RuBackup

Система резервного копирования
и восстановления данных

МОДУЛЬ POSTGRESQL И TANTOR

ВЕРСИЯ 2.4.0, 02.07.2025

Содержание

Модуль PostgreSQL Universal предназначен для резервного копирования и восстановления СУБД PostgreSQL версий 11, 12, 13, 14, 15, 16 и СУБД Tantor Special Edition версий 15, 1С 15, 16 в режиме полного резервного копирования и резервного копирования архивных WAL. Возможно выбрать один из подмодулей PostgreSQL Universal при настройке правил и стратегий резервного копирования:

- postgresql — подмодуль, использующий низкоуровневый API СУБД PostgreSQL для выполнения резервного копирования;
- pg_probackup — подмодуль, использующий утилиту pg_probackup для выполнения резервного копирования СУБД Postgres Pro (поддерживаются версии PostgreSQL с 11 и выше);
- superb — подмодуль, предназначенный для резервного копирования и восстановления СУБД PostgreSQL в режиме непрерывного резервного копирования и резервного копирования архивных WAL.

Модуль PostgreSQL Universal обеспечивает поддержку дедупликации данных в ходе резервного копирования RuBackup.

Принцип резервного копирования СУБД PostgreSQL с использованием RuBackup состоит в периодическом создании полных резервных копий экземпляра СУБД и резервному копированию архивированных файлов WAL по определенному расписанию.

В СРК RuBackup поддерживается создание базовых резервных копий (полных резервных копий (full)), а также создание инкрементальных резервных копий (incremental, создаются на основе базовой резервной копии). Дифференциальное резервное копирование данных СУБД PostgreSQL не предусмотрено.

После успешного выполнения резервного копирования архивные файлы WAL могут быть автоматически удалены клиентом RuBackup из каталога с архивными WAL-файлами (параметр `archive_catalog` в конфигурационном файле модуля), если включен параметр `auto_remove_wal` в конфигурационном файле модуля.

Невозможно запустить одновременно две операции резервного копирования или восстановления для модуля PostgreSQL на одном хосте. Это предопределено тем, что выполнение резервного копирования СУБД производится по особой методике, не предусматривающей корректную возможность параллельных задач резервного копирования и восстановления для одной и той же СУБД. В том случае, если параллельная задача все-таки будет запущена, то она завершится с ошибкой.

Модуль PostgreSQL Universal поддерживает безопасное хранение аутентификационной информации для подключения к СУБД PostgreSQL с помощью интеграции с внешним хранилищем секретов HashiCorp Vault 1.16.3.

Глава 1. Ограничения

При поступлении задачи на создание инкрементальной резервной копии в СРК RuBackup производится анализ WAL-файлов. Если анализ показывает, что в текущем WAL-файле изменилась линия времени по сравнению с последним WAL-файлом из предыдущей итерации создания резервной копии, то вместо инкрементальной копии создается полная резервная копия. Это актуально для подмодулей postgresql и superb.

Настоящее руководство является описанием функционала и не является точной инструкцией по восстановлению СУБД в любой ситуации, которая может произойти!

При выполнении операции восстановления с развертыванием существующий кластер баз данных СУБД PostgreSQL будет уничтожен, а на его месте будет восстановлен кластер баз данных из резервной копии. Перед операцией восстановления рекомендуется принудительно остановить работу всех клиентов с СУБД и выполнить базовое резервное копирование!

Рекомендуется отключить возможность централизованного восстановления СУБД на клиенте и выполнять восстановление из резервной копии только со стороны клиента под контролем администратора СУБД.

Централизованное восстановление и восстановление с развертыванием рекомендуется предварительно выполнять на резервном хосте (виртуальной машине) для проверки корректности восстановления СУБД.

Модуль PostgreSQL Universal поддерживает работу с хранилищем секретов HashiCorp Vault 1.16.3 на платформах Astra Linux 1.7, Ubuntu 20.04, RHEL 8.

Глава 2. Установка клиента RuBackup

Для возможности резервного копирования при помощи RuBackup на защищаемый сервер с БД должен быть установлен клиент RuBackup и модуль резервного копирования PostgreSQL Universal.

Для корректной работы модуля PostgreSQL Universal на узел с клиентом RuBackup необходимо установить утилиты `sudo`, `lsof`, `grep`, `awk`.

Установка пакетов клиента RuBackup производится из-под учетной записи с административными правами при помощи следующих команд (имена пакетов могут отличаться в зависимости от используемой операционной системы):

```
sudo dpkg -i rubackup-common.deb
sudo dpkg -i rubackup-client.deb
sudo dpkg -i rubackup-postgresql.deb
```

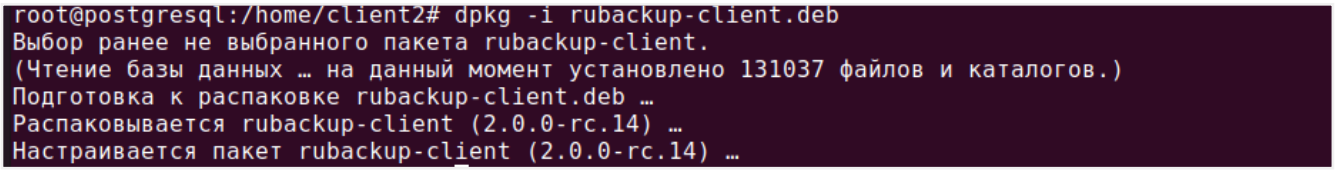

Рисунок 1. Установка пакета rubackup-client.deb

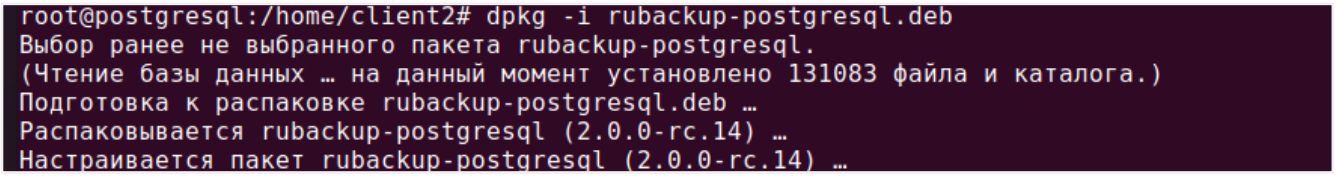

Рисунок 2. Установка пакета rubackup-postgresql.deb

При настройке клиента рекомендуется активировать функцию централизованного восстановления в тех случаях, когда предполагается восстановление СУБД из средства управления RBM.

В ходе инсталляции пакета модуля PostgreSQL Universal в системе будет создан файл настроек доступа к СУБД PostgreSQL `/opt/rubackup/etc/rb_module_postgresql.conf`. Измените в этом файле настройки для возможности подключения модуля к СУБД и выполнения резервного копирования (см. соответствующий раздел ниже).

При старте клиента RuBackup, в случае правильной настройки доступа к СУБД и корректной настройки самой СУБД для выполнения задач резервного копирования и восстановления, в журнальном файле `/opt/rubackup/log/RuBackup.log` на клиенте появится следующая запись:

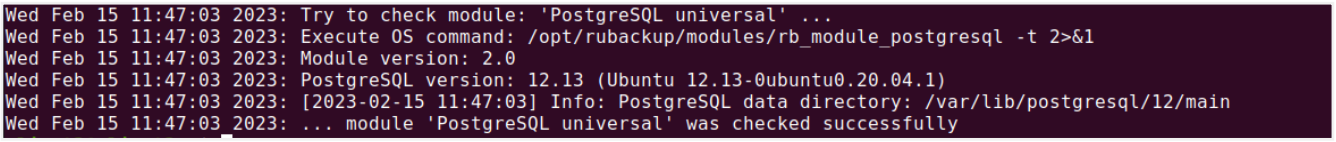

Рисунок 3. Старт клиента RuBackup

Подробно процедуры подготовки к установке, инсталляция, настройка и запуск клиента СРК описаны в документе «Руководство по установке и обновлению серверов резервного копирования и Linux клиентов RuBackup».

2.1. Обновление конфигурационного файла

Новая версия модуля содержит конфигурационный файл, параметры которого могут отличаться от текущей версии, поэтому при обновлении модуля на новую версию также обновляется и его конфигурационный файл. Для переноса значений параметров настроек из старого конфигурационного файла в новый предусмотрен механизм слияния конфигурационных файлов.

Может существовать 3 версии конфигурационного файла:

- `/opt/rubackup/etc/rb_module_postgresql.conf` — текущий конфигурационный файл модуля. После слияния будет переименован в rb_module_postgresql_old.conf.
- `/opt/rubackup/etc/rb_module_postgresql_old.conf`` — старый конфигурационный файл, который был загружен в предыдущее обновление или при установке модуля.
- `/opt/rubackup/etc/rb_module_postgresql_upgrade.conf`` — конфигурационный файл обновления. Должен быть создан вручную. Данный конфигурационный файл используется для замены значения параметров при обновлении модуля. При обновлении модуля обновится и его конфигурационный файл, а значения параметров будут взяты из старого конфигурационного файла. Если же в `rb_module_postgresql_upgrade.conf` записано другое значение, то будет использовано оно. Это единственный способ подменить значения параметров модуля при обновлении.

Механизм слияния конфигурационных файлов запускается автоматически при обновлении пакета deb или rpm.

2.2. Автоматическое обновление конфигурационного файла

Автоматическое обновление конфигурационного файла выполняется при обновлении пакетов deb или rpm и не требует действий от пользователя.

Порядок автоматического обновления:

1. Текущий конфигурационный файл `rb_module_postgresql.conf` переименовывается в `rb_module_postgresql_old.conf`.
2. Создается файл `/opt/rubackup/etc/rb_module_postgresql.conf`, который далее будет использован в качестве текущего.
3. В созданный файл `rb_module_postgresql.conf` добавляются параметры конфигурационного файла, которые поставляются в пакете deb или rpm. При этом все параметры закомментированы (выставлен символ `#` перед каждой строкой).
4. Происходит слияние старого конфигурационного файла, конфигурационного файла обновления, и нового конфигурационного файла, который поставляется в пакете, при этом:
    - Значение каждого параметра берется из конфигурационного файла обновления.
    - Если в конфигурационном файле обновления параметра нет, то значение берется из старого конфигурационного файла.
    - Если в старом конфигурационном файле значение параметра отсутствует, то такое значение:
        - Добавляется, если это обязательная настройка. Добавляется без значения.
        - Не добавляется, если настройка не обязательная.
    - Если у обязательного параметра нет значения, то при установке пакета возникнет ошибка. Информацию об ошибке можно посмотреть в логе установки:

```
[2024-03-18 12:11:52] Info: UpgradeConfig options.configs_list: /media/nik/Special/resource/test/ol
[2024-03-18 12:11:52] Error: Variable 'host' is mandatory and has not value. Module cannot be used
[2024-03-18 12:11:52] Error: Variable 'port' is mandatory and has not value. Module cannot be used
```

Рисунок 4. Информация об ошибке

В результате автоматического обновления будет обновлен конфигурационный файл `rb_module_postgresql.conf`. Модуль PostgreSQL Universal будет готов к работе.

При слиянии конфигурационных файлов будут удалены все комментарии из `rb_module_postgresql.conf`.

Если при обновлении конфигурационного файла возникли ошибки, то пользователю необходимо проверить корректность `/opt/rubackup/etc/rb_module_postgresql.conf` и при необходимости заполнить параметры вручную.

Глава 3. Конфигурационный файл модуля

В ходе инсталляции пакета модуля в системе создается конфигурационный файл `/opt/rubackup/etc/rb_module_postgresql.conf`.

Содержимое конфигурационного файла

```
# Symbol "#" at the beginning of the line treats as a comment
# "#" in the middle of the line treats as a parameter value
# So please do not use comments in one line with parameter

dbname postgres
username rubackup_backuper
password 12345
host localhost
port 5432

#Enable interaction with centralized secret repositories
use_secret_storage no
archive_catalog /opt/rubackup/mnt/postgresql_archives
# Specify this path according to the installed version
pg_ctl /usr/lib/postgresql/12/bin/pg_ctl
postgresql_service_name postgresql
pg_waldump /usr/lib/postgresql/12/bin/pg_waldump
num_threads_for_wal_processing 8
# Specify if custom built PostgreSQL binary is required
#pg_binary /usr/bin/custom/postgres
# Specify if server output should be redirected to the file
#pg_log /tmp/postgres.log
auto_remove_wal yes
postgresql_admin postgres
# Timeout period for the last WAL file generated during backup(in seconds)
wal_wait_timeout 10
# Availability check period for last WAL file generated during backup(in
seconds)
wal_check_period 1
# Patroni parameters are optional
# and may be needed for the module work in a patroni cluster
patroni_host localhost
patroni_port 8008
patroni_node_type_for_backup leader
# Далее идут параметры для подмодуля PgProbackup
# Возможные значения для restore_target_action: [pause | promote | shutdown]
restore_target_action pause
# Возможные значения для restore_target: [immediate | latest]
```

```
restore_target immediate
# Абсолютный путь до утилиты pg_probackup
pg_probackup /opt/pgpro/std-13/bin/pg_probackup
# Абсолютный путь до каталога, в котором хранятся резервные копии
probackup_catalog_copies /opt/rubackup/mnt/pg_probackup
# Имя инстанса. Имя подкаталогов, в которых будут храниться копии
probackup_instance_name data
# Возможные значения для s3_interface: [minio | vk]
s3_interface minio
# Путь до утилиты pg_receivewal (физическая репликация), начиная с 10 версии
pg_receivewal /usr/lib/postgresql/10/bin/pg_receivewal
# Использовать стрим режим. Возможные значения "yes", "no"
stream no
# Выполнить drop физического слота с именем slot_name после создания РК
drop_slot no
# Использовать физический слот, заданный в системе, иначе создать с указанным
именем
slot_name my_slot_name
# Директория для сохранения репликационных данных
replication_catalog /opt/rubackup/mnt/postgresql_replica
# Использовать move для переноса WAL из архива вместо копирования
move_on_archive_get yes
# Количество потоков для архивации WAL
num_threads_for_wal_archiving 1
# Количество файлов обрабатываемых за один вызов архивации
batch_size_for_wal_archiving 1
# Макс. размер хранимых локально архивных WAL файлов (0 - нет лимита)
wal_archive_files_size 0
# Макс. время в секундах которое процесс снятия РК ждет освобождения
архивного каталога
cleanup_wait_timeout 1000
# Выполнять проверку работы команд архивации перед снятием РК
make_archiving_check yes
```

Параметры из конфигурационного файла `rb_module_postgresql.conf` представлены в таблице.

Таблица 1. Параметры файла конфигурации модуля резервного копирования PostgreSQL

Параметр	Назначение	Значение по умолчанию
`dbname`	Имя базы данных, резервное копирование которой будет выполняться	`postgres`
`username`	Имя пользователя в СУБД PostgreSQL, обладающего правами выполнять резервное копирование	`rubackup_backuper`

Параметр	Назначение	Значение по умолчанию
`password`	Пароль для пользователя, указанного в параметре `username`	
`host`	IP-адрес или доменное имя локального хоста, на котором СУБД принимает подключения. Используется для взаимодействия с СУБД, резервное копирование которой выполняется. Параметр необязательный, т.е. его можно не указывать в конфигурационном файле	`localhost`
`port`	Порт для соединения с СУБД. Параметр необязательный, т.е. его можно не указывать в конфигурационном файле	`5432`
`use_secret_storage`	Использование хранилища секретов HashiCorp vault v1.16.3	`no`
`archive_catalog`	Каталог для хранения архивных WAL	`/opt/rubackup/mnt/postgresql_archives`
`pg_ctl`	Используется для запуска и остановки СУБД PostgreSQL во время восстановления с развертыванием. Местонахождение `pg_ctl` зависит от используемой версии.	`/usr/lib/postgresql/12/bin/pg_ctl`
`pg_binary`	Используется при вызове утилиты `pg_ctl` для запуска PostgreSQL во время восстановления с развертыванием. Указывает путь к исполняемому файлу `postgres`. Параметр `pg_binary` не является обязательным. По умолчанию исполняемый файл `postgres` берется из того же каталога, из которого запускался `pg_ctl`, а если найти файл невозможно, то из жёстко заданного каталога инсталляции.	–
`pg_log`	Используется при вызове утилиты `pg_ctl` для запуска PostgreSQL во время восстановления с развертыванием. В указанный файл будет направляться вывод сообщений сервера. Файл создаётся, если он ещё не существует. Параметр не является обязательным.	–
`postgresql_service_name`	Служебное имя базы данных	`postgresql`
`pg_waldump`	Путь до утилиты `pg_waldump`. Параметр необходимо задать для работы подтипа инкрементального резервного копирования page (при использовании подмодуля postgresql). Местонахождение `pg_waldump` зависит от используемой версии PostgreSQL.	`/usr/lib/postgresql/12/bin/pg_waldump`
`num_threads_for_wal_processing`	Количество процессов, выделенных для обработки архивных WAL файлов	`8`
`auto_remove_wal`	В случае значения `yes` архивные WAL будут удалены из каталога `archive_catalog` после выполнения резервного копирования (если они включены в резервную копию)	`yes`
`postgresql_admin`	Login администратора PostgreSQL в операционной системе	`postgres`

Параметр	Назначение	Значение по умолчанию
`execute_only_on_leader`	В случае значения `yes` резервное копирование выполняется только на лидере кластера Patroni. В случае активации параметра модуль возвращает отрицательный ответ серверу на запрос о наличии ресурса, если хост, на котором производится проверка, не является лидером кластера Patroni. Параметр применяется только при работе в кластере Patroni и используется только в версии модуля 2.0 и ниже. В конфигурационном файле модуля версии 2.1 параметр заменён на `patroni_node_type_for_backup`.	`no`
`patroni_node_type_for_backup`	В случае указания значения `leader`, ресурс будет доступен только при условии, что узел на котором установлен модуль с таким значением имеет роль leader в кластере Patroni. В случае указания значения `sync`, ресурс будет доступен только при условии, что узел, на котором установлен модуль с таким значением, имеет роль sync stanby в кластере Patroni. В случае указания значения `async`, ресурс будет доступен только при условии, что узел, на котором установлен модуль с таким значением, имеет роль replica в кластере Patroni. Параметр `patroni_node_type_for_backup` заменяет в версии модуля 2.1 параметр `execute_only_on_leader`.	
`wal_wait_timeout`	Период ожидания окончания архивации последнего WAL-файла, сгенерированного во время создания резервной копии.	`10`
`wal_check_period`	Период проверки окончания архивации последнего WAL-файла, сгенерированного во время создания резервной копии	`1`
`patroni_host`	IP-адрес, на котором Patroni принимает входящие запросы Rest API. Параметр необязательный (т.е. его можно не указывать в конфигурационном файле) и необходим только для взаимодействия модуля с Rest API локального процесса Patroni. Если значение параметра не указано, будет предпринята попытка автоматически определить значение для этого параметра через утилиту `lsof`.	`localhost`
`patroni_port`	Порт, на котором локальный процесс Patroni слушает запросы Rest API. Параметр необязательный (т.е. его можно не указывать в конфигурационном файле) и необходим только для взаимодействия модуля с Rest API локального процесса Patroni. Если значение параметра не указано, будет предпринята попытка автоматически определить значение для этого параметра через утилиту `lsof`.	`8008`
`restore_target_action`	Восстановление целевого действия: В случае указания значения `pause`, восстановление будет приостановлено. В случае указания значения `promote`, восстановление будет продолжено. В случае указания значения `shutdown` восстановление не производится. Параметр используется только для подмодуля pg_probackup.	`pause`

Параметр	Назначение	Значение по умолчанию
restore_target	Какое восстановление будет выполнено: В случае указания значения immediate, будет восстановлена срочное РК. В случае указания значения latest, будет восстановлена последняя РК. Параметр используется только для подмодуля pg_probackup.	immediate
pg_probackup	Абсолютный путь до утилиты pg_probackup. Параметр используется только для подмодуля pg_probackup.	/opt/pgpro/std-13/bin/pg_probackup
probackup_catalog_copies	Абсолютный путь до каталога, хранящего резервные копии. Параметр используется только для подмодуля pg_probackup.	/opt/rubackup/mnt/pg_probackup
probackup_instance_name	Имя подкаталога для хранения резервных копий. Параметр используется только для подмодуля pg_probackup.	data
s3_interface minio	Интерфейс облачного хранилища В случае указания значения minio, будет использоваться объектное хранилище MinIO, совместимое с облачным хранилищем S3. В случае указания значения vk, будет использоваться хранилище vk. Параметр используется только для подмодуля pg_probackup.	minio
pg_receivewal	Путь до утилиты pg_receivewal. Утилита обрабатывает ошибки и сохраняет файлы. Параметр используется только для подмодуля pg_probackup.	/usr/lib/postgresql/10/bin/pg_receivewal
stream	Параметр, указывающий на использование режима потокового резервного копирования. В случае указания значения yes, будет включен режим STREAM. В случае указания значения no, режим STREAM будет выключен. Параметр используется только для подмодуля pg_probackup.	no
drop_slot	Выполняет drop физического слота. В случае указания значения yes, drop выполнится. В случае указания значения no, drop не выполнится. Параметр используется только для подмодуля pg_probackup.	no
slot_name	Имя физического слота	my_slot_name
replication_catalog	Абсолютный путь до каталога с репликационными данными. Параметр используется только для подмодуля pg_probackup.	/opt/rubackup/mnt/postgresql_replica
move_on_archive_get	Переносит WAL файлы вместо копирования. В случае указания значения yes, WAL файлы будут перенесены. В случае указания значения no, будут созданы копии WAL файлов. Параметр используется только для подмодуля pg_probackup.	yes
num_threads_for_wal_archiving	Количество потоков для архивации WAL файлов. Параметр используется только для подмодуля pg_probackup.	1
batch_size_for_wal_archiving	Количество обрабатываемых WAL файлов, которые одновременно отправляются в архивный каталог. Параметр используется только для подмодуля pg_probackup.	1

Параметр	Назначение	Значение по умолчанию
`wal_archive_files_size`	Максимальный размер архивных WAL файлов, хранимых локально (в Мб). Если установлено значение 0, то лимита нет. При превышении размера архива запускается фоновая очистка. Во время резервного копирования очистка блокируется. Параметр используется только для подмодуля pg_probackup.	`0`
`cleanup_wait_timeout`	Максимальное время (в секундах) ожидания освобождения архивного каталога, которое ждет процесс снятия PK. Параметр используется только для подмодуля pg_probackup.	`1000`
`make_archiving_check`	Проверка работы архивации до запуска длительного резервного копирования. В случае указания значения yes, будет выполнена проверка команд архивации перед снятием PK. При успешной проверке резервное копирование проходит в штатном режиме. При неуспешной проверке в журнале появляется запись и резервное копирование останавливается. В случае указания значения no, проверка команд архивации перед снятием PK выполнена не будет. Параметр используется только для подмодуля pg_probackup.	`yes`

Глава 4. Удаление клиента RuBackup

Порядок удаления клиента RuBackup изложен в документе «Руководство по установке серверов резервного копирования и Linux клиентов RuBackup».

Глава 5. Подготовка СУБД PostgreSQL

Подготовка СУБД PostgreSQL к выполнению резервного копирования при помощи СРК RuBackup включает:

1. Подготовку сервера с СУБД PostgreSQL;
2. Создание пользователя СУБД для безопасного выполнения резервной копии PostgreSQL.

Для подготовки pg_probackup перейдите к разделу Резервное копирование с использованием подмодуля pg_probackup.

