackup_backuper} | 127.0.0.1 | 255.255.255.255                        | md5         |         |
(8 строк)

postgres@postgresPro-client:~$ psql -c 'select pg_reload_conf()'
Пароль пользователя postgres:
 pg_reload_conf
----------------
 t
(1 строка)
```

После реализации данных настроек модуль сможет выполнять **полное** резервное копирование и **дифференциальное** резервное копирование в режиме **DELTA** используя режим доставки WAL по умолчанию (ARCHIVE).

Настройка потокового резервного копирования

Для настройки потокового резервного копирования (STREAM) требуется произвести изменения в файле postgresql.conf, который находится внутри копируемого кластера. Например, директорией кластера СУБД Postgres Pro 13 является /var/lib/pgpro/std-13/data/. Обратите внимание на то, что расположение файла может отличаться в зависимости от дистрибутива Linux, версии Postgres Pro и кластера баз данных.

Выполните следующие действия:

1) установите для параметра max_wal_senders достаточно большое значение, предусматривающее минимум одно подключение для процесса резервного копирования;

2) задайте для параметра wal_level значение выше minimal.

Если вы не планируете производить дальнейшую настройку, то после внесённых изменений в файл postgresql.conf необходимо перезагрузить сервер Postgres Pro при помощи команды:

```
# sudo systemctl restart postgrespro-std-13.service
```

После реализации данных настроек модуль сможет выполнять **полное** резервное копирование и **дифференциальное** резервное копирование в режимах **DELTA** используя потоковую доставку WAL (STREAM).

Настройка непрерывного архивирования WAL

Для выполнения копирования в режиме PAGE и восстановления резервных копий на момент времени (recovery-target) должно осуществляться непрерывное архивирование WAL.

Чтобы настроить непрерывное архивирование, выполните следующие действия:

1) задайте для параметра **wal_level** значение выше **minimal**.

2) если вы настраиваете резервное копирование на ведущем сервере, параметр **archive_mode** должен иметь значение on или **always**. Для выполнения резервного копирования на **ведомом** требуется значение **always**.

3) установите параметр **archive_command**:

archive_command = '**/путь_инсталляции/**pg_probackup archive-push -B **/opt/rubackup/mnt/pg_probackup** --instance **имя_экземпляра** --wal-file-path %p --wal-file-name %f [параметры_удалённого_режима]'

где **путь_инсталляции** — путь к каталогу установленной версии pg_probackup, которую вы хотите использовать,

имя_экземпляра должно указывать на уже проинициализированный для данного кластера БД копируемый экземпляр,

параметры_удалённого_режима должны задаваться только в случае расположения архива WAL в удалённой системе.

Внимание! Если вы планируете выполнять страничное копирование и/или делать копии с ведомого сервера, используя режим доставки WAL ARCHIVE, при недостаточной транзакционной активности может потребоваться долго ждать заполнения очередного сегмента WAL. Чтобы ограничить время ожидания, вы можете воспользоваться параметром archive_timeout на ведущем сервере. Значение этого параметра должно быть меньше значения --archive-timeout (по умолчанию 5 минут), чтобы заполненный сегмент успел передаться ведомому серверу и попасть в архив WAL, прежде чем копирование прервётся по тайм-ауту, заданному параметром --archive-timeout.

Если вы не планируете производить дальнейшую настройку, то после внесённых изменений в файл postgresql.conf необходимо перезагрузить сервер Postgres Pro при помощи команды:

```
# sudo systemctl restart postgrespro-std-13.service
```

После реализации данных настроек модуль сможет выполнять **дифференциальное** резервное копирование в режиме **PAGE** используя режимы доставки ARCHIVE и STREAM.

Настройка копирования в режиме PTRACK

Перед выполнением дифференциальной резервной копии в режиме PTRACK выполните следующие подготовительные действия:

1. Зайдите от имени администратора БД в «bakupdb»:

```
# sudo -u postgres psql -d backupdb
```

и выполните следующий запрос:

```
# CREATE EXTENSION ptrack;
```

2. Далее перейдите к редактированию конфигурационного файла postgresql.conf:

1) задайте для параметра **shared_preload_libraries** значение **ptrack**:

2) добавьте в конец конфигурационного файла параметр **ptrack.map_size** и установите его значение по следующим правилам:

Для оптимальной производительности рекомендуется задавать **ptrack.map_size** равным N / 1024, где N — объём кластера Postgres Pro в мегабайтах. Увеличивать значение **ptrack.map_size** сверх рекомендуемого не имеет большого практического смысла. Максимально допустимое значение — 1024.

Внимание! Если до этих изменений была сделана полная резервная копия, то после вступления изменений в силу необходимо сделать новую полную резервную копию, иначе дифференциальное резервное копирование в режиме PTRACK прервётся ошибкой.

3. Выполните команду для перезапуска сервиса:

```
# sudo systemctl restart postgrespro-std-13.service
```

После реализации данных настроек модуль сможет выполнять **дифференциальное** резервное копирование в режиме **PTRACK** используя режимы доставки ARCHIVE и STREAM.

Завершение настройки кластера

После выполнения подготовки целевого кластера к выполнению резервного копирования необходимо перезапустить клиента RuBackup:

```
# rubackup_client stop
# rubackup_client start
```

В результате клиент должен сообщить о том, что модуль резервного копирования Postgres Pro готов к работе:

```
Try to check module: Postgres Pro ...
Execute OS command:
/opt/rubackup/modules/rb_module_postgres_pro -t 2>&1
... module Postgres Pro was checked successfully
```

Мастер-ключ

В ходе установки клиента RuBackup будет создан мастер-ключ для защитного преобразования резервных копий, а также ключи для электронной подписи, если предполагается использовать электронную подпись.

Внимание! При утере ключа вы не сможете восстановить данные из резервной копии, если она была преобразована с помощью защитных алгоритмов.

Важно! Ключи рекомендуется после создания скопировать на внешний носитель, а также распечатать бумажную копию и убрать эти копии в надёжное место.

Мастер-ключ рекомендуется распечатать при помощи утилиты hexdump, так как он может содержать неотображаемые на экране символы:

```
$ hexdump /opt/rubackup/keys/master-key
0000000 79d1 4749 7335 e387 9f74 c67e 55a7 20ff
0000010 6284 54as 83a3 2053 4818 e183 1528 a343
0000020
```

Защитное преобразование резервных копий

При необходимости, сразу после выполнения резервного копирования ваши резервные копии могут быть преобразованы на хосте клиента. Таким образом, важные данные будут недоступны для администратора RuBackup или других лиц, которые могли бы получить доступ к резервной копии (например, на внешнем хранилище картриджей ленточной библиотеки или на площадке провайдера облачного хранилища для ваших резервных копий).

Защитное преобразование осуществляется входящей в состав RuBackup утилитой rbcrypt. Ключ для защитного преобразования резервных копий располагается на хосте клиента в файле /opt/rubackup/keys/master-key. Защитное преобразование данных при помощи rbcrypt возможно с длиной ключа 256 бит (по умолчанию), а также 128, 512 или 1024 бита в зависимости от выбранного алгоритма преобразования.

Автоматическое защитное преобразование и обратное преобразование резервных копий клиентом RuBackup возможны при помощи ключей длиной 256 бит, однако утилита rbcryt поддерживает ключи длиной 128, 256, 512 и 1024 бита (в зависимости от выбранного алгоритма преобразования). Если необходимо для правила глобального расписания выбрать особый режим преобразования, с длиной ключа, отличной от 256 бит и с ключом, располагающемся в другом месте, то вы можете воспользоваться возможностью сделать это при помощи скрипта, выполняющегося после выполнения резервного копирования (определяется в правиле глобального расписания администратором RuBackup). При этом необходимо, чтобы имя преобразованного файла осталось таким же, как и ранее, иначе задача завершится с ошибкой. Провести обратное преобразование такого файла после восстановления его из резервной копии следует вручную при помощи утилиты преобразования. При таком режиме работы нет необходимости указывать алгоритм преобразования в правиле резервного копирования, либо архив будет преобразован ещё раз автоматически с использованием мастер-ключа.

Для выполнения защитного преобразования доступны алгоритмы, представленные в таблице 1.

Таблица 1 – Алгоритмы защитного преобразования, доступные в утилите rbcrypt

Алгоритм	Длина ключа, бит	Примечание
Anubis	128, 256	
Aria	128, 256	
CAST6	128, 256	
Camellia	128, 256	
Kalyna	128, 256, 512	Украинский национальный стандарт ДСТУ 7624:2014
Kuznyechik	256	Российский национальный стандарт ГОСТ Р 34.12-2015
MARS	128, 256	
Rijndael	128, 256	Advanced Encryption Standard (AES)
Serpent	128, 256	
Simon	128	
SM4	128	Китайский национальный стандарт для беспроводных сетей
Speck	128, 256	
Threefish	256, 512, 1024	
Twofish	128, 256	

Удаление клиента RuBackup

При необходимости вы можете удалить с хоста клиент RuBackup и установленные модули резервного копирования.

Удаление клиента RuBackup возможно из учётной записи с административными правами.

Для удаления сервиса rubackup-client используйте команды:

```
# systemctl disable rubackup-client
# systemctl daemon-reload
```

Для удаления клиента RuBackup и модуля **rubackup-postgres-pro-13** используйте команды:

```
# apt remove rubackup-postgres-pro-13
# apt remove rubackup-client
```

При необходимости удалить клиент RuBackup из конфигурации системы резервного копирования, это может сделать системный администратор RuBackup с помощью оконного Менеджера Администратора (RBM).

После удаления клиента RuBackup в ОС Astra Linux SE 1.6 с активированным режимом защитной программной среды следует:

1. Выполнить команду:

```
$ sudo update-initramfs -u -k all
```

2. Перезагрузить операционную систему

```
$ sudo init 6
```

Принцип базового резервного копирования кластера Postgres Pro

В ходе базового резервного копирования выполняется взаимодействие с утилитой pg_probackup.

Команда на выполнение резервного копирования в режиме доставки WAL ARCHIVE:

 # pg_probackup backup -B /opt/rubackup/mnt/pg_probackup --instance=имя_экземпляра --backup-mode=FULL -j 1 --pguser=rubackup_backuper --pgdatabase=backupdb --no-password,

где опция -j - количество потоков, в которые выполняется программа;

- pguser - роль для резервного копирования кластера;
- pgdata - база данных резервного копирования;
- no-password – опция, исключающая пользовательский ввод пароля во время выполнения резервного копирования.

Команда на выполнение резервного копирования в режиме доставки WAL STREAM:

 # pg_probackup backup -B /opt/rubackup/mnt/pg_probackup --instance=имя_экземпляра --backup-mode=FULL --stream -j 1 --pguser=rubackup_backuper --pgdatabase=backupdb --no-password

Внимание! Если планируется, что резервная копия будет сделана на одном сервере Postgres Pro, а восстанавливаться на другом, то необходимо, чтобы принимающий сервер имел такие же значения параметров block_size и wal_blocksize, что и на основном сервере и одинаковую основную версию. В зависимости от конфигурации кластера, Postgres Pro может накладывать дополнительные ограничения, например, по архитектуре процессора и версии libc/libicu.

Принцип дифференциального резервного копирования кластера Postgres Pro

Выполнение дифференциального резервного копирования кластера Postgres Pro может производиться в трёх разных режимах:

1. В режиме **DELTA**:

Команда на выполнение дифференциального резервного копирования в режиме доставки WAL ARCHIVE:

```
# pg_probackup backup -B
/opt/rubackup/mnt/pg_probackup --
instance=имя_экземпляра --backup-mode=DELTA -j 1 --
pguser=rubackup_backuper --pgdatabase=backupdb --no-
password
```

Команда на выполнение дифференциального резервного копирования в режиме доставки WAL STREAM:

```
# pg_probackup backup -B
/opt/rubackup/mnt/pg_probackup --
instance=имя_экземпляра --backup-mode=DELTA --
stream -j 1 --pguser=rubackup_backuper --
pgdatabase=backupdb --no-password
```

2. В режиме **PAGE**:

Внимание! Перед выполнением резервного копирования произведите настройку непрерывного архивирования WAL.

Команда на выполнение дифференциального резервного копирования в режиме доставки WAL ARCHIVE:

 # pg_probackup backup -B /opt/rubackup/mnt/pg_probackup --instance=имя_экземпляра --backup-mode=PAGE -j 1 --pguser=rubackuper --pgdatabase=backupdb --no-password

Команда на выполнение дифференциального резервного копирования в режиме доставки WAL STREAM:

 # pg_probackup backup -B /opt/rubackup/mnt/pg_probackup --instance=имя_экземпляра --backup-mode=PAGE --stream -j 1 --pguser=rubackuper --pgdatabase=backupdb --no-password

3. В режиме **PTRACK**:

Внимание! Перед выполнением резервного копирования произведите настройку резервного копирования в режиме PTRACK.

Команда на выполнение дифференциального резервного копирования в режиме доставки WAL ARCHIVE:

 # pg_probackup backup -B /opt/rubackup/mnt/pg_probackup --instance=имя_экземпляра --backup-mode=PTRACK -j 1 --pguser=rubackuper --pgdatabase=backupdb --no-password

Команда на выполнение дифференциального резервного копирования в режиме доставки WAL STREAM:

 # pg_probackup backup -B /opt/rubackup/mnt/pg_probackup --instance=имя_экземпляра --backup-mode=PAGE --PTRACK -j 1 --pguser=rubackuper --pgdatabase=backupdb --no-password

Принцип восстановления резервной копии кластера Postgres Pro

Данный метод может быть использован при ручном восстановлении служебной базы данных сервера RuBackup, если для её работы используется СУБД Postgres Pro и выполнялось её резервное копирование в составе кластера.

Внимание! Перед восстановлением базы данных рекомендуется сделать резервную копию всех имеющихся файлов в каталоге кластера баз данных.

Для восстановления кластера СУБД Postgres Pro необходимо выполнить следующие действия:

1. Остановить сервер Postgres Pro, если он работает:

```
# sudo systemctl stop postgrespro-std-13.service
```

2. Сделать резервную копию файлов каталога кластера баз данных, для возможности отката (в примере ниже использован каталог ~/emergency_copy, в нём должно быть достаточно места для выполнения данной операции):

```
# sudo -iu postgres (cd /var/lib/pgpro/std-13/data && tar cfv - *) | (cd ~/emergency_copy && tar xf - )
```

3. Очистить каталог кластера баз данных:

```
# sudo -iu postgres rm -rf /var/lib/pgpro/std-13/data/*
```

4. Восстановить данные из резервных копий в директорию. Для этого сперва потребуется изменить значение параметра **direct_restore** на **no** в конфигурационном файле модуля **/opt/rubackup/etc/rb_module_postgres_pro_13.conf**, а затем выполнить восстановление резервной копии в какой-либо каталог при помощи Менеджера Клиента RuBackup (RBC) или утилиты командной строки rb_archives.

5. Запустить восстановление кластера Postgres Pro:

   # pg_probackup restore -B /postgreBackups/ --instance=postgres -i ид_резервной_копии --no-validate -I режим_восстановления

6. Восстановить права на владение файлами кластера для группы и пользователя postgres:

   # sudo chown -R postgres:postgres /var/lib/pgpro/std-13/data/

7. Запустить сервер Postgres Pro:

   # sudo systemctl start postgrespro-std-13.service

Менеджер администратора RuBackup (RBM)

Оконное приложение «Менеджер администратора RuBackup» (RBM) предназначено для общего администрирования серверной группировки RuBackup, управления клиентами резервного копирования, глобальным расписанием резервного копирования, хранилищами резервных копий и пр.

RBM может быть запущено администратором на основном сервере резервного копирования RuBackup.

Для запуска менеджера администратора RBM необходимо выполнить команду:

```
# ssh -X user@rubackup_server

# /opt/rubackup/bin/rbm&
```

Пользователь, запускающий RBM, должен входить в группу rubackup.

На вкладке **Объекты** в левой части представлен список клиентов системы резервного копирования, в котором указано имя, уникальный HWID и описание. Клиенты, которые в данный момент находятся в online, будут отмечены зеленым цветом. Клиенты в состоянии offline – красным (рисунок 1).

Для резервного копирования кластера Postgres Pro на хосте должен быть установлен клиент RuBackup и необходимые модули. Клиент должен быть авторизован администратором RuBackup.

В том случае, если клиент RuBackup был установлен, но не авторизован, в нижней части окна RBM будет сообщение о том, что найдены неавторизованные клиенты. Все новые клиенты должны быть авторизованы в системе резервного копирования.

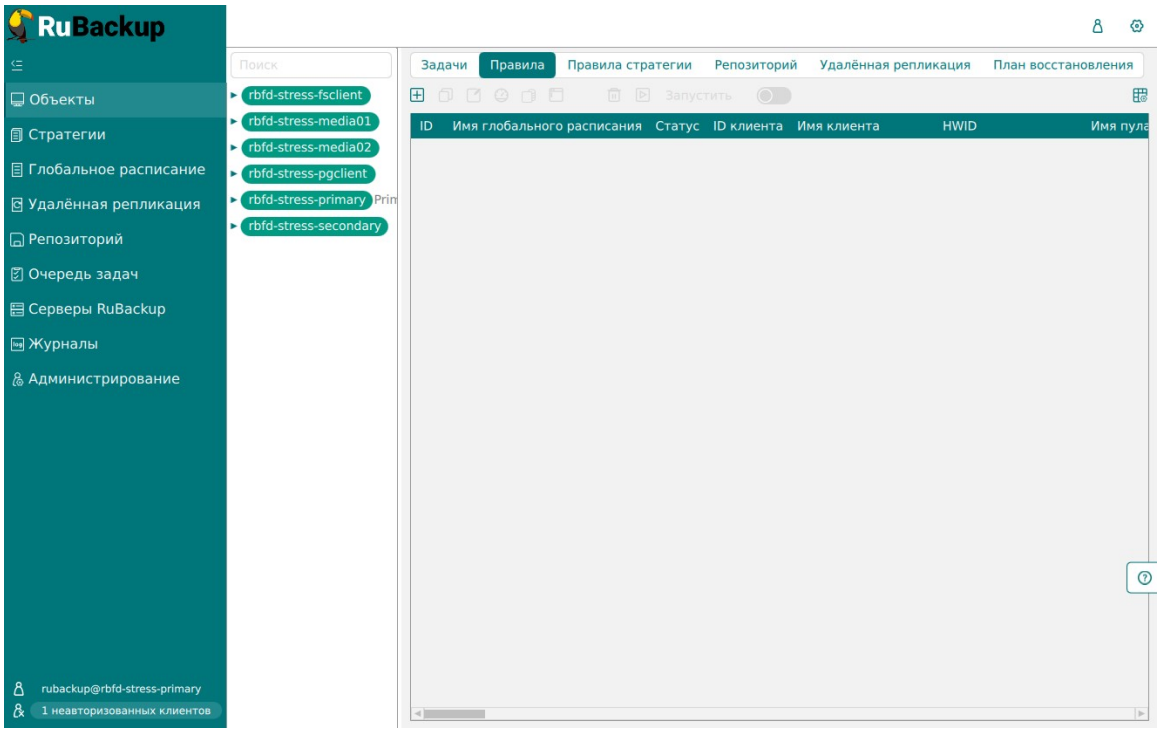


Рисунок 1

Для авторизации неавторизованного клиента в RBM выполните следующие действия:

1) Нажмите на вкладку «**Администрирование**» и выберите иконку «**Клиенты**» (рисунок 2):

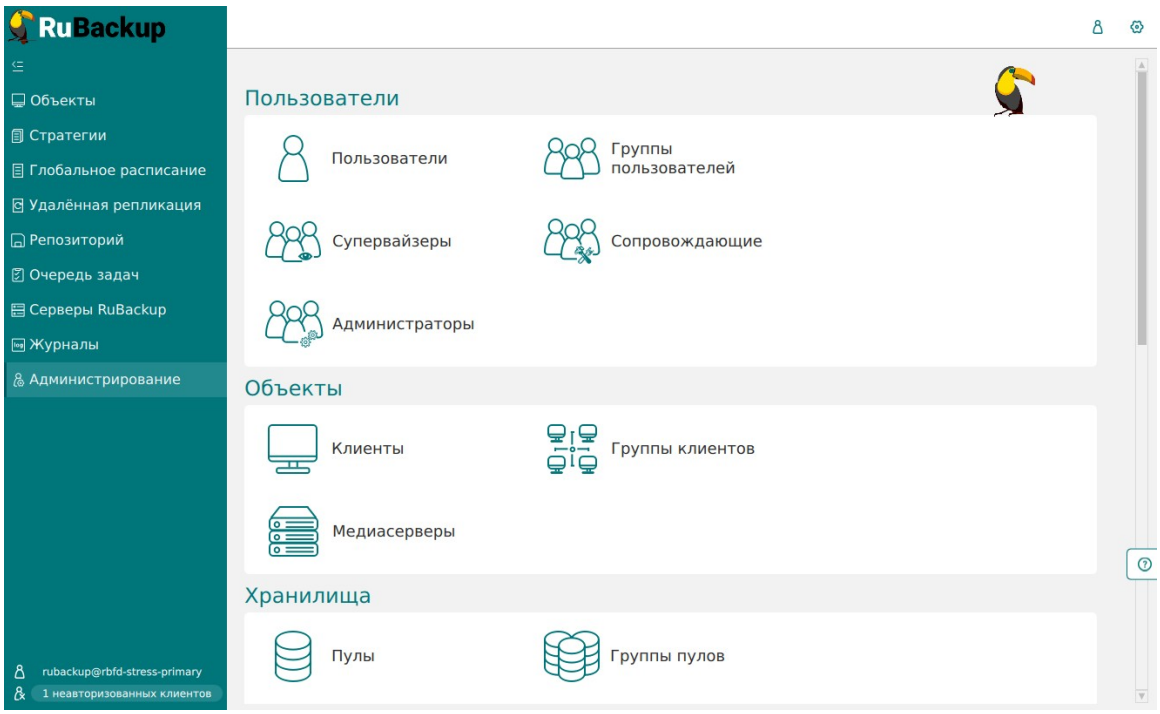


Рисунок 2

2) На верхней панели перейдите на вкладку «**Неавторизованные клиенты**» (Рисунок 3).

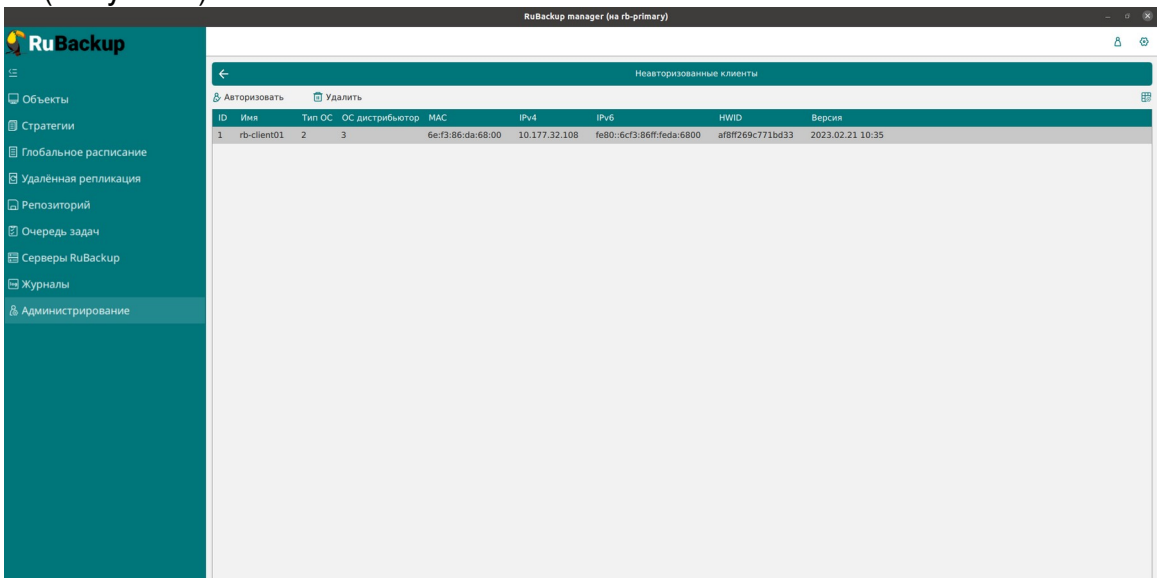


Рисунок 3

3) Выберите нужного неавторизованного клиента и нажмите **Авторизовать** (рисунок 4):

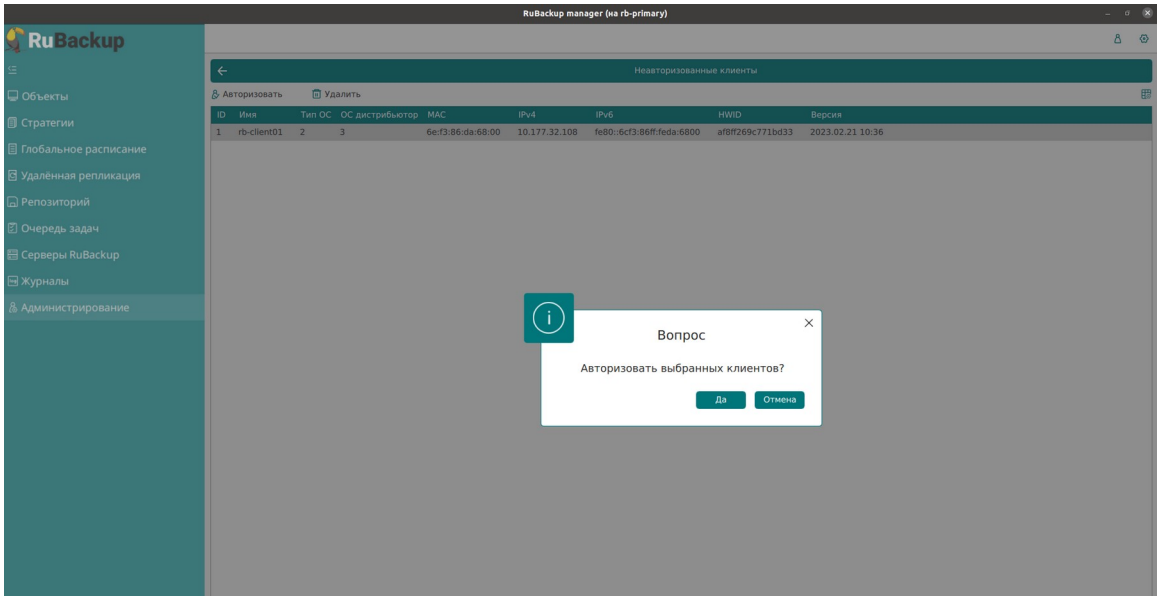


Рисунок 4

После авторизации новый клиент будет виден в главном окне RBM (рисунок 5):

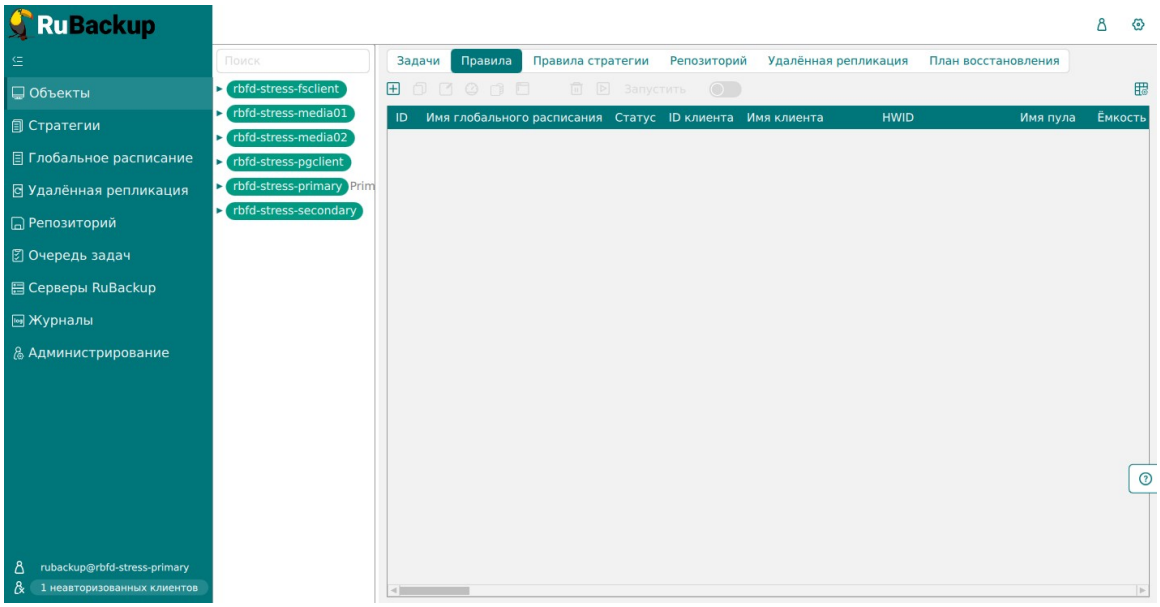


Рисунок 5

Клиенты могут быть сгруппированы администратором по какому-либо общему признаку. В случае необходимости восстанавливать резервные копии на другом хосте клиенты должны принадлежать к разделяемой группе (такая группа отмечается шрифтом italic).

Чтобы выполнять регулярное резервное копирование кластера СУБД Postgres Pro, необходимо создать правило в глобальном расписании. Для этого выполните следующие действия:

1. Находясь в разделе «**Объекты**», выберите вкладку «**Правила**» и нажмите на иконку «**+**» (рисунок 6).

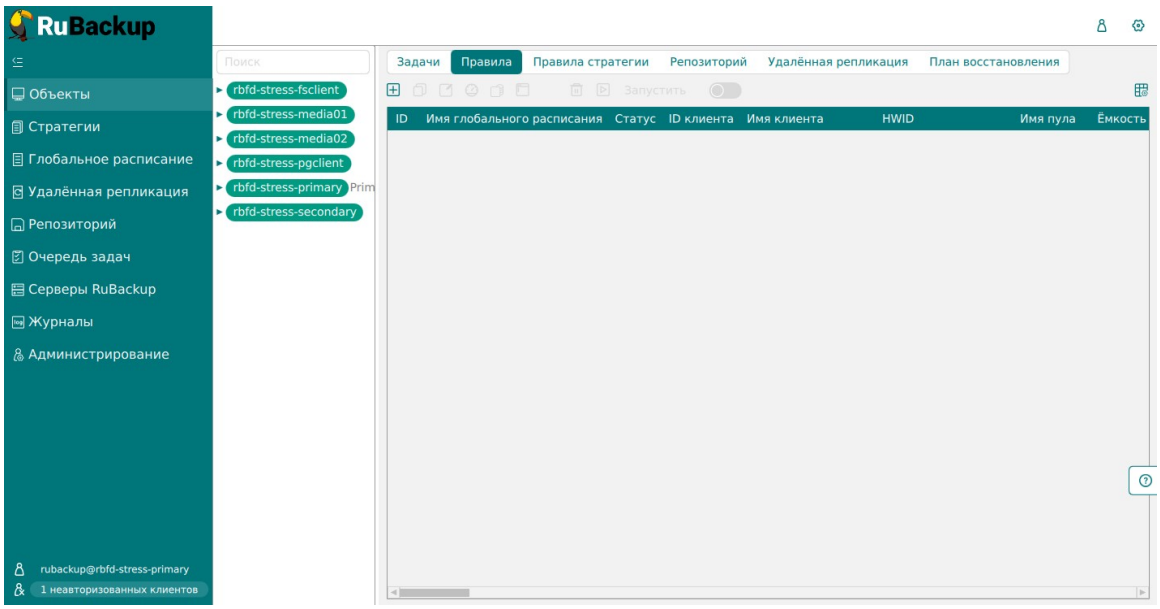


Рисунок 6

