RuBackup
Система резервного копирования и восстановления данных

Руководство по установке и взаимодействию с программным интерфейсом RuBackup REST API

RuBackup

Версия 2.0

31.05. 2023 г.

Содержание

Введение

Система резервного копирования и восстановления данных RuBackup предоставляет пользователю возможность взаимодействия с активным сервером резервного копирования посредством HTTP-запросов к его ресурсам.

RuBackup REST API разработан в соответствии со следующими руководящими принципами написания RESTful-интерфейсов:

1. Client-Server. Хранилище данных отделено от клиента, а информация из него предоставляется только при помощи разрешённых HTTP-методов для конкретного ресурса. Клиент может взаимодействовать со строго определёнными URI ресурсов.

2. Uniform interface (единообразие интерфейса). Архитектура интерфейса должна быть единообразной. Ресурс в системе должен иметь только один логический URI, что обеспечивает явный способ извлечения связанных или дополнительных данных.

3. Stateless (без состояния). Все запросы от клиента к серверу должны содержать внутри себя всю необходимую информацию и не учитывать какое-либо состояние, хранящееся на стороне сервера. Таким образом, сервер не должен хранить ничего о последнем HTTP-запросе, сделанном клиентом и обрабатывать каждый запрос как новый.

4. Layered system (многоуровневая система). Системная архитектура RESTful-интерфейса должна быть многоуровневой, чтобы обеспечить независимое и безопасное взаимодействие клиента и сервера. Клиент не должен «понимать» подключён он к серверу или нет, а также, влиять на другие компоненты системы.

5. Cacheable (кэшируемость). Если ресурс, к которому обратился клиент позволяет производить кэширование своих данных, то клиент сохранит полученную от сервера информацию на определённый срок. При следующем обращении данные уже будут готовы к выдаче, что ускоряет взаимодействие между клиентом и сервером.

Программный интерфейс RuBackup реализован и документирован с использованием набора инструментов Swagger. Для описания RESTful API Swagger использует формат JSON. Swagger используется вместе с набором программных средств с открытым исходным кодом для проектирования, создания, документирования и использования веб-служб RESTful.

Настоящее руководство описывает базовые шаги установки, настройки и эксплуатации RuBackup REST API. Руководство предназначено для системных администраторов, отвечающих за сопровождение СРК .

Перед установкой RuBackup REST API

Базовые требования

Перед установкой REST API убедитесь, что вы уже установили и настроили сервер RuBackup. Также, убедитесь в том, что база данных rubackup инициализирована в системе и функционирует в штатном режиме.

Установка сервера и клиента системы резервного копирования RuBackup описана в «Руководстве по установке серверов резервного копирования и Linux клиентов».

Также, проверьте наличие файлов-сертификатов RuBackup, которые используются программным интерфейсом для создания защищённого соединения. Расположение сертификатов в файловой системе:

1. /opt/rubackup/keys/server/serverCert.crt
2. /opt/rubackup/keys/server/serverKey.key

Особенности установки пакетов в Linux

Дистрибутив RuBackup REST API поставляется в виде deb и rpm пакетов. Для разных дистрибутивов Linux, по причине их отличий друг от друга, предусмотрены специально подготовленные пакеты RuBackup.

В зависимости от типа используемого пакетного менеджера в вашем дистрибутиве Linux, процедура установки и удаления пакетов может использовать команды `dpkg`, `rpm`, `apt`, `yum` и пр. В настоящем руководстве процедуры установки описаны для пакетного менеджера, который оперирует пакетами deb. Например, процедура установки пакета в операционной системе Ubuntu 20.04 выглядит следующим образом:

```
$ sudo dpkg -i rubackup-rest-api.deb
```

Для установки API в ОС с пакетным менеджером, который оперирует rpm пакетами, вместо вышеуказанной команды следует выполнить команду:

```
$ sudo rpm -i rubackup-rest-api.rpm
```

Процедуры удаления пакетов в настоящем руководстве описаны для пакетного менеджера, который оперирует пакетами deb. Например, процедура удаления пакета RuBackup API выглядит следующим образом:

```
$ sudo apt remove rubackup-rest-api
```

Для удаления RuBackup API в операционной системе с пакетным менеджером, который оперирует rpm пакетами, вместо вышеуказанной команды следует выполнить:

```
$ sudo yum remove rubackup-rest-api
```

либо:

```
$ sudo rpm -e rubackup-rest-api
```

Некоторые операционные системы, такие как Alt Linux, используют пакетную систему rpm, но вместо `yum` используют `apt`. Перед установкой или удалением пакета RuBackup REST API следует уточнить, какие команды необходимо использовать для вашего дистрибутива Linux.

