RuBackup
Система резервного копирования и восстановления данных

Резервное копирование и восстановление отдельных баз данных и таблиц PostgreSQL

Версия 2.1

05.04.2024 г.

Содержание

Введение

Система резервного копирования (СРК) RuBackup позволяет осуществлять резервное копирование и восстановление таблиц и баз данных СУБД PostgreSQL. Модули резервного копирования поддерживают СУБД PostgreSQL версий 9.6, 10, 11, 12, 13, 14.

Принцип резервного копирования отдельной БД или таблицы с использованием RuBackup состоит в периодическом создании базовых полных резервных копий экземпляра СУБД по определённому расписанию.

В репозитории RuBackup базовые резервные копии будут храниться как полные резервные копии (*full*). Если производится резервное копирование отдельной БД, файл с расширением «.snap» содержит в себе имена всех таблиц, которые были внутри скопированной БД.

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

При необходимости архивный файл может быть преобразован при помощи алгоритма защитного преобразования на клиенте и сжат. Снимок состояния не преобразовывается.

Для выполнения резервного копирования на хосте клиента должно быть достаточно свободного места для создания резервной копии.

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

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

Подготовка клиента

Для возможности резервного копирования отдельной БД или таблицы СУБД PostgreSQL при помощи СРК RuBackup на хост с СУБД следует установить следующие пакеты:

1) `rubackup-client.deb` – клиент резервного копирования;

2) `rubackup-pg-dump.deb` – модуль резервного копирования.

Установка клиента RuBackup

Для осуществления резервного копирования и восстановления данных СУБД PostgreSQL при помощи RuBackup на сервер должен быть установлен клиент RuBackup со всеми необходимыми модулями. Клиент RuBackup представляет собой фоновое системное приложение (демон или сервис), обеспечивающее взаимодействие с серверной группировкой RuBackup.

Для выполнения резервного копирования ресурсов СУБД PostgreSQL клиент RuBackup должен работать от имени суперпользователя (root в Linux и Unix).

Подробно процедуру установки клиента RuBackup см. «Руководство по установке серверов резервного копирования и Linux клиентов RuBackup».

Установка пакета модулей резервного копирования

Установка пакета модулей резервного копирования RuBackup производится из учётной записи root на хосте с СУБД PostgreSQL **после** установки на него клиента RuBackup.

Для установки пакета модулей используйте следующий вызов:

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

Обновление конфигурационного файла

При необходимости вы можете обновить модуль резервного копирования PostgreSQL. При этом обновится конфигурационный файл модуля.

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_pg_dump_database.conf — текущий конфигурационный файл модуля. После слияния будет переименован в rb_module_pg_dump_database_old.conf.
- /opt/rubackup/etc/rb_module_pg_dump_database_old.conf — старый конфигурационный файл который был загружен в предыдущее обновление или при установке модуля.
- /opt/rubackup/etc/rb_module_pg_dump_database_upgrade.conf — конфигурационный файл обновления. Должен быть создан вручную.

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

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_pg_dump_database.conf переименовывается в rb_module_pg_dump_database_old.conf.
2. Создается файл /opt/rubackup/etc/rb_module_pg_dump_database.conf, который далее будет использован в качестве текущего.
3. В созданный файл rb_module_pg_dump_database.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 обязательный параметр. 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
```

В результате автоматического обновления будет обновлен конфигурационный файл rb_module_pg_dump_database.conf. Модуль PostgreSQL будет готов к работе.

При слиянии конфигурационных файлов будут удалены все комментарии из старого конфигурационного файла.

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

Удаление клиента RuBackup

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

Удаление клиента RuBackup возможно из учётной записи с административными правами.

Для удаления сервиса rubackup-client используйте команды:

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

Для удаления клиента RuBackup и модуля **rubackup-pg-dump** используйте команды:

```
# apt remove rubackup-pg-dump
# apt remove rubackup-client
```

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

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

1. Выполнить команду:

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

2. Перезагрузить операционную систему

```
$ init 6
```

Настройка PostgreSQL

Подготовка к использованию модулей резервного копирования

Для выполнения резервного копирования баз данных и таблиц при помощи СРК RuBackup в СУБД выполните следующие действия:

1. Создайте роль RuBackup — например, **rubackup_backuper**;
2. Создайте базу данных для выполнения процедур резервного копирования — например, **backupdb**;
3. Выполните настройку конфигурации сервера PostgreSQL;
4. Создайте расширение **dblink** для роли RuBackup.

Создание роли RuBackup

Для обеспечения безопасного выполнения операций с базами данных и таблицами необходимо создать роль с ограниченным набором прав. Чтобы создать роль, выполните следующие действия:

1. Подключитесь к СУБД от имени администратора:

   $ sudo -u postgres psql

2. Создайте роль RuBackup:

   # CREATE ROLE <role_name> WITH LOGIN password <role_password>;

Создание специальной базы данных

Для того, чтобы у ранее созданного пользователя была точка входа, необходимо создать отдельную базу данных. Чтобы создать данную БД выполните следующие действия:

1. Подключитесь к СУБД от имени администратора:

   $ sudo -u postgres psql

2. Создайте базу данных:

   # CREATE DATABASE <database_name>;

3. Назначьте ранее созданную роль владельцем базы данных:

   # ALTER DATABASE <database_name> OWNER TO <role_name>;

Настройка конфигурации сервера PostgreSQL

Для подготовки СУБД PostgreSQL к выполнению резервного копирования при помощи СРК RuBackup необходимо выполнить следующие действия:

1. Установите метод подключения peer для пользователя postgres. Для этого необходимо внести изменения в файл pg_hba.conf. Чтобы найти этот файл используйте вызовы:

   $ su postgres
   $ psql -c 'SHOW hba_file;'

   Пример установки метода в hba_file.conf:

   # Database administrative login by Unix domain socket

   local all postgres peer

2. Установите для роли RuBackup метод подключения md5. Данный метод должен распространяться только на ранее созданную базу данных. Пример настройки:

   host backupdb rubackup_backuper 127.0.0.1/32 md5

3. Чтобы изменения вступили в силу без перезагрузки сервера используйте данные вызовы:

Проверить файл на наличие опечаток:

$ psql -c 'SELECT * from pg_hba_file_rules;'

Применить изменения:

$ psql -c ' SELECT pg_reload_conf();'

Включение расширения dblink для роли RuBackup

Для обеспечения безопасности модули используют сквозное подключение к целевой БД, осуществляемое при помощи расширения **dblink**. Чтобы включить расширение для роли RuBackup выполните подключение к ранее созданной базе данных от имени администратора:

$ sudo -u postgres psql -d <database_name>

 # CREATE EXTENSION dblink;

Управление правами роли RuBackup

Для управления правами роли RuBackup используйте следующие команды:

- Перед запуском модулем rb_module_pg_dump процесса восстановления резервной копии нужно предоставить пользователю rubackup_backuper привилегии superuser, выполнив команду:

  alter user rubackup_backuper with superuser

  По завершении восстановления выданные ранее привилегии superuser необходимо отнять, выполнив команду:

  alter user rubackup_backuper with nosuperuser

- Для выполнения резервного копирования БД, содержащей большие объекты, нужно в файле postgresql.conf раскомментировать параметр lo_compat_privileges и установить для него значение **on**:

  lo_compat_privileges = on

- В некоторых случаях роли RuBackup требуется выдать временные права на выполнение команд SELECT и CREATE. Например, когда восстанавливаемая БД уже существует и необходимо создать ее копию вместо аварийного завершения работы.

В СРК RuBackup предусмотрены автоматический и ручной способы выдачи прав роли:

- Автоматическое управление правами роли на основе sql-скриптов, заполняемых вручную.

Для использования автоматического режима управления правами роли необходимо заполнить скрипт */opt/*rubackup/scripts/rb_pg_dump_script.sql . Пример наполнения:

```
/opt/rubackup/scripts/rb_pg_dump_script.sql   [-M--]  0 L:[  1+ 4   5/
GRANT SELECT ON ALL TABLES IN SCHEMA public TO rubackup_backuper;
GRANT SELECT ON ALL SEQUENCES IN SCHEMA public TO rubackup_backuper;
ALTER ROLE rubackup_backuper WITH createdb;
```

- Управление правами роли в ручном режиме при помощи консольной утилиты. При помощи этой утилиты можно выдать права ранее созданному пользователю на выполнение ‘SELECT’ и ‘CREATE’

Для того чтобы управлять правами роли в ручном режиме выполните следующее:

1. Запустить консольную утилиту */opt/rubackup/bin/rb_pg_dump_script*. Утилита находится по пути */opt/rubackup/bin/rb_pg_dump_script.* :

$ /opt/rubackup/scripts/rb_pg_dump_script -1 -U *admin_name* -p *admin_password* -B *backup_user* -H *db_host* -P *db_port* -S *schema*

Параметры команды:

-1 – выдать права ранее созданному пользователю на выполнение команд SELECT и CREATE в целевой БД; параметр -2 снимает эти права;

-U *admin_name* – имя администратора кластера Postgres;

-p *admin_password* – пароль администратора кластера;

-B *backup_user* – имя пользователя RuBackup;

-H *db_host* – хост сервера базы данных или каталог сокетов;

-P *db_port* – порт базы данных;

-S *schema* – схема БД, к которой принадлежат копируемые таблицы.

2. При необходимости отзыва у роли ранее выданных прав, запустите утилиту с опцией -2:

 # /opt/rubackup/scripts/rb_pg_dump_script -2 -U admin_name -p admin_password -B backup_user -H db_host -P db_port -S schema

Завершение подготовки хоста с СУБД PostgreSQL

Перед созданием правил и дальнейшей работой с модулями заполните их конфигурационные файлы актуальной информацией.

1. Для резервного копирования отдельной БД заполните файл /opt/rubackup/etc/rb_module_pg_dump_database.conf следующим образом (данные приведены для примера):

```
backup_user: ‘rubackup_backuper’
password: ‘12345’
backup_db: ‘backupdb’
host: ‘localhost’
port: ‘5432’
pg_dump: ‘/user/bin/pg_dump’
pg_restore: ‘/usr/bin/pg_restore’
```

Параметры конфигурации:

backup_user – имя ранее созданной роли RuBackup;

password – пароль роли RuBackup;

backup_db – имя базы данных роли RuBackup;

host – хост сервера базы данных или каталог сокетов;

port – порт базы данных;

pg_dump – полный путь к утилите pg_dump;

pg_restore – полный путь к утилите pg_restore.

3. Для резервного копирования отдельной таблицы заполните файл /opt/rubackup/etc/rb_module_pg_dump_table.conf следующим образом:

```
backup_user: ‘rubackup_backuper’
password: ‘12345’
backup_db: ‘backupdb’
host: ‘localhost’
```

port: ‘5432’
pg_dump: ‘/user/bin/pg_dump’
pg_restore: ‘/usr/bin/pg_restore’

Параметры конфигурации:

backup_user – имя ранее созданной роли RuBackup;

password – пароль роли RuBackup;

backup_db – имя базы данных роли RuBackup;

host – хост сервера базы данных или каталог сокетов;

port – порт базы данных;

pg_dump – полный путь к утилите pg_dump;

pg_restore – полный путь к утилите pg_restore.

Внимание! Во избежание раскрытия пароля роли RuBackup для конфигурационных файлов рекомендуется определить строгие права только для суперпользователя (chmod 0600).

После выполнения подготовки сервера СУБД PostgreSQL к выполнению резервного копирования необходимо перезапустить клиент RuBackup:

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

В результате клиент должен сообщить о том, что модули rb_module_pg_dump_database и rb_module_pg_dump_table готовы к работе.

Принцип базового резервного копирования при помощи pg_dump

В ходе базового резервного копирования выполняется взаимодействие с утилитой pg_dump.

В модуле предусмотрена настройка формата dump-файла, то есть dump может быть выполнен в формате custom или в формате plain text.

Команда на выполнение полного резервного копирования отдельной базы данных:

```
# /usr/bin/pg_dump --username=<role_name> --
dbname=<target_db> --host=localhost --port=5432 --
format=c > backup.dump
```

Параметры команды:

--username – ранее созданный пользователь RuBackup;

--dbname – имя копируемой базы данных;

--host – хост сервера базы данных или каталог сокетов;

--port – порт базы данных;

--format – формат дампа.

Команда для создания резервной копии отдельной таблицы:

```
# /usr/bin/pg_dump —username=<role_name> --
dbname=<target_db> --table=my_table --host=localhost
--port=5432 --format=c > backup.dump
```

Параметры команды:

--dbname – база данных, которая содержит в себе копируемую таблицу;

--table – имя копируемой таблицы.

Восстановление резервной копии

Описываемые методы могут быть использованы при ручном восстановлении отдельной БД или таблицы.

Если dump был выполнен в формате plain text, то для восстановления воспользуйтесь утилитой psql:

$ psql -U <role_name> -f backup.dump

Если dump был выполнен в формате custom, то возможно применение утилиты pg_restore. Для восстановления резервной копии отдельной базы данных или таблицы PostgreSQL выполните команду:

```
# /usr/bin/pg_restore --username=<role_name> --
dbname=<target_db> --host=localhost --port=5432
backup.dump
```

Внимание! При ручном восстановлении нужно указать существующую базу данных для параметра --dbname.

Внимание! Если в целевой БД уже есть таблица с тем же именем что и восстанавливаемая, то восстановление завершится ошибкой. Рекомендуется либо удалить таблицу, не позволяющую выполнить восстановление, либо выполнить восстановление в другую базу данных.

При восстановлении базы данных из резервной копии на исходный хост происходит замена имеющейся базы данных на восстановленную. При необходимости восстановить базу данных из резервной копии на другой хост, необходимо выбрать хост с установленным на нем PostgreSQL, табличным пространством и пользователем, соответствующим оригинальным в восстанавливаемой базе данных, и после этого выполнить восстановление.

Восстановление резервной копии возможно в базу данных с новым именем, но происходит в режиме data-only. Это означает, что восстанавливаются табличные данные, большие объекты и значения последовательностей, если они присутствуют в резервной копии.

После того как база данных из резервной копии восстановлена с новым именем, Вы можете поменять её настройки вручную. Например, изменить владельца базы данных можно с помощью команды:

```
# ALTER DATABASE <database_name> OWNER TO <new_owner>;
```

Информацию об изменении прочих настроек можно найти в документации по PostgreSQL.

Мастер-ключ

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

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

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

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

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

Защитное преобразование резервных копий

При необходимости, сразу после выполнения резервного копирования 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).

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

Если для правила глобального расписания необходимо выбрать особый режим защитного преобразования с длиной ключа, отличной от 256 бит, и с ключом, расположенным в другом месте, то вы можете сделать это при помощи скрипта, выполняющегося после выполнения резервного копирования (определяется в правиле глобального расписания администратором RuBackup). При этом необходимо, чтобы имя преобразованного файла осталось таким же, как и ранее, иначе задача завершится с ошибкой. Провести обратное преобразование такого файла после восстановления его из архива следует вручную при помощи утилиты rbcrypt. При таком режиме работы нет необходимости указывать алгоритм преобразования в правиле резервного копирования, иначе архив будет повторно преобразован с использованием мастер-ключа.

Алгоритмы защитного преобразования

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

Таблица 1 – Алгоритмы защитного преобразования, доступные в утилите rbcrypt.

Наименование алгоритма	Поддерживаемая rbcrypt длина ключа, бит	Примечание
Anubis	128, 256	
Aria	128, 256	
CAST6	128, 256	
Camellia	128, 256	
Kalyna	128, 256, 512	Украинский национальный стандарт ДСТУ 7624:2014
Kuznyechik	256	Российский национальный стандарт ГОСТ Р 34.12-2015
MARS	128, 256	
Rijndael	128, 256	Advanced Encryption Standard (AES)
Serpent	128, 256	
Simon	128	
SM4	128	Chinese national standard for Wireless LAN
Speck	128, 256	
Threefish	256, 512, 1024	
Twofish	128, 256	

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. Системный администратор RuBackup может запустить RBM на основном сервере резервного копирования RuBackup.

Для запуска RBM следует выполнить команду:

```
# ssh -X user@rubackup_server
# /opt/rubackup/bin/rbm&
```

Пользователь, запускающий RBM, должен входить в группу rubackup.

Для резервного копирования отдельной БД или таблицы PostgreSQL на хосте должен быть установлен клиент RuBackup и необходимые модули. Клиент должен быть авторизован администратором RuBackup.

Если клиент RuBackup установлен, но не авторизован, в нижней части окна RBM появится сообщение о том, что найдены неавторизованные клиенты (рисунок 1). Все новые клиенты должны быть авторизованы в системе резервного копирования RuBackup.

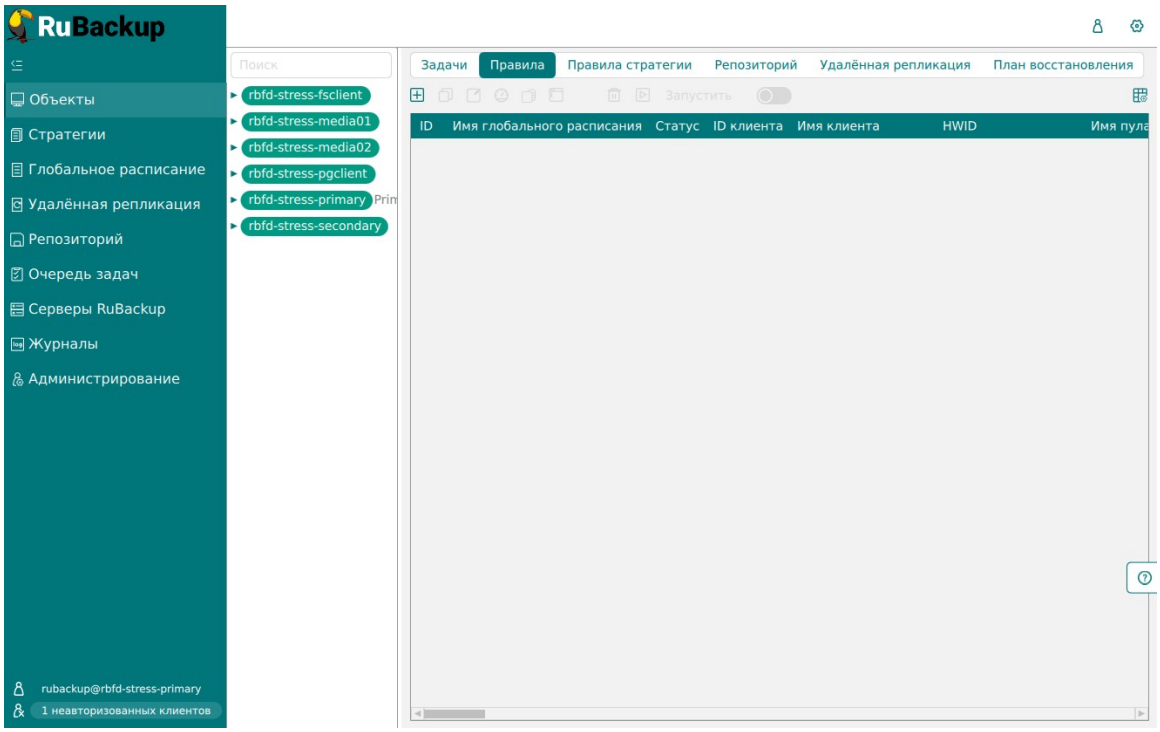


Рисунок 1

Для авторизации неавторизованного клиента в RBM выполните следующие действия:

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

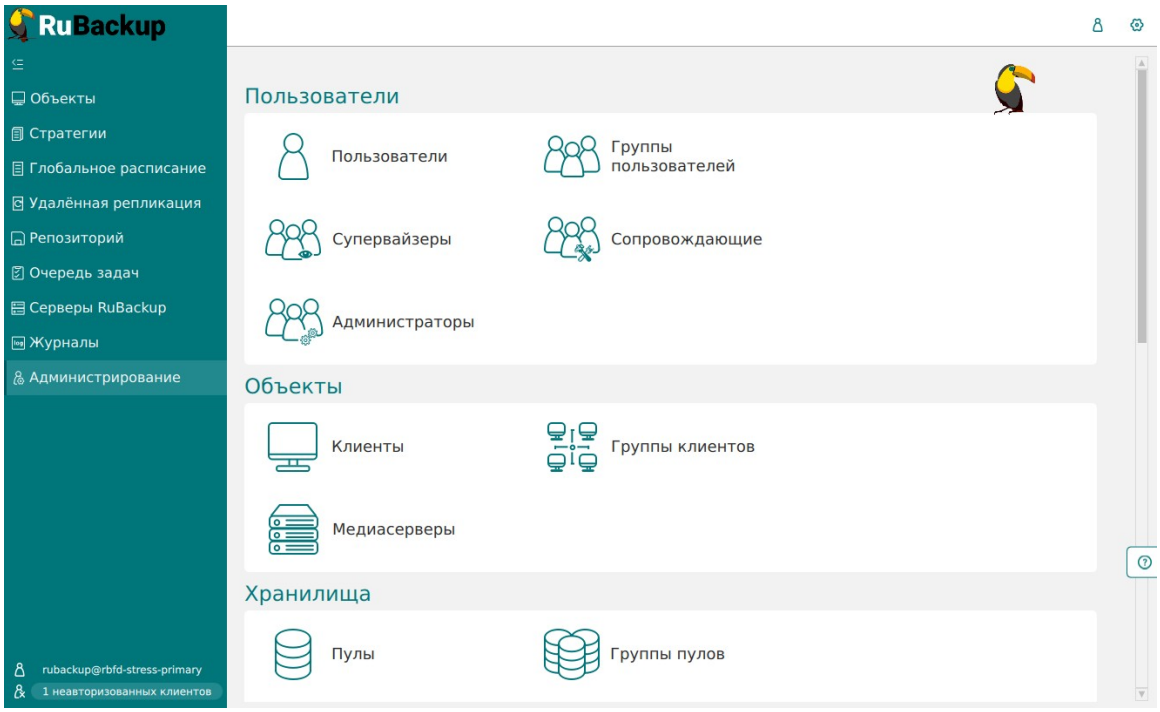


Рисунок 2

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

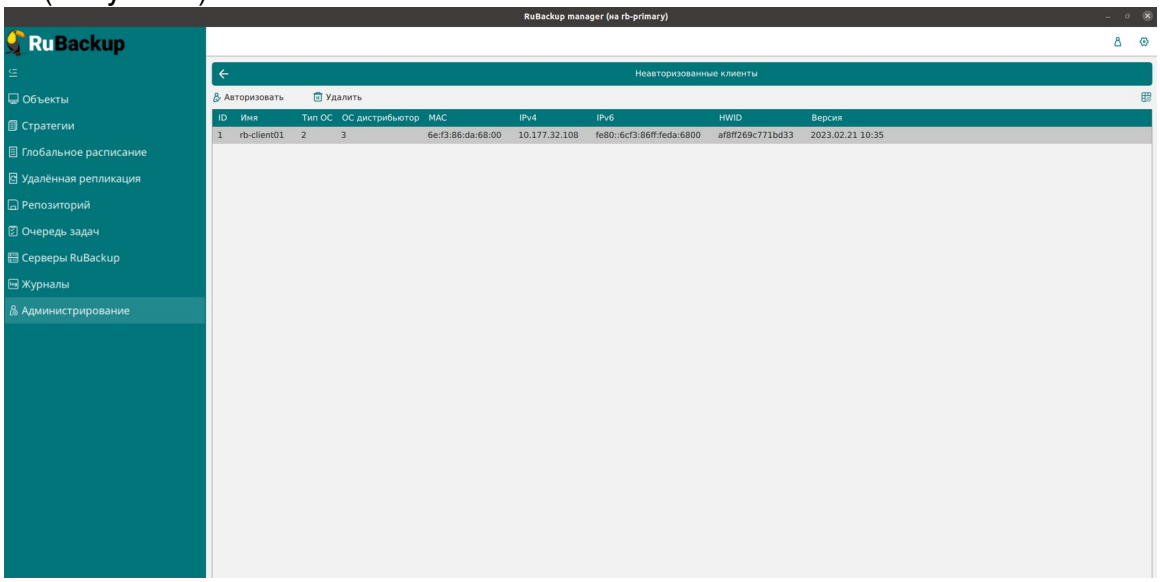


Рисунок 3

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

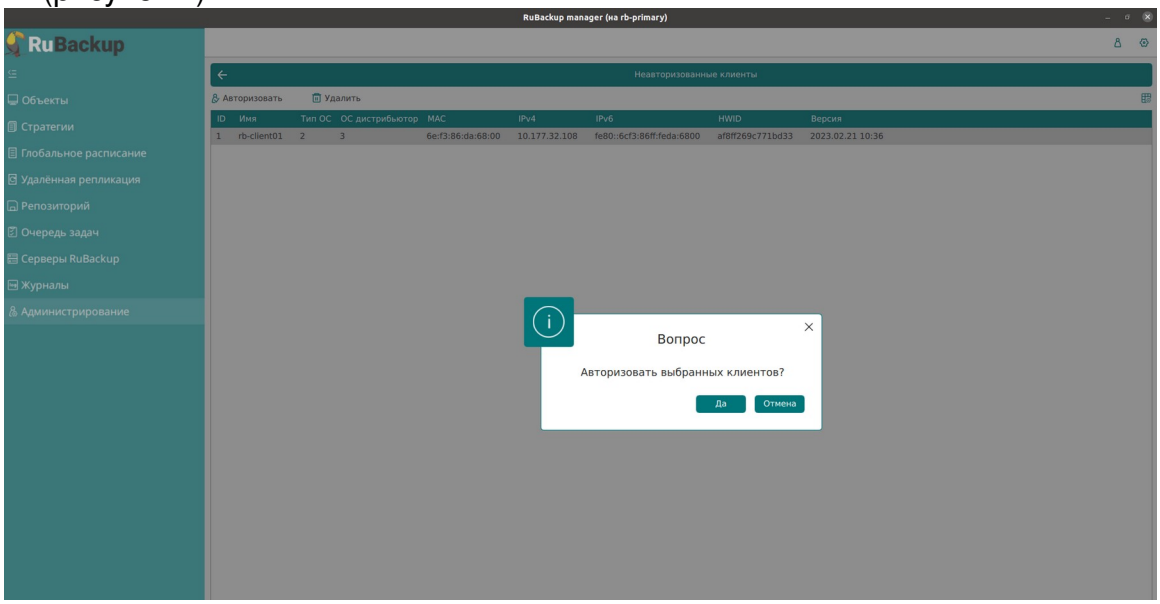


Рисунок 4

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

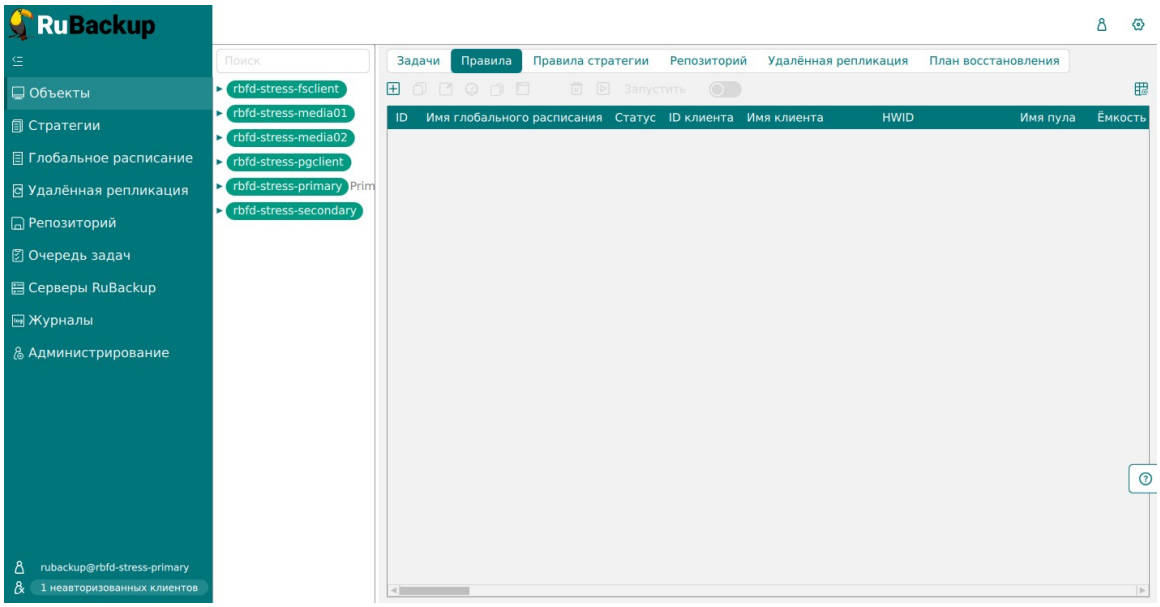


Рисунок 5

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

Чтобы выполнять регулярное резервное копирование отдельной БД или таблицы, необходимо создать правило в глобальном расписании. Для этого выполните следующие действия:

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

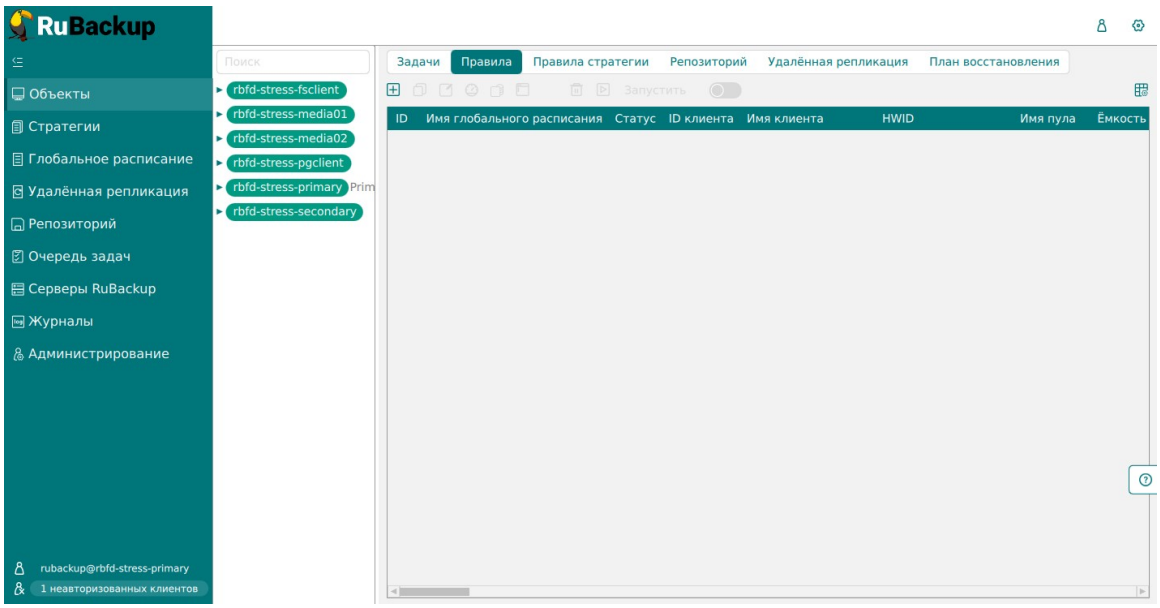


Рисунок 6

