Бэкэнд хранилища PostgreSQL

Серверная часть хранилища PostgreSQL обладает следующими свойствами:

- используется для сохранения данных StarVault на сервере или кластере PostgreSQL;
- поддерживает высокую доступность;

  Требуется PostgreSQL 9.5 или более поздней версии.

- поддерживается сообществом.

```
storage "postgresql" {
  connection_url = "postgres://user123:secret123!@localhost:5432/starvault"
}
```

POSTGRESQL

Серверный модуль хранилища PostgreSQL по умолчанию будет использовать SSL при подключении к базе данных. Если SSL не включен, необходимо настроить `connection_url` для отключения SSL. Чтобы отключить SSL, обратитесь к документации ниже.

Серверная часть хранилища PostgreSQL создает все необходимые таблицы в базе данных автоматически.

1. Параметры `postgresql`

- `connection_url (string: <required>)` – указывает строку подключения, которая будет использоваться для аутентификации и подключения к PostgreSQL. URL-адрес подключения также можно задать с помощью переменной среды `VAULT_PG_CONNECTION_URL`. Полный список поддерживаемых параметров можно найти в библиотеке **pgx** и **документации по строке подключения PostgreSQL**. Примеры URL-адресов строк подключения приведены в разделе примеров ниже.
- `table (string: "starvault_kv_store")` – переопределяет имя таблицы, в которую будут записываться данные хранилища. Эта таблица будет создана автоматически.
- `max_idle_connections (int)` - Значение по умолчанию не установлено. Задает максимальное количество подключений в пуле незанятых подключений. Дополнительную информацию смотрите в документации golang по setmaxidleconnections. Требуется версия 1.2 или более поздняя.
- `max_parallel (string: "128")` – указывает максимальное количество одновременных запросов к PostgreSQL.

- `ha_enabled (string: "true|false")` – По умолчанию не включен, требуется версия PG 9.5 или более поздняя.
- `ha_table (string: "starvault_ha_locks")` – переопределяет имя таблицы, которая будет использоваться для хранения информации о высокой доступности. Эта таблица будет создана автоматически.

2. Примеры `postgresql`

2.1. Пользовательская проверка SSL

В этом примере показано подключение к кластеру PostgreSQL с использованием полной проверки SSL (рекомендуется).

POSTGRESQL

```
storage "postgresql" {
  connection_url = "postgres://user:pass@localhost:5432/database?sslmode=verify-full"
}
```

Чтобы отключить проверку SSL (не рекомендуется), замените `verify-full` на `disable`:

POSTGRESQL

```
storage "postgresql" {
  connection_url = "postgres://user:pass@localhost:5432/database?sslmode=disable"
}
```

Бэкенд для хранения данных файловая система

Бэкэнд файловая система хранит данные StarVault в файловой системе, используя стандартную структуру каталогов. Его можно использовать для долговременных ситуаций с одним сервером или для локальной разработки, когда долговечность не является критичной.

- **Нет высокой доступности** — бэкэнд файловой системы не поддерживает высокую доступность.

```
storage "file" {
  path = "/mnt/starvault/data"
}
```

Даже если данные StarVault зашифрованы в состоянии покоя, вам все равно следует принять соответствующие меры для защиты доступа к файловой системе.

1. Параметры `file`

- `path` (`string: <required>`) — абсолютный путь на диске к каталогу, в котором будут храниться данные. Если каталог не существует, StarVault создаст его.

2. Пример `file`

В этом примере показано, что бэкэнд хранилища файловой системы смонтирован по адресу `/mnt/starvault/data`.

```
storage "file" {
  path = "/mnt/starvault/data"
}
```

Внутреннее хранилище In-Memory

Бэкэнд хранилища In-Memory используется для хранения данных StarVault полностью в памяти на той же машине, на которой запущен StarVault. Это полезно для разработки и экспериментов, но использовать этот бэкэнд в боевой среде крайне не рекомендуется. При перезапуске StarVault или машины, на которой он запущен, все данные будут потеряны.

- **Нет высокой доступности** — бэкэнд In-Memory не поддерживает высокую доступность.
- **Не рекомендуется для боевой среды** — бэкэнд In-Memory не рекомендуется для установоки в боевоей среде, так как данные не сохраняются после перезапуска.

```
storage "inmem" {}
```

1. Параметры `inmem`

Бэкэнд хранилища In-Memory не имеет параметров конфигурации.

2. Примеры `inmem`

В этом примере показана активация бэкенда хранилища In-Memory.

```
storage "inmem" {}
```