Установка RuBackup REST API

Для инсталляции RuBackup API установите пакет **rubackup-rest-api:**

```
$ sudo dpkg -i rubackup-rest-api.deb
```

Имя файла пакета может отличаться в зависимости от сборки.

После установки пакета вы можете сразу запустить API если у вас уже определён статический адрес для хоста и он явно указан в файле /**etc**/**hosts.** Если это не так, то перейдите к ручной настройке API.

Настройка RuBackup REST API

Настройка RuBackup API осуществляется пользователем при помощи конфигурационного файла *rubackup_api.conf*, расположенного по следующему пути:

/opt/rubackup/etc/rubackup_api.conf

Структура конфигурационного файла выглядит следующим образом:

```
# Конфигурационный файл заполняется пользователем вручную

DEBUG: False

# FQDN/IP адрес хоста
app_host: '192.168.10.10'
# Порт хоста
app_port: '443'

# FQDN/IP адрес сервера Postgres с базой данных rubackup
database_host: 'localhost'
# Порт сервера Postgres
database_port: '5432'
# Использование SSL при подключении к БД
database_ssl_mode: 'require'
```

Заполните конфигурационный файл актуальной информацией, не меняя его структуру и не удаляя одинарные кавычки.

Важно! Если вы удалили конфигурационный файл, то вы можете либо создать его вручную, но при этом соблюдая его структуру и используя программные средства для создания YAML-файлов, либо переустановите пакет.

Настройка запуска RuBackup REST API

Для штатной эксплуатации RuBackup API запустите интерфейс как сервис. Для этого выполните следующие действия:

1. Включите сервис RuBackup API:

```
$ sudo systemctl enable \
  /opt/rubackup/etc/systemd/system/rubackup_api.service
```

2. Перезагрузите `systemctl`:

```
$ sudo systemctl daemon-reload
```

3. Запустите сервис `rubackup_client`:

```
$ sudo systemctl start rubackup_api
```

Уточнить статус RuBackup API можно при помощи команды:

```
$ sudo systemctl status rubackup_api
```

```
● rubackup_api.service - RuBackup API
     Loaded: loaded (/etc/systemd/system/rubackup_api.service; enabled;
vendor preset: enabled)
     Active: active (running) since Sat 2022-10-01 02:14:15 MSK; 7s ago
   Main PID: 4109 (rubackup_api)
      Tasks: 2 (limit: 4626)
     Memory: 122.3M
     CGroup: /system.slice/rubackup_api.service
             ├─4109 /opt/rubackup/bin/rubackup_api start
             └─4111 /opt/rubackup/bin/rubackup_api start

окт 01 02:14:15 rubackup-server systemd[1]: Started RuBackup API.
окт 01 02:14:16 rubackup-server rubackup_api[4111]: The current instant
is powered by "sqlite://'memory'", this will change after authentication
окт 01 02:14:17 rubackup-server rubackup_api[4111]: Starting RuBackup
API, available at https://192.168.10.10:443
```

Использование RuBackup REST API

Этот раздел описывает процесс аутентификации, авторизации и взаимодействия пользователя с программным интерфейсом RuBackup.

Аутентификация пользователя

Перед тем как пользователь сможет обратиться к ресурсам сервера RuBackup, он должен пройти аутентификацию и получить токены доступа: *access_token* и *refresh_token*.