2. Выберите тип ресурса: **«PG_dump database»** (для резервного копирования отдельной БД) или **«PG_dump table»** (для резервного копирования таблицы) (рисунок 7):

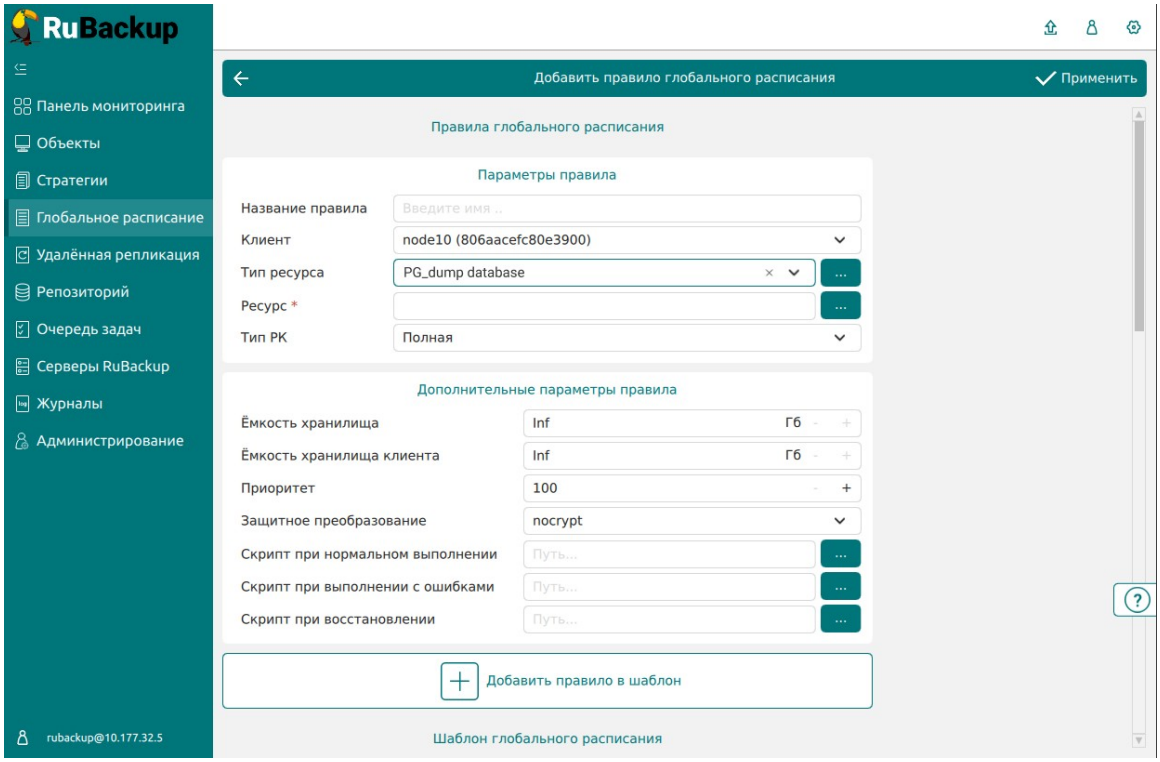


Рисунок 7

3. Выберите ресурс, нажав кнопку **Выбрать**.

Для типа ресурса: «**PG_dump database**» выберите целевую БД.

Для типа ресурса «**PG_dump table**» выберите целевую таблицу.

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

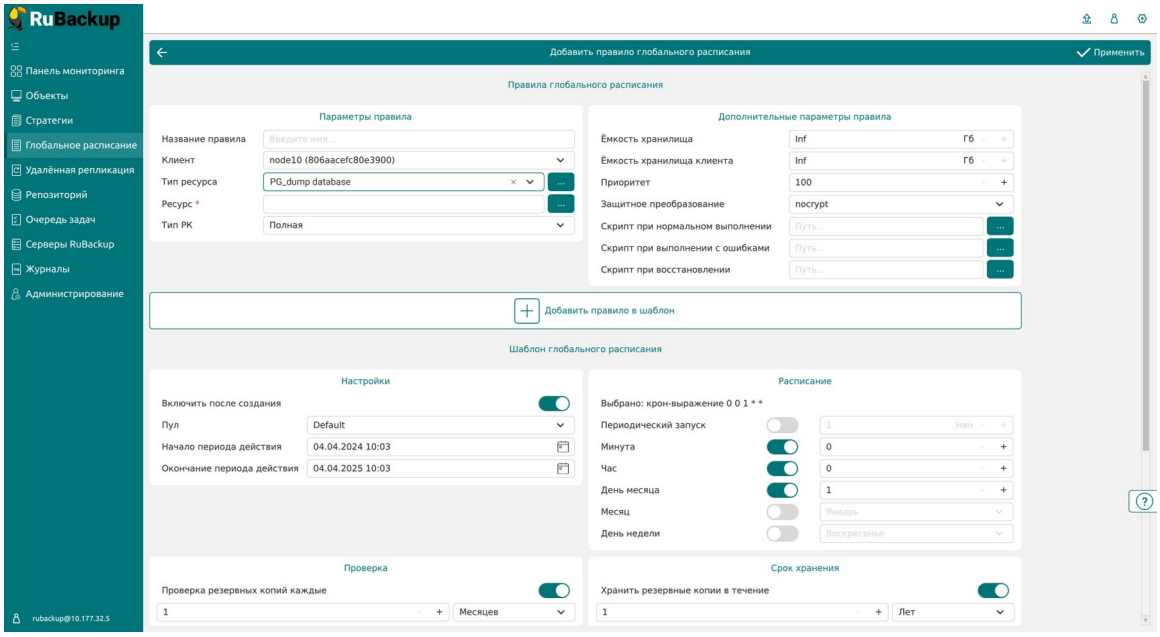


Рисунок 8

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

Дополнительные параметры ресурса представлены в таблице 2.

Таблица 2 – Дополнительные параметры ресурса

Параметр	**Описание**	**Значение по умолчанию**	**Допустимые значения**
threads	Количество потоков выполнения резервной копии	1	Положительные целые числа
dump_format	Формат выводимых данных	1	1(custom), 2(plain text)
serializable_deferrable	Дождаться момента для выгрузки данных без аномалий	true	true, false
no_owner	Не восстанавливать владение объектами	false	true, false
no_sync	Не ждать надёжного сохранения изменений на диске	false	true, false

Параметр	Описание	Значение по умолчанию	Допустимые значения
data_only	Выгрузить только данные, без схемы	false	true, false
schema_only	Выгрузить только схему, без данных	false	true, false
blobs	Выгрузить также большие объекты	false	true, false
no_blobs	Исключить из выгрузки большие объекты	false	true, false
no_publications	Не выгружать публикации	false	true, false
no_security_labels	Не выгружать назначения меток безопасности	false	true, false
no_subscriptions	Не выгружать подписки	false	true, false
no_tablespaces	Не выгружать назначения табличных пространств	false	true, false
no_unlogged_table_data	Не выгружать данные нежурналируемых таблиц	false	true, false

6. Нажмите на кнопку «**Применить**» в правом-верхнем углу для завершения настройки и создания правила.

Вновь созданное правило будет иметь статус wait. Это означает, что оно не будет порождать задач на выполнение резервного копирования, пока администратор RuBackup не запустит его (тогда его статус сменится на run). При необходимости, администратор может приостановить работу правила или немедленно запустить его (т. е. инициировать немедленное создание задачи при статусе правила wait).

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 определённый срок, по окончании которого удалять их из хранилища резервных копий и из записей репозитория, либо уведомлять клиента об окончании срока хранения;

– через определённый срок после создания резервной копии автоматически переместить её 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.

При создании задачи RuBackup она появляется в главной очереди задач. Отслеживать выполнение правил может как администратор (при помощи RBM или утилит командной строки), так и клиент (при помощи RBC или 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.

Менеджер Клиента RuBackup (RBC)

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

Графический интерфейс RBC поддерживает русский и английский языки.

Для запуска RBC следует выполнить команды:

```
# ssh -X user@postgresql-host
# /opt/rubackup/bin/rbc&
```

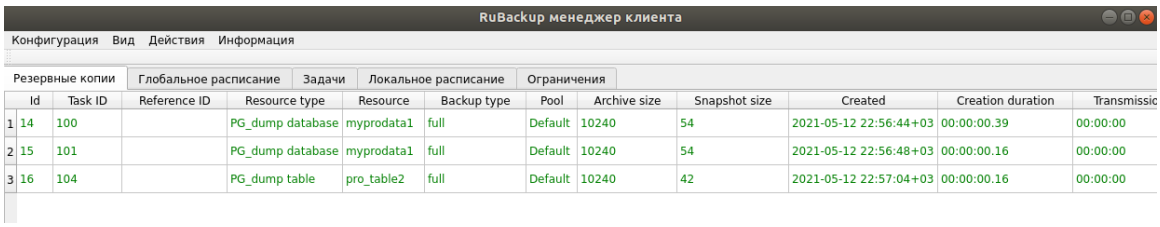
Пользователь, запускающий RBC, должен входить в группу rubackup.

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

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

Вкладка «Резервные копии»

Вкладка «Резервные копии» содержит таблицу с информацией обо всех резервных копиях клиента, которые хранятся в репозитории RuBackup (рисунок 9):



	Id	Task ID	Reference ID	Resource type	Resource	Backup type	Pool	Archive size	Snapshot size	Created	Creation duration	Transmissio
1	14	100		PG_dump database	myprodata1	full	Default	10240	54	2021-05-12 22:56:44+03	00:00:00.39	00:00:00
2	15	101		PG_dump database	myprodata1	full	Default	10240	54	2021-05-12 22:56:48+03	00:00:00.16	00:00:00
3	16	104		PG_dump table	pro_table2	full	Default	10240	42	2021-05-12 22:57:04+03	00:00:00.16	00:00:00

Рисунок 9

На этой вкладке клиенту доступны следующие действия:

– **Удалить выбранную резервную копию.** Это действие возможно в том случае, если в правиле глобального расписания есть соответствующее разрешение. При удалении резервной копии потребуется вести пароль клиента.

– **Восстановить резервную копию.** Это действие запускает процесс восстановления резервной копии на локальной файловой системе клиента. При восстановлении резервной копии клиент должен выбрать место для восстановления файлов резервной копии. Рекомендуется использовать временный каталог для операций с резервными копиями (например, /rubackup-tmp).

Если при восстановлении PK базы данных было выбрано восстановление с развертыванием (without_deployment_restore=no), то произойдёт восстановление таблиц, которые были в БД на момент создания PK. Восстановление происходит в базу данных с таким же названием, как и у копируемой БД.

Если при восстановлении PK отдельной таблицы было выбрано восстановление с развертыванием (without_deployment_restore=no), то произойдёт восстановление в базу данных с таким же названием, как и у той, к которой принадлежала таблица на момент выполнения резервного копирования.

Внимание! Если при восстановлении обнаружится, что хотя бы одна таблица, с таким же именем, как и у восстанавливаемой, присутствует в целевой БД, то модуль создаст новую базу данных с суффиксом «_rbcopy_[index]» и восстановит таблицы в неё.

RBC не ожидает окончания восстановления всех резервных копий. На вкладке «задачи» клиент должен проконтролировать, что созданные задачи на восстановление данных завершились успешно (статус done). Для успешного выполнения этого действия требуется наличие достаточного свободного места в каталоге, предназначенном для создания и временного хранения резервных копий (см. параметр use-local-backup-directory).

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

Вкладка «Глобальное расписание»

Вкладка «Глобальное расписание» содержит таблицу с информацией обо всех правилах в глобальном расписании RuBackup для этого клиента (рисунок 10):

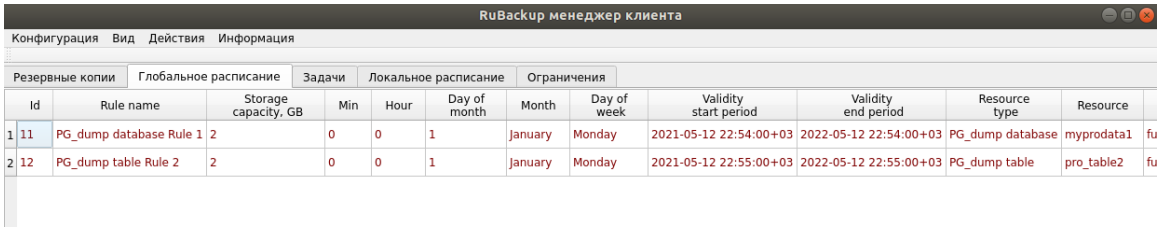


Рисунок 10

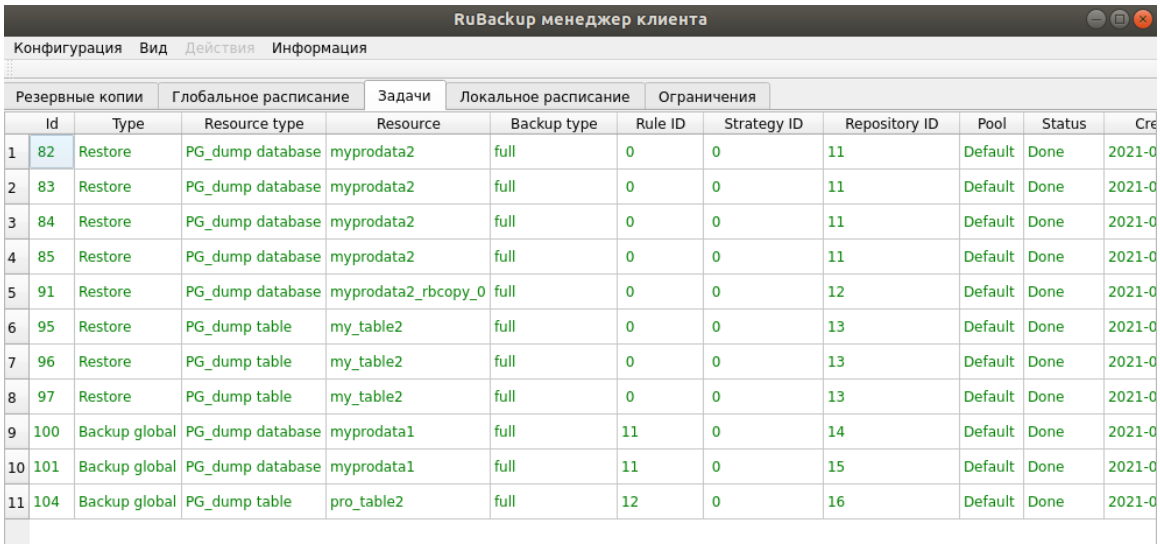
На этой вкладке клиенту доступны следующие действия:

– **Запросить новое правило.** Это действие вызывает диалог подготовки нового правила в глобальном расписании RuBackup для клиента. Запрос на добавление правила требует одобрения администратора RuBackup, одобрение может быть сделано в RBM.

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

Вкладка «Задачи»

Вкладка «Задачи» содержит таблицу информацией обо всех задачах в главной очереди заданий RuBackup для этого клиента (рисунок 11):



	Id	Type	Resource type	Resource	Backup type	Rule ID	Strategy ID	Repository ID	Pool	Status	Cre
1	82	Restore	PG_dump database	myprodata2	full	0	0	11	Default	Done	2021-0
2	83	Restore	PG_dump database	myprodata2	full	0	0	11	Default	Done	2021-0
3	84	Restore	PG_dump database	myprodata2	full	0	0	11	Default	Done	2021-0
4	85	Restore	PG_dump database	myprodata2	full	0	0	11	Default	Done	2021-0
5	91	Restore	PG_dump database	myprodata2_rbcopy_0	full	0	0	12	Default	Done	2021-0
6	95	Restore	PG_dump table	my_table2	full	0	0	13	Default	Done	2021-0
7	96	Restore	PG_dump table	my_table2	full	0	0	13	Default	Done	2021-0
8	97	Restore	PG_dump table	my_table2	full	0	0	13	Default	Done	2021-0
9	100	Backup global	PG_dump database	myprodata1	full	11	0	14	Default	Done	2021-0
10	101	Backup global	PG_dump database	myprodata1	full	11	0	15	Default	Done	2021-0
11	104	Backup global	PG_dump table	pro_table2	full	12	0	16	Default	Done	2021-0

Рисунок 11

В зависимости от настроек сервера RuBackup выполненные задачи и задачи, завершившиеся неудачно, через какое-то время могут быть автоматически удалены из главной очереди задач. Информация о выполнении заданий фиксируется в специальном журнале задач сервера RuBackup. При необходимости статус любой задачи, даже удалённой из очереди, можно уточнить у администратора RuBackup. Также информация о выполнении задач клиента заносится в локальный файл журнала на хосте клиента. В RBC можно открыть окно отслеживания журнального файла (меню «**Информация**» → «**Журнальный файл**»).

Примечание – Информация о выполнении служебных задач в данной вкладке не отображается. Служебными являются задачи проверки, удаления, перемещения резервных копий, а также их копирования в другой пул.

Вкладка «Локальное расписание»

На вкладке «Локальное расписание» можно определить правила, задаваемые клиентом для каких-либо локальных ресурсов. Для работы локального расписания эта возможность должна быть включена для клиента администратором RuBackup.

Вкладка «Ограничения»

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

Утилиты командной строки клиента

RuBackup

Для управления RuBackup со стороны клиента, помимо RBC, можно использовать утилиты командной строки. Пользователь, запускающий утилиты командной строки, должен входить в группу rubackup.

Подробнее ознакомиться с функциями утилит командной строки можно при помощи команды man и в руководстве «Утилиты командной строки RuBackup».

rb_archives

Эта утилита предназначена для просмотра списка резервных копий клиента в системе резервного копирования, создания срочных резервных копий, их удаления, проверки и восстановления. Ниже представлен пример.

 # rb_archives

