Приемники автоматической аутентификации StarVault Agent и StarVault Proxy

Каждый раз, когда автоматическая аутентификация проходит успешно, токен записывается во включенные приемники в зависимости от их конфигурации. На сегодняшний день мы поддерживаем только один тип приемников - файловые приемники.

1. Файловый приемник

Файловый приемник (`file sink`) записывает токены, по желанию обернутые в ответ и/или зашифрованные, в файл. Это может быть локальный файл или файл, созданный с помощью какого-либо другого процесса (NFS, Gluster, CIFS и т. д.).

После того как сервер записывает файл, клиент должен контролировать его жизненный цикл; как правило, лучше всего удалять файл, как только он будет замечен.

Также лучше всего записывать файл на ramdisk, в идеале - на зашифрованный ramdisk, и использовать соответствующие права доступа к файловой системе. По умолчанию файл записывается с правами `0640` , но их можно изменить с помощью дополнительного параметра 'mode'.

2. Конфигурация

- `path` (string: `required`) - путь для записи файла с маркером.
- `mode` (int: `optional`) - строка, содержащая восьмеричное число, представляющее битовый шаблон для режима файла, аналогично chmod. Установите значение `0000` , чтобы запретить StarVault изменять режим файла.

  Параметр `mode` доступен только в StarVault 1.3.0 и выше.

Параметры конфигурации для упаковки ответов и шифрования файла приемника находятся в опциях, общих для всей документации по приемникам.

Агент. Режим супервизора

Режим супервизора процессов StarVault Agent позволяет внедрять секреты StarVault в процесс через переменные среды с использованием разметки шаблона Consul.

Если вы запускаете свои приложения в кластере Kubernetes, мы рекомендуем оценить Vault Secrets Operator и Vault Agent Sidecar Injector.

Режим супервизора процесса StarVault Agent находится в стадии публичной бета-версии. Пожалуйста, отправьте свой отзыв, открыв вопрос GitHub здесь.

1. Функциональность

StarVault Agent внедрит секреты, указанные в блоках конфигурации `env_template`, в качестве переменных среды в дочерний процесс, указанный в блоке `exec`.

При запуске StarVault Agent в режиме супервизора процесса он будет ждать, пока каждый шаблон переменной среды не будет отрендерен хотя бы один раз, прежде чем запускать процесс. Если `restart_on_secret_changes` установлен на `always` (по умолчанию), Agent перезапустит процесс всякий раз, когда будет обнаружено обновление внедренного секрета. Это может быть как статическое обновление секрета (выполненное на `static_secret_render_interval`), так и динамический секрет, срок действия которого близок к истечению.

Во многих отношениях StarVault Agent будет зеркалировать дочерний процесс. Стандартные потоки ввода и вывода (`stdin` / `stdout` / `stderr`) перенаправляются в дочерний процесс. Кроме того, StarVault Agent завершит работу, когда дочерний процесс завершит работу самостоятельно с тем же кодом выхода.

2. Конфигурация

Инструмент генерации конфигурации агента поможет вам начать работу с создания действительного файла конфигурации агента из предоставленных входных данных.

Режим супервизора процесса требует по крайней мере одного блока `env_template` и ровно одного блока `exec` верхнего уровня. Он несовместим с обычными записями `шаблона` файла.

2.1. env_template

Блок конфигурации `env_template` сопоставляет шаблон, указанный в поле содержимого или указанный в поле `источника`, с именем переменной среды в заголовке блока конфигурации. Он использует тот же язык шаблонов, что и шаблоны файлов, но допускает только подмножество своих параметров конфигурации:

- environment variable name `(string: <required>)` — имя переменной среды, с которой должно сопоставляться содержимое шаблона.
- `contents (string: "")` — эта опция позволяет встраивать содержимое шаблона в файл конфигурации, а не указывать исходный `путь` к файлу шаблона. Это полезно для коротких шаблонов. Эта опция является взаимоисключающей с опцией источника.
- `source (string: "")` — путь на диске для использования в качестве входного шаблона. Эта опция обязательна, если не используется опция `содержимого`.
- `error_on_missing_key (bool: false)` — выход с ошибкой при доступе к несуществующему полю/ключу структуры или карты. Поведение по умолчанию будет печатать `<no value>` при доступе к несуществующему полю. Настоятельно рекомендуется установить это значение в «true». Также см. `exit on retry failure` глобальной конфигурации шаблона агента хранилища.
- `left_delimiter (string: "{{")` — разделитель для использования в шаблоне. Значение по умолчанию — «{{», но для некоторых шаблонов может быть проще использовать другой разделитель, который не конфликтует с самим выходным файлом.
- `right_delimiter (string: "}}")` — разделитель для использования в шаблоне. Значение по умолчанию — «}}», но для некоторых шаблонов может быть проще использовать другой разделитель, который не конфликтует с самим выходным файлом.