Выпуск AcessToken и RefreshToken

Необходимо отправить POST-запрос с помощью консольной утилиты *curl* или любым другим удобным способом. В данном примере используется *curl*:

```
$ curl -k -X POST https://192.168.10.10:443/api/v0.1/login \
-H 'Content-Type: application/json' \
-d '{"username":"rubackup","password":"secret"}'
```

Результат вернётся в формате JSON:


```
{
    "data": {
        "access_token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJmcmVzaCI6ZmFsc2UsImlhdCI6MTY4NTA5MDYxNiwianRpIjoiNjhkNzI5OGQtNDIyZi00ZjE5LWExNGUtNmQ0MDg5NzZmZGY5IiwidHlwZSI6ImFjY2VzcyIsInN1YiI6InJ1YmFja3VwIiwibmJmIjoxNjg1MDkwNjE2LCJleHAiOjE2ODUxNzcwMTZ9.IURpehv7czAiQGzuY2CDfQtsOQIvTU1yilGTNmgN8-w",
        "refresh_token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJmcmVzaCI6ZmFsc2UsImlhdCI6MTY4NTA5MDYxNiwianRpIjoiMjQyYTUwOGMtOGZmNi00M2U2LThmOWYtMDEwNTlmMmJmNmMyIiwidHlwZSI6InJlZnJlc2giLCJzdWIiOiJydWJhY2t1cCIsIm5iZiI6MTY4NTA5MDYxNiwiZXhwIjoxNjg3NjgyNjE2fQ.a3pInerrmOJFcGqLWLa14hZrZhud8NJwU4JY11I34fY"
    },
    "is_error": false,
    "msg": ""
}
```


Сгенерированный a*ccess_token* будет действовать в течение 15 минут с момента получения, а *refresh_token* — 24 часа с момента получения.

По истечении срока жизни *access_token* его можно перевыпустить с помощью *refresh_token*. Если истек срок жизни *refresh_token*, необходимо перевыпустить новую пару токенов с помощью логина и пароля.

Перевыпуск AccessToken на основе RefreshToken

Необходимо отправить POST-запрос с помощью консольной утилиты *curl* или любым другим удобным способом. В данном примере используется *curl*:

```
$ curl -k -X POST https://192.168.10.10:443/api/v0.1/refresh
-H 'Content-Type: application/json'
-H 'X-Access-Token: <refresh_token>'
```

Результат вернётся в формате JSON:


```
{
	"data": {
		"access_token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJmcmVzaCI6ZmFsc2UsImlhdCI6MTY4NTA5OTY4OCwianRpIjoiYWEwNTNlYTUtYzg5YS00OTBmLWJjMTItYTM3MTQzOThkMDdlIiwidHlwZSI6ImFjY2VzcyIsInN1YiI6InJ1YmFja3VwIiwibmJmIjoxNjg1MDk5Njg4LCJleHAiOjE2ODUxODYwODh9.qasqwxJHVxOoiporRRz686HDSneKzB5Vq5S_UgpDmvE",
		"refresh_token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJmcmVzaCI6ZmFsc2UsImlhdCI6MTY4NTA5OTY4OCwianRpIjoiYTQ2YjFkNDUtNTE1ZS00NTQzLWIxNjktMjZkZDBkMGU5NGYxIiwidHlwZSI6InJlZnJlc2giLCJzdWIiOiJydWJhY2t1cCIsIm5iZiI6MTY4NTA5OTY4OCwiZXhwIjoxNjg3NjkxNjg4fQ.HlYMM-SazdNqR8jL52wa5ELZVuFTdIzjwJLSvNk-Jsw"
	},
	"is_error": false,
	"msg": ""
}
```


Отзыв токенов

Если ваши токены были скомпрометированы, то можно сделать отзыв токенов (каждого по отдельности). Для этого необходимо отправить POST-запрос с помощью консольной утилиты *curl* или любым другим удобным для вас способом. В данном примере используется *curl*:

```
$ curl -k -X POST https://192.168.10.10:443/api/v0.1/logout \
-H 'Content-Type: application/json' \
-H 'X-Access-Token: <refresh_token>'
```

Результат вернётся в формате JSON. В зависимости от переданного токена в сообщение будет указан тот или иной тип токена:

```
{
```

```
    "data": {},
    "is_error": false,
    "msg": "Refresh token was revoked!"
}
```

