



RuBackup

Система резервного копирования
и восстановления данных

ОТДЕЛЬНЫЕ ТАБЛИЦЫ И БД POSTGRESQL

ВЕРСИЯ 3.0.0.0.0

Содержание

1. Подготовка клиента	3
1.1. Установка клиента RuBackup	3
1.2. Установка пакета модулей резервного копирования	3
1.3. Обновление конфигурационного файла	3
1.3.1. Автоматическое обновление конфигурационного файла	4
1.4. Удаление клиента RuBackup	5
2. Настройка PostgreSQL	6
2.1. Подготовка к использованию модулей резервного копирования	6
2.1.1. Создание роли RuBackup	6
2.1.2. Создание специальной базы данных	6
2.1.3. Настройка конфигурации сервера PostgreSQL	7
2.1.4. Включение расширения dblink для роли RuBackup	8
2.2. Управление правами роли RuBackup	8
2.3. Завершение подготовки хоста с СУБД PostgreSQL	10
3. Режим базового резервного копирования при помощи pg_dump	12
4. Резервное копирование	13
4.1. Резервное копирование в RBM	13
4.1.1. Резервное копирование БД	13
4.1.2. Резервное копирование таблиц БД	13
4.2. Резервное копирование из командной строки	13
4.2.1. Резервное копирование БД	13
4.2.2. Резервное копирование таблиц БД	14
Приложение А: Тонкие настройки для резервного копирования	14
5. Восстановление резервной копии	18
6. Мастер-ключ	20
7. Менеджер Клиента RuBackup (RBC)	21
7.1. Вкладка «Резервные копии»	22
7.2. Вкладка «Глобальное расписание»	23
7.3. Вкладка «Задачи»	23
7.4. Вкладка «Локальное расписание»	24
7.5. Вкладка «Ограничения»	24
8. Восстановление резервной копии со стороны клиента	25
8.1. Восстановление резервной копии в RBC	25
8.2. Восстановление при помощи утилиты rb_archives	26

Система резервного копирования (СРК) RuBackup позволяет осуществлять резервное копирование и восстановление таблиц и баз данных СУБД PostgreSQL. Модули резервного копирования поддерживают СУБД PostgreSQL версий 11, 12, 13, 14, 15, 16.

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.

Глава 1. Подготовка клиента

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

- `rubackup-client.deb` — клиент резервного копирования;
- `rubackup-pg-dump.deb` — модуль резервного копирования.

1.1. 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».

1.2. 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 следующую команду

```
sudo dpkg -i rubackup-pg-dump.deb
```

1.3. 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`.

1.3.1. 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 воз-

никнет 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/rubakup/etc/rb_module_pg_dump_database.conf` и при необходимости заполнить параметры вручную.

1.4. 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 `rubakup-client` используйте команды:

```
systemctl disable rubakup-client
systemctl daemon-reload
```

Для удаления клиента RuBackup и модуля `rubakup-pg-dump` используйте команды:

```
apt remove rubakup-pg-dump
apt remove rubakup-client
```

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

Глава 2. Настройка PostgreSQL

2.1. Подготовка к использованию модулей резервного копирования

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

1. Убедитесь, что утилита `psql` располагается в каталоге `/usr/bin`. Если `psql` располагается в другом каталоге, то создайте в каталоге `/usr/bin` символическую ссылку на утилиту:

```
ln -s /<путь_до_утилиты>/psql /usr/bin/psql
```

2. Создайте роль RuBackup (например, `rubackup_backuper`) (см. [Создание роли RuBackup](#)).
3. Создайте базу данных для выполнения процедур резервного копирования (например, `backupdb`) (см. [Создание специальной базы данных](#)).
4. Выполните настройку конфигурации сервера PostgreSQL (см. [Настройка конфигурации сервера PostgreSQL](#)).
5. Включите расширение `dblink` для роли RuBackup (см. [Включение расширения dblink для роли RuBackup](#)).

2.1.1. Создание роли RuBackup

Для безопасного выполнения операций с базами данных и таблицами необходимо создать роль с ограниченным набором прав.

1. Подключитесь к СУБД от имени администратора:

```
sudo -u postgres psql
```

2. Создайте роль RuBackup:

```
CREATE ROLE rubackup_backuper WITH LOGIN password '12345'; ❶
```

❶ Вместо `12345` укажите желаемый пароль.

