# Телеметрия секретов

Телеметрия секретов предоставляет информацию о настроенных операциях механизма секретов.

## 1. Стандартные метрики

В таблице далее представлены стандартные метрики секретов и их описания:

| Название метрики | Тип метрики | Единицы измерения | Описание | Примечание |
|---|---|---|---|---|
| vault.secret.kv.count | датчик | число | Количество записей в каждом механизме секретов ключ-значение | StarVault организует количество пар ключ-значение по кластеру и точке монтирования. |
| vault.secret.lease.creation | счетчик | число | Количество аренд, созданных механизмами секретов | StarVault организует количество аренд по кластеру, механизму секретов, точке монтирования и времени жизни (TTL). |

## 2. Метрики PKI

В таблице далее представлены метрики PKI и их описания:

| Название метрики | Тип метрики | Единицы измерения | Описание | Примечание |
|---|---|---|---|---|
| secrets.pki.tidy.cert_store_current_entry | датчик | число | Индекс текущей записи в хранилище сертификатов, проверяемой операцией очистки сертификатов | --- |

| Название метрики | Тип метрики | Единицы измерения | Описание | Примечание |
|---|---|---|---|---|
| secrets.pki.tidy.cert_store_deleted_count | счетчик | число | Количество записей, удалённых из хранилища сертификатов | --- |
| secrets.pki.tidy.cert_store_total_entries_remaining | датчик | число | Количество записей в хранилище сертификатов, проверенных, но не удалённых в ходе операции очистки сертификатов | --- |
| secrets.pki.tidy.cert_store_total_entries | датчик | число | Количество записей в хранилище сертификатов, подлежащих проверке в ходе операции очистки сертификатов | --- |
| secrets.pki.tidy.duration | сводная | мс | Время, необходимое для завершения операции очистки PKI | --- |
| secrets.pki.tidy.failure | счетчик | число | Количество раз, когда операция очистки PKI не была завершена из-за ошибок | --- |
| secrets.pki.tidy.revoked_cert_current_entry | датчик | число | Индекс текущей записи в хранилище отозванных сертификатов, проверяемой операцией очистки | --- |

| Название метрики | Тип метрики | Единицы измерения | Описание | Примечание |
|---|---|---|---|---|
| secrets.pki.tidy.revoked_cert_deleted_count | счетчик | число | Количество записей, удалённых из хранилища для отозванных сертификатов | --- |
| secrets.pki.tidy.revoked_cert_total_entries_fixed_issuers | датчик | число | Количество записей в хранилище сертификатов, у которых были обнаружены и исправлены некорректные данные об издателе в ходе операции очистки | --- |
| secrets.pki.tidy.revoked_cert_total_entries_incorrect_issuers | датчик | число | Общее количество записей в хранилище сертификатов, у которых были обнаружены некорректные данные об издателе | --- |
| secrets.pki.tidy.revoked_cert_total_entries_remaining | датчик | число | Количество отозванных сертификатов в хранилище, проверенных, но не удалённых в ходе операции очистки сертификатов | --- |
| secrets.pki.tidy.revoked_cert_total_entries | датчик | число | Количество записей об отозванных сертификатах в хранилище, подлежащих проверке в ходе операции очистки сертификатов | --- |

| Название метрики | Тип метрики | Единицы измерения | Описание | Примечание |
|---|---|---|---|---|
| secrets.pki.tidy.start_time_epoch | датчик | секунды | Время начала операции очистки PKI в формате эпохи (секунды с 1970-01-01) | Показатель времени начала будет равен 0, если операция очистки PKI не активна в данный момент. |
| secrets.pki.tidy.success | счетчик | число | Количество успешных завершений операции очистки PKI | --- |

## 3. Метрики базы данных секретов

Метрики, связанные с настроенными механизмами секретов, включая метрики, специфичные для базы данных, для каждого именованного механизма секретов. Например, если вы включили механизм секретов PostgreSQL под названием `postgresql-prod`, связанная с ним метрика `CreateUser.error` будет `database.postgresql-prod.CreateUser.error`.