Авторизация пользователя в веб-интерфейсе

Для начала работы с ресурсами сервера перейдите по адресу *https://<ваш ip-адрес>:443/* в Вашем браузере и нажмите на кнопку *«Authorize»* (рисунок 1):

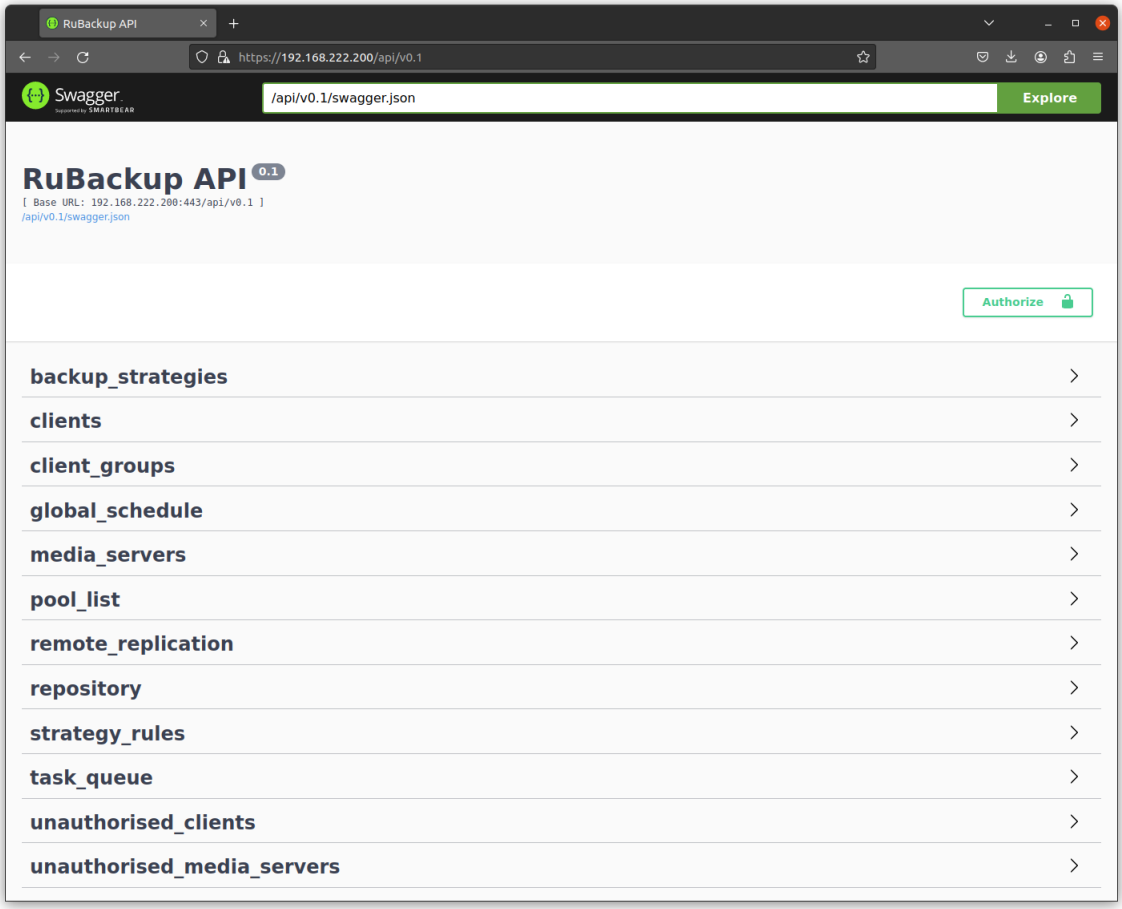

Рисунок 1

В открывшейся форме заполните поле *«Value»* ранее полученным *access_token* и нажмите кнопкку *«Authorize»* (рисунок 2):