2. Выберите тип ресурса «**Postgres Pro**» (рисунок 7):

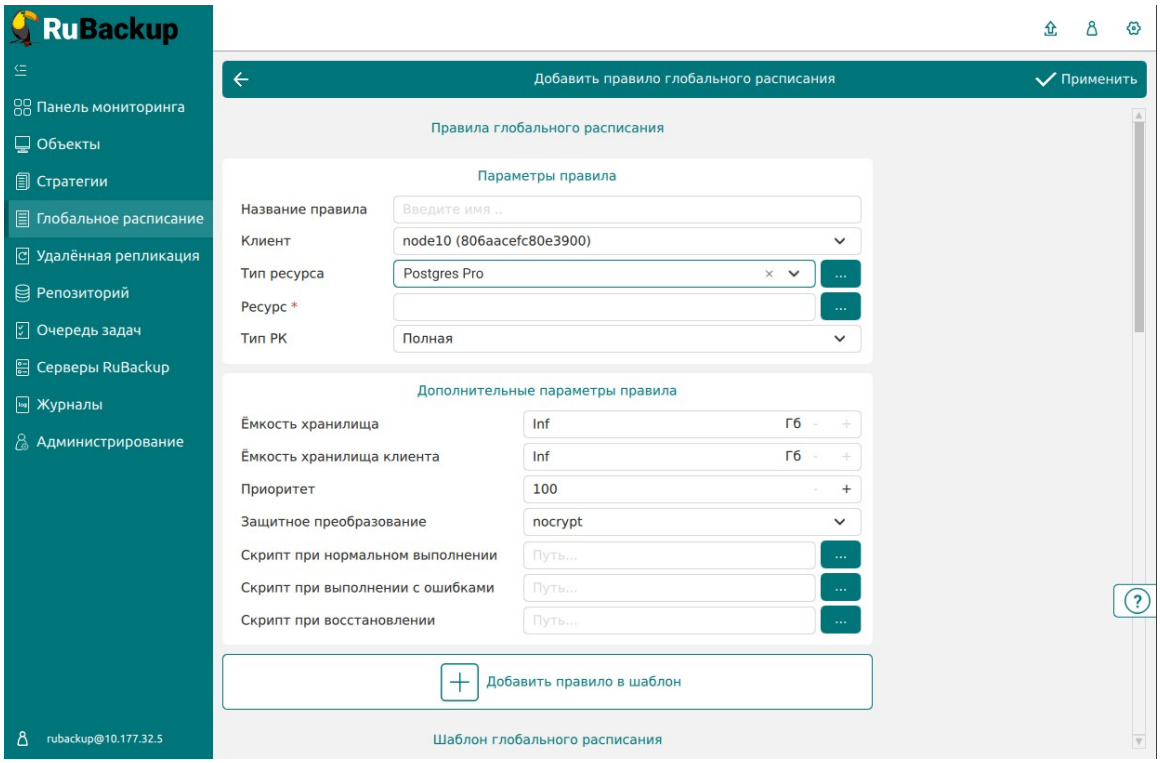


Рисунок 7

В качестве ресурса будет автоматически подставлен путь до каталога кластера. Если вам необходимо сделать резервную копию другого ресурса - укажите полный путь до него вручную или воспользуйтесь кнопкой «Выбрать...».

3. Установить настройки правила: название правила, пул хранения данных, максимальный объём для резервных копий правила (в ГБ), тип резервного копирования, расписание резервного копирования, срок хранения и необязательный временной промежуток проверки резервной копии (рисунок 8):

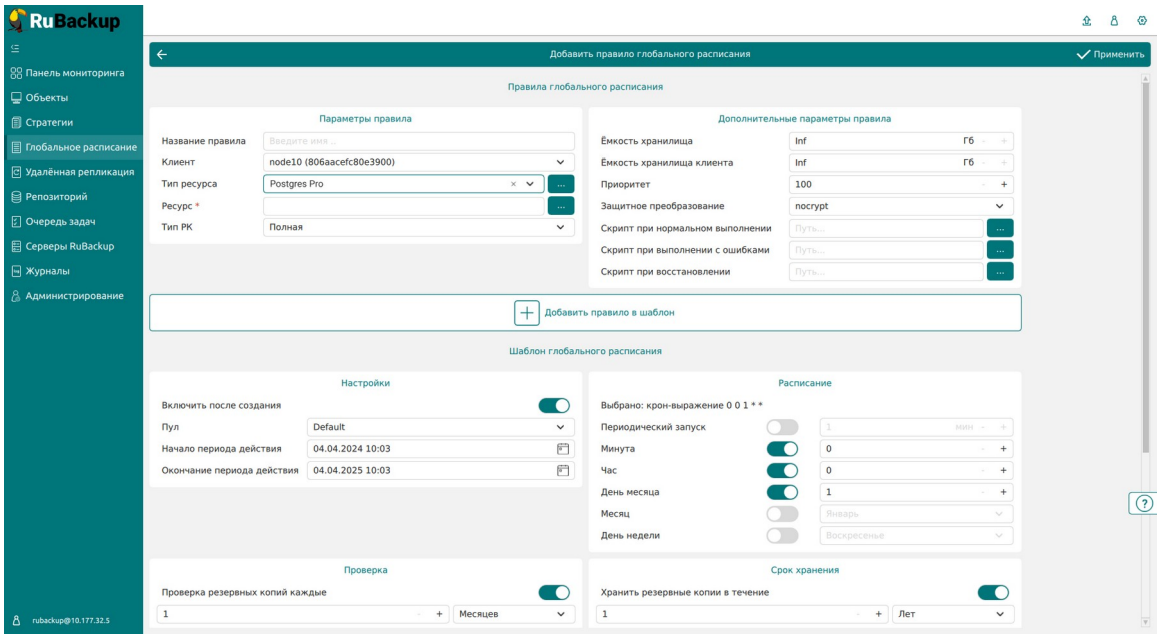


Рисунок 8

При необходимости, администратор может приостановить работу правила или немедленно запустить его (т.е. инициировать немедленное создание задачи при статусе правила wait).

4. Нажав на иконку «**...**» рядом с выбранным типом ресурса, установите дополнительные настройки правила резервного копирования.

Доступны следующие параметры:

– threads - количество потоков в которые будет выполняться резервное копирование или восстановление. По умолчанию количество потоков равно 1;

– differential_mode - выбор режима резервного копирования (**0 - DELTA, 1 - PAGE, 2 - PTRACK**). По умолчанию всегда **DELTA**;

– stream - выбор режима доставки WAL файлов. Если параметр имеет значение **True** (включен), то режимом доставки при выполнении резервного копирования будет **STREAM**, если значение примет **False** (выключен), то режимом будет **ARCHIVE**. По умолчанию значение равно **True** (включен).

Правила глобального расписания имеют срок жизни, определяемый при их создании, а также предоставляют следующие возможности:

1) Выполнить защитное преобразование резервной копии на клиенте.

2) Периодически выполнять проверку целостности резервной копии.

3) Хранить резервные копии определённый срок, а после его окончания удалять их из хранилища резервных копий и из записей репозитория, либо просто уведомлять пользователей системы резервного копирования об окончании срока хранения.