| Название метрики | Тип метрики | Единицы измерения | Описание | Примечание |
|---|---|---|---|---|
| database.Close | сводная | мс | Время, необходимое для закрытия хранилища секретов баз данных (по всем хранилищам секретов баз данных) | --- |
| database.Close.error | счетчик | счетчик | Количество ошибок, возникших при закрытии соединений с базой данных во всех хранилищах секретов баз данных | --- |

| Название метрики | Тип метрики | Единицы измерения | Описание | Примечание |
|---|---|---|---|---|
| database.NewUser | сводная | мс | Время, необходимое для создания пользователя во всех хранилищах секретов баз данных | --- |
| database.New-User.error | счетчик | число | Количество ошибок, возникших при создании пользователей во всех хранилищах секретов баз данных | --- |
| database.Initialize | сводная | мс | Время, необходимое для инициализации хранилища секретов баз данных (по всем хранилищам секретов баз данных) | --- |
| database.Initialize.er-ror | счетчик | число | Количество ошибок, возникших при инициализации базы данных во всех хранилищах секретов баз данных. | --- |
| database. {NAME}.Close | сводная | мс | Время, необходимое для закрытия хранилища секретов базы данных с именем {NAME} | --- |

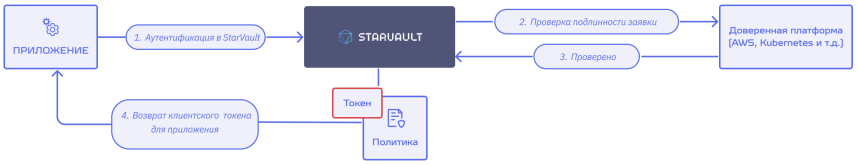
| Название метрики | Тип метрики | Единицы измерения | Описание | Примечание |
|---|---|---|---|---|
| database.{NAME}.Close.error | счетчик | число | Количество ошибок, возникших при закрытии соединений с базой данных в хранилище секретов базы данных с именем {NAME} | --- |
| database.{NAME}.NewUser | сводная | мс | Время, необходимое для создания пользователя в хранилище секретов базы данных с именем {NAME} | --- |
| database.{NAME}.NewUser.error | счетчик | число | Количество ошибок, возникших при создании пользователей в хранилище секретов базы данных с именем {NAME} | --- |
| database.{NAME}.Initialize | сводная | мс | Время, необходимое для инициализации хранилища секретов базы данных для базы данных с именем {NAME} | --- |
| database.{NAME}.Initialize.error | счетчик | число | Количество ошибок, возникших при инициализации базы данных в хранилище секретов с именем {NAME} | --- |

| Название метрики | Тип метрики | Единицы измерения | Описание | Примечание |
|---|---|---|---|---|
| database.{NAME}.UpdateUser | сводная | мс | Время, необходимое для обновления пользователя в хранилище секретов базы данных с именем {NAME} | --- |
| database.{NAME}.Update-User.error | счетчик | число | Количество ошибок, возникших при обновлении пользователей в хранилище секретов базы данных с именем {NAME} | --- |
| database.{NAME}.DeleteUser | сводная | мс | Время, необходимое для аннулирования пользователя в хранилище секретов базы данных с именем {NAME} | --- |
| database.{NAME}.Delete-User.error | счетчик | число | Количество ошибок, возникших при аннулировании пользователей в хранилище секретов базы данных с именем {NAME} | --- |
| database.Update-User | сводная | мс | Время, необходимое для обновления пользователя во всех хранилищах секретов баз данных | --- |

| Название метрики | Тип метрики | Единицы измерения | Описание | Примечание |
| --- | --- | --- | --- | --- |
| database.Update-User.error | счетчик | число | Количество ошибок, возникших при обновлении пользователей во всех хранилищах секретов баз данных | --- |
| database.DeleteUser | сводная | мс | Время, необходимое для аннулирования пользователя во всех хранилищах секретов баз данных | --- |
| database.Delete-User.error | счетчик | число | Количество ошибок, возникших при аннулировании пользователей во всех хранилищах секретов баз данных | --- |

# Агент и прокси

Все запросы к StarVault должны сопровождаться действительным клиентским токеном. Это относится ко всем запросам API, а также к запросам через StarVault CLI и другие библиотеки. Поэтому клиенты StarVault должны сначала пройти аутентификацию в StarVault, чтобы получить токен. StarVault предоставляет несколько методов аутентификации, чтобы помочь в получении этого начального маркера.