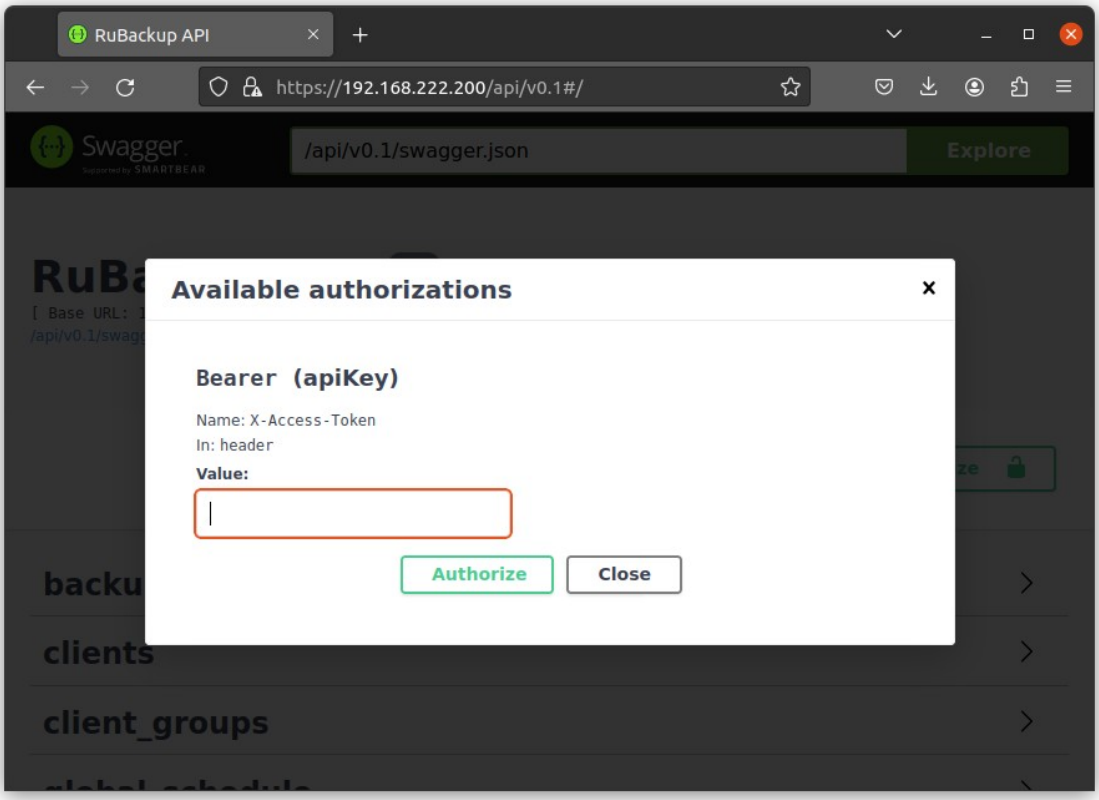

Рисунок 2

Закройте форму нажатием на кнопку *«Close»* (рисунок 3). Теперь вы авторизованы и можете работать с ресурсами сервера RuBackup.