```
root@postgresPro-client:~# rb_archives
Id | Ref ID | Resource   | Resource type    | Backup type | Created               | Crypto  | Signed | Status
---+--------+------------+------------------+-------------+-----------------------+---------+--------+--------------------
14 |        | myprodata1 | PG_dump database | full        | 2021-05-12 22:56:44+03 | nocrypt | True   | Trusted
15 |        | myprodata1 | PG_dump database | full        | 2021-05-12 22:56:48+03 | nocrypt | True   | Trusted
16 |        | pro_table2 | PG_dump table    | full        | 2021-05-12 22:57:04+03 | nocrypt | True   | Trusted
```

rb_schedule

Эта утилита предназначена для просмотра имеющихся правил клиента в глобальном расписании резервного копирования. Ниже представлен пример.

#rb_schedule

```
root@postgresPro-client:~# rb_schedule
Id | Name                     | Resource type    | Resource   | Backup type | Status
---+--------------------------+------------------+------------+-------------+-------
11 | PG_dump database Rule 1  | PG_dump database | myprodata1 | full        | wait
12 | PG_dump table Rule 2     | PG_dump table    | pro_table2 | full        | wait
```

rb_tasks

```
root@postgresPro-client:~# rb_tasks
Id  | Task type     | Resource             | Backup type | Status
----+---------------+----------------------+-------------+--------
82  | Restore       | myprodata2           | full        | Done
83  | Restore       | myprodata2           | full        | Done
84  | Restore       | myprodata2           | full        | Done
85  | Restore       | myprodata2           | full        | Done
91  | Restore       | myprodata2_rbcopy_0  | full        | Done
95  | Restore       | my_table2            | full        | Done
96  | Restore       | my_table2            | full        | Done
97  | Restore       | my_table2            | full        | Done
100 | Backup global | myprodata1           | full        | Done
101 | Backup global | myprodata1           | full        | Done
104 | Backup global | pro_table2           | full        | Done
```

Восстановление резервной копии отдельной БД или таблицы

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

Внимание! Если резервная копия была выполнена в формате plain text, то модуль не сможет выполнить восстановление в автономном режиме. В таком формате возможно только ручное восстановление PK. Для этого сначала необходимо восстановить PK без развертывания (without_deployment_restore=yes), а затем воспользоваться консольной утилитой psql.

Внимание! Для успешного восстановления содержимого существующей базы данных необходимо завершить все активные соединения с ней.

Восстановление резервной копии в RBC

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

1. Выделите нужную резервную копию и в контекстном меню выберите **Восстановить**.
2. Для восстановления потребуется ввести пароль клиента. Затем RBC выведет информационное сообщение о дальнейших действиях.
3. Укажите в качестве временного места восстановления резервных копий любой каталог (например, /rubackup-tmp).
4. RBC выведет информационное сообщение о создании задачи на восстановление.

Для контроля процесса восстановления RBC автоматически переключится на вкладку «Задачи», в которой можно проконтролировать результат (рисунок 12):

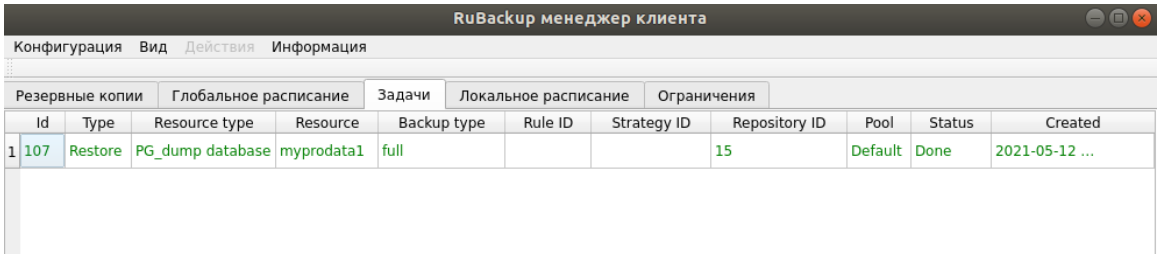


Рисунок 12

Восстановление при помощи утилиты rb_archives

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

 # rb_archives

```
root@postgresPro-client:~# rb_archives
Id | Ref ID | Resource   | Resource type    | Backup type | Created               | Crypto  | Signed | Status
---+--------+------------+------------------+-------------+-----------------------+---------+--------+--------------------
14 |        | myprodata1 | PG_dump database | full        | 2021-05-12 22:56:44+03 | nocrypt | True   | Trusted
15 |        | myprodata1 | PG_dump database | full        | 2021-05-12 22:56:48+03 | nocrypt | True   | Trusted
16 |        | pro_table2 | PG_dump table    | full        | 2021-05-12 22:57:04+03 | nocrypt | True   | Trusted
```

В приведённом примере в системе резервного копирования присутствуют три резервные копии с идентификаторами 14, 15 и 16. Для восстановления резервной копии 15 необходимо выполнить команду:

 # rb_archives -x 15

```
root@postgresPro-client:~# rb_archives -x 15
Password:
----> Restore archive chain: 15  < ----
Record ID: 15 has status: Trusted
TASK WAS ADDED TO QUEUE:108
```

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

Проконтролировать процесс восстановления можно при помощи утилиты rb_tasks:

#rb_tasks

```
root@postgresPro-client:~# rb_tasks
Id  | Task type | Resource   | Backup type | Status | Created
----+-----------+------------+-------------+--------+----------------------
107 | Restore   | myprodata1 | full        | Done   | 2021-05-12 23:30:26+03
108 | Restore   | myprodata1 | full        | Done   | 2021-05-12 23:32:06+03
```

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

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

```
root@postgresPro-client:~# tail -f /opt/rubackup/log/RuBackup.log
Fri May 14 17:52:32 2021: [RBC] Request to restore next archive(s) ID from repository: 15  to: /root
Fri May 14 17:52:33 2021: RuBackup server commands: Run task ID: 108 Resource type: 31 Module: PG_dump database Resource: myprodata1 Media server: ruba
ckup-server.rubackup.local
Fri May 14 17:52:33 2021: Set unlimited bandwidth for task ID: 108
Fri May 14 17:52:33 2021: Create a file: /root/postgresPro-client_TaskID_101_RuleID_11_D2021_5_14H17_17_14_BackupType_1_ResourceType_31.tar
Fri May 14 17:52:34 2021: md5sum of transferred file is ok: a73b6123a93a6155bdaaf7309a675690
Fri May 14 17:52:34 2021: Transfer file is succesed: /root/postgresPro-client_TaskID_101_RuleID_11_D2021_5_14H17_17_14_BackupType_1_ResourceType_31.tar
Fri May 14 17:52:34 2021: Execute OS command: /opt/rubackup/modules/rb_module_pg_dump_database -r /root/postgresPro-client_TaskID_101_RuleID_11_D2021_5
_14H17_17_14_BackupType_1_ResourceType_31.tar -z 1 -e last:true,tmp_catalog:/rubackup1,rbd_hash_algorithm:sha,rbd_hash_length:512,rbd_block_size:16384,
granular_restore:no,without_deployment_restore:no,threads:1,serializable_deferrable:t,no_owner:f,no_sync:f,data_only:f,schema_only:f,blobs:f,no_blobs:f
,no_publications:f,no_security_labels:f,no_subscriptions:f,no_synchronized_snapshots:f,no_tablespaces:f,no_unlogged_table_data:f -d /root 2>&1
Fri May 14 17:52:34 2021: postgresPro-client_TaskID_101_RuleID_11_D2021_5_14H17_17_14_BackupType_1_ResourceType_31.dump
Fri May 14 17:52:34 2021: postgresPro-client_TaskID_101_RuleID_11_D2021_5_14H17_17_14_BackupType_1_ResourceType_31.snap
Fri May 14 17:52:34 2021: Task was done. ID: 108
```