Если клиент может безопасно получить токен, все последующие запросы (например, запрос учетных данных базы данных, чтение секретов ключей/значений) обрабатываются на основе доверия, установленного в результате успешной аутентификации.

Это означает, что клиентское приложение должно вызывать StarVault API для аутентификации в StarVault и управления полученным токеном, в дополнение к вызову API для запроса секретов из StarVault. Это предполагает внесение изменений в код клиентских приложений, а также дополнительное тестирование и сопровождение приложения.

Следующий пример кода реализует StarVault API для аутентификации в StarVault с помощью метода AppRole auth, а затем использует полученный клиентский токен для чтения секретов в `kv-v2/data/creds`:

```
package main

import (
    ...snip...
    vault "github.com/hashicorp/vault/api"
)

// Получение секретного ключа (kv-v2) после аутентификации через AppRole
func getSecretWithAppRole() (string, error) {
    config := vault.DefaultConfig()

    client := vault.NewClient(config)
    wrappingToken := ioutil.ReadFile("path/to/wrapping-token")
    unwrappedToken :=
client.Logical().Unwrap(strings.TrimSuffix(string(wrappingToken), "\n"))

    secretID := unwrappedToken.Data["secret_id"]
```

```
    roleID := os.Getenv("APPROLE_ROLE_ID")

    params := map[string]interface{}{
        "role_id":   roleID,
        "secret_id": secretID,
    }
    resp := client.Logical().Write("auth/approle/login", params)
    client.SetToken(resp.Auth.ClientToken)

    secret, err := client.Logical().Read("kv-v2/data/creds")
    if err != nil {
        return "", fmt.Errorf("unable to read secret: %w", err)
    }

    data := secret.Data["data"].(map[string]interface{})

    ...snip...
}
```

## 1. Включение агента StarVault Agent и прокси StarVault Proxy в рабочий процесс

StarVault Agent и StarVault Proxy призваны устранить это первоначальное препятствие на пути к внедрению StarVault, обеспечив более масштабируемый и простой способ интеграции приложений с StarVault. StarVault Agent может получать ключи и предоставлять их приложениям, а StarVault Proxy может выступать в качестве прокси между StarVault и приложением, по необходимости упрощая процесс аутентификации и кэшируя запросы.

Как и большинство других команд CLI для двоичных файлов StarVault, StarVault Agent и StarVault Proxy доступны во всех двоичных файлах и образах StarVault.

**Пример вывода:**

| Возможности | StarVault Agent | StarVault Proxy |
|---|---|---|
| `Автоматическая автоавторизация для аутентификации в StarVault` | ✅ | ✅ |
| `Запуск в качестве службы Windows` | ✅ | ✅ |
| `Кэширование недавно созданных токенов и лицензий` | ✅ | ✅ |
| `Шаблонизация для отображения пользовательских шаблонов` | ✅ | ❌ |