2.1.2. Создание специальной базы данных

Для того, чтобы у ранее созданного пользователя была точка входа, создайте

отдельную базу данных.

1. Подключитесь к СУБД от имени администратора:

```
sudo -u postgres psql
```

2. Создайте базу данных:

```
CREATE DATABASE backupdb;
```

3. Назначьте ранее созданную роль владельцем базы данных:

```
ALTER DATABASE backupdb OWNER TO rubakup_backuper;
```

2.1.3. Настройка конфигурации сервера PostgreSQL

Для подготовки СУБД PostgreSQL к выполнению резервного копирования при помощи CPK RuBackup:

1. Установите метод подключения `peer` для пользователя `postgres` в файле `pg_hba.conf`:

```
local    all             postgres                    peer
```

2. Установите для роли RuBackup метод подключения `md5` в файле `pg_hba.conf`. Данный метод должен распространяться только на ранее созданную базу данных:

```
host     backupdb        rubakup_backuper    127.0.0.1/32    md5
```

3. Чтобы изменения вступили в силу без перезагрузки сервера:
 - a. Проверьте файл на наличие опечаток:

```
psql -c 'SELECT * from pg_hba_file_rules;'
```

- b. Примените изменения:

```
psql -c ' SELECT pg_reload_conf();'
```

2.1.4. Включение расширения dblink для роли RuBackup

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

```
sudo -u postgres psql -d backupdb -c 'CREATE EXTENSION dblink;'
```

2.2. Управление правами роли RuBackup

Для управления правами роли RuBackup используйте следующие команды:

1. Перед запуском процесса восстановления резервной копии предоставьте пользователю `rubackup_backuper` права `superuser`:

```
ALTER USER rubackup_backuper WITH superuser;
```

2. По завершении восстановления отзовите выданные ранее пользователю `rubackup_backuper` права `superuser`:

```
ALTER USER rubackup_backuper WITH nosuperuser;
```

3. Для выполнения резервного копирования БД, содержащей большие объекты, в файле `postgresql.conf` установите значение `on` параметру `lo_compat_privileges`:

```
lo_compat_privileges = on
```

4. 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`

Чтобы управлять правами роли в ручном режиме используйте утилиту `/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
```

-1

выдает права ранее созданному пользователю на выполнение команд `SELECT` и `CREATE` в целевой БД;

-2

снимает права ранее созданному пользователю;

-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.

Чтобы отозвать у роли ранее выданные права, 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
```

2.3. 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 параметры файла `/opt/rubackup/etc/rb_module_pg_dump_database.conf`:

Пример заполнения файла `/opt/rubackup/etc/rb_module_pg_dump_database.conf`

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

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`.

2. Для резервного копирования отдельной таблицы заполните параметры файла `/opt/rubackup/etc/rb_module_pg_dump_table.conf`:

Пример заполнения файла `/opt/rubackup/etc/rb_module_pg_dump_table.conf`

```
backup_user: 'rubackup_backuper'
```

```
password: '12345'
backup_db: 'backupdb'
host: 'localhost'
port: '5432'
pg_dump: '/usr/bin/pg_dump'
pg_restore: '/usr/bin/pg_restore'
```

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`.



Во избежание раскрытия пароля роли RuBackup для конфигурационных файлов рекомендуем определить строгие права только для суперпользователя (`chmod 0600`).

3. Перезапустите 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.

Глава 3. Режим 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
```

`--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
```

`--dbname`

база данных, которая содержит в себе копируемую таблицу;

`--table`

имя копируемой таблицы.

Глава 4. Резервное копирование

4.1. Резервное копирование в RBM

Выберите способ выполнения резервного копирования в приложении [Менеджер администратора RuBackup \(RBM\)](#) и произведите настройку, следуя указаниям из соответствующего документа:

- [Срочное резервное копирование](#)
- [Глобальное расписание](#)
- [Стратегии](#)

4.1.1. Резервное копирование БД

Для резервного копирования отдельной БД:

1. Из списка **Клиент** выберите клиента, который установлен на узел с СУБД.
2. Из списка **Тип ресурса** выберите `PG_dump database`.
3. Из списка **Ресурс** выберите базу данных.

При необходимости нажмите **[...]** и определите тонкие настройки модуля (см. [Тонкие настройки для резервного копирования](#)).