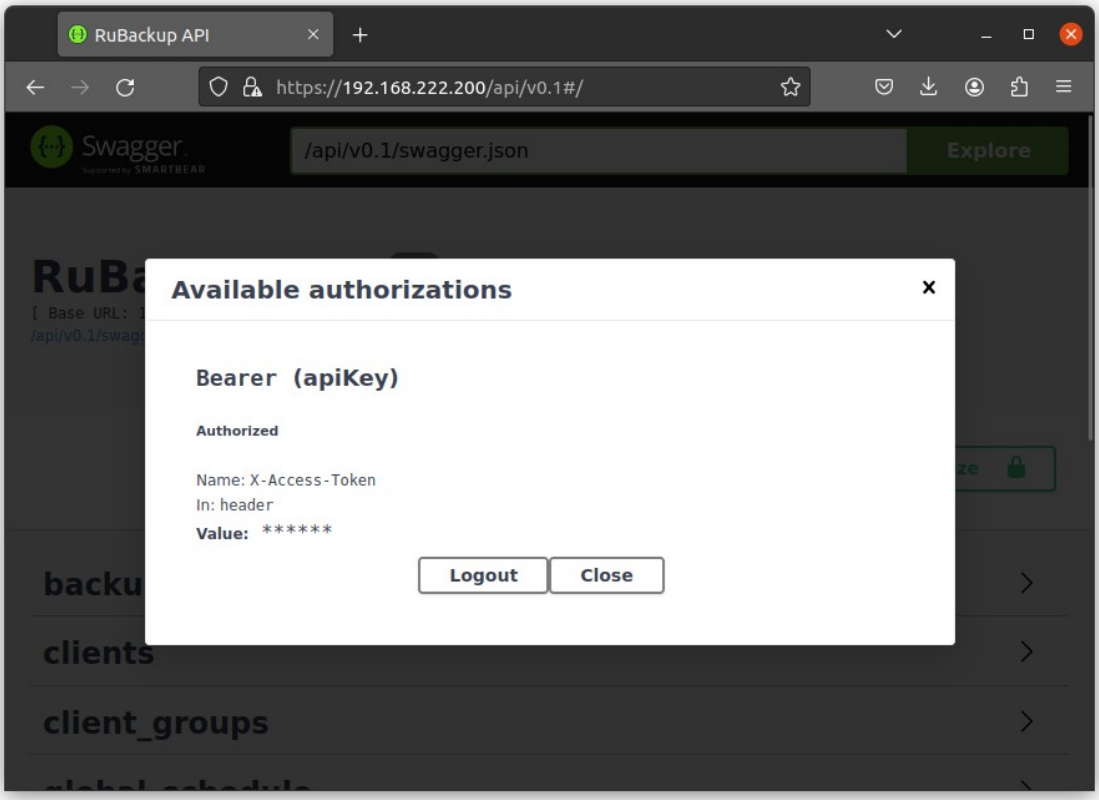

Рисунок 3

Доступные для использования ресурсы и их методы

В программном интерфейсе RuBackup перечислены необходимые ресурсы для взаимодействия с сервером резервного копирования (таблица 1). Пользователь может обращаться к каждому представленному ресурсу либо через браузер, *используя спецификацию, которая предоставляется Swagger*, либо через консольную утилиту *curl*.

Таблица 1 – Доступные для использования ресурсы и их методы

Название ресурса	Доступные методы
backup_strategies	**GET** */<resource>/* **POST** */<resource>/* **GET** */<resource>/{BackupStrategyId}* **PATCH** */<resource>/{BackupStrategyId}* **DELETE** */<resource>/{BackupStrategyId}*
clients	**GET** */<resource>/* **GET** */<resource>/{ClientId}*
client_groups	**GET** */<resource>/* **POST** */<resource>/* **GET** */<resource>/{ClientGroupId}* **PATCH** */<resource>/{ClientGroupId}* **DELETE** */<resource>/{ClientGroupId}*
global_schedule	**GET** */<resource>/* **POST** */<resource>/* **GET** */<resource>/{GlobalScheduleId}* **PATCH** */<resource>/{GlobalScheduleId}* **DELETE** */<resource>/{GlobalScheduleId}*
media_servers	**GET** */<resource>/* **GET** */<resource>/{MediaServerId}*
pool_list	**GET** */<resource>/* **POST** */<resource>/* **GET** */<resource>/{PoolListId}* **PATCH** */<resource>/{PoolListId}* **DELETE** */<resource>/{PoolListId}*
remote_replication	**GET** */<resource>/* **POST** */<resource>/*

Название ресурса	Доступные методы
	GET */<resource>/{RemoteReplicationId}* **PATCH** */<resource>/{RemoteReplicationId}* **DELETE** */<resource>/{RemoteReplicationId}*
repository	**GET** */<resource>/* **GET** */<resource>/{BackupId}*
strategy_rules	**GET** */<resource>/* **POST** */<resource>/* **GET** */<resource>/{StrategyRuleId}* **PATCH** */<resource>/{StrategyRuleId}* **DELETE** */<resource>/{StrategyRuleId}*
task_queue	**GET** */<resource>/* **POST** */<resource>/* **GET** */<resource>/{TaskId}*
unauthorised_clients	**GET** */<resource>/* **GET** */<resource>/{UnauthorisedClientId}*
unauthorised_media_servers	**GET** */<resource>/* **GET** */<resource>/{UnauthorisedMediaServerId}*