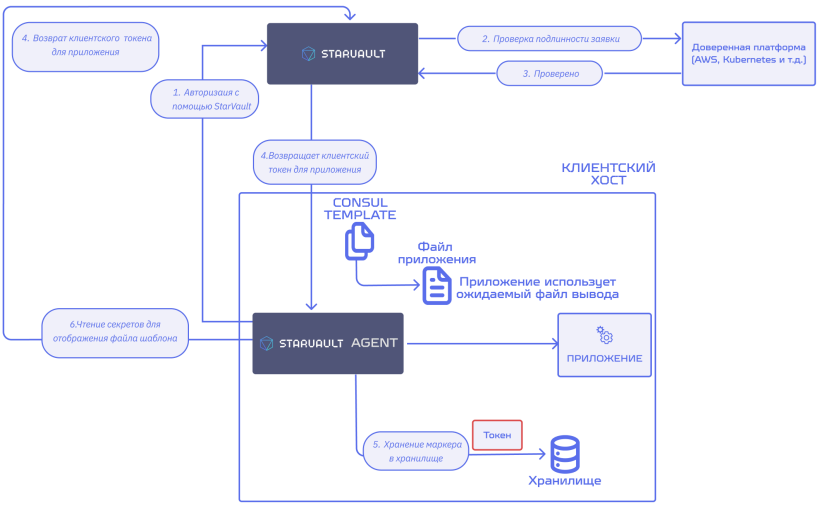
| Возможности | StarVault Agent | StarVault Proxy |
| --- | --- | --- |
| `Супервизор процесса для внедрения ключей в качестве переменных окружения в процесс` | ✅ | ❌ |
| `API-прокси для работы в качестве прокси для API StarVault` | Не актуально | ✅ |
| `Статическое кэширование секретов для секретов KV` | ❌ | ✅ |

Чтобы узнать больше, обратитесь к странице документации StarVault Agent или StarVault Proxy.

---

# Агент хранилища StarVault

StarVault Agent призван устранить первоначальное препятствие для внедрения StarVault, обеспечивая более масштабируемый и простой способ интеграции приложений с StarVault, предоставляя возможность визуализации шаблонов, содержащих секреты, необходимые вашему приложению, без необходимости внесения изменений в ваше приложение.




StarVault Agent - это демон-клиент, который предоставляет следующие возможности:

- **Auto-auth** - автоматическая аутентификация в StarVault и управление процессом обновления токенов для локально извлекаемых динамических секретов.
- **Кэширование** - позволяет кэшировать на стороне клиента ответы, содержащие вновь созданные токены, и ответы, содержащие арендованные секреты, созданные на основе этих вновь созданных токенов. Агент также управляет обновлением кэшированных токенов и арендованных секретов.
- **Служба Windows** - позволяет запускать StarVault Agent в качестве службы Windows.
- **Templating** - позволяет StarVault Agent отрисовывать пользовательские шаблоны, используя токен, сгенерированный на этапе auto-auth.
- **Режим супервизора процесса** - запускает дочерний процесс с секретами StarVault, введенными в качестве переменных окружения.

## 1. Автоматическая авторизация

StarVault Agent позволяет легко аутентифицироваться в StarVault в самых разных средах. Пожалуйста, обратитесь к документации по Auto-auth для получения информации.

Функциональность Auto-auth находится в строке конфигурации `auto_auth`.

## 2. Кэширование

StarVault Agent позволяет кэшировать на стороне клиента ответы, содержащие вновь созданные токены, и ответы, содержащие арендованные секреты, сгенерированные на основе этих вновь созданных токенов. Пожалуйста, обратитесь к документации по кэшированию для получения информации.

## 3. API

### 3.1. Вывод

Эта конечная точка запускает выключение агента. По умолчанию она отключена, но может быть включена для каждого слушателя с помощью строфы `agent_api`. Рекомендуется включать его только на доверенных интерфейсах, так как он не требует авторизации для использования.

| Метод | Расположение |
| --- | --- |
| `POST` | `/agent/v1/quit` |

### 3.2. Кэш

Подробности об API кэширования смотрите на странице «Кэширование».