2.2. exec

Блок `exec` верхнего уровня имеет следующие записи конфигурации:

- `command (string array: required)` — укажите команду для дочернего процесса с необязательными аргументами. Путь к исполняемому файлу должен быть либо абсолютным, либо относительным к текущему рабочему каталогу.
- `restart_on_secret_changes (string: "always")` — управляет тем, будет ли агент перезапускать дочерний процесс при изменении секрета. Существует два типа изменений секрета, относящихся к этой конфигурации: статическое обновление секрета (на `static_secret_render_interval`) и динамический секрет, срок действия которого близок к истечению. Конфигурация поддерживает два варианта: `always` и `never`.
- `restart_stop_signal (string: "SIGTERM")` — сигнал для отправки дочернему процессу, когда секрет был обновлен и процесс необходимо перезапустить. У процесса

есть 30 секунд после отправки этого сигнала, пока не будет отправлен сигнал `SIGKILL`, чтобы принудительно остановить дочерний процесс.

3. Пример конфигурации

Следующий пример был сгенерирован с помощью `starvault agent generate-config`, инструмента-помощника конфигурации. При такой конфигурации StarVault Agent запустит дочерний процесс (`./my-app arg1 arg2`) с двумя дополнительными переменными среды (`FOO_USER` и `FOO_PASSWORD`), заполненными секретами из StarVault:

```
auto_auth {

  method {
    type = "token_file"

    config {
      token_file_path = "/Users/avean/.vault-token"
    }
  }
}

template_config {
  static_secret_render_interval = "5m"
  exit_on_retry_failure         = true
}

vault {
  address = "http://localhost:8200"
}

env_template "FOO_PASSWORD" {
  contents             = "{{ with secret \"secret/data/foo\" }}{{
.Data.data.password }}{{ end }}"
  error_on_missing_key = true
}
env_template "FOO_USER" {
  contents             = "{{ with secret \"secret/data/foo\" }}{{
.Data.data.user }}{{ end }}"
  error_on_missing_key = true
}

exec {
  command                   = ["./my-app", "arg1", "arg2"]
  restart_on_secret_changes = "always"
  restart_stop_signal       = "SIGTERM"
}
```

Агент. Шаблоны

Функциональность шаблона StarVault Agent позволяет преобразовывать секреты StarVault в файлы или переменные среды (через режим супервизора процессов) с использованием разметки шаблона Consul.

1. Функциональность

Блок конфигурации `template_config` настраивает общее поведение по умолчанию для шаблонизатора. Обратите внимание, что `template_config` можно определить только один раз, и он отличается от шаблонного блока конфигурации. В отличие от `template`, который фокусируется на том, где и как визуализируется определенный секрет, `template_config` содержит параметры, влияющие на поведение шаблонизатора в целом и его взаимодействие с остальной частью Agent. Это включает, помимо прочего, поведение выхода из программы. Со временем могут быть добавлены и другие параметры, которые применяются к шаблонизатору в целом.

`Шаблон` блока конфигурации настраивает агента StarVault для рендеринга секретов в файлы с использованием языка разметки шаблонов Consul. Можно определить несколько `шаблонов` блоков конфигурации для рендеринга нескольких файлов.

Когда агент запускается с включенным шаблонированием, он попытается получить токен StarVault с помощью настроенного метода автоматической аутентификации. В случае неудачи он отступит на короткое время (включая некоторую случайность для предотвращения сценариев Thundering Herd) и повторит попытку. В случае успеха секреты, определенные в шаблонах, будут извлечены из StarVault и рендерятся локально.

2. Язык шаблонов

Выходное содержимое шаблона может быть предоставлено непосредственно как часть параметра `содержимого` в блоке конфигурации `шаблона` или как отдельный файл `.ctmpl` и указывается в исходном `источнике` блока конфигурации `шаблона`.

Чтобы извлечь секреты из StarVault, будь то статические секреты, динамические учетные данные или сертификаты, шаблоны StarVault Agent требуют использования `секретной` функции или функции `pkiCert` из Шаблона Consul.

Функция `secret` работает для всех типов секретов, и в зависимости от типа секрета, который отображается этой функцией, шаблон будет иметь разное поведение обновления, как подробно описано в разделе «Продления». Функция `pkiCert` предназначена для

работы специально с сертификатами, выпущенными PKI Secrets Engine. Обратитесь к разделу **Сертификаты** за информацией о различиях в поведении обновления сертификатов между `secret` и `pkiCert`.