Для корректной работы восстановления с развертыванием в ОС Alt Linux необходимо в файле `/etc/passwd` добавить строку `postgres:x:46:46:PostgreSQL Server:/var/lib/pgsql:/bin/bash` и закомментировать строку `postgres:x:46:46:PostgreSQL Server:/var/lib/pgsql:/dev/null`

5.1. Подготовка сервера с СУБД PostgreSQL

Для подготовки сервера с СУБД PostgreSQL необходимо выполнить следующие действия:

1. Для обеспечения доступа пользователя rubackup_backuper к СУБД измените метод доступа в конфигурационном файле СУБД PostgreSQL `/etc/postgresql/12/main/pg_hba.conf` (расположение файла может отличаться в зависимости от дистрибутива Linux и версии PostgreSQL) на `md5`.

```
# 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
```

Рисунок 5. Пример итогового файла

2. Для непрерывного архивирования и восстановления СУБД PostgreSQL необхо-

димо включить архивирование WAL, для чего:

- в конфигурационном файле СУБД PostgreSQL `/etc/postgresql/12/main/postgresql.conf` (расположение файла может отличаться в зависимости от дистрибутива Linux и версии PostgreSQL) настройте следующие параметры:

```
wal_level = replica
archive_mode = on
archive_command '/opt/rubackup/modules/rb_module_postgresql
pgsql-archive-push %p'
```

  Также возможно использовать команду:

```
archive_command = 'cp %p /opt/rubackup/mnt/ postgresql_archives/%f'
```

- там же установите значение параметра `data_directory` (если оно не определено), иначе модуль резервного копирования не сможет определить местоположение файлов СУБД:

```
data_directory = '/var/lib/postgresql/12/main'
```

- в файле `postgresql.conf` для версий PostgreSQL 12 и более новых, должна быть прописана строка, определяющая порядок развертывания СУБД из резервной копии:

```
restore_command '/opt/rubackup/modules/rb_module_postgresql pgsql-
archive-get %f %p'
```

  Также возможно использовать команду:

```
restore_command = 'cp /opt/rubackup/mnt/ postgresql_archives/%f %p'
```

  Без добавления этой строки для версий PostgreSQL 12 и более новых модуль будет отказываться стартовать и будет выдавать сообщение об ошибке:

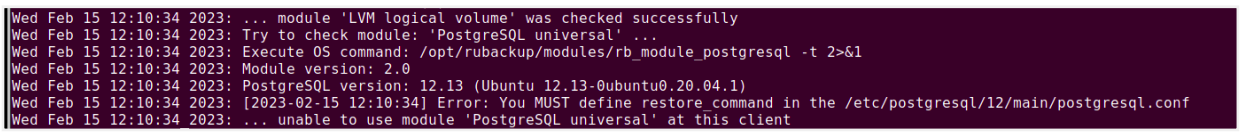

Рисунок 6. Сообщение об ошибке

  Если размер архива слишком большой, произведите сжатие архивных фай-

лов WAL утилитой gzip:

```
archive_command = 'gzip < %p >
/opt/rubackup/mnt/postgresql_archives/%f.gz'
```

Для сжатия архивных файлов можно воспользоваться любой другой утилитой, но таким образом, чтобы имя WAL-файлов после сжатия было формата TTTTTTTTXXXXXXXXYYYYYYYY.extension, где extension – это расширение файла.

Для восстановления архива используйте утилиту gunzip или любую другую подходящую утилиту:

```
restore_command = 'gunzip < /opt/rubackup/mnt/postgresql_archives/%f.gz
> %p'
```

3. После внесения изменений в конфигурационный файл перезапустите PostgreSQL командой:

```
sudo service postgresql restart
```

Значение параметра `archive_command` должно содержать каталог в файловой системе сервера PostgreSQL, в который будут копироваться архивируемые сегменты WAL.

В настройках RuBackup для каждой СУБД PostgreSQL в файле `/opt/rubackup/etc/rb_module_postgresql.conf` определен параметр archive_catalog, содержащий значение каталога, в котором предполагается временное хранение архивных WAL-файлов. Значение этого параметра по умолчанию:

```
/opt/rubackup/mnt/postgresql_archives/
```

При планировании установки СРК RuBackup вы можете назначить для хранения архивных WAL-файлов выделенное хранилище требуемого размера и сделать на него ссылку на том сервере PostgreSQL, где это требуется.

Объем необходимого пространства под архивные WAL-файлы зависит от нагруженности базы данных и периодичности бэкапов, а также значения параметра `auto_remove_wal` в конфигурационном файле.

Указанный каталог должен быть доступен для записи и чтения пользова-

телю postgres, а также пользователю, под контролем которого работает клиент RuBackup!

Обеспечить это можно командой:

```
sudo chown postgres:postgres /opt/rubackup/mnt/postgresql_archives/
```

Для правильной работы клиента RuBackup параметр `archive_catalog` в конфигурации RuBackup и параметр `archive_command` в конфигурационном файле PostgreSQL должны иметь одинаковое значение для одной и той же СУБД.

После изменения параметров конфигурационного файла необходимо перезагрузить PostgreSQL при помощи команды:

```
sudo systemctl restart postgresql
```

При настройке резервного копирования PostgreSQL в ОС Astra Linux SE 1.6 и 1.7 необходимо в файле `/etc/parsec/mswitch.conf` для параметра `zero_if_notfound` установить значение yes и затем перезагрузить сервис PostgreSQL:

```
sudo service postgresql restart
```

5.2. Создание пользователя СУБД для безопасного выполнения базовой резервной копии PostgreSQL

Пользователь для выполнения операции создания базовой резервной копии должен обладать правами на выполнение функций начала и окончания резервного копирования экземпляра PostgreSQL. Для настройки выполните следующие действия:

1. Вызовите `psql` при помощи команды:

   ```
   sudo -u postgres psql
   ```

2. В `psql` создайте пользователя `rubackup_backuper` (в качестве пароля укажите желаемый пароль вместо `12345`):

   ```
   create user rubackup_backuper password '12345';
   alter role rubackup_backuper with login;
   ```

В PostgreSQL версии 14 и ниже используются функции pg_stop_backup и pg_start_backup, а в версии 15 и выше - pg_backup_stop и pg_backup_start.

```
grant execute on function pg_backup_start to rubackup_backuper;
grant execute on function pg_backup_stop(bool, bool) to rubackup_backuper;
grant execute on function pg_switch_wal to rubackup_backuper;
grant pg_read_all_settings to rubackup_backuper;
```

Вместо пользователя rubackup_backuper вы можете создать пользователя с другим именем и с таким же набором прав. В файле конфигурации модуля `/opt/rubackup/etc/rb_module_postgresql.conf` необходимо указать имя пользователя и его пароль:

```
sudo cat /opt/rubackup/etc/ rb_module_postgresql.conf
username rubackup_backuper
password 12345
```

Для параметра `pg_ctl` необходимо указать абсолютный путь для используемой версии PostgreSQL.

5.3. Настройка SELinux

В некоторых случаях SELinux может блокировать выполнение резервного копирования модуля PostgreSQL. На это может указывать следующая ошибка в логах postgresql:

```
cp: cannot create regular file
'/opt/rubackup/mnt/postgresql_archives/00000001000004B500000077':
Permission denied
```

Для того, чтобы устранить данную ошибку, выполните следующие шаги:

1. Установите необходимые инструменты управления SELinux:

   Убедитесь, что у вас установлен пакет policycoreutils-python-utils (или policycoreutils-python в зависимости от вашего дистрибутива):

   ```
   sudo yum install policycoreutils-python-utils # For RHEL/CentOS/Fedora
   sudo apt install policycoreutils-python-utils # For Debian/Ubuntu
   ```

2. Создайте пользовательский модуль политики SELinux:

   Сначала создайте файл для определения вашей пользовательской политики. Например, создайте файл с именем `my_custom_policy.te`:

   ```
   vi my_custom_policy.te
   ```

   Добавьте в этот файл следующее содержимое, заменив `/path/to/your/folder` фактическим путем к каталогу, который вы хотите исключить, и именем `my_custom_t1` пользовательского типа:

   ```
   module my_custom_policy 1.0;
   require
   {
     type unconfined_t;
     type my_custom_t;
   }
   type my_custom_t;
   allow unconfined_t my_custom_t:file { read write execute };
   allow unconfined_t my_custom_t:dir { read write add_name remove_name };
   ```

3. Скомпилируйте и установите модуль политики:
   - Скомпилируйте модуль политики:

     ```
     checkmodule -M -m -o my_custom_policy.mod my_custom_policy.te
     semodule_package -o my_custom_policy.pp -m my_custom_policy.mod
     ```

   - Установите модуль политики:

     ```
     sudo semodule -i my_custom_policy.pp
     ```

4. Пометьте каталог пользовательским типом SELinux:

   Используйте команду `semanage fcontext`, чтобы добавить контекст, и команду restorecon, чтобы его применить:

   ```
   sudo semanage fcontext -a -t my_custom_t "/path/to/your/folder(/.*)?"
   sudo restorecon -R -v /path/to/your/folder
   ```

Глава 6. Подготовка к использованию централизованного хранилища секретов HashiCorp Vault 1.16.3

Аутентификационная информация (далее по тексту — секрет) для подключения к СУБД PostgreSQL, с целью резервного копирования или восстановления данных СУБД, хранится в конфигурационном файле, что не является безопасным подходом, т.к. злоумышленники могу получить доступ к содержимому файла, если он недостаточно защищён.

Более безопасным подходом к хранению секретов является использование инструмента управления секретами, таким как HashiCorp Vault. Эта система позволяет шифровать секреты и хранить их в безопасном хранилище, к которому имеют доступ авторизованные пользователи СРК RuBackup с соответствующими правами доступа.

Интеграция СРК RuBackup с хранилищем секретов HashiCorp Vault происходит через основной сервер (при его недоступности — через резервный сервер) посредством интерфейса REST API.

Интеграция СРК RuBackup с хранилищем секретов HashiCorp Vault поддерживается только при создании и восстановлении полных и инкрементальных резервных копий.

6.1. Права доступа

Права доступа к секретам хранилища предоставляются пользователям СРК RuBackup в соответствии с таблицей.

Таблица 2. Права доступа пользователей СРК RuBackup к секретам хранилища HashiCorp Vault

Редактирование данных хранилища секретов	+	—	—	—	—
Добавление данных хранилища секретов	+	—	—	—	—
Удаление данных хранилища секретов	+	—	—	—	—
Добавление методов получения секретов в СРК	+	—	—	—	—
Просмотр методов получения секретов в СРК	+	C	—	—	C
Редактирование методов получения секретов в СРК	+	—	—	—	—
Удаление методов получения секретов в СРК	+	—	—	—	—
Назначение доступа к методам получения секретов в СРК	+	—	—	—	—

Примечание: C — доступ назначает Суперпользователь на выбранный метод

6.2. Подготовка к использованию

Предварительно следует:

1. Получите следующие данные от Администратора хранилища секретов:
   - подтверждение, что секрет создан в хранилище секретов HashiCorp Vault. Один секрет может содержать несколько наборов аутентификационной информации в формате .json;
   - токен для доступа к хранилищу секретов HashiCorp Vault;
   - метод доступа к секрету.
2. На хосте с развёрнутым RBM, посредством которого будет производится резервное копирование и восстановление данных СУБД PostgreSQL, необходимо:
   - добавить запись о сервере хранилища секретов HashiCorp в файл `/etc/hosts`, например, выполнив в терминале команду:

     ```
     sudo echo "ip-address hostname" >> /etc/hosts
     ```

     где:

     - `ip-address` — ip адрес сервера хранилища секретов HashiCorp.
     - `hostname` — имя сервера хранилища секретов HashiCorp.
3. сгенерировать запрос на сертификат и подписать его на сервере хранилища секретов HashiCorp (для использования метода https), выполнив в терминале команду:

   ```
   sudo openssl s_client -showcerts -servername hostname -connect
   hostname:8200 > /tmp/cacert.pem
   ```

   где:

   - `hostname` — имя сервера хранилища секретов HashiCorp;
   - `8200` — порт (по умолчанию) для прослушивания запросов к хранилищу секретов HashiCorp Vault по безопасному протоколу https;
   - `/tmp/` — папка, в которую будет сохранён сгенерированный и подписанный сертификат;
   - `cacert.pem` — имя файла сертификата в формате .pem, подписанного на сервере хранилища секретов HashiCorp.
4. Суперпользователь СРК RuBackup должен произвести следующие настройки

для использования секретов хранилища HashiCorp Vault:

- в конфигурационном файле модуля `rb_module_postgresql.conf` добавьте параметр использования Хранилища `use_secret_storage` со значением `yes`;
- в конфигурационном файле модуля `rb_module_postgresql.conf` удалите параметры, например, username и/или password, которые содержит используемый секрет;
- после авторизации в RBM с правами суперпользователя произведите настройки для работы с хранилищем секретов:
  - добавьте хранилище секретов;
  - добавьте метод получения секрета в хранилище;
  - назначьте Администратору и/или Супервайзеру доступный метод получения секрета.

Глава 7. 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.

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

Ключи рекомендуется после создания скопировать на внешний носитель, а также распечатать бумажную копию и убрать эти копии в надежное 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
```

Глава 8. Защитное преобразование резервных копий

При необходимости, сразу после выполнения резервного копирования архивы 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).

Выбрать тип защитного преобразования можно в Менеджере администратора Rubackup при создании правила или стратегии (подробнее — в документе «Руководство системного администратора RuBackup»).

Глава 9. Менеджер администратора RuBackup (RBM)

Оконное приложение Менеджер администратора RuBackup (RBM) предназначено для администрирования серверной группировки RuBackup, включая управление клиентами, глобальным расписанием, хранилищами резервных копий и другими параметрами RuBackup.

В RuBackup 2.1 RBM располагается в отдельном пакете и может быть установлен как на сервер резервного копирования, так и на удаленном автоматизированном рабочем месте администратора (подробнее — в документе «Руководство системного администратора RuBackup»).

RuBackup 2.1 предоставляет многопользовательскую модель доступа к системе резервного копирования. При запуске RBM вам потребуется пройти аутентификацию. Уточните login/password для вашей работы у главного администратора СРК. Если вы главный администратор, то используйте для авторизации суперпользователя rubackup и тот пароль, который вы задали ему при инсталляции.

Для запуска RBM выполните команду:

```
rbm
```

После этого в открывшемся окне введите наименование сервера RuBackup, имя пользователя и пароль (Рисунок 7).

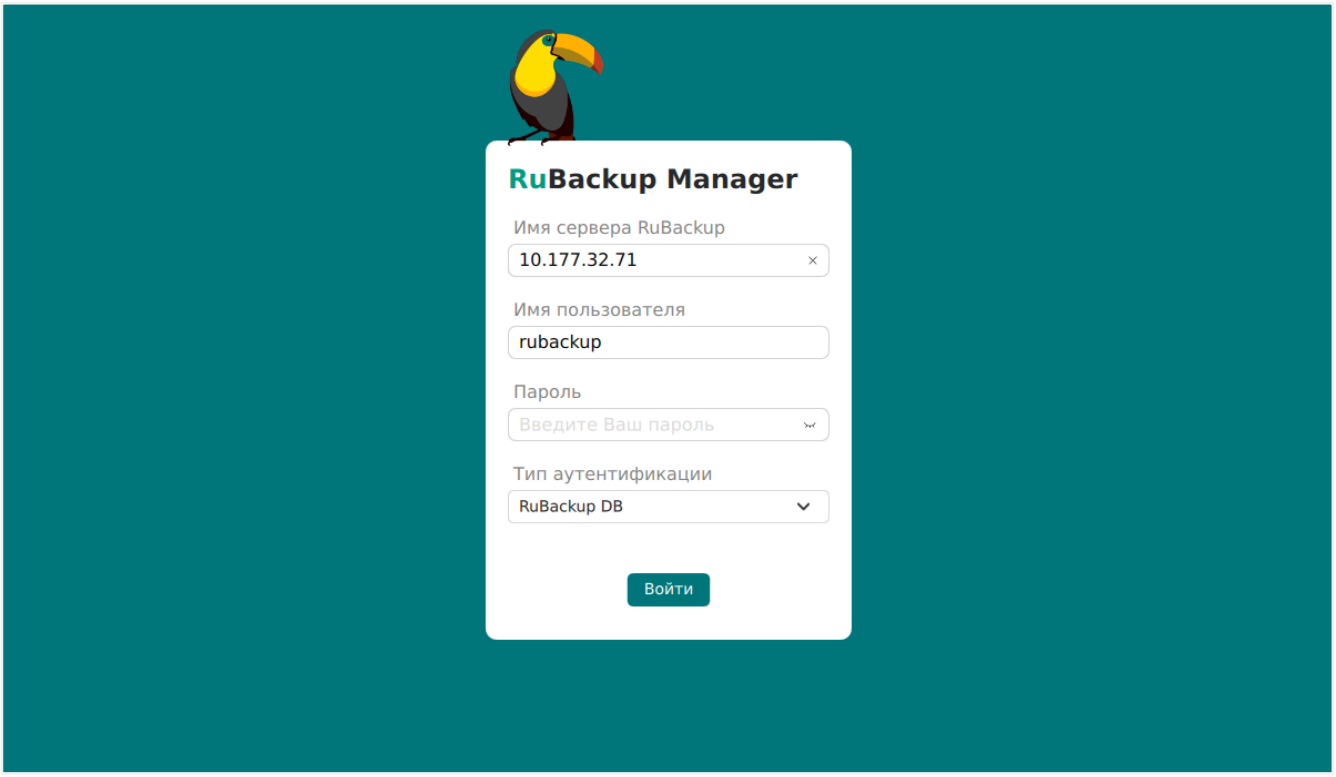

Рисунок 7. Запуск RBM

Для определения статуса клиента необходимо перейти в раздел Администрирование → Клиенты (Рисунок 8):

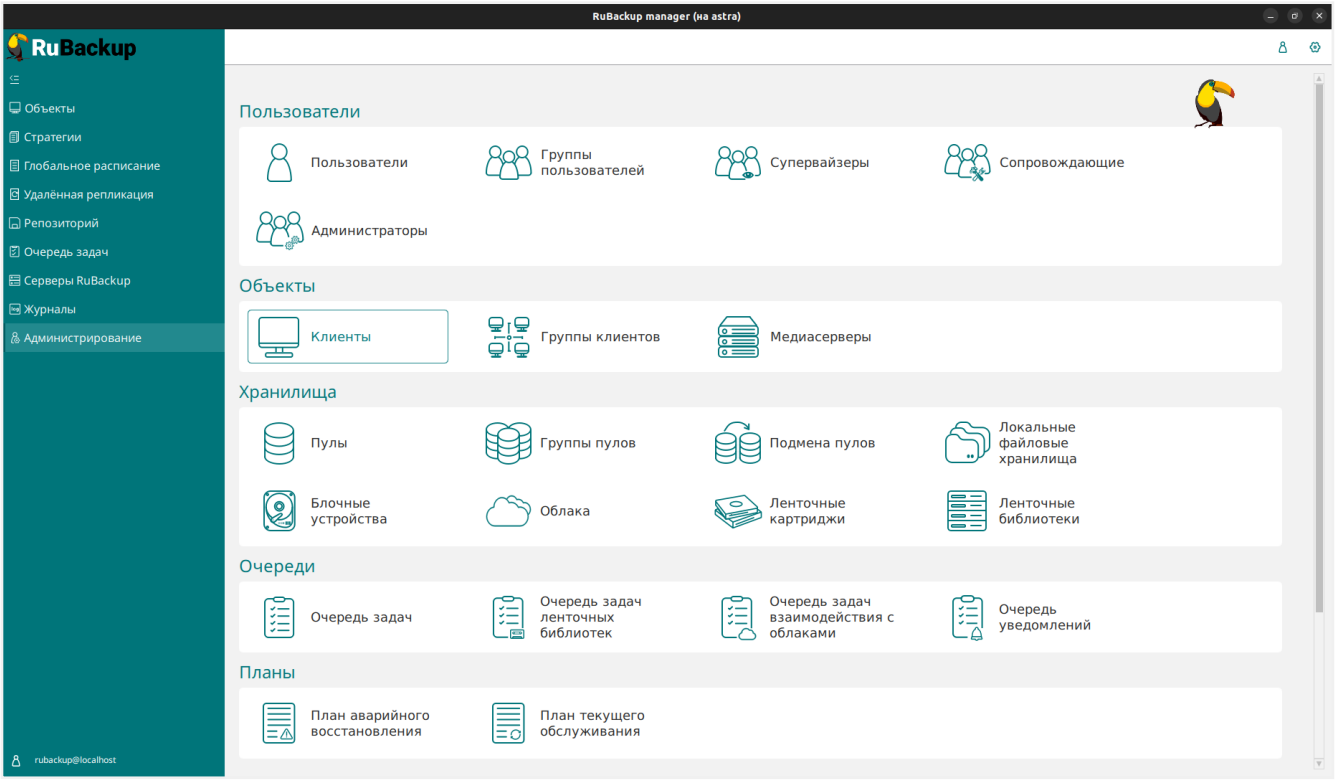

Рисунок 8. Раздел "Администрирование"

При этом откроется окно (Рисунок 9).

Если клиент RuBackup установлен, но не авторизован, в верхней части окна RBM

кнопка «Неавторизованные клиенты» будет активна, а в нижней левой части будет указано количество неавторизованных клиентов.

Все новые клиенты должны быть авторизованы в системе резервного копирования RuBackup.

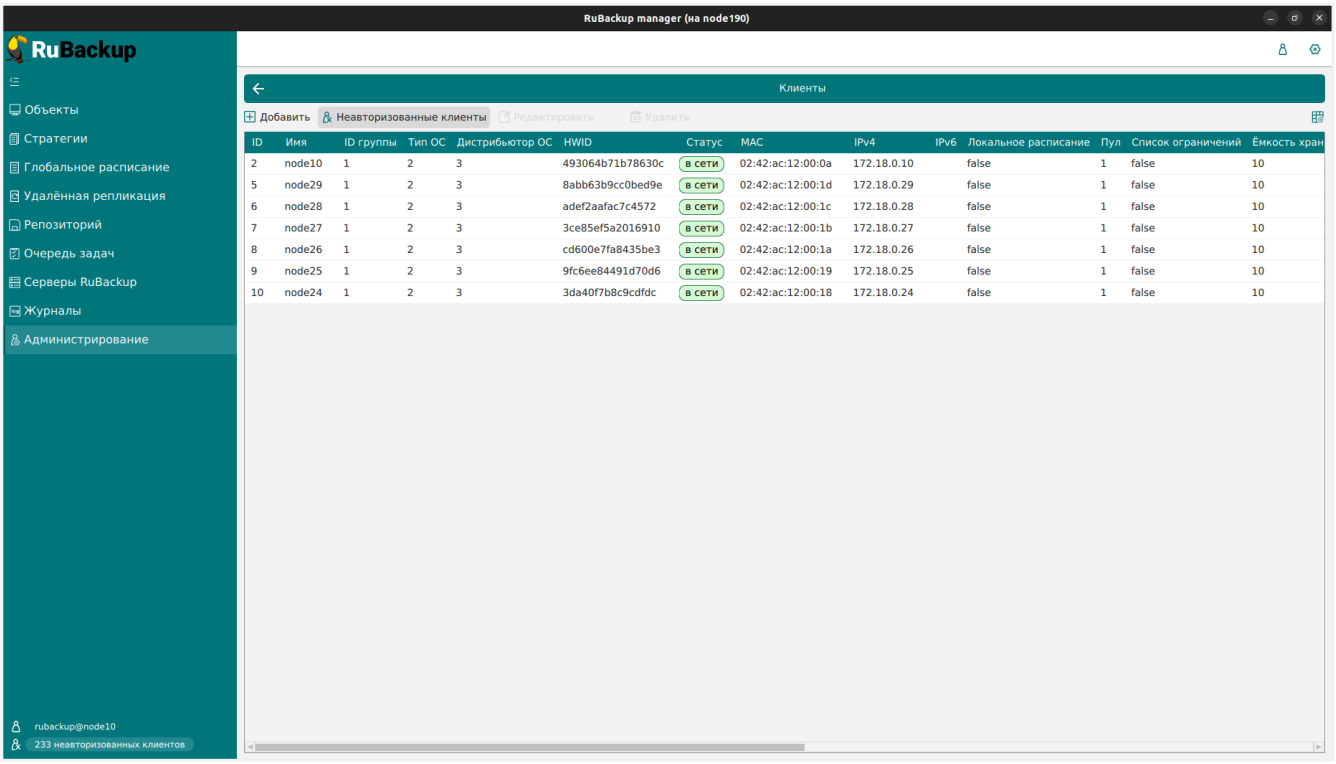

Рисунок 9. Кнопка "Неавторизованные клиенты"

Для авторизации неавторизованного клиента в RBM выполните следующие действия:

1. Нажмите кнопку Неавторизованные клиенты. При этом откроется окно (Рисунок 10):

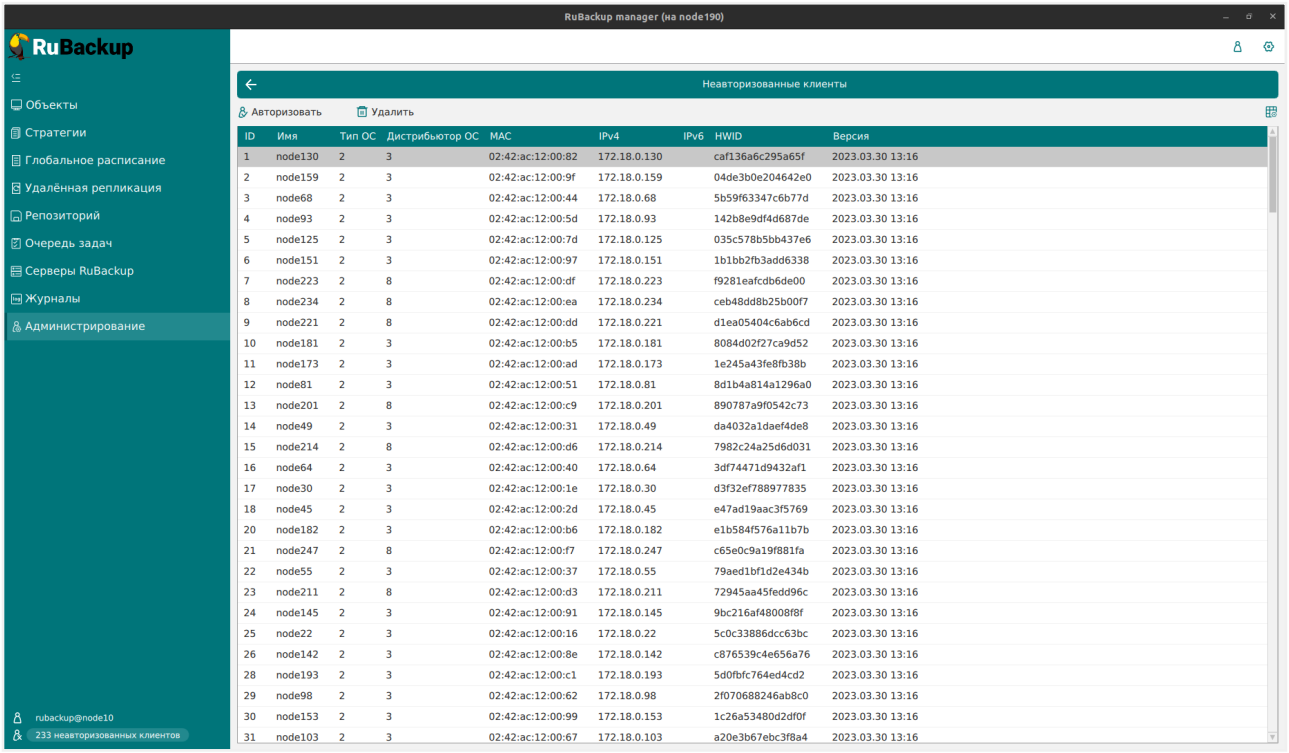

Рисунок 10. Окно "Неавторизованные клиенты"

2. Выберите нужный неавторизованный клиент и нажмите Авторизовать (Рисунок 11):

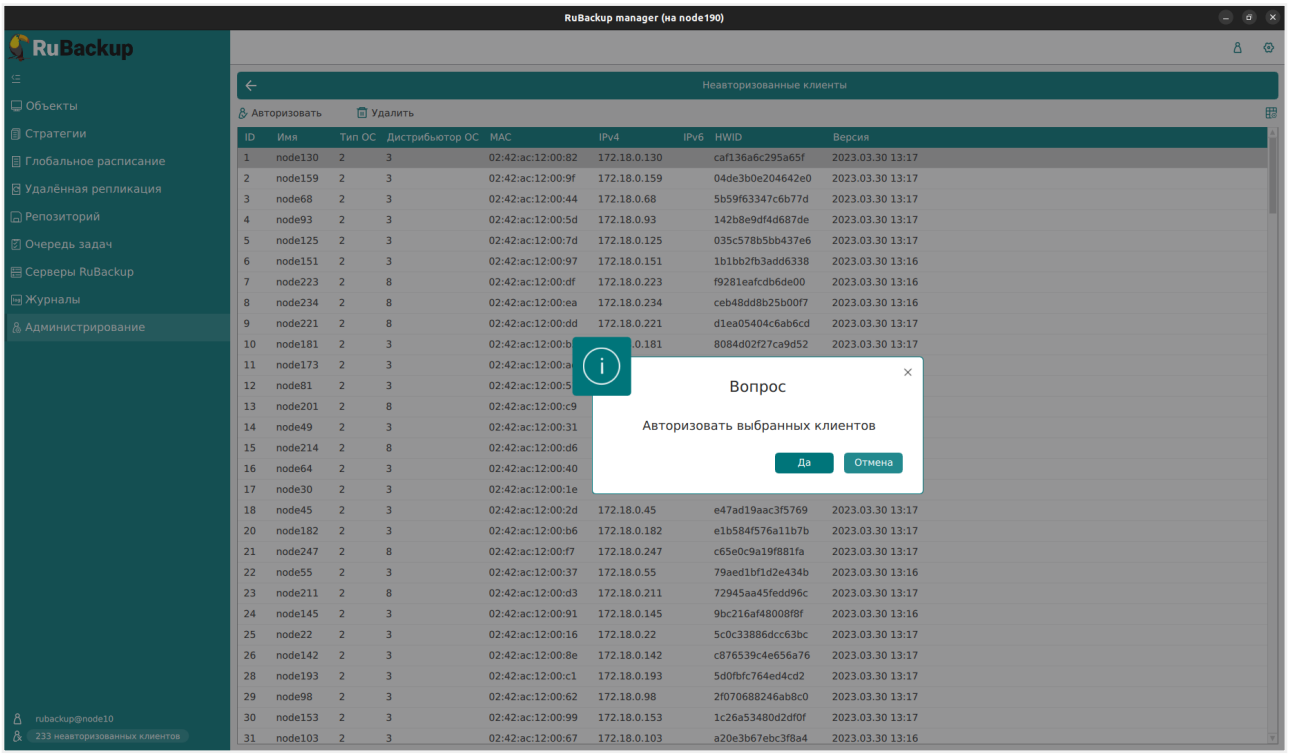

Рисунок 11. Авторизация клиентов

После авторизации новый клиент будет виден в главном окне RBM (Рисунок 12):

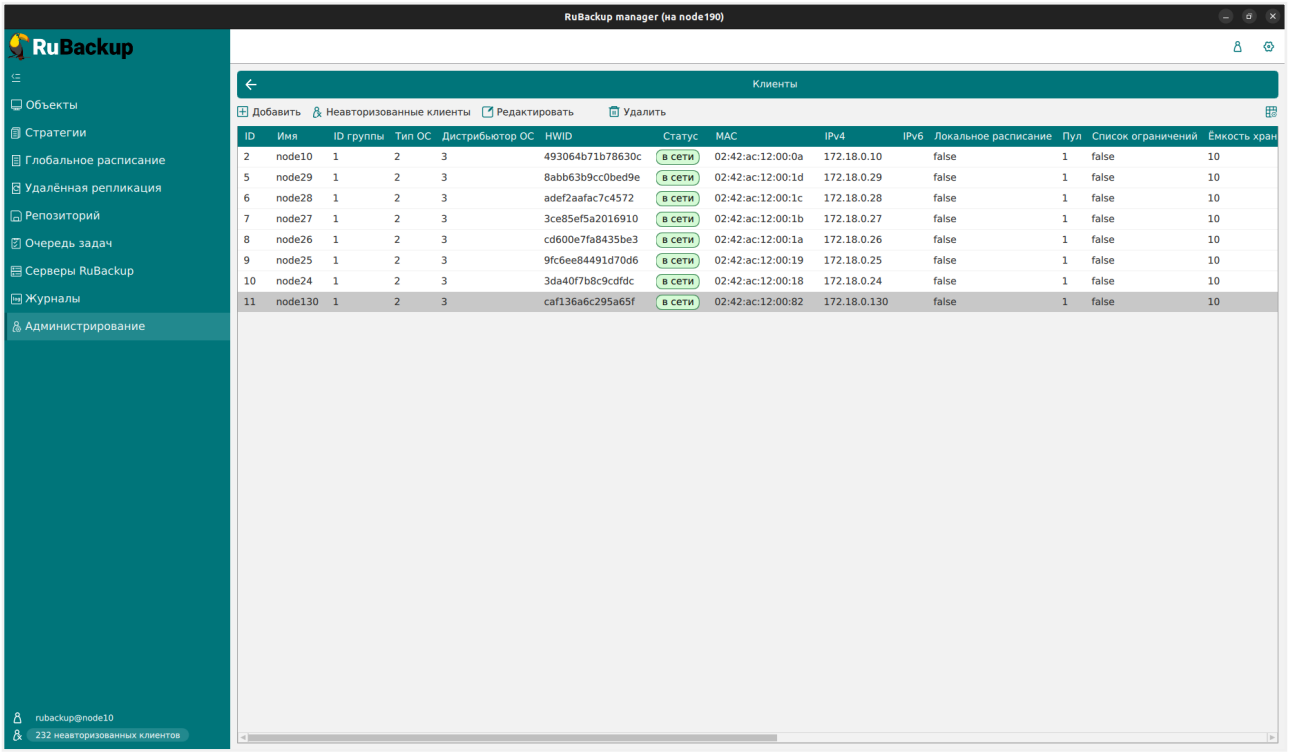

Рисунок 12. Успешная авторизация клиента

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 выделяется курсивом).

9.1. Подготовка к использованию хранилища секретов HashiCorp Vault

Также возможно использование консольной утилиты администратора RuBackup `rb_secret_storage` для управления доступом к аутентификационной информации СУБД PostgreSQL в хранилище секретов HashiCorp Vault (подробнее смотри документ «Утилиты командной строки RuBackup»).

После авторизации в RBM с правами Суперпользователя СРК RuBackup выполните приведённые ниже настройки (добавление хранилища секретов, добавление метода получения секрета, назначение пользователям прав на метод получения секрета) в разделе «Администрирование» — подраздел «Настройки хранилища секретов» (Рисунок 13).

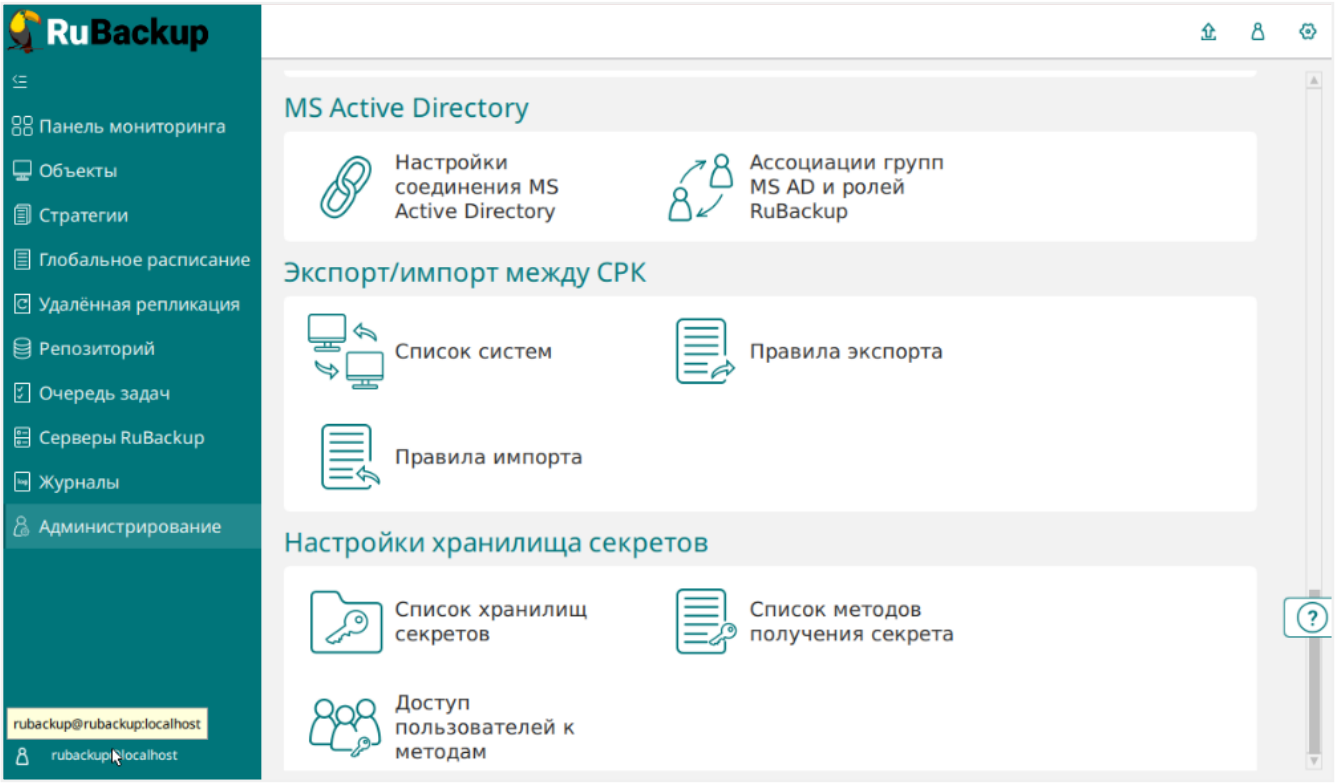

Рисунок 13. Настройка хранилища секретов

9.1.1. Добавление хранилища

Перейдите в раздел «Администрирование» — подраздел «Настройки хранилища секретов» — блок «Список хранилищ секретов» (Рисунок 13).

В открывшемся окне нажмите кнопку 

и заполните форму «Добавление хранилища секретов» (Рисунок 14):

- в поле «Имя хранилища секретов» укажите отображаемое имя хранилища;
- «Тип хранилища секретов» — доступно только хранилище секретов типа HashiCorp Vault;
- при использовании https-запросов в хранилище секретов добавьте сертификат стандарта x.509 в текстовом формате, предварительно полученный на шаге b. При использовании http-запросов в хранилище секретов добавлять сертификат не требуется;
- введите описание хранилища секретов;
- сохраните результат, нажав кнопку 

.

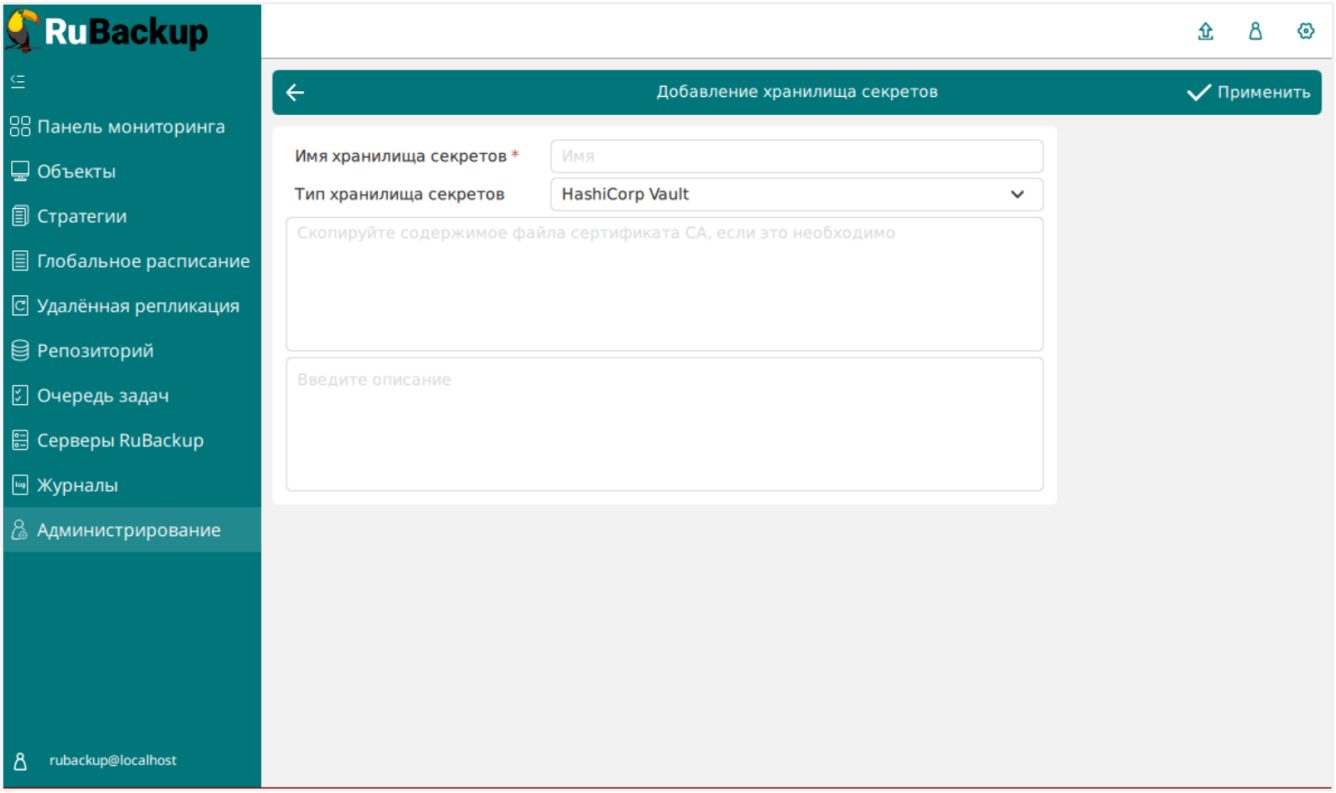

Рисунок 14. Сохранение настроек

Добавленное хранилище будет отображено в Списке хранилищ секретов.

9.1.2. Добавление метода получения секрета

Перейдите в раздел «Администрирование» — подраздел «Настройки хранилища секретов» — блок «Список методов получения секрета» (Рисунок 13).

В открывшемся окне нажмите кнопку 

и заполните форму «Добавление метода получения секрета» (Рисунок 15):

- в поле «Имя метода получения секрета» укажите отображаемое в «Списке методов получения секретов» имя метода;
- в поле «Имя хранилища секретов» выберите из выпадающего списка доступное (созданное на предыдущем шаге) хранилище секретов, к которому будет подключение при выборе создаваемого метода для запроса секрета;
- в поле «Токен» введите токен (идентификатор для получения секрета), предварительно полученный у Администратора хранилища секретов;
- в поле «Метод получения секретов» укажите путь до секрета, предварительно полученный у Администратора хранилища секретов.

Формат пути к секрету представлен в таблице.

Таблица 3. Методы получения аутентификационной информации для подключения к СУБД PostgreSQL в хранилище секретов

Метод получения секретов	Использование https-запросов (указан сертификат при создании хранилища)	Использование http-запросов (сертификат не указан при создании хранилища)
Формат пути к секрету	`hostname:8200/v1/vault/data/postgresql`	`<ip-address or hostname>:8201/v1/vault/data/postgresql`
Описание формата метода	где: * `hostname` — имя указанное в сертификате и используемом для подключения к хранилищу, хоста на котором развёрнуто хранилище секретов HashiCorp Vault; * `8200` — порт (по умолчанию) для прослушивания запросов к хранилищу секретов HashiCorp Vault по безопасному протоколу https; * `/v1/vault/data/postgresql` — путь до папки, содержащей секрет	где: * <ip-address or hostname> — ip адрес или имя хоста, на котором развёрнуто хранилище секретов HashiCorp Vault, секрет которого запрашивается; * `8201` — порт (по умолчанию) для прослушивания запросов к хранилищу секретов HashiCorp Vault, по протоколу http; * `/v1/vault/data/postgresql` — путь до папки, содержащей секрет

- введите описание метода получения секрета;
- сохраните результат, нажав кнопку 

.

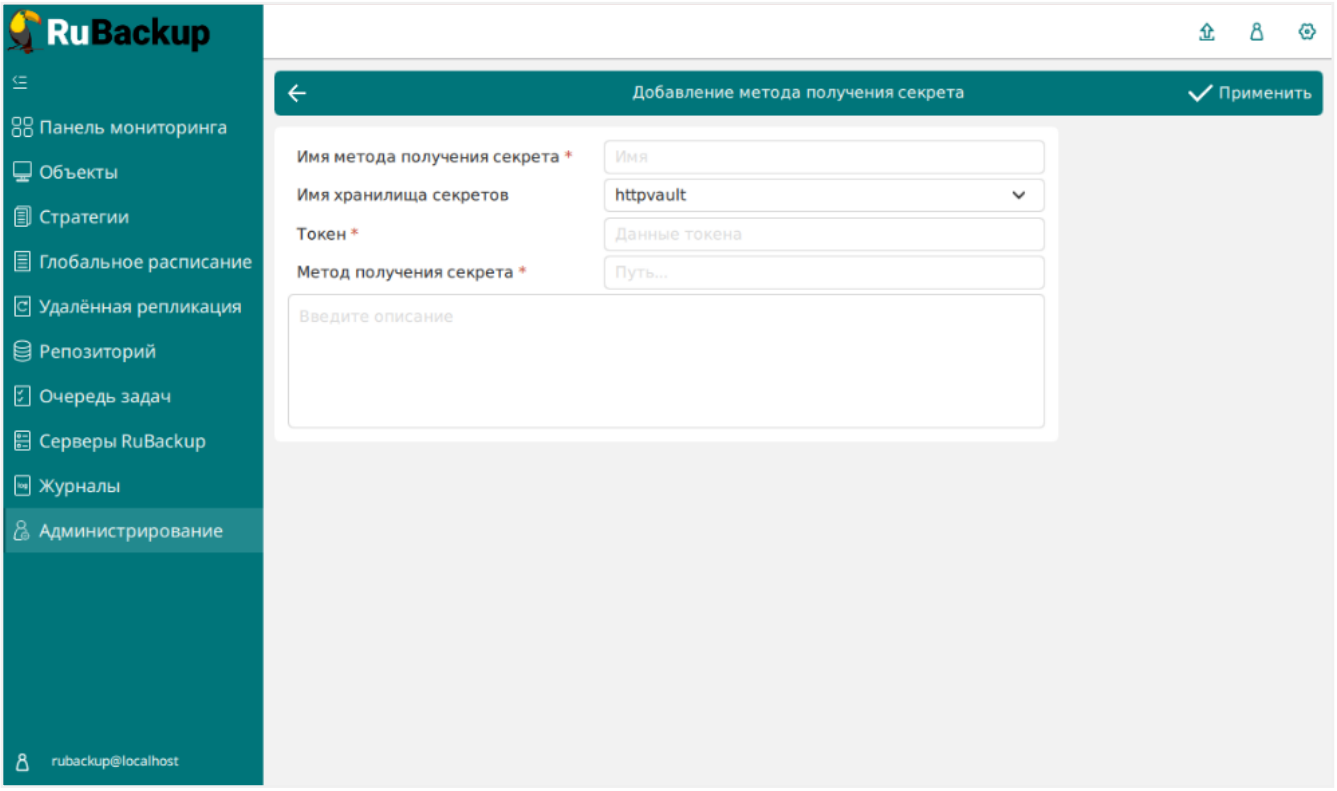

Рисунок 15. Сохранение результата

Добавленный метод будет отображён в Списке методов получения секрета.

9.1.3. Политика доступа к хранилищу секретов

Суперпользователь СРК RuBackup может назначить Супервайзеру или Администратору СРК RuBackup доступ к выбранному секрету посредством ассоциации пользователя с методом получения секрета.

Перейдите в раздел «Администрирование» — подраздел «Настройки хранилища секретов» — блок «Доступ пользователей к методам» (Рисунок 13).

В открывшемся окне нажмите кнопку 

и заполните форму «Добавление метода получения секрета» (Рисунок 16):

- в поле «Имя хранилища секретов» выберите из выпадающего списка добавленное хранилище;
- в поле «Имя метода получения секрета» выберите из выпадающего списка добавленный метод, с которым будет ассоциирован данный пользователь;
- далее показан список пользователей (с назначенной ролью Супервайзер и Администратор), которым следует назначить доступ к методу получения секрета в выбранном хранилище секретов;
- сохраните результат, нажав кнопку 

.

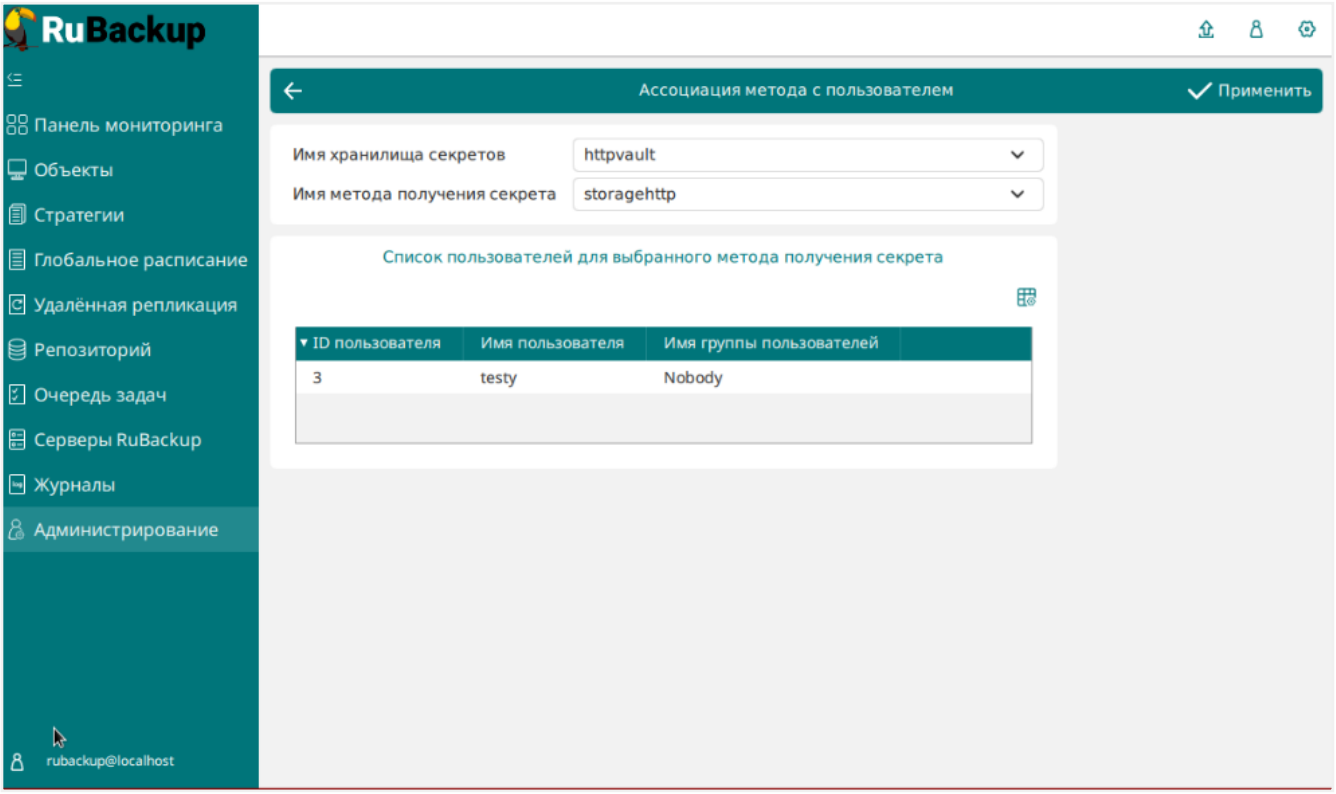

Рисунок 16. Сохранение результата

Добавленный пользователь и ассоциированный с ним метод будет отображён в окне «Доступ пользователей к методам».

Глава 10. Настройка правил резервного копирования СУБД PostgreSQL

Чтобы выполнять регулярное резервное копирование СУБД PostgreSQL, необходимо создать правило в глобальном расписании (в случае групповых операций можно так же использовать стратегии резервного копирования).

Рекомендуемая стратегия резервного копирования баз данных — осуществлять полное резервное копирование раз в неделю и инкрементальное резервное копирование — каждый день.

Для создания правила в глобальном расписании выполните следующие действия:

1. Находясь в разделе «Глобальное расписание», нажмите на кнопку «Добавить», чтобы добавить правило глобального расписания (Рисунок 17).

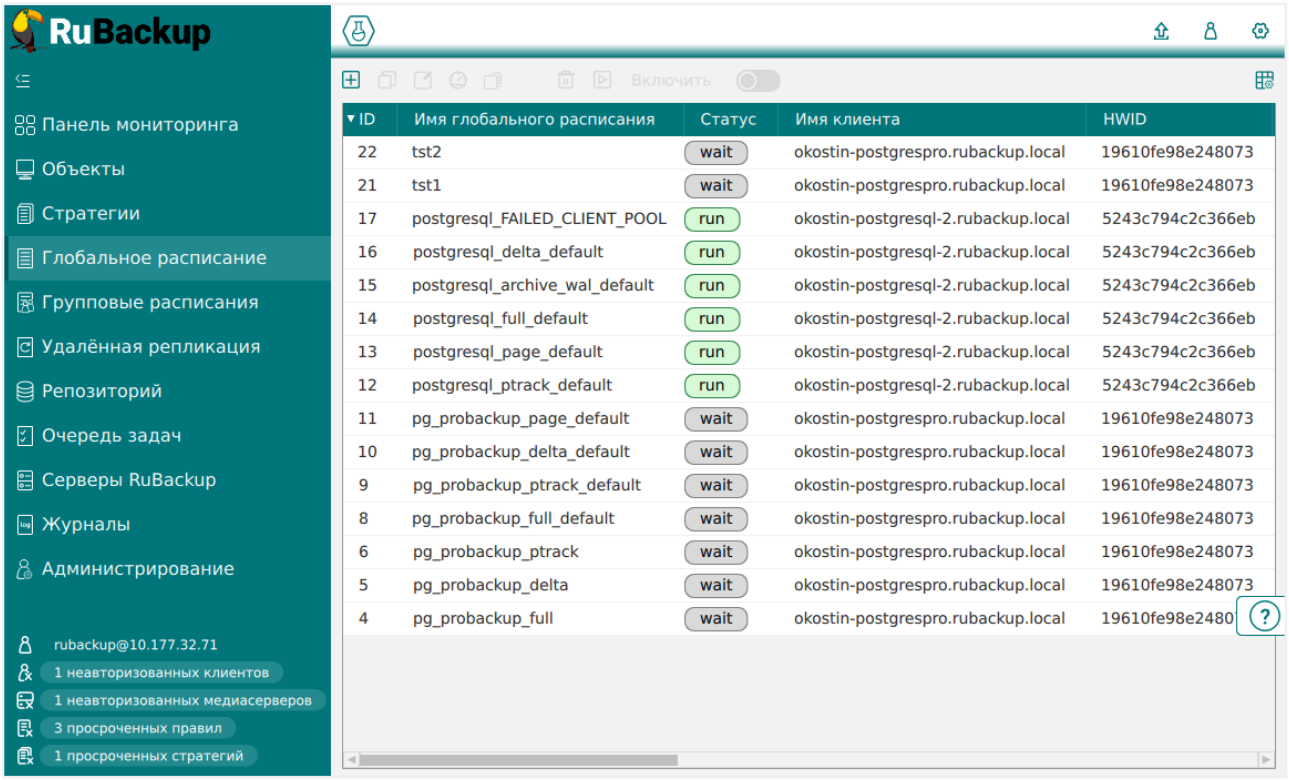

Рисунок 17. Глобальное расписание

2. Выберите клиент и тип ресурса «PostgreSQL Universal» (Рисунок 18).

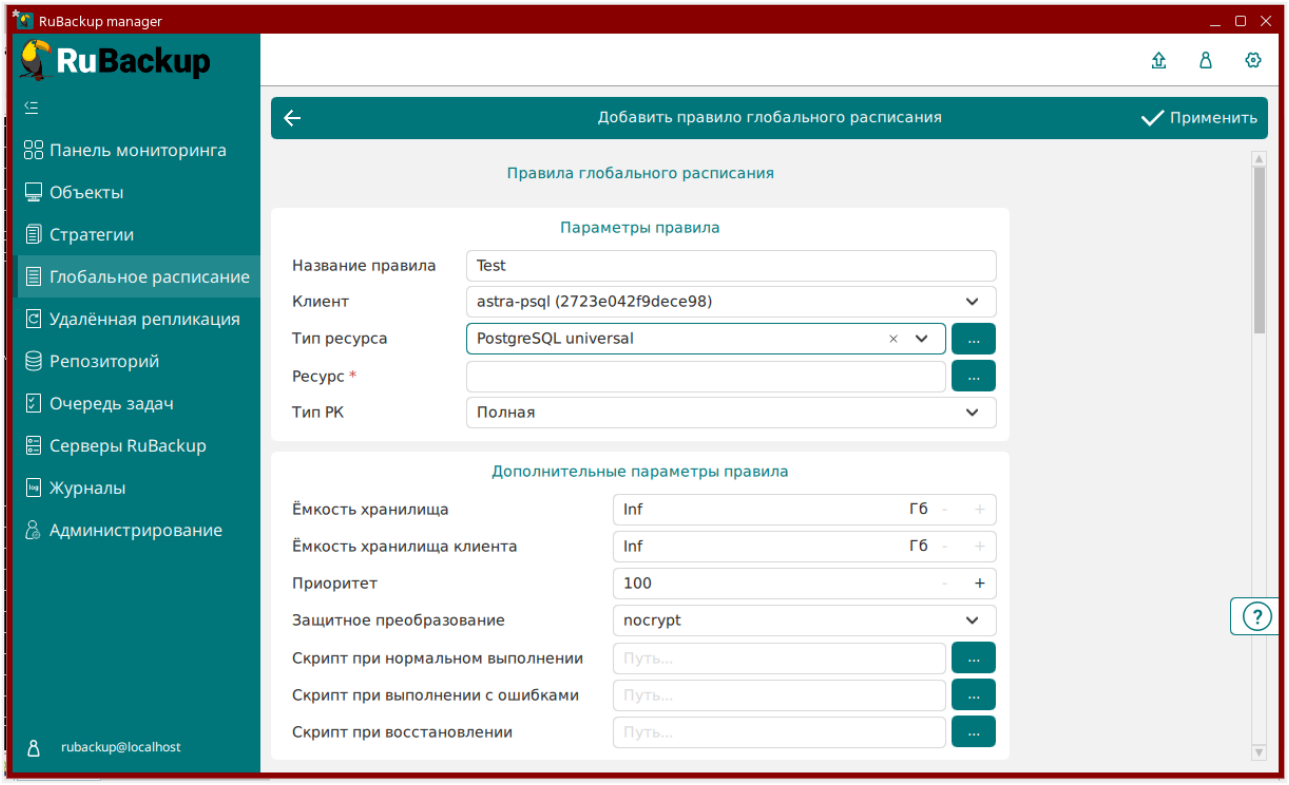

Рисунок 18. Выбор клиента и типа ресурса

3. Нажмите на иконку «...» рядом с надписью «Тип ресурса», чтобы выбрать следующие параметры (Рисунок 19):

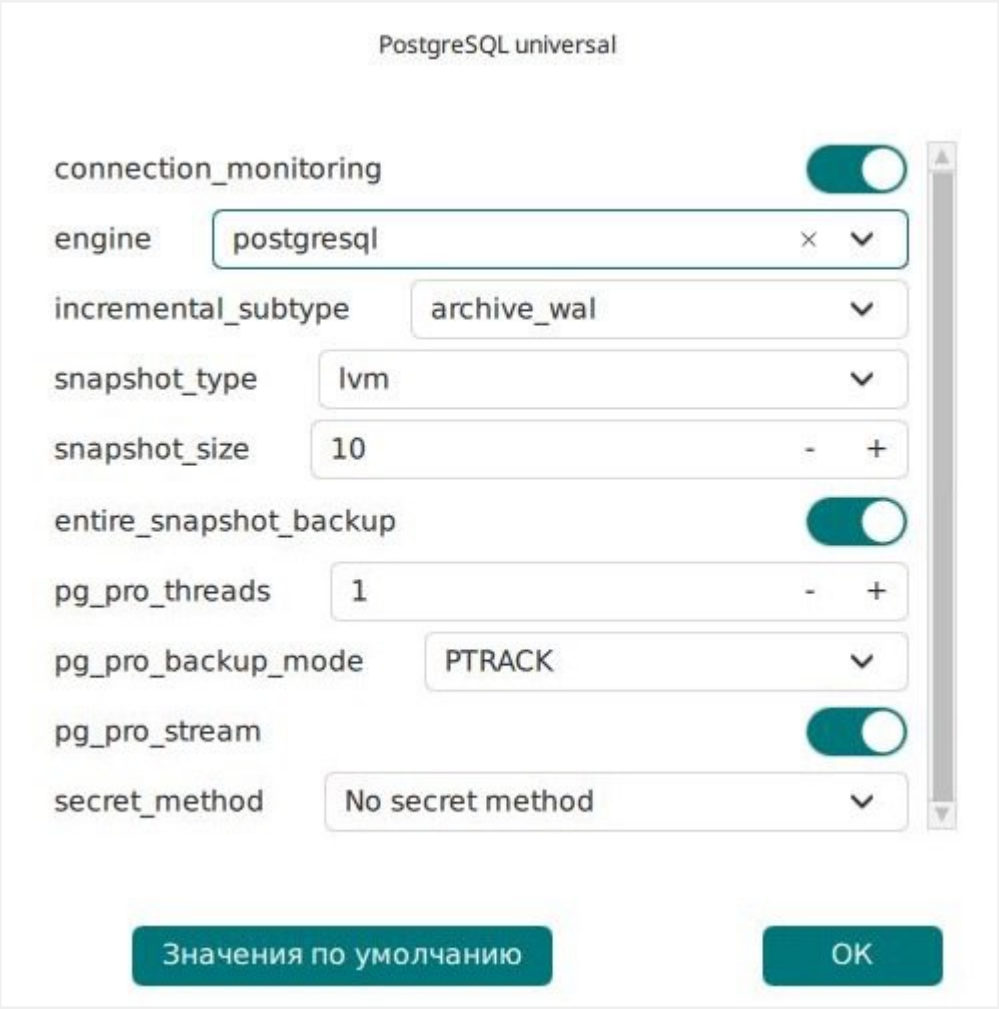

Рисунок 19. Выбор типа ресурса

- connection_monitoring — мониторинг соединения с базой данных (параметр используется только для подмодуля postgresql);
- engine — выбор подмодуля для резервного копирования (postgresql, pg_probackup, superb):
  - postgresql — подмодуль, использующий низкоуровневый API СУБД PostgreSQL для выполнения резервного копирования;
  - pg_probackup — подмодуль, использующий утилиту pg_probackup для выполнения резервного копирования СУБД Postgres Pro;
  - superb — подмодуль, использующий снапшоты для выполнения резервного копирования СУБД PostgreSQL.
- incremental_subtype — выбор подтипа инкрементального резервного копирования (параметр используется только для подмодуля postgresql):
  - archive_wal — режим инкрементального резервного копирования, при котором модуль PostgreSQL Universal выполняет резервное копирование архивных WAL-файлов;
  - page (страничное копирование) — режим инкрементального копирования, при котором Модуль PostgreSQL Universal сканирует все файлы WAL

в архиве с момента создания предыдущей полной или инкрементальной копии. Новая резервная копия будет содержать только страницы, упомянутые в записях WAL;

Для работы подтипа page необходимо настроить параметр конфигурационного файла pg_waldump (подробнее см. в разделе «Конфигурационный файл»).

- delta (разностное копирование) — режим инкрементального копирования, при котором Модуль PostgreSQL Universal считывает все файлы данных в каталоге данных и копирует только те страницы, которые изменились со времени предыдущего копирования;
- ptrack (копирование изменений) — режим инкрементального копирования, при котором Модуль PostgreSQL Universal использует функционал отслеживания изменения страниц на лету. При каждом изменении страницы отношения она помечается в специальной карте PTRACK.

- snapshot_type. Выбор типа снимка (параметр используется только для подмодуля superb):
  - lvm — использование снапшотов LVM;

    СУБД PostgreSQL должна располагаться в файловой системе, которая использует том LVM.

    Модуль поддерживает СУБД PostgreSQL с дополнительными табличными пространствами (tablespaces). Табличные пространства так же должны располагаться в файловых системах, которые используют тома LVM.

  - dattobd — использование модуля ядра dattobd, позволяющего делать снимки блочного устройства;
  - tatlin – использование Tatlin Unified Storage, позволяющего делать мгновенные снимки состояния данных СУБД PostgreSQL, которая располагается в системе хранения данных Tatlin Unified Storage.

    Перед выполнением резервного копирования создаётся мгновенный снимок состояния (в зависимости от способа мгновенного снимка состояния: логического тома LVM, данных на блочном устройстве, системы хранения данных Tatlin Unified Storage), по окончанию резервного копирования мгновенный снимок состояния будет удалён.

    

    При использовании Tatlin Unified Storage необходимо предварительно на хосте, на котором развёрнут модуль резервного копирования и восстановления данных файловых систем, установить утилиты `multipath` и `sg3_utils`.

- snapshot_size — выбор размера снимка в процентах от размера Logical Volume тома, на котором расположены файлы базы данных, для которой выполняется резервное копирование, в случае LVM. Либо размер снимка в процентах от размера устройства, на котором расположены файлы базы данных, для которой выполняется резервное копирование, в случае использования dattobd. Параметр используется только для подмодуля superb.

  В LVM volume groups, в которых расположены тома LVM, должно быть не менее 10% свободного места для возможности создания моментальных снимков LVM.

  В ходе выполнения задания резервного копирования в журнальном файле задания резервного копирования (файл с номером задачи в `/opt/rubackup/log`) можно проконтролировать реальную утилизацию созданного снэпшота:

```
Snapshot '/dev/mapper/vg0-var--lib.snap' was used for:   0.92  %
The snapshot was removed: /dev/mapper/vg0-var--lib.snap
```

  Рисунок 20. Утилизация созданного снэпшота

  В том случае, если это значение при реальном резервном копировании близко к 100%, то необходимо увеличить размер свободного места в LVM группе и увеличить `lvm_snapshot_size`.

- entire_snapshot_backup — при выставлении этого параметра будет выполняться резервное копирование всего снэпшота диска (логического тома lvm), на котором располагается СУБД PostgreSQL. Если параметр не выставлен, то будет выполняться резервное копирование только данных СУБД PostgreSQL. Параметр используется только для подмодуля superb.
- pg_pro_threads — выбор количества параллельных потоков при резервном копировании. Параметр используется только для подмодуля pg_probackup;
- pg_pro_backup_mode — выбор подтипа инкрементального резервного копирования. Параметр используется только для подмодуля pg_probackup;
- pg_pro_stream — выбор режима доставки WAL. При включенном переключателе — режим STREAM. Копии типа STREAM включают все сегменты WAL, необходимые для восстановления согласованного состояния кластера на момент создания копии. При выключенном переключателе действует режим ARCHIVE — в таком режиме целостность копий обеспечивается посредством непрерывного архивирования (подробнее см. на https://postgrespro.ru/docs/enterprise/). Параметр используется только для подмодуля pg_probackup;
- secret_method — выбор метода получения аутентификационных данных (секрета) для подключения к резервируемой базе данных. Для выбора метода необходимо предварительно его добавить в соответствии с описанием в пункте "Добавление метода получения секрета". Выбор предостав-

ляется в соответствии с назначенными правами доступа авторизованного пользователя СРК RuBackup. Возможные значения: «No secret method» — учётные данные пользователя для подключения к резервируемой базе данных будут считаны из конфигурационного файла `/opt/rubackup/etc/rb_module_postgresql.conf`; «Название метода» — метод, который предварительно добавлен и права на который назначены пользователю, учётные данные пользователя для подключения к резервируемой базе данных будут запрошены в хранилище секретов HashiCorp Vault.

4. Выберите ресурс, нажав на иконку «...» рядом с надписью «Ресурс» (Рисунок 21).

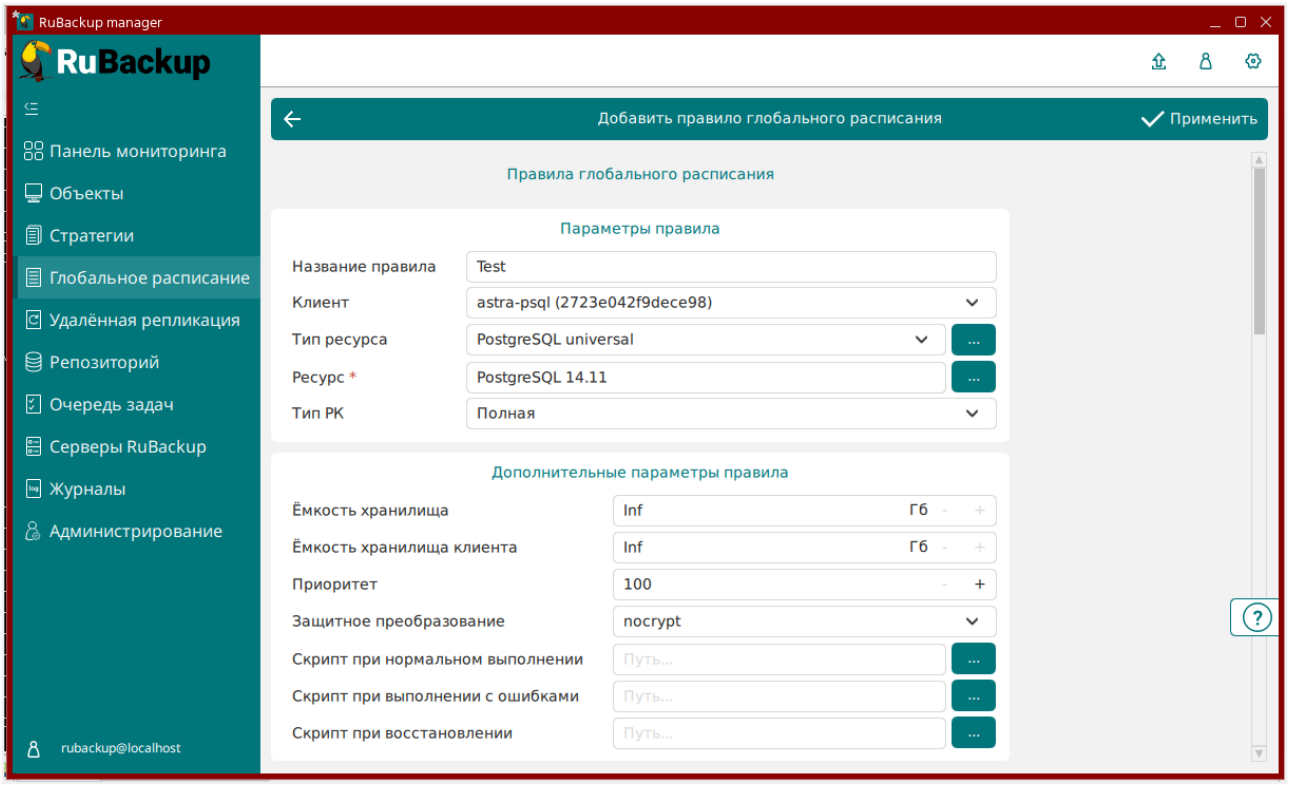

Рисунок 21. Выбор ресурса

В окне выбора всегда будет предложен только один вариант: PostgreSQL с номером версии. Модуль rb_module_postgresql будет пытаться подключиться к прописанной в файле настроек `/opt/rubackup/etc/rb_module_postgresql.conf` базе данных по выбранному методу получения аутентификационной информации (секрета): из файла настроек модуля или хранилища секретов HashiCorp Vault.

Если на хосте, который не является лидером, в поле «Ресурс» отсутствуют необходимые ресурсы, то выберите клиент, который является лидером.

5. Установите настройки правила: название правила, тип резервного копирования (full — для базового резервного копирования, incremental — для резервного

копирования архивных WAL), ёмкость хранилища и ёмкость хранилища клиента (в ГБ), приоритет, алгоритм защитного преобразования, скрипт при нормальном выполнении и при выполнении с ошибками, скрипт при восстановлении резервной копии (Рисунок 22).

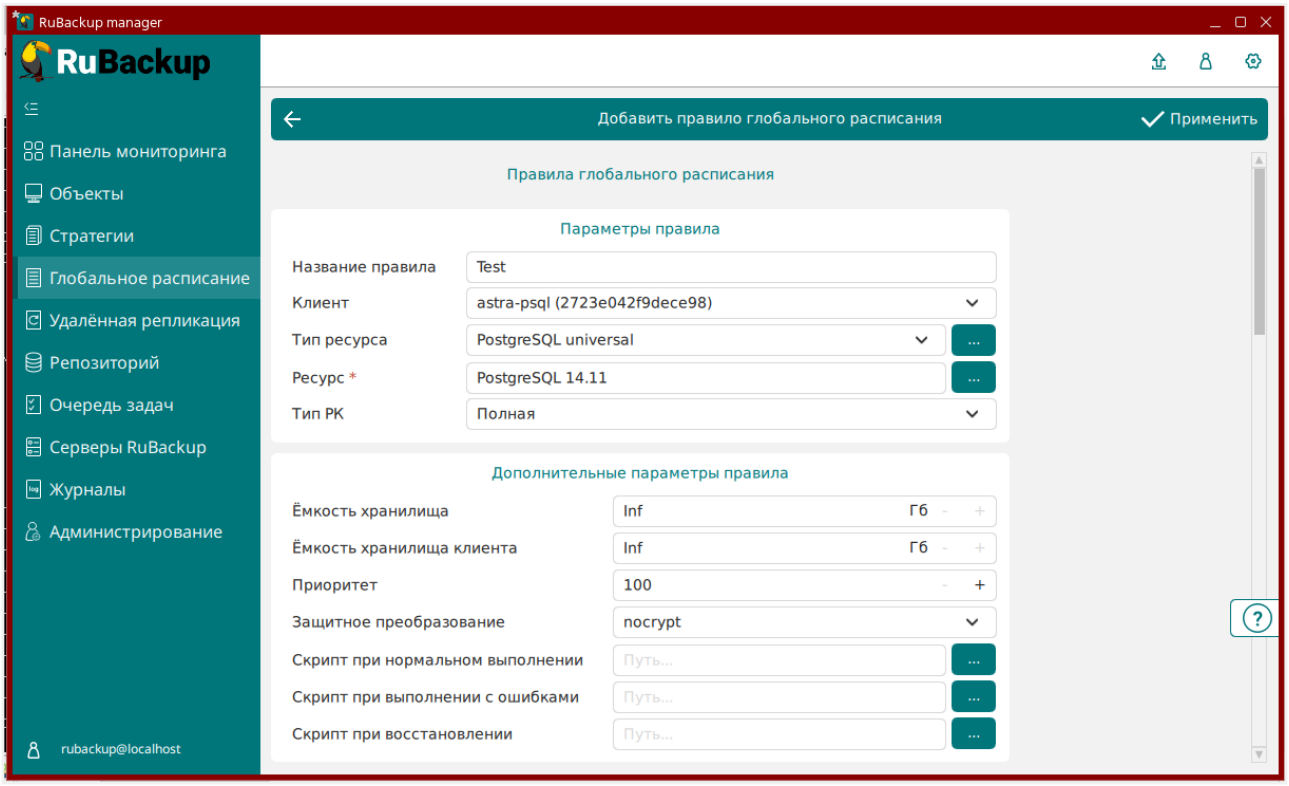

Рисунок 22. Настройки правила

Сделать инкрементальную резервную копию одного ресурса в разные пулы при включенном параметре `auto_remove_wal` невозможно. Чтобы создать инкрементальную резервную копию одного ресурса в разных пулах, установите в конфигурационном файле `/opt/rubackup/etc/rb_module_postgresql.conf` параметр `auto_remove_wal` в значении `no`. По умолчанию этот параметр имеет значение `yes`.

Опционально, при использовании хранилища секретов: при создании инкрементальной резервной копии ресурса следует учитывать метод получения секрета при выполнении предыдущих резервных копий выбранного ресурса. Созданные РК будут находится в одном репозитории при условии использования одного и того же метода получения секрета для подключения к резервируемому ресурсу. В случае выполнения инкрементальной РК ресурса с отличным методом получения секрета, который был использован для получения предыдущей РК, будет выполнено полное резервное копирование ресурса и созданная РК будет помещена в другой репозиторий.

6. После выбора настроек правила глобального расписания нажмите на кнопку «Добавить правило в шаблон», если хотите создать сразу несколько правил —

правило для выбранного типа ресурса и выбранного ресурса появится в списке правил под кнопкой (Рисунок 23). Таким образом создайте столько правил, сколько требуется. Для создания одного правила нажимать на кнопку не нужно.

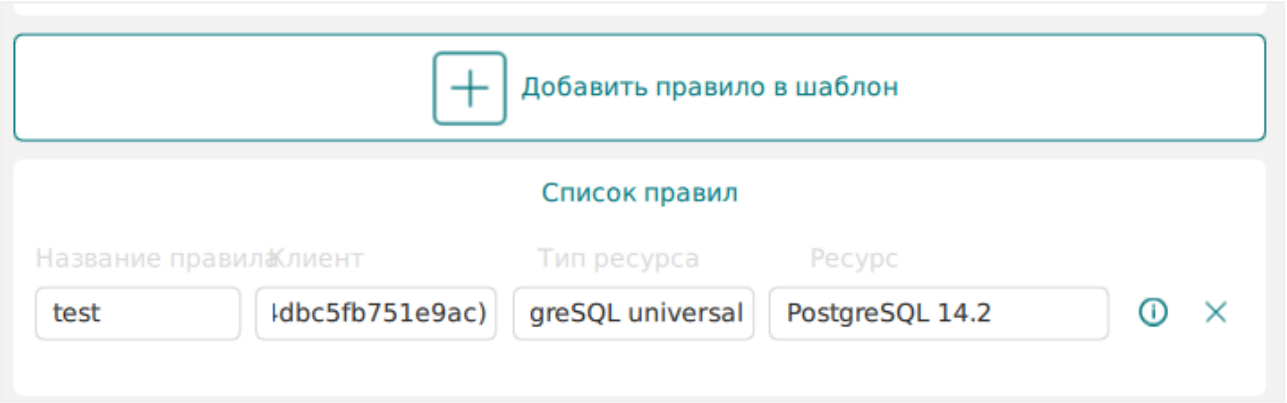

Рисунок 23. Добавление правила в шаблон

7. Заполните раздел «Шаблон глобального расписания» (подробнее см. в документе «Руководство системного администратора RuBackup»).
8. Если необходимо создать правило, которое пока не должно порождать задач резервного копирования, нужно убрать отметку «Включить после создания».
9. Правила глобального расписания имеют срок жизни, определяемый при их создании, а также предоставляют следующие возможности:
   - выполнить скрипт на клиенте перед началом резервного копирования;
   - выполнить скрипт на клиенте после успешного окончания резервного копирования;
   - выполнить скрипт на клиенте после неудачного завершения резервного копирования;
   - выполнить защитное преобразование резервной копии на клиенте;
   - периодически выполнять проверку целостности резервной копии;
   - хранить резервные копии определенный срок, по окончании которого удалять их из хранилища резервных копий и из записей репозитория, либо уведомлять клиента об окончании срока хранения;
   - через определенный срок после создания резервной копии автоматически переместить ее в другой пул хранения резервных копий, например, на картридж ленточной библиотеки;
   - уведомлять пользователей системы резервного копирования о результатах выполнения тех или иных операций, связанных с правилом глобального расписания.
10. Нажмите на кнопку «Применить» в правом-верхнем углу для завершения настройки и создания правила/правил.

Вновь созданное правило будет иметь статус run. При необходимости, администратор может приостановить работу правила или немедленно запустить его (т.е. инициировать немедленное создание задачи при статусе правила wait).