4) Через определённый срок после создания резервной копии автоматически переместить её на другой пул хранения резервных копий, например на картридж ленточной библиотеки.

5) Уведомлять пользователей системы резервного копирования о результатах выполнения тех или иных операций, связанных с правилом глобального расписания.

При создании задачи RuBackup она появляется в главной очереди задач. Отслеживать исполнение правил может как администратор, с помощью RBM, так клиент при помощи RBC.

После успешного завершения резервного копирования резервная копия будет размещена в хранилище резервных копий, а информация о ней будет размещена в репозитории RuBackup.

Менеджер клиента RuBackup (RBC)

Принцип взаимодействия клиентского менеджера (RBC) с системой резервного копирования состоит в том, что пользователь может сформировать ту или иную команду (желаемое действие) и отправить его серверу резервного копирования RuBackup. Взаимодействие пользователя с сервером резервного копирования производится через клиента (фоновый процесс) резервного копирования. Клиентский менеджер отправляет команду пользователя клиенту, клиент отправляет её серверу. В том случае, если действие допустимо, то сервер RuBackup отдаст обратную команду клиенту и/или перенаправит её медиасерверу RuBackup для дальнейшей обработки. Это означает, что, как правило, клиентский менеджер обычно не ожидает завершения того или иного действия, но ожидает ответа от клиента, что задание принято. Это позволяет инициировать параллельные запросы клиента к серверу резервного копирования, но требует от пользователя самостоятельно контролировать чтобы не было «встречных» операций, когда происходит восстановление данных, и в этот же момент эти же данные требуются для создания новой резервной копии. После того, как клиент отдал какую-либо команду при помощи RBC, он может просто закрыть приложение, все действия будут выполнены системой резервного копирования (тем не менее, стоит дождаться сообщения о том, что задание принято к исполнению, и проконтролировать это на вкладке «Задачи»).

Графический интерфейс клиентского менеджера поддерживает русский и английский языки.

Для запуска RBC следует выполнить команды:

```
# ssh -X user@postgreshost

# /opt/rubackup/bin/rbc&
```

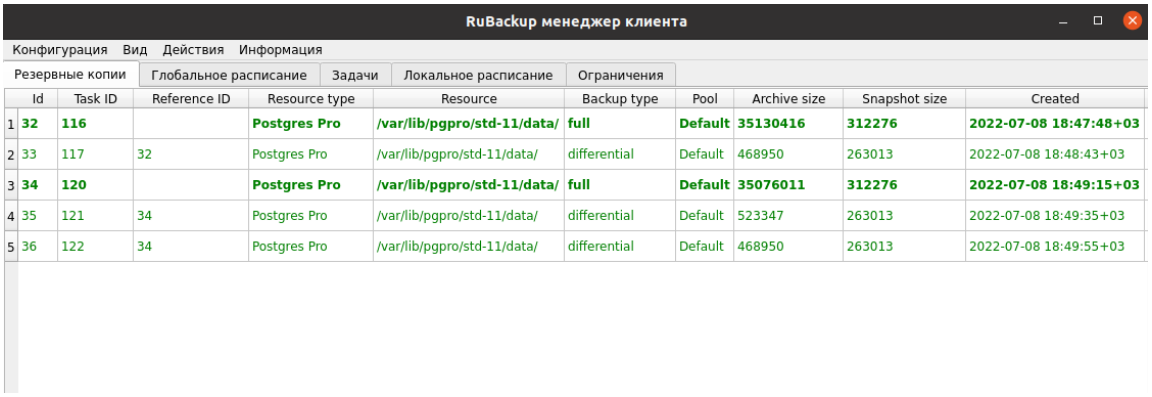
Пользователь, запускающий RBC, должен входить в группу rubackup.

При первом запуске клиентского менеджера необходимо задать пароль, при помощи которого впоследствии можно будет запросить восстановление резервной копии. Без ввода пароля получить резервную копию для клиента из хранилища невозможно. Хэш пароля восстановления хранится в базе данных RuBackup сервера. При необходимости можно изменить пароль при помощи клиентского менеджера (меню «**Конфигурация**» → «**Изменить пароль**»).

Главная страница RBC содержит переключающиеся вкладки, позволяющие управлять резервными копиями, расписанием резервного копирования, а также просматривать текущие задачи клиента, локальное расписание и ограничения.

Вкладка «Резервные копии»

В таблице вкладки «Резервные копии» содержится информация обо всех резервных копиях клиента, которые хранятся в репозитории RuBackup (рисунок 9). Дифференциальные резервные копии ссылаются на полные резервные копии, инкрементальные резервные копии ссылаются на полные резервные копии или предыдущие инкрементальные, так что при необходимости восстановить данные можно одной командой инициировать восстановление всей цепочки резервных копий.



	Id	Task ID	Reference ID	Resource type	Resource	Backup type	Pool	Archive size	Snapshot size	Created
1	**32**	**116**		**Postgres Pro**	**/var/lib/pgpro/std-11/data/**	**full**	**Default**	**35130416**	**312276**	**2022-07-08 18:47:48+03**
2	33	117	32	Postgres Pro	/var/lib/pgpro/std-11/data/	differential	Default	468950	263013	2022-07-08 18:48:43+03
3	**34**	**120**		**Postgres Pro**	**/var/lib/pgpro/std-11/data/**	**full**	**Default**	**35076011**	**312276**	**2022-07-08 18:49:15+03**
4	35	121	34	Postgres Pro	/var/lib/pgpro/std-11/data/	differential	Default	523347	263013	2022-07-08 18:49:35+03
5	36	122	34	Postgres Pro	/var/lib/pgpro/std-11/data/	differential	Default	468950	263013	2022-07-08 18:49:55+03

Рисунок 9

Во вкладке «Резервные копии» пользователю доступны следующие действия:

Удалить выбранную резервную копию.

Это действие возможно в том случае, если в правиле глобального расписания есть соответствующее разрешение. Кроме того, при необходимости выполнить удаление резервной копии потребуется вести пароль клиента.

Восстановить цепочку резервных копий.

Это действие запускает процесс восстановления цепочки резервных копий на локальной файловой системе клиента.

При восстановлении резервной копии или цепочки резервных копий клиент должен выбрать место для восстановления файлов резервной копии. Рекомендуется использовать временный каталог для операций с резервными копиями (например, /rubackup-tmp).

Если в файле /opt/rubackup/etc/rb_module_postgres_pro_13.conf параметр **direct_restore** имеет значение **yes**, то произойдёт остановка сервиса Postgres Pro, очистка каталога кластера баз данных, перемещение восстановленной полной резервной копии в очищенный каталог кластера баз данных. Если в файле /opt/rubackup/etc/rb_module_postgres_pro_13.conf параметр direct_restore имеет значение no, то восстановленные резервные

копии будут расположены в выбранном для восстановления каталоге и далее вы сможете провести восстановление кластера в ручном режиме. RBC не ожидает окончания восстановления всех резервных копий. Клиент должен проконтролировать на вкладке «Задачи» успешное завершение созданных задач на восстановление данных завершились успешно (статус задач Done). Для успешного выполнения этого действия требуется наличие достаточного свободного места в каталоге, предназначенном для создания и временного хранения резервных копий (см. параметр use-local-backupdirectory).

Проверить резервную копию.

Это действие инициирует создание задачи проверки резервной копии. Если резервная копия была подписана цифровой подписью, то будет проверен размер файлов резервной копии и сама резервная копия.

Вкладка «Глобальное расписание»

В таблице вкладки «Глобальное расписание» содержится информация обо всех правилах в глобальном расписании RuBackup для этого клиента (рисунок 10).

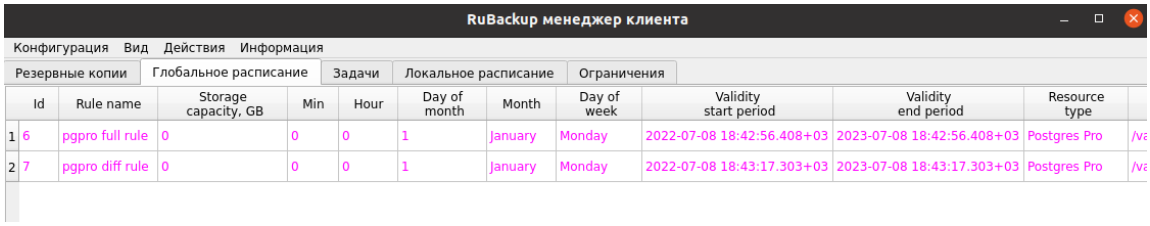


Рисунок 10

Во вкладке «Глобальное расписание» пользователю доступны следующие действия:

Запросить новое правило.

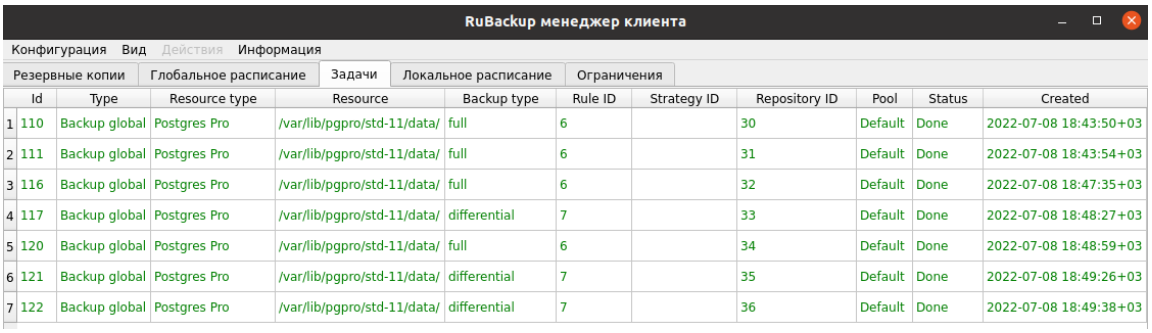
Это действие вызывает диалог подготовки нового правила в глобальном расписании RuBackup для данного клиента. Запрос на добавление правила требует одобрения администратора RuBackup, одобрение может быть сделано в оконном менеджере администратора RuBackup.

Запросить удалить правило из глобального расписания.

Это действие формирует запрос к администратору RuBackup об удалении выбранного пользователем правила из глобального расписания RuBackup. Запрос на удаление правила требует одобрения администратора RuBackup, одобрение может быть сделано в оконном менеджере администратора RuBackup.

Вкладка «Задачи»

В таблице вкладки «Задачи» содержится информация обо всех задачах в главной очереди заданий RuBackup для этого клиента (рисунок 11). В зависимости от настроек резервного сервера RuBackup выполненные задачи и задачи, завершившиеся неудачно, через какое-то время могут быть автоматически удалены из главной очереди задач. Информация о выполнении заданий фиксируется в специальном журнале задач сервера RuBackup, при необходимости статус любой задачи, даже удалённой из очереди, можно уточнить у администратора RuBackup. Так же информация о выполнении задач клиента заносится в локальный журнальный файл на клиенте. В клиентском менеджере можно открыть окно отслеживания журнального файла (меню «**Информация**» → «**Журнальный файл**»).



	Id	Type	Resource type	Resource	Backup type	Rule ID	Strategy ID	Repository ID	Pool	Status	Created
1	110	Backup global	Postgres Pro	/var/lib/pgpro/std-11/data/	full	6		30	Default	Done	2022-07-08 18:43:50+03
2	111	Backup global	Postgres Pro	/var/lib/pgpro/std-11/data/	full	6		31	Default	Done	2022-07-08 18:43:54+03
3	116	Backup global	Postgres Pro	/var/lib/pgpro/std-11/data/	full	6		32	Default	Done	2022-07-08 18:47:35+03
4	117	Backup global	Postgres Pro	/var/lib/pgpro/std-11/data/	differential	7		33	Default	Done	2022-07-08 18:48:27+03
5	120	Backup global	Postgres Pro	/var/lib/pgpro/std-11/data/	full	6		34	Default	Done	2022-07-08 18:48:59+03
6	121	Backup global	Postgres Pro	/var/lib/pgpro/std-11/data/	differential	7		35	Default	Done	2022-07-08 18:49:26+03
7	122	Backup global	Postgres Pro	/var/lib/pgpro/std-11/data/	differential	7		36	Default	Done	2022-07-08 18:49:38+03

Рисунок 11

Вкладка «Локальное расписание»

Во вкладке «Локальное расписание» можно определить правила, задаваемые клиентом для тех или иных локальных ресурсов. Для работы локального расписания эта возможность должна быть включена администратором RuBackup для клиента.

Вкладка «Ограничения»

Во вкладке «Ограничения» могут быть определены локальные ресурсы, резервное копирование которых нежелательно. Для работы локальных ограничений эта возможность должна быть включена администратором RuBackup для клиента.

Утилиты командной строки клиента

RuBackup

Для управления RuBackup со стороны клиента, помимо клиентского оконного менеджера, можно воспользоваться утилитами командной строки:

rb_archives

Утилита предназначена для просмотра списка резервных копий клиента в системе резервного копирования, создания срочных резервных копий, их удаления, проверки и восстановления.

 # rb_archives