## 4. Конфигурация

### 4.1. Параметры команды

- `-log-level (string: «info»)` - уровень подробности журнала. Поддерживаемые значения (в порядке убывания подробности): `trace`, `debug`, `info`, `warn` и `error`. Это значение также может быть задано с помощью переменной окружения `STARVAULT_LOG_LEVEL`.
- `-log-format (string: «standard»)` - формат журнала. Поддерживаются следующие значения: `standard` и `json`. Также может быть задан через переменную окружения `STARVAULT_LOG_FORMAT`.
- `-log-file` - абсолютный путь, по которому StarVault Agent должен сохранять сообщения журнала. Пути, заканчивающиеся разделителем путей, используют имя

файла по умолчанию, `agent.log`. Пути, которые не заканчиваются расширением файла, используют расширение `.log` по умолчанию. Если файл журнала вращается, StarVault Agent добавляет к имени файла текущую временную метку в момент вращения. Например:

| `log-file` | **Полная версия журнала** | **Ротируемый файл журнала** |
| --- | --- | --- |
| `/var/log` | `/var/log/agent.log` | `/var/log/agent-{timestamp}.log` |
| `/var/log/my-diary` | `/var/log/my-diary.log` | `/var/log/my-diary-{timestamp}.log` |
| `/var/log/my-diary.txt` | `/var/log/my-diary.txt` | `/var/log/my-diary-{timestamp}.txt` |

## 5. Параметры файла конфигурации

- `-log-rotate-bytes` - указание количества байт, которые должны быть записаны в журнал, прежде чем его нужно будет повернуть. Если не указано, количество байт, которые могут быть записаны в файл журнала, не ограничено.
- `-log-rotate-duration` - указание максимального времени, в течение которого журнал должен быть записан, прежде чем его нужно будет повернуть. Должно быть значение длительности, например 30c. По умолчанию 24 часа.
- `-log-rotate-max-files` - указание максимального количества старых архивов файлов журнала, которые следует сохранять. По умолчанию `0` (файлы никогда не удаляются). Установите значение `-1`, чтобы отбрасывать старые файлы журнала при создании нового.

## 6. Параметры файла конфигурации

Это доступные на данный момент варианты общей конфигурации:

- `vault` (vault: <optional>) - указывает удаленный сервер StarVault, к которому подключается агент.
- `auto_auth` (auto_auth: <optional>) - Указывает метод и другие опции, используемые для функции auto-auth.
- `api_proxy` (api_proxy: <optional>) - Указывает опции, используемые для функциональности API-прокси.
- `cache` (cache: <optional>) - Указывает опции, используемые для функциональности кэширования.

- `listener` (listener: <optional>) - Указывает адреса и порты, на которых агент будет отвечать на запросы.

При `SIGHUP (kill -SIGHUP $(pidof starvault))` StarVault Agent попытается перезагрузить конфигурацию TLS списков. Этот метод можно использовать для обновления сертификатов, используемых StarVault Agent, без необходимости перезапускать его процесс.

- `pid_file (string: "")` - Путь к файлу, в котором должен храниться идентификатор процесса (PID) агента.
- `exit_after_auth (bool: false)` - если установлено значение `true`, агент завершит работу с кодом `0` после одной успешной аутентификации, где успех означает, что токен был получен и все поглотители успешно записали его. Если в конфигурации агента определены строфы `шаблонов`, то перед выходом агент будет ждать успешного рендеринга настроенных шаблонов. Если вы используете шаблоны окружения (`env_template`) и установили `exit_after_auth` в true, агент StarVault не будет запускать дочерние процессы, определенные в вашей строфе `exec`.
- `disable_idle_connections (string array: [])` - список строк, которые отключают простаивающие соединения для различных функций StarVault Agent. Допустимые значения: `auto-auth`, `caching`, `proxying` и `templating`. `proxying` настраивается для API proxy, который по историческим причинам идентичен по функциям `caching`. Можно также настроить переменную окружения `VAULT_AGENT_DISABLE_IDLE_CONNECTIONS` в виде строки, разделенной запятой. Эта переменная окружения будет переопределять любые значения, найденные в конфигурационном файле.
- `disable_keep_alives (string array: [])` - список строк, отключающих keep alives для различных функций StarVault Agent. Допустимые значения: `auto-auth`, `caching`, `proxying` и `templating`. `proxying` настраивается для API proxy, который по историческим причинам идентичен по функциям `caching`. Кроме того, можно настроить переменную окружения `VAULT_AGENT_DISABLE_KEEP_ALIVES` в виде строки, разделенной запятыми. Эта переменная окружения будет переопределять любые значения, найденные в конфигурационном файле.
- `template` (template: <optional>) - Указывает опции, используемые для шаблонирования секретов StarVault в файлы.
- `template_config` (template_config: <optional>) - определяет поведение механизма шаблонизации.
- `exec` (exec: <optional>) - указывает опции агента хранилища для запуска дочернего процесса, который внедряет секреты (через строфы `env_template`) в качестве переменных окружения.
- `env_template` (env_template: <optional>) - принимается несколько блоков. Каждый блок содержит опции, используемые для шаблонирования секретов StarVault в качестве