При создании задачи RuBackup она появляется во вкладке «Очередь задач». Отслеживать выполнение правил может как администратор (при помощи RBM или утилит командной строки), так и клиент (при помощи утилиты командной строки rb_tasks).

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

10.1. Срочное резервное копирование при помощи RBM

Для выполнения срочного резервного копирования любого источника данных на клиенте необходимо в RBM во вкладке «Объекты» выбрать нужный клиент СРК и нажать кнопку «Срочное РК» (Рисунок 24):

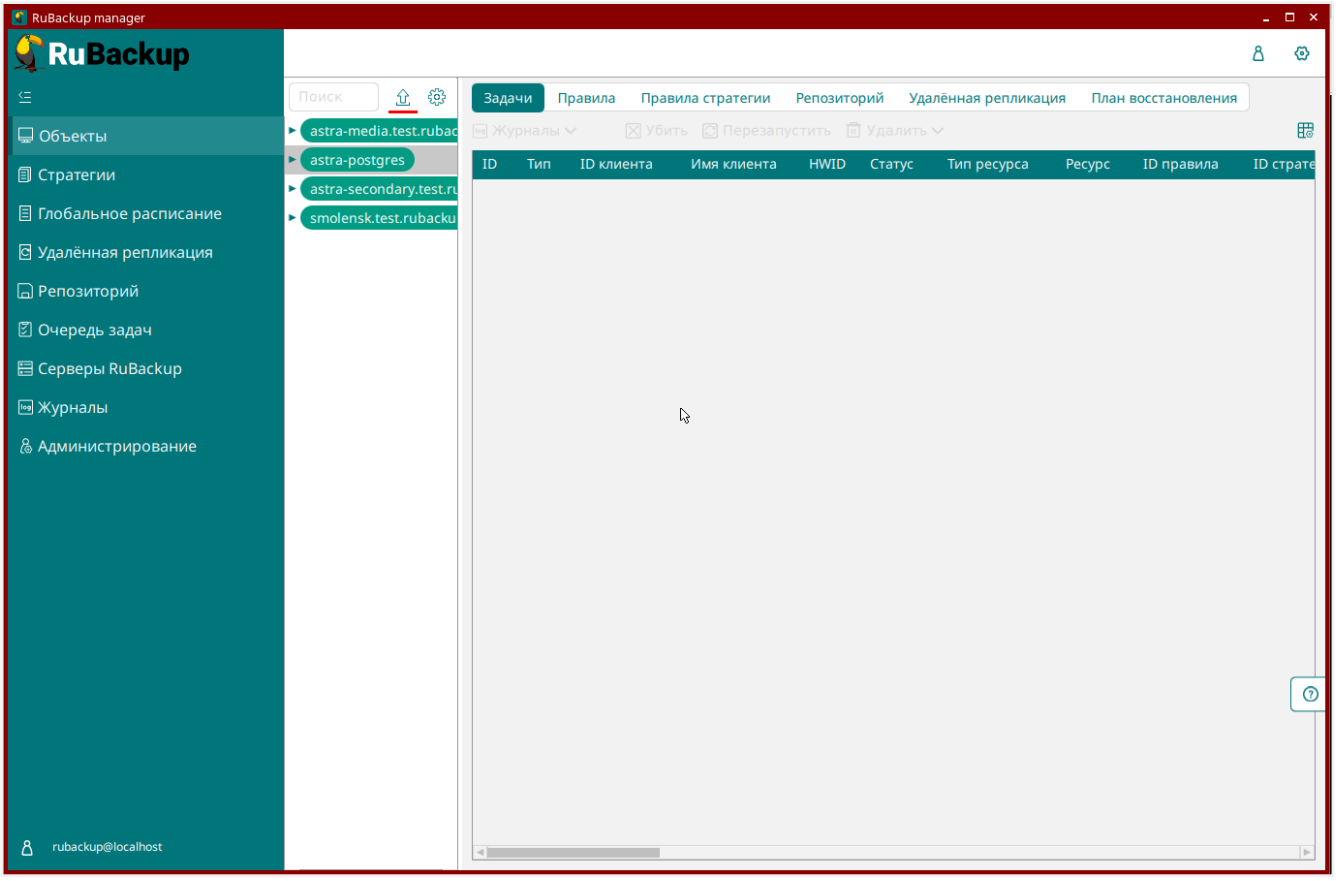

Рисунок 24. Выполнение срочного резервного копирования

Появится окно, в котором можно будет выбрать нужный источник данных для выполнения срочного резервного копирования (Рисунок 25):

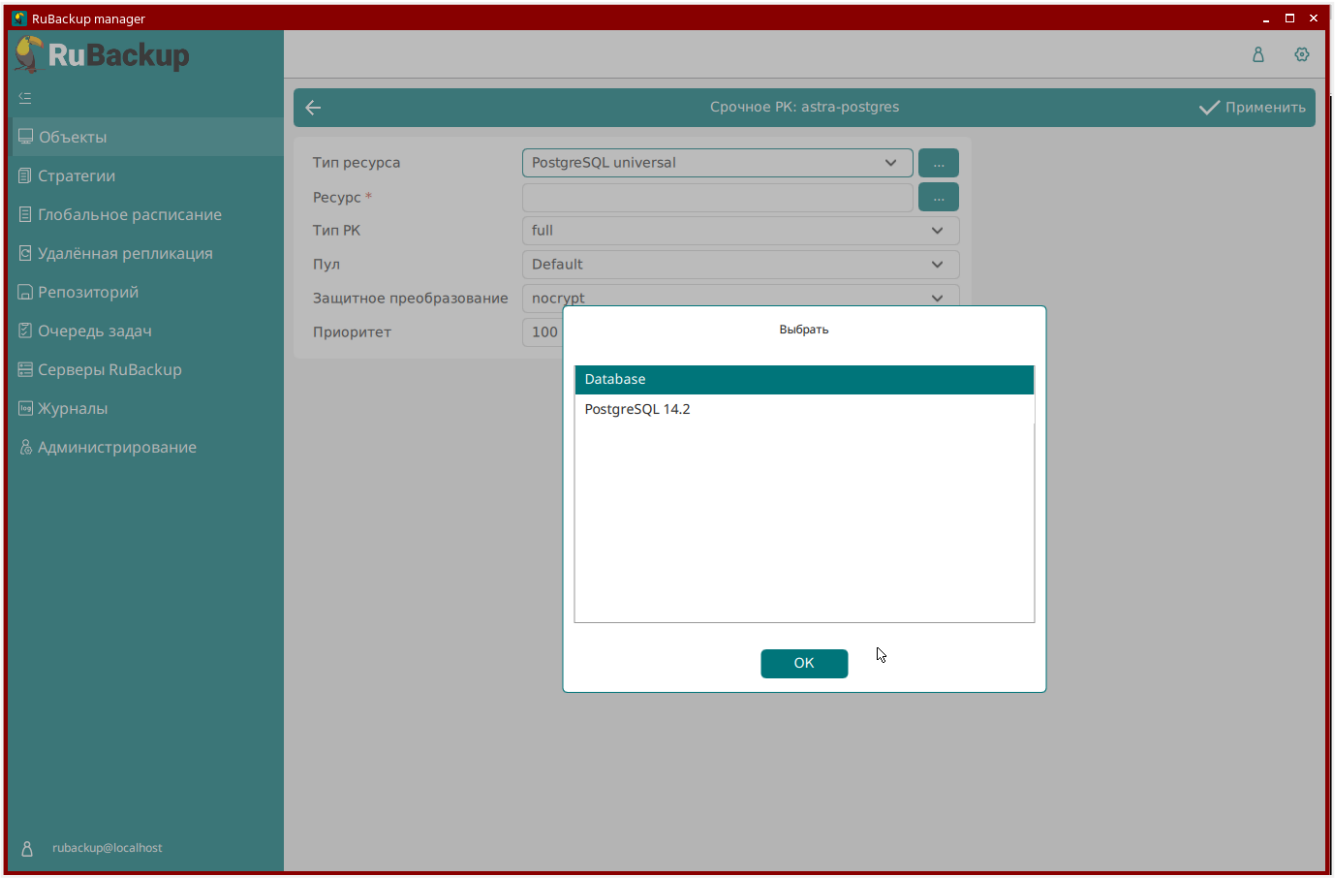

Рисунок 25. Источник данных для выполнения срочного резервного копирования

После выбора настроек нажать кнопку «Применить» в правом верхнем углу (Рисунок 26):

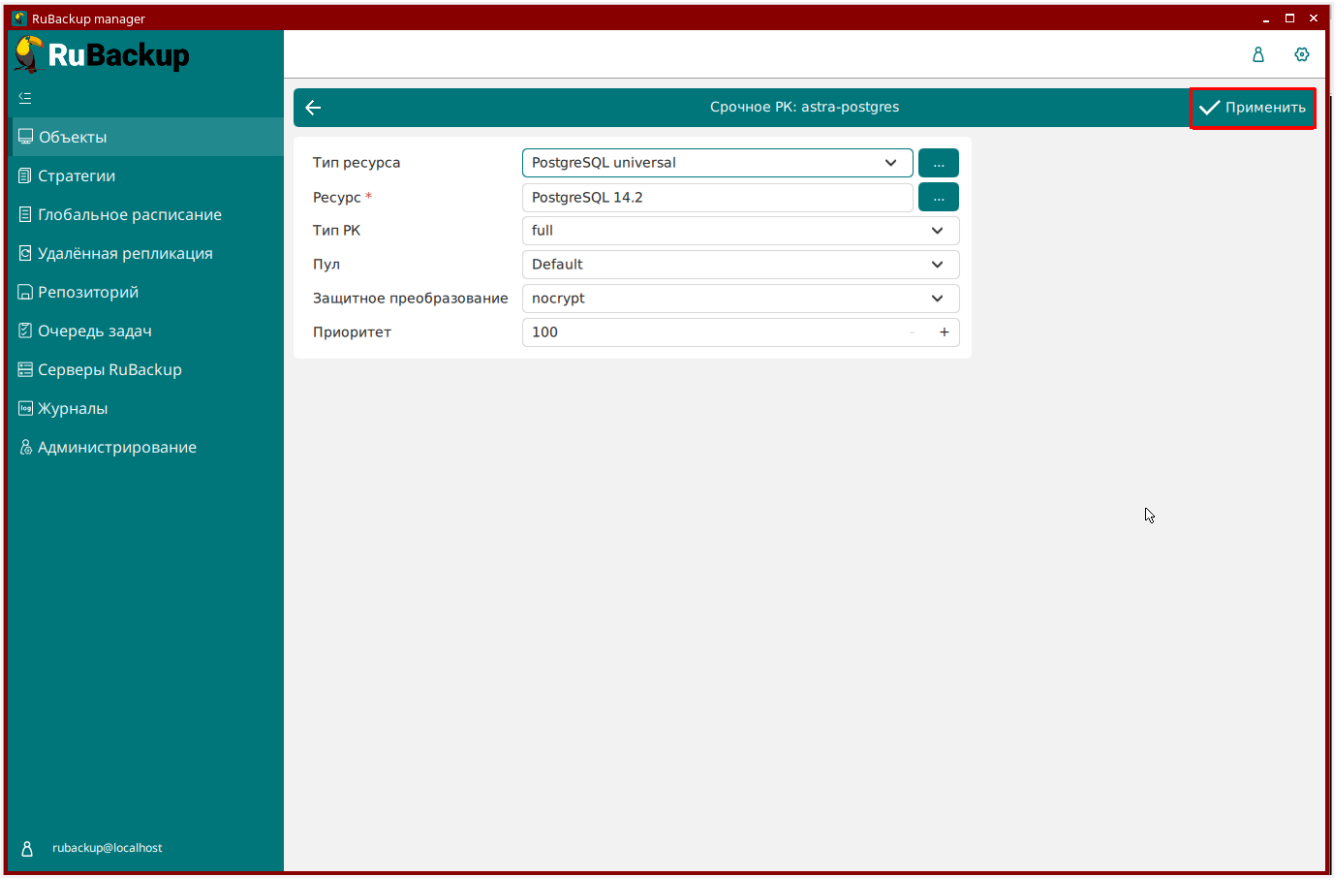

Рисунок 26. Кнопка "Применить"

Проверить ход выполнения резервного копирования можно в окне «Очередь задач» (Рисунок 27):

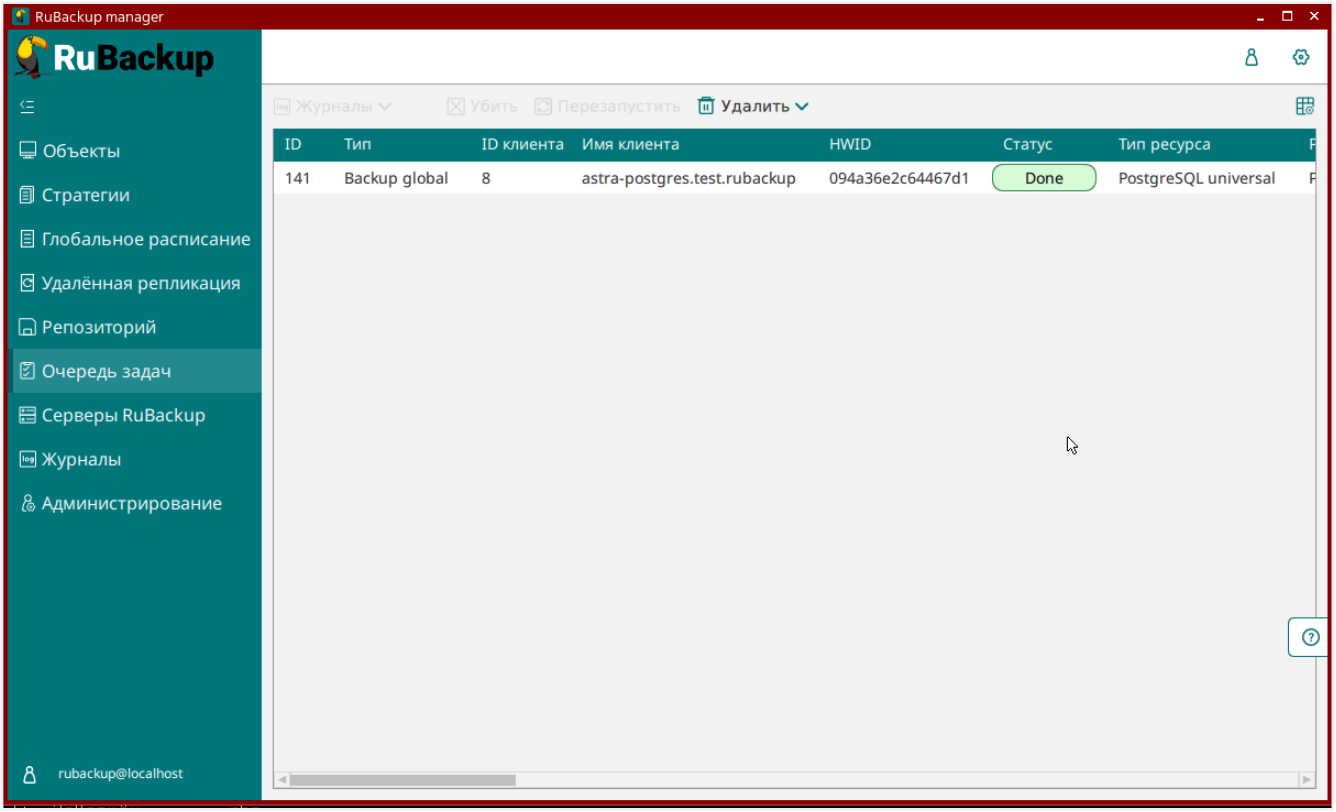

Рисунок 27. Очередь задач

При успешном завершении резервного копирования задача переходит в статус «Done».

Глава 11. Централизованное восстановление резервных копий с помощью RBM

При выполнении операции восстановления с развертыванием существующий кластер баз данных СУБД PostgreSQL будет уничтожен, а на его месте будет восстановлен кластер баз данных из резервной копии. Перед операцией восстановления рекомендуется принудительно остановить работу всех клиентов с СУБД и выполнить полное резервное копирование!

Рекомендуется отключить в конфигурационном файле (подробнее об включении и отключении централизованного восстановления см. в документе «Руководство системного администратора RuBackup») возможность централизованного восстановления СУБД на клиенте и выполнять восстановление из резервной копии только со стороны клиента под контролем администратора СУБД.

Централизованное восстановление и восстановление с развертыванием рекомендуется предварительно выполнять на резервном хосте (виртуальной машине) для проверки корректности восстановления СУБД.

Система резервного копирования RuBackup предусматривает возможность восстановления резервных копий как со стороны клиента системы, так и со стороны администратора СРК.

В тех случаях, когда централизованное восстановление на клиенте включено, его можно инициировать, вызвав правой кнопкой мыши контекстное меню «Восстановить» во вкладке «Репозиторий» (Рисунок 28):

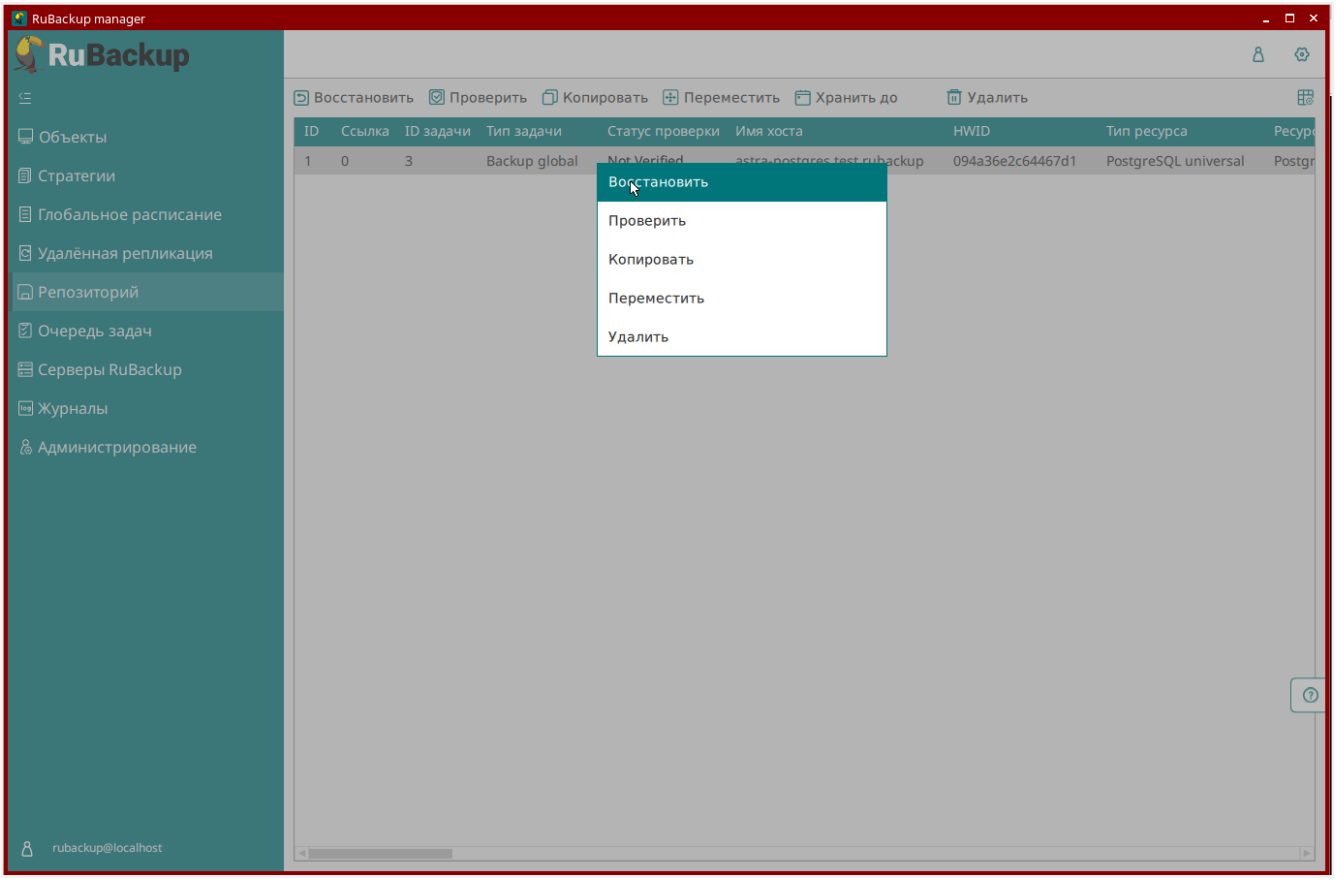

Рисунок 28. Восстановление резервной копии

В окне централизованного восстановления можно увидеть основные параметры резервной копии (Рисунок 29):

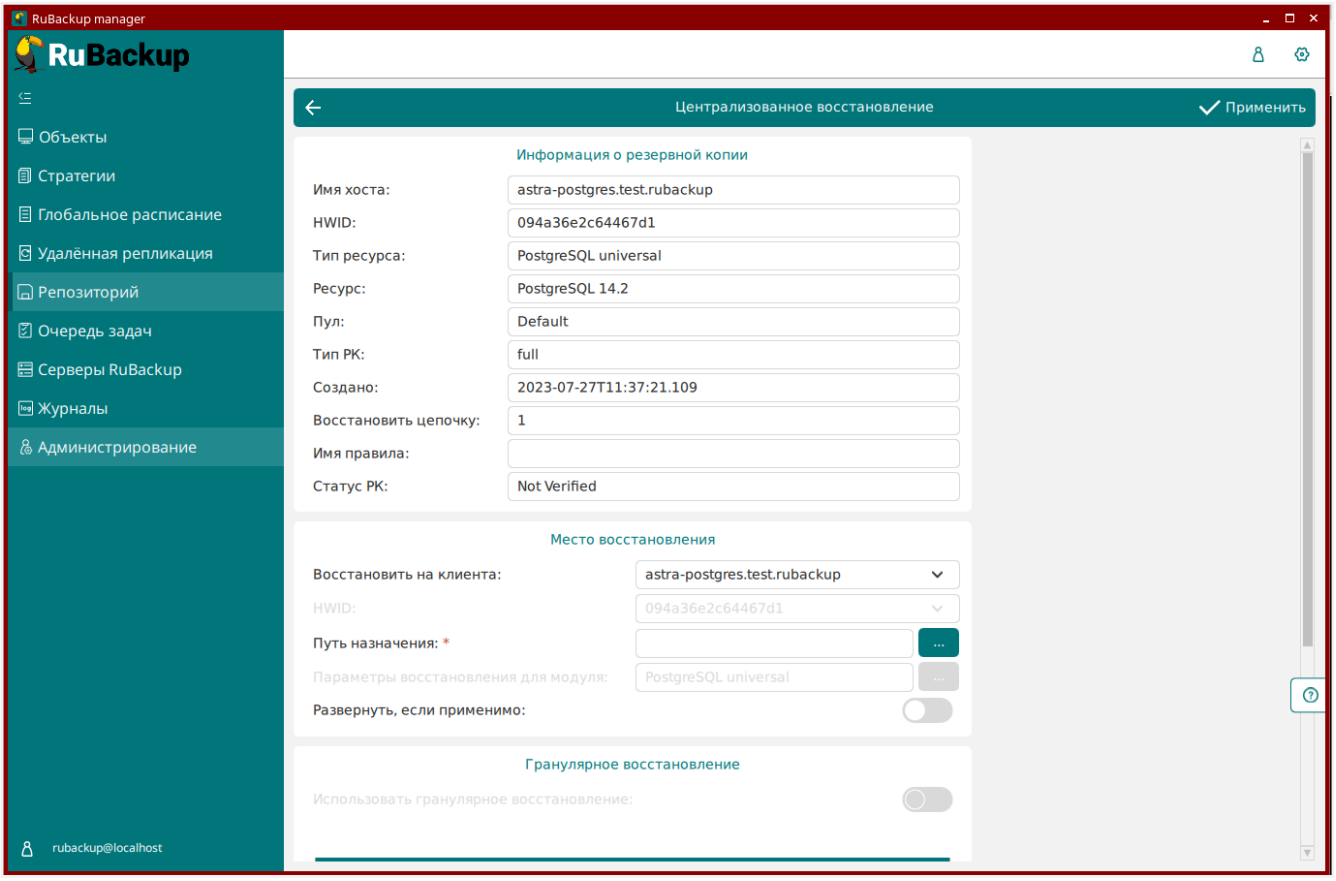

Рисунок 29. Основные параметры резервной копии

Также в разделе «Репозиторий» можно увидеть дополнительные параметры резервных копий, сделанных для модуля PostgreSQL universal. Для этого нажмите кнопку «Дополнительно» и выберите нужный модуль (Рисунок 30):

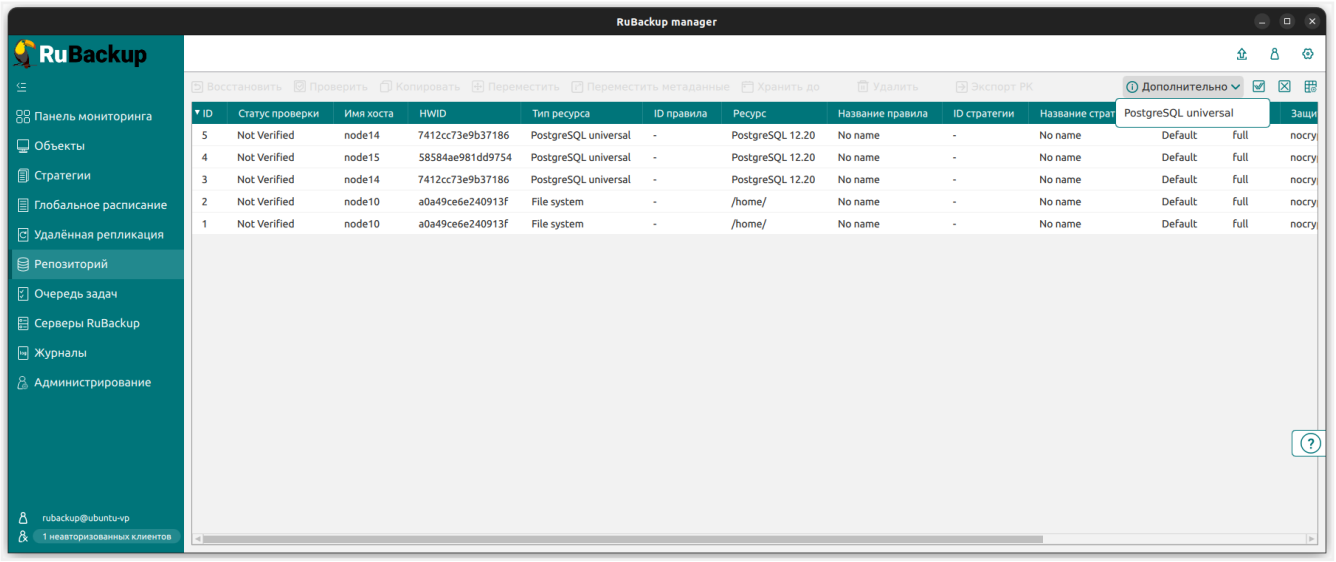

Рисунок 30. Раздел «Репозиторий»

Откроется таблица с дополнительными параметрами резервных копий (Рисунок 31):

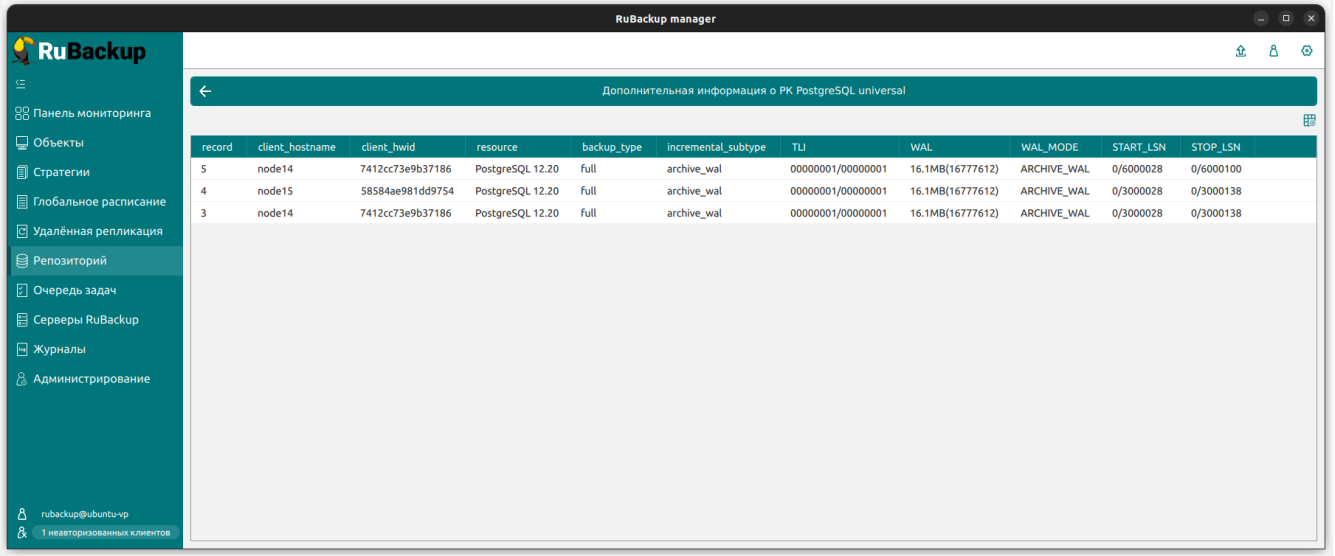

Рисунок 31. Таблица с дополнительными параметрами резервных копий

Значение параметров:

- record — id резервной копии.
- client_hostname — имя (хост) клиента, с которого сделана резервная копия.
- client_hwid — уникальный id клиента.
- resource — ресурс, для которого сделана резервная копия.
- backup_type — тип резервной копии (full — полная, incremental — инкрементальная).
- incremental_subtype — подтип инкрементального резервного копирования. Колонка может принимать значения archive_wal, page, delta или ptrack в зависимости от режима, в котором сделана резервная копия (подробнее см. Глава 10).
- TLI — timeline id начального и конечного WAL-файлов в резервной копии, разделенные символом `/`.
- WAL — общий объём WAL-файлов в резервной копии, выражается в байтах, килобайтах, мегабайтах и т.д.
- WAL_MODE — принимает значение ARCHIVE_WAL (режим непрерывной архивации) или STREAM (стрим режим).
- START_LSN — начальный LSN при вызове pg_start_backup().
- STOP_LSN — конечный LSN при вызове pg_stop_backup().

11.1. Режим восстановления с развертыванием

Существует два режима восстановления резервной копии — с развертыванием и без. Чтобы восстановить резервную копию с развертыванием, включите опцию «Развернуть, если применимо» (Рисунок 32). В том случае если опция включена,

восстановление базы данных из резервной копии будет выполнено автоматически — дополнительные действия со стороны пользователя не требуются.

11.2. Режим восстановления без развертывания

Если выбран режим без развертывания, при восстановлении необходимо указать каталог для восстановления резервной копии (Рисунок 32).

После завершения задачи по восстановлению необходимо:

1. Перенести файлы из каталога для восстановления в целевые каталоги, т.е. из каталога для восстановления `/restore_dir/number.rest/var/lib/postgresql/11/main` (где `number` — это номер резервной копии) в `/var/lib/postgresql/11/main` с заменой файлов, а также из каталога для восстановления `/restore_dir/number.rest/opt/rubackup/mnt/postgresql_archives` перенести wal-файлы (где number — это номер резервной копии) для восстановления в `/opt/rubackup/mnt/postgresql_archives/`;
2. В RBM в разделе Репозиторий выберите восстанавливаемую РК и нажмите кнопку «Восстановить» .
3. В открывшемся окне в блоке «Место восстановления» (Рисунок 32) нажмите на иконку ... рядом с полем «Параметры восстановления для модуля»

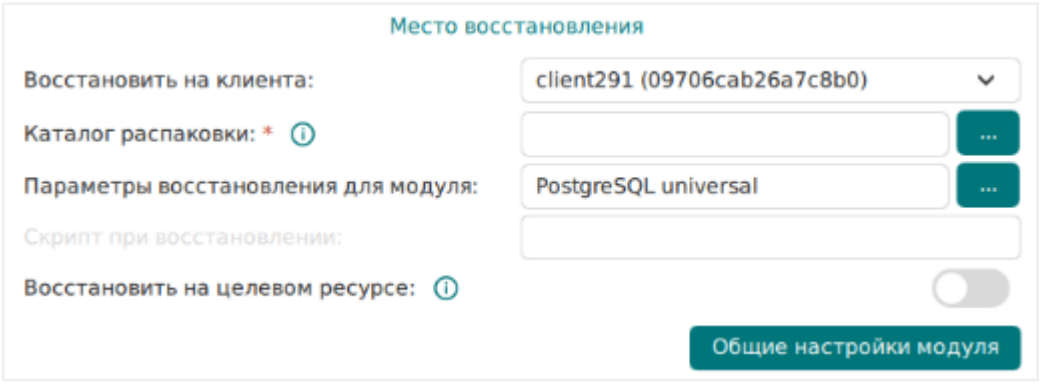

Рисунок 32. Блок «Место восстановления»

4. Опционально, при использовании хранилища секретов: в параметрах восстановления деактивируйте переключатель «Использовать настройки по умолчанию» [1] и выберите метод получения аутентификационной информации для подключения к восстанавливаемой базе данных (Рисунок 33). При восстановлении данных СУБД PostgreSQL можно использовать любой доступный секрет.

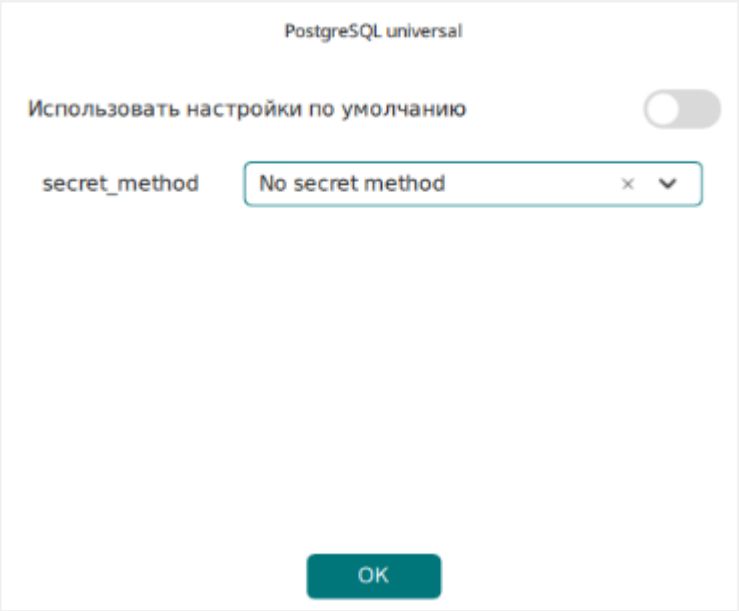

Рисунок 33. Выбор метода получения аутентификационной информации

5. Нажмите кнопку Применить для применения выбранных настроек и восстановления данных.
6. Проверить ход выполнения восстановления резервной копии можно в окне «Очередь задач» (Рисунок 34):

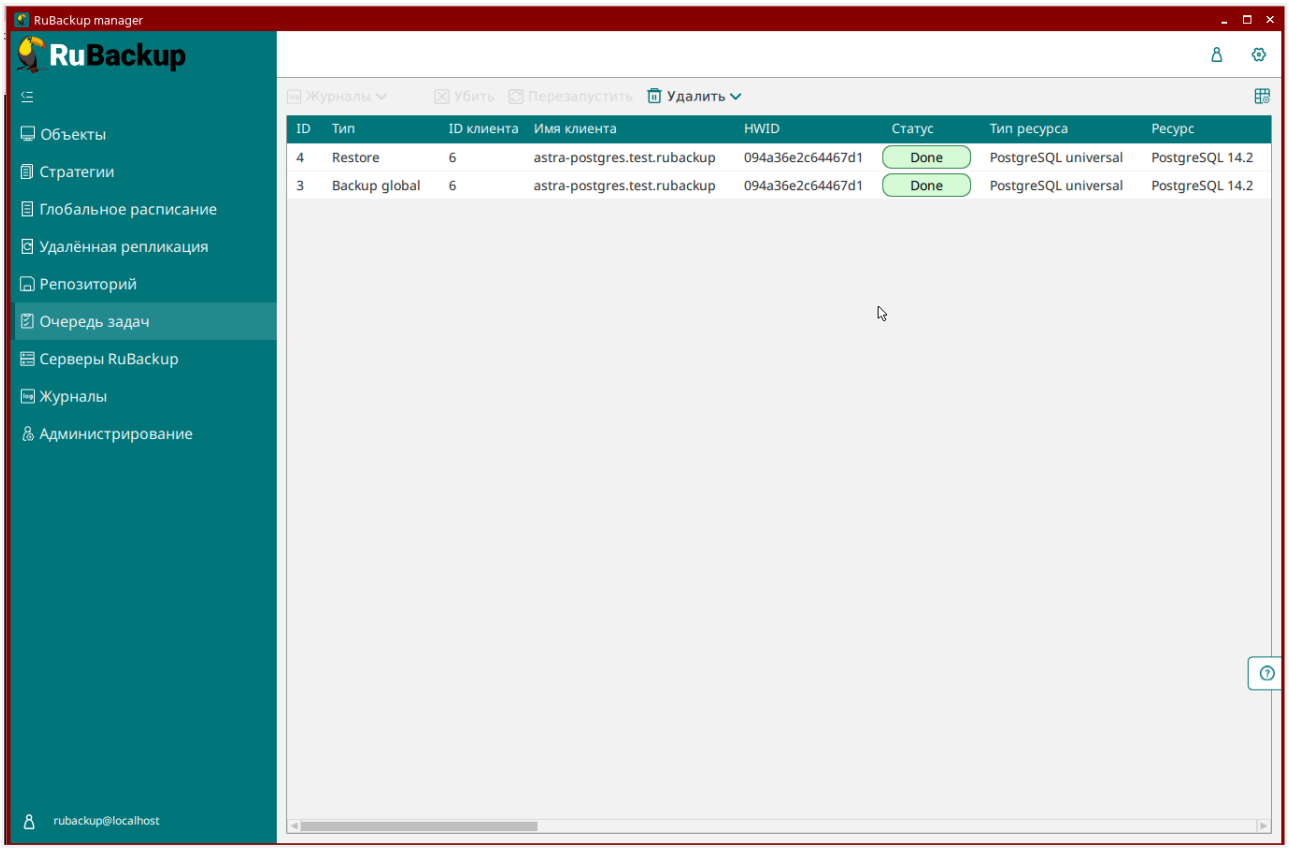

Рисунок 34. Окно «Очередь задач»

При успешном завершении восстановления задача переходит в статус «Done».

Так же можно проконтролировать ход восстановления резервной копии в журнальном файле:

```
root@ubuntu-server:~# cat /opt/rubackup/log/task_2.log
Wed Feb 15 12:30:18 2023: Media server q1 has 'New' task in the queue. Task ID: 2. Task type: Backup global
Wed Feb 15 12:30:18 2023: Task ID: 2. New status: Assigned
Wed Feb 15 12:30:18 2023: Task ID: 2. New status: At_Client
Wed Feb 15 12:30:18 2023: Task ID: 2. New status: Execution
Wed Feb 15 12:30:19 2023: Set unlimited bandwidth for task ID: 2
Wed Feb 15 12:30:22 2023: Task ID: 2. New status: Start_Transfer
Wed Feb 15 12:30:22 2023: Set unlimited bandwidth for task ID: 2
Wed Feb 15 12:30:23 2023: Transfer of snapshot client2_TaskID_2_NORuleOrStrategy_0_D2023_2_15H09_30_18_BackupType_1_ResourceType_11 has succeeded. Task ID: 2
Wed Feb 15 12:30:23 2023: Task ID: 2. New record ID was created in repository: 2
Wed Feb 15 12:30:23 2023: Task ID: 2. New status: Transmission
Wed Feb 15 12:30:24 2023: Task ID: 2. New status: Done
Thu Feb 16 11:21:20 2023: Media server ubuntu-server has 'New' task in the queue. Task ID: 2. Task type: Restore
Thu Feb 16 11:21:20 2023: Task ID: 2. New status: Assigned
Thu Feb 16 11:21:21 2023: Task ID: 2. New status: At_Client
Thu Feb 16 11:21:21 2023: Task ID: 2. New status: Start_Transfer
Thu Feb 16 11:21:21 2023: Task ID: 2. New status: Transmission
Thu Feb 16 11:21:21 2023: Set unlimited bandwidth for task ID: 2
Thu Feb 16 11:21:24 2023: Blocks are ready. time: 2
Thu Feb 16 11:21:26 2023: Task ID: 2. New status: Done
```

Рисунок 35. Журнальный файл

Права на папку, которая была создана для восстановления резервной копии после завершения восстановления будет иметь права `root:root`.

[1] Настройки по умолчанию будут добавлены в последующих релизах программного продукта в это же окно для поля «Параметры восстановления для модуля»

Глава 12. Восстановление со стороны клиента

Для операции восстановления можно использовать утилиту командной строки `rb_archives`.

Использование утилиты командной строки `rb_archives` позволяет посмотреть список резервных копий:

```
root@postgresql:~# rb_archives
Id | Ref ID | Resource          | Resource type        | Backup type | Created                | Crypto  | Signed | Status
---+--------+-------------------+----------------------+-------------+------------------------+---------+--------+-------------------
1  |        | PostgreSQL 12.13 | PostgreSQL universal | full        | 2023-02-16 11:11:03+03 | nocrypt | True   | Not Verified
root@postgresql:~#
```

Рисунок 36. Просмотр списка резервных копий с помощью утилиты `rb_archives`

В первой колонке указаны идентификаторы резервных копий. Чтобы восстановить резервную копию без развертывания, нужно использовать команду:

```
sudo rb_archives -X -d /path_to_restore
```

Опция `-X` указывает, что нужно выполнить операцию восстановления без развертывания

Опция `-d` указывает путь, в который нужно восстановить резервную копию. Если не используется опция `-d`, резервная копия будет восстановлена в каталог для временных операций с резервными копиями, либо, если клиент настроен на использование временной NFS-папки от сервера резервного копирования, восстановление произойдет в эту NFS-папку. В случае восстановления резервной копии без развертывания всегда рекомендуется использовать опцию `-d` с указанием каталога на клиенте, в котором есть достаточно места для восстановления резервной копии.

В том случае, если необходимо выполнить восстановление резервной копии с развертыванием, выполните команду:

```
sudo rb_archives -x -d /path_to_restore
```

Опция `-x` указывает, что нужно восстановить резервную копию с развертыванием.

Для восстановления резервной копии необходимо ввести пароль клиента (задается при первом использовании `rb_archives` со стороны клиента. В том случае если вы не знаете пароль, обратитесь к системному администратору СРК чтобы его сбросить и задать заново).

Проконтролировать выполнение задачи восстановления можно при помощи ути-

литы командной строки `rb_tasks`:

```
root@postgresql:~# rb_tasks
Id | Task type     | Resource          | Backup type | Status | Created
---+---------------+-------------------+-------------+--------+----------------------
1  | Backup global | PostgreSQL 12.13 | full        | Done   | 2023-02-16 11:10:42+03
2  | Restore       | PostgreSQL 12.13 | full        | Done   | 2023-02-16 11:21:20+03
root@postgresql:~#
```

Рисунок 37. Получение информации о выполнении задачи восстановления с помощью утилиты `rb_tasks`

Так же можно получить детальную информацию о ходе восстановления из журнального файла задачи:

```
root@ubuntu-server:~# cat /opt/rubackup/log/task_2.log
Wed Feb 15 12:30:18 2023: Media server q1 has 'New' task in the queue. Task ID: 2. Task type: Backup global
Wed Feb 15 12:30:18 2023: Task ID: 2. New status: Assigned
Wed Feb 15 12:30:18 2023: Task ID: 2. New status: At_Client
Wed Feb 15 12:30:18 2023: Task ID: 2. New status: Execution
Wed Feb 15 12:30:19 2023: Set unlimited bandwidth for task ID: 2
Wed Feb 15 12:30:22 2023: Task ID: 2. New status: Start_Transfer
Wed Feb 15 12:30:22 2023: Set unlimited bandwidth for task ID: 2
Wed Feb 15 12:30:23 2023: Transfer of snapshot client2_TaskID_2_NORuleOrStrategy_0_D2023_2_15H09_30_18_BackupType_1_ResourceType_11 has succeeded. Task ID: 2
Wed Feb 15 12:30:23 2023: Task ID: 2. New record ID was created in repository: 2
Wed Feb 15 12:30:23 2023: Task ID: 2. New status: Transmission
Wed Feb 15 12:30:24 2023: Task ID: 2. New status: Done
Thu Feb 16 11:21:20 2023: Media server ubuntu-server has 'New' task in the queue. Task ID: 2. Task type: Restore
Thu Feb 16 11:21:20 2023: Task ID: 2. New status: Assigned
Thu Feb 16 11:21:21 2023: Task ID: 2. New status: At_Client
Thu Feb 16 11:21:21 2023: Task ID: 2. New status: Start_Transfer
Thu Feb 16 11:21:21 2023: Task ID: 2. New status: Transmission
Thu Feb 16 11:21:21 2023: Set unlimited bandwidth for task ID: 2
Thu Feb 16 11:21:24 2023: Blocks are ready. time: 2
Thu Feb 16 11:21:26 2023: Task ID: 2. New status: Done
```

Рисунок 38. Получение информации о ходе восстановления из журнального файла задачи

Глава 13. Восстановление на определенный момент времени (Point in time recovery (PITR))

Рекомендуется заранее подготовить инструкцию по восстановлению именно вашей инфраструктуры в контексте PITR, проверить эту инструкцию, провести обучение персонала и проводить регулярные учения по восстановлению СУБД из сделанных резервных копий!

Настоящее руководство является описанием функционала и не является точной инструкцией во восстановлению СУБД в любой ситуации, которая может произойти!

Восстановление на определенный момент времени (Point in time recovery (PITR)) невозможно для подмодуля pg_probackup.

В случае, когда требуется восстановление на определенный момент времени или на определенную транзакцию, необходимо:

1. Восстановить резервную копию без развертывания.
2. В зависимости от версии PostgreSQL добавить необходимую метку в конфигурационный файл (версии PostgreSQL > 12) или в файл `recovery.conf` в соответствии с документацией PostgreSQL требуемой версии, например: https://www.postgresql.org/docs/12/continuous-archiving.html#BACKUP-PITR-RECOVERY

Конкретная точка восстановления должна быть установлена в соответствии с https://postgrespro.ru/docs/postgrespro/12/runtime-config-wal#RUNTIME-CONFIG-WAL-RECOVERY-TARGET

Глава 14. Резервное копирование и восстановление СУБД PostgreSQL в кластере Patroni

Все действия по копированию выполняются только на клиенте с ролью, указанной в параметре `patroni_node_type_for_backup`, по умолчанию это «Лидер».

14.1. Создание группы Patroni на сервере RuBackup

При регистрации в RuBackup все клиенты помещаются в группу No group. Для корректной работы с кластером нужно создать отдельную группу клиентов, переместить в неё все клиенты кластера, а также установить атрибуты группы «Разделяемая группа» и «Кластерная группа» в `true` (Рисунок 39). Более подробно создание и добавление клиентов в группу описано в документе «Руководство системного администратора RuBackup».

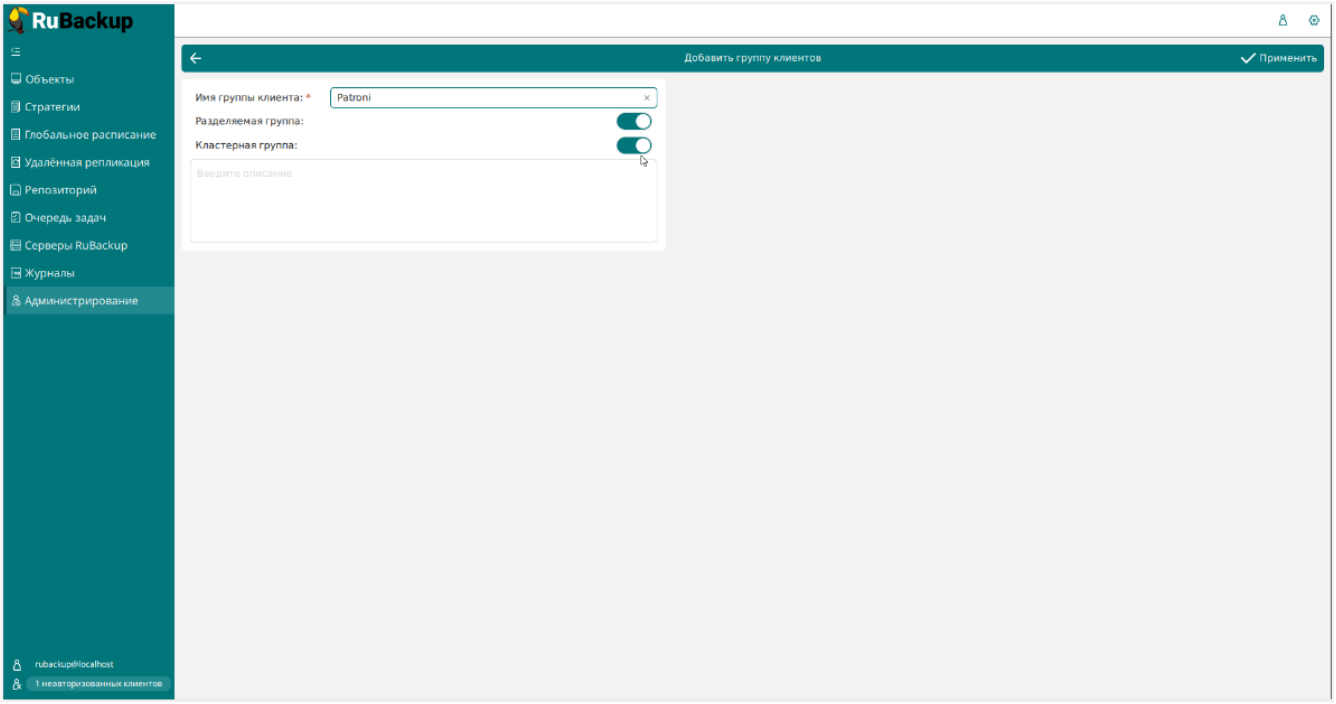

Рисунок 39. Атрибуты «Разделяемая группа» и «Кластерная группа»

14.2. Выполнение полной и/или инкрементальной копии кластера Patroni

Единственное условие выполнения полной и/или инкрементальной копии кластера Patroni — выполнение копирования только на клиенте с ролью, указанной в параметре `patroni_node_type_for_backup`, по умолчанию это «Лидер».

Для выполнения полного и/или инкрементального копирования в кластере Patroni необходимо на каждом клиенте, входящем в кластер, настроить следующие пара-

метры: `patroni_host`, `patroni_port`, `patroni_node_type_for_backup`.

14.3. Восстановление без развертывания

1. Определить лидера кластера Patroni командой

   ```
   patronictl -d etcd://адрес_хоста_etcd list patroni_cluster_1
   ```

   выполненной от пользователя `postgres` на клиенте, входящем в кластер;

2. Отключить элементы кластера с ролью Replica;
3. Отключить элемент кластера с ролью Leader;
4. Запустить процесс восстановления без развертывания в каталог для восстановления на клиенте с ролью Leader;
5. После завершения задачи по восстановлению необходимо перенести файлы из каталога для восстановления в целевые каталоги, т. е. из каталога для восстановления `/restore_dir/number.rest/var/lib/postgresql/11/main` (где `number` — это номер резервной копии) в `/var/lib/postgresql/11/main` с заменой файлов, а также из каталога для восстановления `/restore_dir/number.rest/opt/rubackup/mnt/postgresql_archives` перенести wal-файлы (где `number` — это номер резервной копии) для восстановления в `/opt/rubackup/mnt/postgresql_archives/`;
6. Убедиться, что у перенесенных файлов владелец и группа назначены `postgres`;
7. На клиенте с ролью Leader от пользователя postgres необходимо удалить кластер командой:

   ```
   patronictl -c /etc/patroni/config.yml remove patroni_cluster_1
   ```

   Потребуется подтверждение удаления кластера: на первый запрос повторно ввести имя кластера, на второй запрос ввести фразу Yes I am aware;

   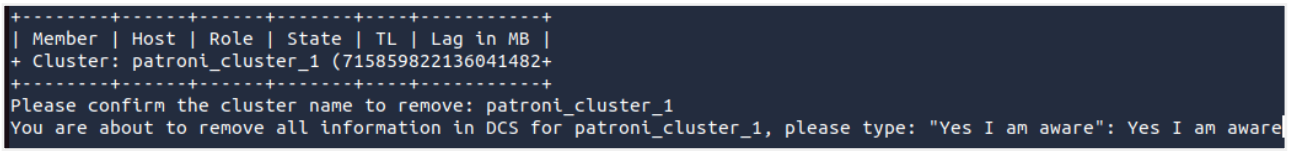

   Рисунок 40. Подтверждение удаления кластера

8. Запустить элемент кластера на клиенте с ролью Leader;
9. Запустить элементы кластера на клиенте с ролью Replica;
10. Убедиться, что все элементы имеют статус running командой:

```
patronictl -d etcd://адрес_хоста_etcd list patroni_cluster_1
```

выполненной от пользователя postgres на клиенте, входящем в кластер.

14.4. Восстановление в режиме Point in Time Recovery (PITR)

1. Определить лидера кластера Patroni командой

```
patronictl -d etcd://адрес_хоста_etcd list patroni_cluster_1
```

выполненной от пользователя postgres на клиенте, входящем в кластер;

1. Отключить элементы кластера с ролью Replica;
2. Отключить элемент кластера с ролью Leader;
3. Запустить процесс восстановления без развертывания в каталог для восстановления на ВМ с ролью Leader;
4. После завершения задачи по восстановлению необходимо скопировать файлы из каталога для восстановления в целевые каталоги, т. е. из каталога для восстановления `/restore_dir/number.rest/var/lib/postgresql/11/main` (где number — это номер резервной копии) скопировать файлы в `/var/lib/postgresql/11/main`, предварительно удалив файлы из целевого каталога, и из каталога для восстановления `/restore_dir/number.rest/opt/rubackup/mnt/postgresql_archives` (где `number` — это номер резервной копии) перенести wal-файлы для восстановления в `/opt/rubackup/mnt/postgresql_archives/`
5. Убедиться, что у скопированных файлов назначены владелец и группа `postgres`;
6. На клиенте с ролью Leader от пользователя postgres необходимо удалить кластер командой:

```
patronictl -c /etc/patroni/config.yml remove patroni_cluster_1
```

Потребуется подтверждение удаления кластера: на первый запрос повторно ввести имя кластера, на второй запрос ввести фразу `Yes I am aware`.

7. Запустить элемент кластера на клиенте с ролью Leader;
8. Запустить элементы кластера на клиенте с ролью Replica;
9. Убедиться, что все элементы имеют статус running командой:

```
patronictl -d etcd://адрес_хоста_etcd list patroni_cluster_1
```

выполненной от пользователя `postgres` на клиенте, входящем в кластер;

10. Отключить элементы кластера с ролью Replica;
11. Отключить элемент кластера с ролью Leader;
12. Еще раз скопировать файлы из каталога для восстановления `/restore_dir/number.rest/var/lib/postgresql/11/main` (где `number` — это номер резервной копии) в `/var/lib/postgresql/11/main`, предварительно удалив файлы из целевого каталога;
13. В файле `postgresql.conf` внести значения в параметр: `recovery_target_time = '2023-03-27 15:29:00'` и закомментировать остальные параметры recovery_target, если они не используются;

```
# recovery.conf
#recovery_target = ''
#recovery_target_lsn = ''
#recovery_target_name = ''
recovery_target_time = '2023-03-27 15:29:00'
#recovery_target_timeline = 'latest'
#recovery_target_xid = ''
```

Рисунок 41. Редактирование файла `postgresql.conf`

14. В файле `postgresql.auto.conf` внести значения в следующие параметры: `restore_command = 'cp /opt/rubackup/mnt/postgresql_archives/%f %p'`, `recovery_target_time = '2023-03-27 15:29:00'` и `recovery_target_action = 'promote'`

```
recovery_target_time = '2023-03-27 15:29:00'
recovery_target_action='promote'
restore_command = 'cp /opt/rubackup/mnt/postgresql_archives/%f %p'
~
~
```

Рисунок 42. Редактирование файла `postgresql.auto.conf`

15. Запустить PostgreSQL командой

```
sudo -u postgres путь_к_bin/postgres -D
путь_к_data_содержащей_конфиг_файлы
```

Пример 1.

```
sudo -u postgres /usr/local/pgsql/bin/postgres -D
```

```
/usr/local/pgsql/data/patroni_cluster_1/data/
```

16. После успешного запуска PostgreSQL проверить в логах, что СУБД готова принимать подключения;
17. Остановить PostgreSQL;
18. На клиенте с ролью Leader от пользователя postgres необходимо удалить кластер командой `patronictl -c /etc/patroni/config.yml remove patroni_cluster_1`. Потребуется подтверждение удаления кластера: на первый запрос повторно ввести имя кластера, на второй запрос ввести фразу `Yes I am aware`;

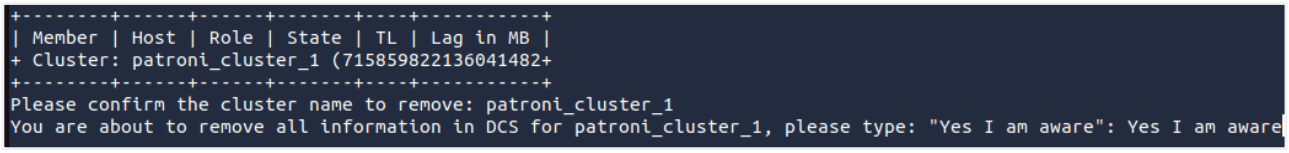

Рисунок 43. Подтверждение удаления кластера

19. Запустить элемент кластера на ВМ с ролью Leader;
20. Запустить элементы кластера на клиенте с ролью Replica;
21. Убедиться, что все элементы имеют статус running командой, выполненной от пользователя postgres на клиенте, входящем в кластер:

```
patronictl -d etcd://адрес_хоста_etcd list patroni_cluster_1
```

Глава 15. Резервное копирование с использованием подмодуля pg_probackup

15.1. Подготовка к использованию pg_probackup

Для корректной работы подмодуля необходимо наличие:

- утилиты pg_probackup версии 2.6 или выше;
- ptrack версии 2.6.0 или выше.

Для выполнения резервного копирования с помощью модуля PostgreSQL с подмодулем pg_probackup выполните следующие действия:

1. Запустите RBM командой:

   ```
   rbm
   ```

   После этого в открывшемся окне (Рисунок 44) введите наименование сервера RuBackup, имя пользователя и пароль.

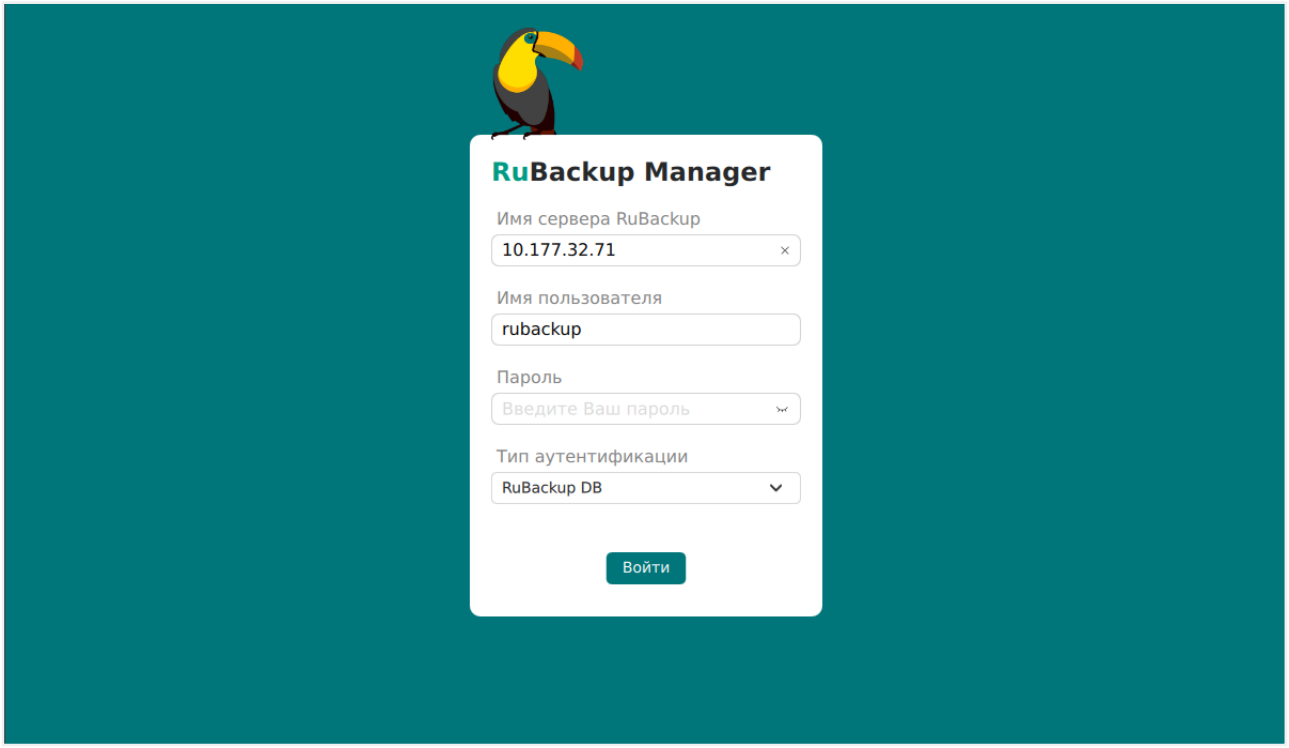

Рисунок 44. Окно входа RBM

2. Добавьте пул типа «Client defined» с помощью RBM либо командной строки (см. подробнее в документе «Руководство системного администратора RuBackup»);
3. Настройте Клиентское хранилище с помощью RBM либо командной строки (см.

подробнее в документе «Руководство системного администратора RuBackup»);

> В случае работы с подмодулем `pg_probackup` при выборе типа хранилища Cloud вам будет предложено ввести данные для работы с облачным хранилищем: хост облака, порт облака, бакет, безопасность облака, ID ключа доступа, секретный ключ доступа. Данная информация будет передаваться модулю клиента при выполнении различных операций и перезаписывать конфигурационный файл `/etc/pg_probackup/s3.config`. Таким образом, указав данные в RBM, вы сможете обновить данные для всех клиентов, которые будут использовать данный пул.

4. Настройте на клиентском хосте базу данных PostgreSQL;
5. Установите на клиентском хосте модуль для PostgreSQL Universal (`rb_module_postgresql`);
6. Разверните клиент резервного копирования, сконфигурируйте и подключите его к серверу СРК на хосте, где установлен Модуль для PostgreSQL Universal;
7. Создайте в S3-хранилище MinIO папку, в которую будут помещаться резервные копии;
8. Настройте PTRACK на Клиенте для создания и восстановления инкрементальных копий табличных пространств CFS;
9. Настройте утилиту `pg_probackup` на Клиенте для работы с S3-хранилищем;

> Для работы с S3-хранилищем MinIO в утилите pg_probackup нужно использовать ключ `--s3=minio`

1. Для использования режима постраничного копирования (PAGE) настройте непрерывное архивирование WAL с сервера БД в СРК RuBackup.

15.2. Инициализация каталога резервных копий

Для начала работы подмодуля pg_probackup выполните последовательно следующие команды от имени пользователя указанного в параметре `postgresql_admin` в конфигурационном файле модуля (`rb_module_postgresql.conf`), по умолчанию это пользователь `postgres`. (подробнее см. в официальной документации):

```
pg_probackup init -B каталог_копий [--skip-if-exists] [параметры_s3] [--help]

pg_probackup add-instance -B каталог_копий -D каталог_данных --instance
имя_экземпляра [--skip-if-exists] [параметры_s3] [--help]
```

> Имя_экземпляра должно совпадать с именем каталог_данных. Например,

если добавляется каталог данных `/var/lib/pgpro/std-13/data/`, значит, именем экземпляра будет `data`.

Директория `/opt/rubackup/mnt/pg_probackup` и все вложенные в неё папки должны быть доступны для записи и чтения пользователю указанному в параметре `postgresql_admin` в конфигурационном файле модуля (`rb_module_postgresql.conf`), по умолчанию этот пользователь `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
```

Ниже описаны параметры, которые необходимо задать для размещения копий в частном облачном хранилище. Эти параметры могут задаваться с любыми командами, которые pg_probackup выполняет через интерфейс S3:

- `--s3=s3_interface_provider` — задаёт провайдера, поддерживающего интерфейс S3. Возможные значения:
  - minio — объектное хранилище MinIO, совместимое с облачным хранилищем S3. С этим провайдером по умолчанию используется протокол HTTP и порт 9000.
- `--s3-config-file=путь_к_файлу_конфигурации`. Если этот параметр опускается, `pg_probackup` ищет файл конфигурации S3 сначала в `/etc/pg_probackup/s3.config`, а затем в `~postgres/.pg_probackup/s3.config`.