```
root@postgresPro-client:~# rb_archives
Id | Ref ID | Resource                  | Resource type | Backup type  | Created               | Crypto  | Signed | Status
---+--------+---------------------------+---------------+--------------+-----------------------+---------+--------+------------
32 |        | /var/lib/pgpro/std-11/data/ | Postgres Pro  | full         | 2022-07-08 18:47:48+03 | nocrypt | True   | Trusted
33 | 32     | /var/lib/pgpro/std-11/data/ | Postgres Pro  | differential | 2022-07-08 18:48:43+03 | nocrypt | True   | Trusted
34 |        | /var/lib/pgpro/std-11/data/ | Postgres Pro  | full         | 2022-07-08 18:49:15+03 | nocrypt | True   | Trusted
35 | 34     | /var/lib/pgpro/std-11/data/ | Postgres Pro  | differential | 2022-07-08 18:49:35+03 | nocrypt | True   | Trusted
36 | 34     | /var/lib/pgpro/std-11/data/ | Postgres Pro  | differential | 2022-07-08 18:49:55+03 | nocrypt | True   | Trusted
```

rb_schedule

Утилита предназначена для просмотра имеющихся правил клиента в глобальном расписании резервного копирования.

#rb_schedule

```
root@postgresPro-client:~# rb_schedule
Id | Name             | Resource type | Resource                    | Backup type  | Status
---+------------------+---------------+-----------------------------+--------------+-------
6  | pgpro full rule  | Postgres Pro  | /var/lib/pgpro/std-11/data/ | full         | wait
7  | pgpro diff rule  | Postgres Pro  | /var/lib/pgpro/std-11/data/ | differential | wait
```

rb_tasks

Утилита предназначена для просмотра задач клиента, которые присутствуют в главной очереди задач системы резервного копирования.

#rb_tasks