переменных окружения в режиме супервизора процесса.

- `telemetry` (telemetry: <optional>) - Указывает систему отчетов о телеметрии. Список метрик, специфичных для Агента, см. в разделе «Телеметрия Stanza» ниже.
- `log_level` - эквивалент флага командной строки `-log-level`.

При `SIGHUP` `(kill -SIGHUP $(pidof starvault))` StarVault Agent обновляет уровень журнала до значения, указанного в конфигурационном файле (включая переопределение значений, заданных с помощью CLI или параметров переменной окружения).

- `log_format` - эквивалентно флагу командной строки `-log-format`.
- `log_file` - эквивалентно флагу командной строки `-log-file`.
- `log_rotate_duration` - эквивалентно флагу командной строки `-log-rotate-duration`.
- `log_rotate_bytes` - эквивалентно флагу командной строки `-log-rotate-bytes`.
- `log_rotate_max_files` - эквивалентно флагу командной строки `-log-rotate-max-files`.

## 6.1. Блок конфигурации хранилища

Блок хранилища верхнего уровня может быть только один, и он имеет следующие конфигурационные записи:

- `address` `(string: <optional>)` - Адрес сервера StarVault, к которому необходимо подключиться. Это должно быть полное доменное имя (FQDN) или IP, например `https://vault-fqdn:8200` или `https://172.16.9.8:8200`. Это значение можно переопределить, задав переменную окружения `VAULT_ADDR`.
- `ca_cert` `(string: <optional>)` - путь на локальном диске к одному сертификату CA в PEM-кодировке для проверки SSL-сертификата сервера StarVault. Это значение можно переопределить, установив переменную окружения `STARVAULT_CACERT`.
- `ca_path` `(string: <optional>)` - путь на локальном диске к каталогу сертификатов CA в PEM-кодировке для проверки SSL-сертификата сервера StarVault. Это значение можно переопределить, задав переменную окружения `STARVAULT_CAPATH`.
- `client_cert` `(string: <optional>)` - путь на локальном диске к одному сертификату CA в PEM-кодировке, который будет использоваться для TLS-аутентификации на сервере StarVault. Это значение можно переопределить, задав переменную окружения `STARVAULT_CLIENT_CERT`.
- `client_key` `(string: <optional>)` - путь на локальном диске к одному закрытому ключу в PEM-кодировке, соответствующему клиентскому сертификату из `client_cert`. Это значение может быть переопределено установкой переменной окружения `STARVAULT_CLIENT_KEY`.

- `tls_skip_verify (string: <optional>)` - отключение проверки сертификатов TLS. Использовать эту опцию крайне не рекомендуется, так как она снижает безопасность передачи данных на сервер StarVault и с него. Это значение можно отменить, установив переменную окружения `STARVAULT_SKIP_VERIFY`.
- `tls_server_name (string: <optional>)` - Имя, которое будет использоваться в качестве SNI-хоста при подключении по TLS. Это значение можно переопределить, задав переменную окружения `STARVAULT_TLS_SERVER_NAME`.
- `namespace (string: <optional>)` - Пространство имен, которое будет использоваться для всех запросов StarVault Agent к StarVault. Его также можно указать в командной строке или переменной окружения. Порядок старшинства таков: ниже всего этот параметр, затем переменная окружения `STARVAULT_NAMESPACE`, а затем опция командной строки `-namespace` с наивысшим приоритетом. Если ни одна из этих опций не указана, по умолчанию используется корневое пространство имен.