4. Из списка **Тип РК** выберите тип резервной копии `Полная`.

4.1.2. Резервное копирование таблиц БД

Для резервного копирования отдельной таблицы БД:

1. Из списка **Клиент** выберите клиента, который установлен на узел с СУБД.
2. Из списка **Тип ресурса** выберите `PG_dump table`.
3. Из списка **Ресурс** выберите таблицу базы данных.

При необходимости нажмите **[...]** и определите тонкие настройки модуля (см. [Тонкие настройки для резервного копирования](#)).

4. Из списка **Тип РК** выберите тип резервной копии `Полная`.

4.2. Резервное копирование из командной строки

4.2.1. Резервное копирование БД

Для выполнения резервного копирования отдельной БД на клиенте РК:

1. Получите список ресурсов (БД):

Команда получения списка ресурсов

```
rb_archives -t pg_dump_database
```

2. Выполните запрос на создание полной резервной копии:

Пример создания полной резервной копии

```
rb_archives \  
  -c <db_name> \ ❶  
  -m pg_dump_database ❷
```

❶ Имя резервируемого ресурса

❷ Используемый модуль

4.2.2. Резервное копирование таблиц БД

Для выполнения резервного копирования отдельной таблицы БД на клиенте РК:

1. Получите список ресурсов (таблиц БД):

Команда получения списка ресурсов

```
rb_archives -t pg_dump_table
```

2. Выполните запрос на создание полной резервной копии:

Пример создания полной резервной копии

```
rb_archives \  
  -c <db_name>:<table_name> \ ❶  
  -m pg_dump_table ❷
```

❶ Имя резервируемого ресурса

❷ Используемый модуль

Подробнее об утилите `rb_archives` читайте [здесь](#).

Приложение А: Тонкие настройки для резервного копирования

В [таблице](#) описаны тонкие настройки модуля для резервного копирования (см.

Резервное копирование в RBM).



Подробную информацию о параметрах смотрите в [официальной документации PostgreSQL](#).

Таблица 1. Тонкие настройки модуля для резервного копирования

Параметр	Описание
threads	Количество потоков выполнения резервной копии. По умолчанию 1
dump_format	Формат, в котором данные сохраняются в РК. Возможные значения custom, plain-text По умолчанию custom При значении: - custom данные будут сохраняться в специальном архивном формате; - plain-text данные будут сохраняться в текстовом SQL-скрипте.
serializable_deferrable	Создать РК в момент, когда данные БД логически согласованы. Возможные значения true, false По умолчанию true
no_owner	Сформировать команды, устанавливающие владельца объектов БД. Возможные значения true, false По умолчанию false Используется, если параметру dump_format установлено значение plain-text. При значении: - true команды, устанавливающие владельца объектов БД, не будут формироваться; - false команды, устанавливающие владельца объектов БД, будут формироваться.

Параметр	Описание
no_sync	Дождаться окончания записи данных БД на диск.
	Возможные значения true, false
	По умолчанию false
no_sync	При значении:
	• true создание РК ускоряется. Не гарантируется полноценная запись данных на диск; • false ожидается завершение записи данных на диск.
data_only	Сохранить в РК данные без схем объектов БД.
	Возможные значения true, false
	По умолчанию false
schema_only	Сохранить в РК схемы объектов БД без данных.
	Возможные значения true, false
	По умолчанию false
blobs	Сохранить в РК большие объекты БД.
	Возможные значения true, false
	По умолчанию false
no_publications	Сохранить в РК публикации БД.
	Возможные значения true, false
	По умолчанию false
no_publications	При значении:
	• true публикации БД не попадают в РК; • false публикации БД попадают в РК.
no_security_labels	Сохранить в РК метки безопасности БД.
	Возможные значения true, false
	По умолчанию false
no_security_labels	При значении:
	• true метки безопасности БД не попадают в РК; • false метки безопасности БД попадают в РК.

Параметр	Описание
no_subscriptions	Сохранить в РК подписки БД.
	Возможные значения true, false
	По умолчанию false
	При значении: • true подписки БД не попадают в РК; • false подписки БД попадают в РК.
no_tablespaces	Сформировать команды для указания табличных пространств БД.
	Возможные значения true, false
	По умолчанию false
	 Используется, если параметру dump_format установлено значение plain-text.
	При значении: • true команды для указания табличных пространств БД не будут формироваться; • false команды для указания табличных пространств БД будут формироваться.
no_unlogged_table_data	Сохранить в РК данные нежурналируемых таблиц БД и последовательностей.
	Возможные значения true, false
	По умолчанию false
	При значении: • true данные нежурналируемых таблиц БД и последовательностей не попадают в РК; • false данные нежурналируемых таблиц БД и последовательностей попадают в РК.