Следующие ссылки содержат дополнительные ресурсы по языку шаблонов, используемому при создании шаблонов StarVault Agent.

- Документация по шаблонам Consul
- Документация по языку шаблонов Go

3. Пример языка шаблона

Ниже приведен пример шаблона, который извлекает общий секрет из хранилища KV StarVault:

```
{{ with secret "secret/my-secret" }}
{{ .Data.data.foo }}
{{ end }}
```

Ниже приведен пример шаблона, который выдает сертификат PKI в движке секретов PKI StarVault. Извлечение сертификата или ключа из роли PKI через эту функцию будет основано на истечении срока действия сертификата.

Чтобы сгенерировать новый сертификат и создать пакет с ключом, сертификатом и центром сертификации, используйте:

```
{{- with secret "pki/cert/ca" -}}
{{ .Data.certificate }}
{{- end -}}
```

В качестве альтернативы можно использовать `pki/cert/ca_chain` для получения полной цепочки CA.

4. Глобальная конфигурация

Блок `template_config` верхнего уровня имеет следующие записи конфигурации, которые влияют на все шаблоны:

- `exit_on_retry_failure (bool: false)` — этот параметр настраивает StarVault Agent на выход после исчерпания количества попыток повтора шаблона из-за сбоев.
- `static_secret_render_interval (string или integer: 5m)` — если указано, настраивает, как часто шаблон StarVault Agent должен отображать неарендуемые

секреты, такие как KV v2. Этот параметр не изменит, как часто шаблон StarVault Agent отображает арендованные секреты. Использует строки формата продолжительности].

4.1. `template_config` пример шаблона конфигурации

```
template_config {
  exit_on_retry_failure = true
  static_secret_render_interval = "10m"
}
```

В другом примере `template_config` с параметром `error_on_missing_key` в блоке конфигурации шаблона, а также `exit_on_retry_failure` приводят к тому, что агент завершает работу в случае отсутствия проблем с ключом/значением вместо поведения повтора по умолчанию.

```
template_config {
  exit_on_retry_failure = true
  static_secret_render_interval = "10m"
}

template {
  source      = "/tmp/agent/template.ctmpl"
  destination = "/tmp/agent/render.txt"
  error_on_missing_key = true
}
```

4.2. Взаимодействие между `exit_on_retry_failure` и `error_on_missing_key`

Параметр `error_on_missing_key` может быть указан в блоке конфигурации шаблона, который определяет, должен ли шаблон выдавать ошибку, если в секрете отсутствует ключ. Если `error_on_missing_key` не указан или установлен в значение `false`, а ключ для визуализации отсутствует в ответе секрета, шаблонизатор проигнорирует его (или выведет `"<нет значения>"`) и продолжит свою визуализацию.

Если требуется, чтобы агент вывел ошибку и вышел из-за отсутствующего ключа, то и `template.error_on_missing_key`, и `template_config`.exit_on_retry_failure` должны быть установлены в значение `true`. В противном случае шаблонизатор выдаст ошибку и выполнит визуализацию в место назначения, но агент не выйдет и будет повторять попытки, пока ключ не будет найден или пока процесс не будет завершен.

Обратите внимание, что отсутствующий ключ из ответа секрета отличается от отсутствующего или несуществующего секрета. Шаблонизатор всегда выдаст ошибку, если секрет отсутствует, но выдаст ошибку только для отсутствующего ключа, если задан

параметр `error_on_missing_key`. Будет ли StarVault Agent выходить из системы при возникновении ошибок в механизме шаблонизации, зависит от значения `exit_on_retry_failure`.

5. Конфигурации шаблона

Блок `шаблона` верхнего уровня имеет несколько записей конфигурации. Параметры, найденные в разделе конфигурации шаблона на странице документации consul-template, можно использовать здесь:

Параметры, отмеченные `Δ` ниже, применимы только к шаблонам файлов и не могут использоваться с записями `env_template` в режиме супервизора процесса.