Подробнее о заполнении конфигурационного файла для работы с S3 см. официальную документацию (раздел «Параметры S3).

Примеры использования команд с S3:

- Пример использования add-instance:

```
pg_probackup add-instance -B /opt/rubackup/mnt/pg_probackup --instance=data
--pgdata=/var/lib/pgpro/ent-16/data --s3=minio
```

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

```
sudo pg_probackup set-config -B /opt/rubackup/mnt/pg_probackup/ --instance
=patroni --pghost ip_кластера
```

где `ip_кластера` – это ip-адрес копируемого экземпляра patroni.

15.3. 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, базы данных и пароль 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`.

   a. Выполните подключение к базе данных backupdb от имени пользователя postgres:

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

   b. 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;
```

Для Postgres Pro 16 команды могут отличаться – см. официальную документацию.

3. Создайте файл `.pgpass` в домашнем каталоге пользователя указанного в параметре postgresql_admin в конфигурационном файле модуля (`rb_module_postgresql.conf`), по умолчанию это пользователь `postgres`. В файле `.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 операции с `pg_probackup`. Этот файл должен содержать строки в таком формате:

   ```
   сервер:порт:база_данных:имя_пользователя:пароль
   ```

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

   Рисунок 45. Пример файла .pgpass

   Файл `.pgpass` обязательно должен находиться в домашнем каталоге пользователя указанного в параметре postgresql_admin в конфигурационном файле модуля (`rb_module_postgresql.conf`), по умолчанию это пользователь postgres (домашний каталог — `/var/lib/postgresql`). 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
```

При использовании HAProxy, необходимо в файл `pg_hba.conf` добавить следующие строки:

```
host backupdb rubackup_backuper <IP-адрес хоста HAProxy> md5
host replication rubackup_backuper <IP-адрес хоста HAProxy> 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
```

Рисунок 46. Пример файла `pg_hba.conf`

Проверяем, не было ли опечаток, и перечитываем конфигурацию:

```
psql -c 'select * from pg_hba_file_rules'
```

```
psql -c 'select pg_reload_conf()'
```

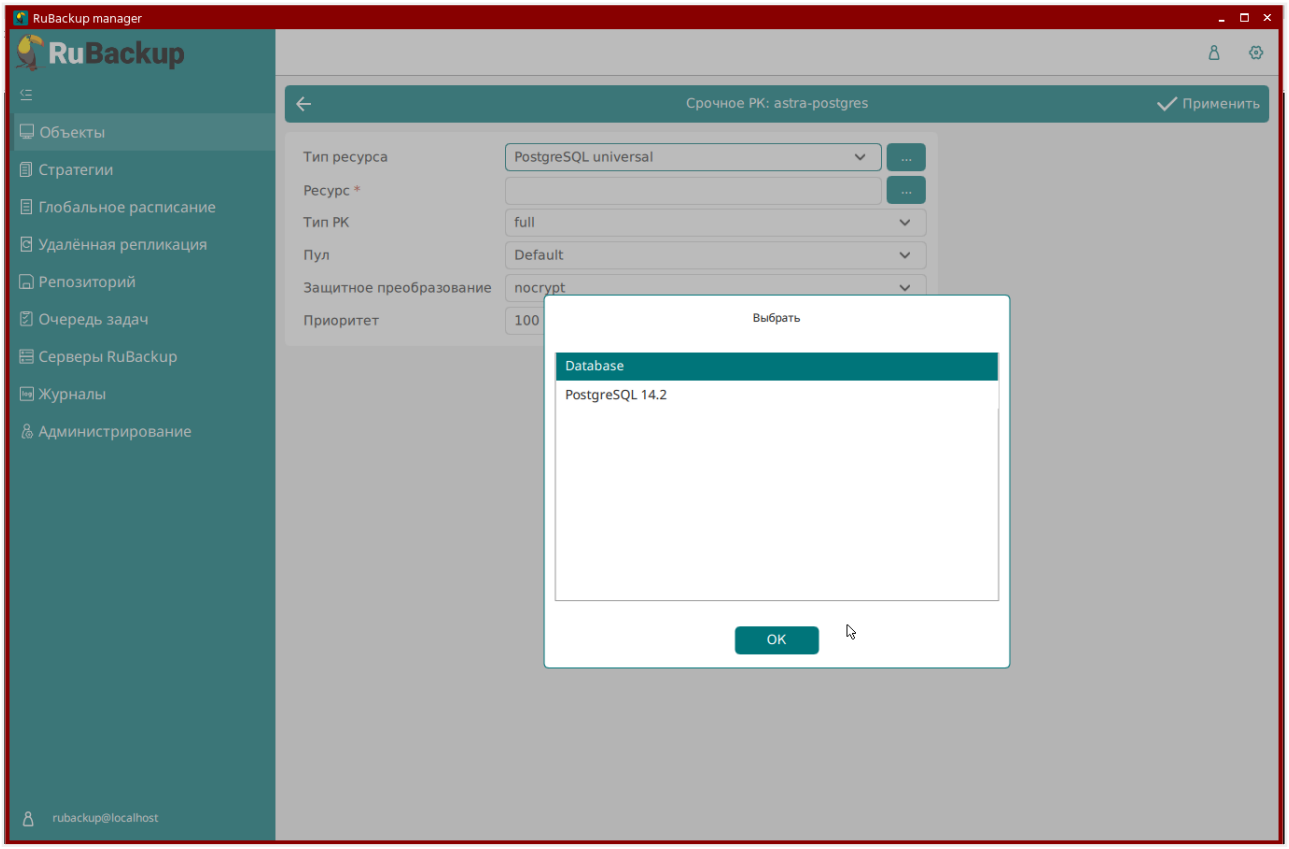

Рисунок 47. Проверка конфигурации

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

15.4. 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).

15.5. 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. если вы настраиваете резервное копирование на ведущем сервере, задайте для параметра `wal_level` значение выше `minimal`. параметр `archive_mode` должен иметь значение `on` или `always`. Для выполнения резервного копирования на ведомом требуется значение `always`.
2. 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 — это 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.

   Пример:

   ```
   archive_command = '/opt/pgpro/ent-16/bin/pg_probackup archive-push -B
   /opt/rubackup/mnt/pg_probackup --instance data --wal-file-path=%p --wal
   -file-name=%f --s3=minio'
   ```

3. установите параметр restore_command:

```
restore_command = 'путь_инсталляции/pg_probackup archive-get -B
каталог_копий --instance имя_экземпляра --wal-file-path=%p --wal-file
-name=%f [параметры_удаленного_режима]
```

где каталог_копий – это каталог, предназначенный для резервных копий.

Пример:

```
restore_command = '/opt/pgpro/ent-16/bin/pg_probackup archive-get -B
/opt/rubackup/mnt/pg_probackup --instance data --wal-file-path=%p --wal
-file-name=%f --s3=minio'
```

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.

15.6. Настройка копирования в режиме PTRACK

Перед выполнением дифференциальной резервной копии в режиме PTRACK выполните подготовительные действия, как указано в разделе Настройка копирования в режиме PTRACK.

15.7. 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: PostgreSQL...
Execute OS command: /opt/rubackup/modules/rb_module_postgres -t 2>&1
... module PostgreSQL was checked successfully
```

15.8. Принцип работы подмодуля pg_probackup

Подробную информацию о принципе работы подмодуля pg_probackup Вы можете посмотреть на официальном сайте Postgres Pro.

15.9. Пример использования подмодуля pg_probackup в Менеджере администратора RuBackup (RBM)

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

1. Создайте правило Глобального расписания, для чего зайдите в раздел «Глобальное расписание» (Рисунок 48) и нажмите на кнопку «Добавить»;

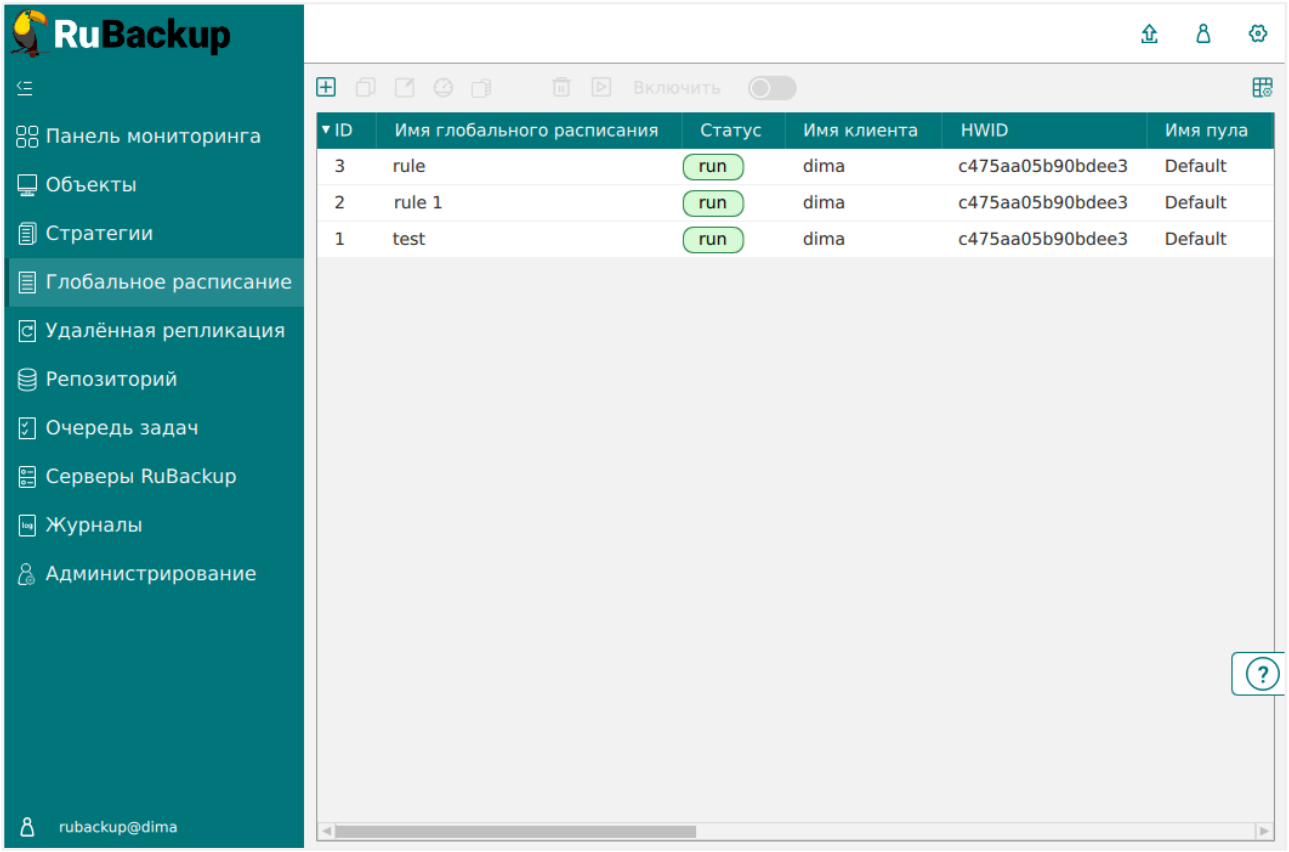

Рисунок 48. Создание правила глобального расписания

2. В открывшемся окне (Рисунок 49) выберите Клиент, на котором установлен Модуль для PostgreSQL Universal;

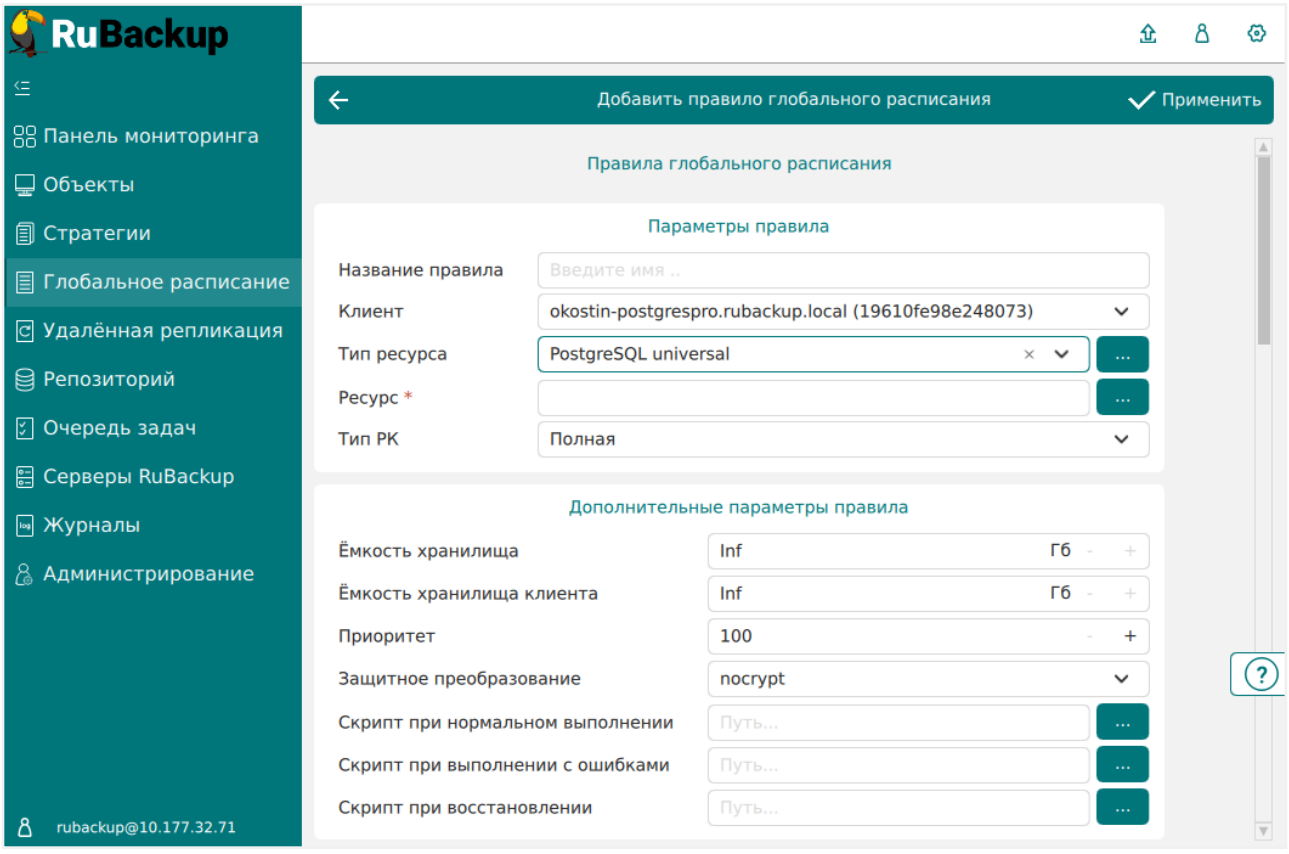

Рисунок 49. Выбор клиента, на котором установлен модуль для PostgreSQL Universal

3. Выберите тип ресурса — «PostgreSQL universal»;
4. Откройте параметры Модуля нажатием кнопки «...» рядом с выбранным типом ресурса;
5. В появившемся окне (Рисунок 50) выберите подмодуль (engine) pg_probackup;

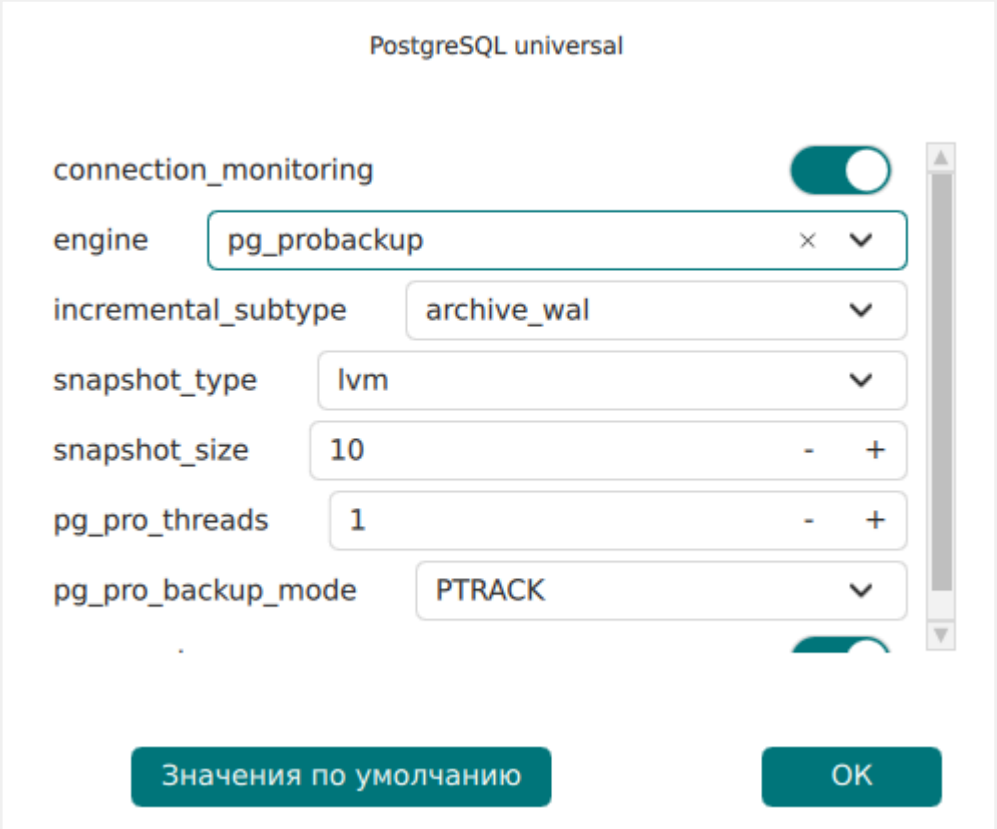

Рисунок 50. Выбор подмодуля

6. Настройте число параллельных потоков (pg_pro_threads), в которые будет выполняться резервное копирование или восстановление. По умолчанию количество потоков равно 1;
7. Выберите режим резервного копирования («pg_pro_backup_mode») — (DELTA, PAGE, PTRACK). По умолчанию — DELTA;

> Перед выполнением резервного копирования в режиме PAGE произведите настройку непрерывного архивирования WAL (см. Глава 15). А перед выполнением резервного копирования в режиме PTRACK произведите настройку согласно Глава 16.

8. Выберите, какой режим доставки WAL использовать (параметр pg_pro_stream, (Рисунок 51)): STREAM (во включенном положении, по умолчанию) или ARCHIVE (в выключенном положении);

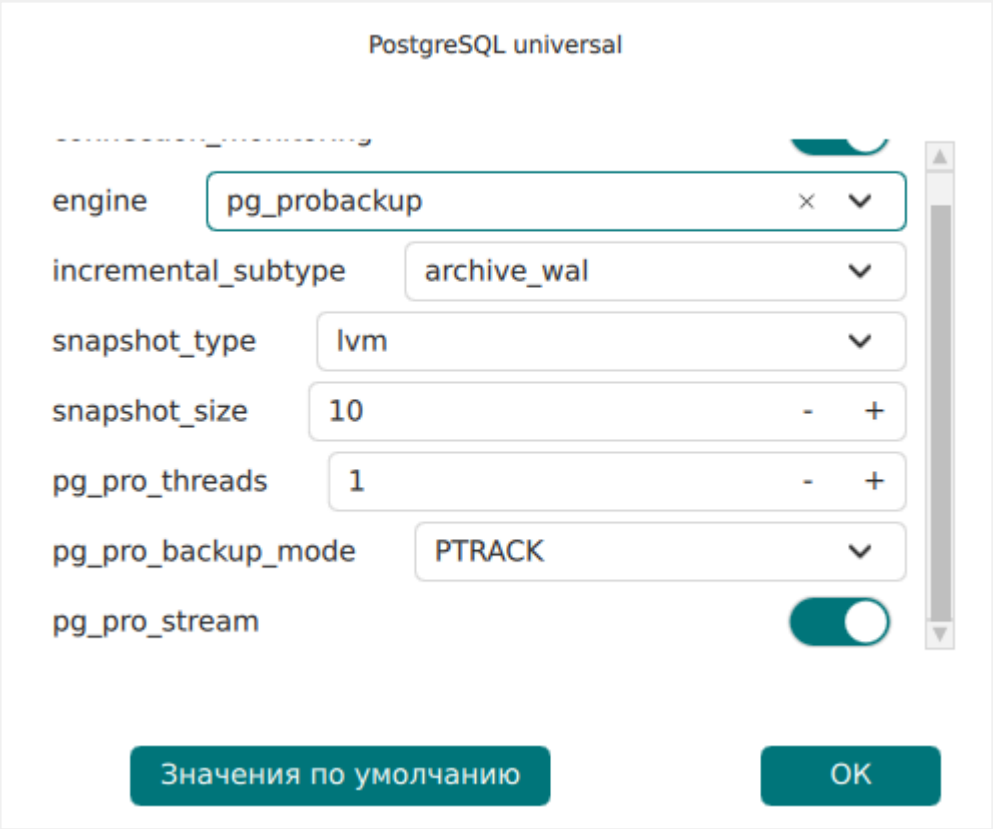

Рисунок 51. Выбор режима доставки WAL

15.10. Настройка копирования в режиме PTRACK

Для корректной работы модуля PostgreSQL необходим ptrack версии 2.6.0 или выше.

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

1. Отредактируйте конфигурационный файл `postgresql.conf`:
2. задайте для параметра `shared_preload_libraries` значение `ptrack`:
3. 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`.

   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.

4. Выполните команду для перезапуска сервиса.

```
sudo systemctl restart postgres[имя_сервиса].service
```

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

1. Зайдите от имени администратора БД в backupdb:

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

   и выполните следующий запрос:

```
CREATE EXTENSION ptrack;
```

2. Завершите настройку параметров Модуля нажатием кнопки «OK»;
3. Выберите ресурс;
4. Выберите тип резервной копии — полная;
5. Выберите пул типа «Client Defined»;

   Если будет выбран пул другого типа, задача завершится с ошибкой.

6. Выберите остальные параметры в окне правила глобального расписания и нажмите на кнопку «Применить».

Не забудьте установить флаг «С развертыванием» во время восстановления резервной копии

Глава 16. Настройка копирования в режиме PTRACK

Для корректной работы модуля PostgreSQL необходим ptrack версии 2.6.0 или выше.

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

1. Отредактируйте конфигурационный файл `postgresql.conf`:
2. задайте для параметра `shared_preload_libraries` значение `ptrack`:
3. 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`.

   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.

4. Выполните команду для перезапуска сервиса.

   ```
   sudo systemctl restart postgres[имя_сервиса].service
   ```

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

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

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

   и выполните следующий запрос:

   ```
   CREATE EXTENSION ptrack;
   ```