**Повторение блока конфигурации**

Конфигурация `StarVault` может содержать `повторение` блока конфигурации хранилища, которая управляет тем, как обрабатываются неудачные запросы StarVault, независимо от того, выдаются ли эти запросы для отрисовки шаблонов или являются прокси-запросами, поступающими от подсистемы api proxy. Auto-auth, однако, имеет свое собственное понятие повторных попыток и не затрагивается этим разделом.

Для запросов от шаблонизатора StarVaul Agent будет сбрасывать счетчик повторных попыток и выполнять повторные попытки, как только все попытки будут исчерпаны. Это означает, что шаблонизатор будет повторять попытки при сбоях бесконечно, если только `exit_on_retry_failure` из строфы `template_config` не установлено в `true`.

Вот опции для `повторяющегося` блока конфигурации хранилища:

- `num_retries (int: 12)` - укажите, сколько раз будет повторен неудачный запрос. Значение `0` соответствует значению по умолчанию, т.е. 12 повторных попыток. Значение `-1` отключает повторные попытки. Переменная окружения `STARVAULT_MAX_RETRIES` переопределяет эту настройку.

Здесь есть несколько тонкостей, о которых следует знать. Во-первых, запросы, исходящие из прокси-кэша, будут повторно выполняться только в том случае, если они привели к определенным кодам результатов HTTP: любой код 50x, кроме 501 («не реализовано»), а также 412 («предварительное условие не выполнено»); 412 используется в StarVault Enterprise 1.7+ для обозначения несвежего чтения из-за возможной согласованности. Запросы, поступающие из подсистемы шаблонов, повторяются независимо от сбоя.

Во-вторых, повторные попытки шаблонизации могут выполняться как шаблонизатором, так и прокси кэша, если включен постоянный кэш StarVault Agent. Это связано с тем, что при включенной персистентности запросы шаблонов проходят через прокси кэша.

В-третьих, алгоритм обратного отсчета, используемый для установки времени между повторными попытками, различается для подсистем шаблонов и кэша. Это техническое ограничение, которое мы надеемся устранить в будущем.

## 6.2. Блок конфигурации слушателя (listener stanza)

StarVault Agent поддерживает одну или несколько групп слушателей. Слушатели могут быть настроены с кэшированием или без него, но они будут использовать кэш, если он был настроен, и будут включать прокси-сервер API. В дополнение к стандартной конфигурации слушателя, конфигурация слушателя агента также поддерживает следующее:

- `require_request_header` `(bool: false)` - Требует, чтобы все входящие HTTP-запросы на этом прослушивателе содержали запись в заголовке `X-Vault-Request: true`. Использование этой опции обеспечивает дополнительный уровень защиты от атак на подделку запросов на стороне сервера. Запросы от слушателя, которые не имеют соответствующего заголовка `X-Vault-Request`, завершатся ошибкой с кодом состояния HTTP-ответа `412: Предварительное условие не выполнено`.
- `role` `(string: default)` - `role` определяет, какие API-интерфейсы обслуживает прослушиватель. Ее можно настроить на `metrics_only`, чтобы она обслуживала только показатели, или на роль по умолчанию, `default`, которая обслуживает все (включая показатели). Параметр `require_request_header` не применяется к слушателем `metrics_only`.
- `agent_api` `(agent_api: <optional>)` - управляет необязательными конечными точками API агента.

## 6.3. Блок конфигурации agent_api

- `enable_quit` `(bool: false)` - Если установлено значение `true`, агент включит API выхода из системы.