Глава 5. 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
```



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



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.

Глава 6. 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 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
00000000 79d1 4749 7335 e387 9f74 c67e 55a7 20ff
00000010 6284 54as 83a3 2053 4818 e183 1528 a343
00000020
```

Глава 7. 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.

7.1. Вкладка «Резервные копии»

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

	Id	Task ID	Reference ID	Resource type	Resource	Backup type	Pool	Archive size	Snapshot size	Created	Creation duration	Transmissi
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 1.

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

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



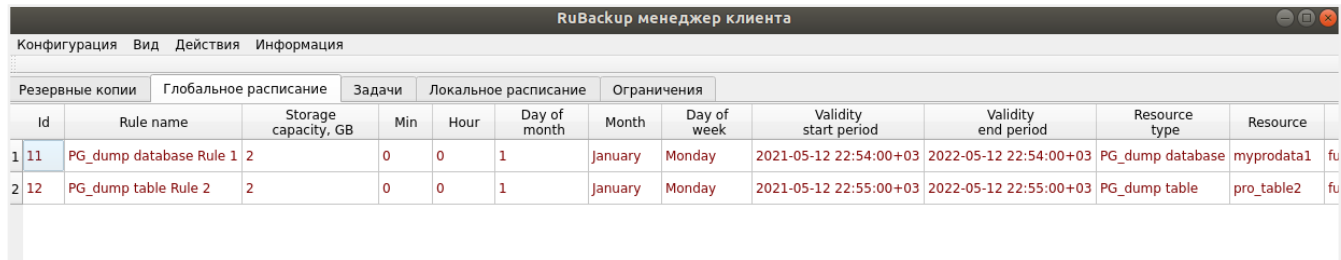
Если при восстановлении обнаружится, что хотя бы одна таблица, с таким же именем, как и у восстанавливаемой, присутствует в целевой БД, то модуль создаст новую базу данных с суффиксом `rbcopy_[index]` и восстановит таблицы в неё.

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. `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.

7.2. Вкладка «Глобальное расписание»

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



RuBackup менеджер клиента												
Конфигурация Вид Действия Информация												
Резервные копии			Глобальное расписание		Задачи		Локальное расписание		Ограничения			
Id	Rule name	Storage capacity, GB	Min	Hour	Day of month	Month	Day of week	Validity start period	Validity end period	Resource type	Resource	
1 11	PG_dump database Rule 1	2	0	0	1	January	Monday	2021-05-12 22:54:00+03	2022-05-12 22:54:00+03	PG_dump database	myprodata1	fu
2 12	PG_dump table Rule 2	2	0	0	1	January	Monday	2021-05-12 22:55:00+03	2022-05-12 22:55:00+03	PG_dump table	pro_table2	fu

Рисунок 2.

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.

7.3. Вкладка «Задачи»

RuBackup менеджер клиента											
Конфигурация Вид Действия Информация											
Резервные копии			Глобальное расписание		Задачи	Локальное расписание		Ограничения			
	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 3.

Вкладка **Задачи** содержит таблицу информацией обо всех задачах в главной очереди заданий RuBackup для этого клиента ().

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**).



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.

7.4. 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.

7.5. 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.

Глава 8. Восстановление резервной копии со стороны клиента

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



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 РК. Для этого сначала необходимо восстановить РК без развертывания (`without_deployment_restore=yes`), а затем воспользоваться консольной утилитой `psql`.



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

8.1. 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 ():

RuBackup менеджер клиента										
Конфигурация Вид Действия Информация										
Резервные копии Глобальное расписание Задачи Локальное расписание Ограничения										
Id	Type	Resource type	Resource	Backup type	Rule ID	Strategy ID	Repository ID	Pool	Status	Created
1 107	Restore	PG_dump database	myprodata1	full			15	Default	Done	2021-05-12 ...

Рисунок 4.

8.2. 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 succeeded: /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_tablespace: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
```