```
root@postgresPro-client:~# rb_tasks
Id  | Task type     | Resource                    | Backup type  | Status | Created
----+--------------+-----------------------------+--------------+--------+----------------------
110 | Backup global | /var/lib/pgpro/std-11/data/ | full         | Done   | 2022-07-08 18:43:50+03
111 | Backup global | /var/lib/pgpro/std-11/data/ | full         | Done   | 2022-07-08 18:43:54+03
116 | Backup global | /var/lib/pgpro/std-11/data/ | full         | Done   | 2022-07-08 18:47:35+03
117 | Backup global | /var/lib/pgpro/std-11/data/ | differential | Done   | 2022-07-08 18:48:27+03
120 | Backup global | /var/lib/pgpro/std-11/data/ | full         | Done   | 2022-07-08 18:48:59+03
121 | Backup global | /var/lib/pgpro/std-11/data/ | differential | Done   | 2022-07-08 18:49:26+03
122 | Backup global | /var/lib/pgpro/std-11/data/ | differential | Done   | 2022-07-08 18:49:38+03
```

Восстановление резервной копии кластера Postgres Pro

Ход восстановления резервной копии кластера Postgres Pro зависит от значения параметра direct_restore в файле конфигурации модуля резервного копирования /opt/rubackup/etc/rb_module_postgres_pro_13.conf.

Если параметр direct_restore имеет значение yes, то произойдёт остановка сервера Postgres Pro, очистка каталога кластера баз данных (*только при значении restore_mode: 'CHECKSUM'*), перемещение разархивированной резервной копии в каталог кластера баз данных и запуск сервера Postgres Pro.

Если параметр direct_restore имеет значение no, то восстановленные резервные копии будут расположены в выбранном для восстановления каталоге, и восстановление СУБД можно будет провести вручную.

Клиент может осуществить восстановление данных резервной копии в оконном Менеджере Клиента RuBackup (RBC), либо при помощи утилиты командной строки rb_archives.

В случае восстановления дифференциальной резервной копии будет сформирована цепочка восстановления: вначале будет восстановлена полная резервная копия, на которую будут наложены изменения из дифференциальной резервной копии.

Восстановление резервной копии в RBC

Для восстановления данных резервной копии в оконном Менеджере Клиента RuBackup (RBC) необходимо выполнить следующие действия:

1. Выделить нужную резервную копию и в контекстном меню выбрать «Восстановить».
2. Ввести пароль клиента и далее RBC выведет информационное сообщение о дальнейших действиях.

3. Указать в качестве временного места восстановления резервных копий каталог, отдельный от копируемого каталога кластера баз данных (/var/lib/pgpro/std-13/data/).

4. Далее появится информационное сообщение о создании задачи на восстановление.

5. Проконтролировать результат процесса восстановления можно после автоматического переключения RBC на вкладку «Задачи» (рисунок 12):

	Id	Type	Resource type	Resource	Backup type	Rule ID	Strategy ID	Repository ID	Pool	Status	Created
1	110	Backup global	Postgres Pro	/var/lib/pgpro/std-11/data/	full	6		30	Default	Done	2022-07-08 ...
2	111	Backup global	Postgres Pro	/var/lib/pgpro/std-11/data/	full	6		31	Default	Done	2022-07-08 ...
3	116	Backup global	Postgres Pro	/var/lib/pgpro/std-11/data/	full	6		32	Default	Done	2022-07-08 ...
4	117	Backup global	Postgres Pro	/var/lib/pgpro/std-11/data/	differential	7		33	Default	Done	2022-07-08 ...
5	120	Backup global	Postgres Pro	/var/lib/pgpro/std-11/data/	full	6		34	Default	Done	2022-07-08 ...
6	121	Backup global	Postgres Pro	/var/lib/pgpro/std-11/data/	differential	7		35	Default	Done	2022-07-08 ...
7	122	Backup global	Postgres Pro	/var/lib/pgpro/std-11/data/	differential	7		36	Default	Done	2022-07-08 ...
8	128	Restore	Postgres Pro	/var/lib/pgpro/std-11/data/	full	6		32	Default	Done	2022-07-08 ...
9	129	Restore	Postgres Pro	/var/lib/pgpro/std-11/data/	full	6		34	Default	Done	2022-07-08 ...
10	130	Restore	Postgres Pro	/var/lib/pgpro/std-11/data/	differential	7		35	Default	Done	2022-07-08 ...

Рисунок 12

Восстановление при помощи утилиты rb_archives

Для восстановления резервных копий клиент может использовать утилиту командной строки rb_archives. Вызов следующий:

 # rb_archives

```
root@postgresPro-client:~# rb_archives
Id | Ref ID | Resource                  | Resource type | Backup type  | Created               | Crypto  | Signed | Status
---+--------+---------------------------+---------------+--------------+-----------------------+---------+--------+-----------
32 |        | /var/lib/pgpro/std-11/data/ | Postgres Pro  | full         | 2022-07-08 18:47:48+03 | nocrypt | True   | Trusted
33 | 32     | /var/lib/pgpro/std-11/data/ | Postgres Pro  | differential | 2022-07-08 18:48:43+03 | nocrypt | True   | Trusted
34 |        | /var/lib/pgpro/std-11/data/ | Postgres Pro  | full         | 2022-07-08 18:49:15+03 | nocrypt | True   | Trusted
35 | 34     | /var/lib/pgpro/std-11/data/ | Postgres Pro  | differential | 2022-07-08 18:49:35+03 | nocrypt | True   | Trusted
36 | 34     | /var/lib/pgpro/std-11/data/ | Postgres Pro  | differential | 2022-07-08 18:49:55+03 | nocrypt | True   | Trusted
```

В приведённом примере в системе резервного копирования присутствуют пять резервных копий с идентификаторами 32, 33, 34, 35 и 36. Для восстановления резервной копии 32 необходимо выполнить команду:

 # rb_archives -x 32

```
root@postgresPro-client:~# rb_archives -x 32
Password:
The archive will be restored in the directory: /rubackup-tmp
----> Restore archive chain: 32  < ----
Record ID: 32 has status: Trusted
TASK WAS ADDED TO QUEUE:131
```

В случае успешно принятой задачи команда вернёт список созданных задач, а восстановление будет происходить в фоновом режиме.

Проконтролировать процесс восстановления можно при помощи утилиты rb_tasks:

#rb_tasks

```
root@postgresPro-client:~# rb_tasks
Id   | Task type     | Resource                  | Backup type  | Status | Created
-----+---------------+---------------------------+--------------+--------+-----------
110  | Backup global | /var/lib/pgpro/std-11/data/ | full         | Done   | 2022-07-08
111  | Backup global | /var/lib/pgpro/std-11/data/ | full         | Done   | 2022-07-08
116  | Backup global | /var/lib/pgpro/std-11/data/ | full         | Done   | 2022-07-08
117  | Backup global | /var/lib/pgpro/std-11/data/ | differential | Done   | 2022-07-08
120  | Backup global | /var/lib/pgpro/std-11/data/ | full         | Done   | 2022-07-08
121  | Backup global | /var/lib/pgpro/std-11/data/ | differential | Done   | 2022-07-08
122  | Backup global | /var/lib/pgpro/std-11/data/ | differential | Done   | 2022-07-08
128  | Restore       | /var/lib/pgpro/std-11/data/ | full         | Done   | 2022-07-08
129  | Restore       | /var/lib/pgpro/std-11/data/ | full         | Done   | 2022-07-08
130  | Restore       | /var/lib/pgpro/std-11/data/ | differential | Done   | 2022-07-08
131  | Restore       | /var/lib/pgpro/std-11/data/ | full         | Done   | 2022-07-08
```

Вы можете проконтролировать процесс восстановления в файле журнала при помощи вызова:

 # tail -f /opt/rubackup/log/RuBackup.log

```
root@postgresPro-client:~# tail -f /opt/rubackup/log/RuBackup.log
Fri Jul  8 20:41:17 2022: INFO: Backup files are restored. Transfered bytes: 32MB, time elapsed: 1s
Fri Jul  8 20:41:17 2022: INFO: Restore incremental ratio (less is better): 6% (32MB/575MB)
Fri Jul  8 20:41:17 2022: INFO: Syncing restored files to disk
Fri Jul  8 20:41:19 2022: INFO: Restored backup files are synced, time elapsed: 2s
Fri Jul  8 20:41:19 2022: INFO: Restore of backup REPL7B completed.
Fri Jul  8 20:41:21 2022: The beginning of the process to clean up the '/opt/rubackup/mnt/pg_probackup/' directory
Fri Jul  8 20:41:21 2022: INFO: Resident data size to free by delete of backup REPL7B : 575MB
Fri Jul  8 20:41:21 2022: INFO: Delete: REPL7B 2022-07-08 18:47:38+03
Fri Jul  8 20:41:21 2022: The clean up process is completed.
Fri Jul  8 20:41:21 2022: Task was done. ID: 131
```