## 6.4. Блок конфигурации телеметрии

StarVault Agent поддерживает конфигурацию телеметрии и собирает различные метрики о своей производительности, автоавторизации и состоянии кэша:

| Метрика | Описание | Тип |
| --- | --- | --- |
| `vault.agent.authenticated` | Текущий статус аутентификации (1 - имеется действительный токен, 0 - нет действительного маркера) | Датчик |

| Метрика | Описание | Тип |
| --- | --- | --- |
| `vault.agent.auth.failure` | Количество сбоев аутентификации | Счетчик |
| `vault.agent.auth.success` | Количество успешных попыток аутентификации | Счетчик |
| `vault.agent.proxy.success` | Количество успешно обработанных запросов | Счетчик |
| `vault.agent.proxy.client_error` | Количество запросов, по которым StarVault вернул сообщение об ошибке | Счетчик |
| `vault.agent.proxy.error` | Количество запросов, которые агенту не удалось обработать через прокси | Счетчик |
| `vault.agent.cache.hit` | Количество обращений к кэшу | Счетчик |
| `vault.agent.cache.miss` | Количество промахов в кэше | Счетчик |

## 7. Важно: использование `STARVAULT_ADDR`

Если вы экспортируете переменную среды `STARVAULT_ADDR` в экземпляр StarVault Agent, это значение будет иметь приоритет над значением в файле конфигурации. Агент хранилища использует это для подключения к StarVault, и это может создать бесконечный цикл, в котором значение `STARVAULT_ADDR` используется для установления соединения, и Агент хранилища в конечном итоге пытается подключиться к самому себе, а не к серверу.

При сбое подключения Агент хранилища увеличивает порт и повторяет попытку. Агент повторяет эти попытки, что приводит к исчерпанию порта.

Эта проблема связана с порядком приоритета трех различных способов настройки адреса хранилища. Они расположены в порядке возрастания приоритета:

1. Файлы конфигурации
2. Переменные среды
3. Флаги CLI

## 8. Запуск агента хранилища

Как запустить StarVault Agent:

1. Загрузите двоичный файл StarVault, в котором выполняется клиентское приложение (виртуальная машина, модуль Kubernetes и т.д.)
2. Создайте файл конфигурации агента StarVault. (Пример конфигурации приведен в разделе **Пример конфигурации**)
3. Запустите агент StarVault с помощью файла конфигурации.

**Пример:**

```bash
$ starvault agent -config=/etc/vault/agent-config.hcl
```

Для получения помощи запустите:

```bash
$ starvault agent -h
```

Как и в случае со StarVault, флаг `-config` можно использовать тремя различными способами:

- Используйте флаг один раз, чтобы указать путь к одному конкретному файлу конфигурации.
- Используйте флаг несколько раз, чтобы присвоить имена нескольким файлам конфигурации, которые будут созданы во время выполнения.
- Используйте этот флаг, чтобы присвоить имя каталогу конфигурационных файлов, содержимое которых будет составлено во время выполнения.

## 9. Пример конфигурации

Ниже приведен пример конфигурации:

```c
pid_file = "./pidfile"

log_file = "/var/log/starvault-agent.log"

vault {
  address = "https://starvault-fqdn:8200"
  retry {
    num_retries = 5
  }
}

auto_auth {
  method {
    type = "approle"
    config = {
      role_id_file_path = "/etc/openbao/roleid"
```

```
      secret_id_file_path = "/etc/openbao/secretid"
    }
  }
  }

  sink "file" {
    config = {
      path = "/tmp/file-foo"
    }
  }

  sink "file" {
    wrap_ttl = "5m"
    aad_env_var = "TEST_AAD_ENV"
    dh_type = "curve25519"
    dh_path = "/tmp/file-foo-dhpath2"
    config = {
      path = "/tmp/file-bar"
    }
  }
}

cache {
  // Пустая строка кэша по-прежнему позволяет выполнять кэширование
}

template_config {
  static_secret_render_interval = "10m"
  exit_on_retry_failure = true
  max_connections_per_host = 20
}

template {
  source = "/etc/starvault/server.key.ctmpl"
  destination = "/etc/starvault/server.key"
}

template {
  source = "/etc/starvault/server.crt.ctmpl"
  destination = "/etc/starvault/server.crt"
}
```