- `source (string: "")` — путь на диске для использования в качестве входного шаблона. Этот параметр требуется, если не используется параметр `содержимого`.
- `destination` `Δ (string: required)` — путь на диске, где должны быть созданы визуализированные секреты. Если родительские каталоги не существуют, StarVault Agent попытается создать их, если параметр `create_dest_dirs` не равен false.
- `create_dest_dirs` `Δ (bool: true)` — этот параметр указывает StarVault Agent создать родительские каталоги целевого пути, если они не существуют.
- `contents (string: "")` — этот параметр позволяет встраивать содержимое шаблона в файл конфигурации, а не указывать исходный `путь` в файл шаблона. Это полезно для коротких шаблонов. Этот параметр является взаимоисключающим с параметром `источника`.
- `command` `Δ (string: "")` — это необязательная команда для запуска при отображении шаблона. Команда будет запущена только в случае изменения результирующего шаблона. Команда должна вернуться в течение 30 с (настраивается) и должна иметь успешный код выхода. StarVault Agent не является заменой монитора процесса или системы инициализации. Это устарело в пользу параметра `exec`.
- `command_timeout` `Δ (duration: 30s)` — это максимальное время ожидания возврата необязательной команды. Это устарело в пользу параметра `exec`.
- `error_on_missing_key (bool: false)` — выход с ошибкой при доступе к несуществующему полю/ключу структуры или карты. Поведение по умолчанию будет выводить `<no value>` при доступе к несуществующему полю. Настоятельно рекомендуется установить это значение в «true». Также см. `exit_on_retry_failure` в глобальной конфигурации шаблона StarVault Agent.
- `exec` `Δ (object: Optional)` — блок `exec` выполняет команду, когда шаблон визуализирован и выходные данные изменились. Параметры блока — `command (string or array: required)` и `timeout (string: optional, по умолчанию 30 c)`. `command` может быть задана как строка или массив строк для выполнения,

например, `"touch myfile"` или `["touch", "myfile"]`. Для защиты от внедрения команд мы настоятельно рекомендуем использовать массив строк и сначала пытаемся выполнить разбор таким образом. Также обратите внимание, что использование запятой со строковым подходом приведет к его интерпретации как массива, что может быть нежелательным.

- `perms` Δ `(string: "")` — это разрешение на рендеринг файла. Если этот параметр не указан, StarVault Agent попытается сопоставить разрешения файла, который уже существует в целевом пути. Если по этому пути нет файла, разрешения будут 0644.
- `backup` Δ `(bool: true)` — этот параметр создает резервную копию ранее рендерингованного шаблона в целевом пути перед записью нового. Он сохраняет ровно одну резервную копию. Этот параметр полезен для предотвращения случайных изменений данных без использования стратегии отката.
- `left_delimiter` `(string: "{{")` — разделитель для использования в шаблоне. Значение по умолчанию — "{{", но для некоторых шаблонов может быть проще использовать другой разделитель, который не конфликтует с самим выходным файлом.
- `right_delimiter` `(string: "}}")` — разделитель для использования в шаблоне. Значение по умолчанию — "}}", но для некоторых шаблонов может быть проще использовать другой разделитель, не конфликтующий с самим выходным файлом.
- `sandbox_path` Δ `(string: "")` — если указан путь песочницы, любой путь, указанный для функции файла, проверяется на предмет попадания в путь песочницы. Относительные пути, которые пытаются выйти за пределы пути песочницы, завершатся с ошибкой.
- `wait` Δ `(object: required)` — это `minimum(:maximum)` ожидания перед рендерингом нового шаблона на диск и запуском команды, разделенные двоеточием (`:`).

6. Пример шаблона блока конфигурации

```
template {
  source      = "/tmp/agent/template.ctmpl"
  destination = "/tmp/agent/render.txt"
  error_on_missing_key = true
}
```

Если вы хотите использовать агент StarVault только для отображения одного или нескольких шаблонов и вам не нужно сохранять полученные учетные данные, вы можете `исключить` блок конфигурации сохранения из блока конфигурации `auto_auth` в конфигурации агента.

7. Продление и обновление секретов

Шаблонизация StarVault Agent автоматически обновляется и извлекает секреты/токены. В отличие от кэширования StarVault Agent, поведение того, как StarVault Agent делает это, зависит от типа секрета или токена. Ниже приведен общий обзор различных поведений.

7.1. Продление секретов

Если секрет или токен подлежит продлению, StarVault Agent обновит секрет по истечении 2/3 срока аренды секрета.

7.2. Невозобновляемые секреты

Если секрет или токен не возобновляемы или не сданы в аренду, StarVault Agent будет извлекать секрет каждые 5 минут. Это можно настроить с помощью Template config static_secret_render_interval. Невозобновляемые секреты включают (но не ограничиваются) KV Version 2.

7.3. Невозобновляемые арендованные секреты

Если секрет или токен не возобновляемы, но сданы в аренду, StarVault Agent извлечет секрет, когда будет достигнуто 85% времени жизни секрета (TTL). Арендованные, невозобновляемые секреты включают (но не ограничиваются) динамические секреты, такие как учетные данные базы данных и KV версии 1.

7.4. Статические роли

Если у секрета есть `rotation_period`, например, статическая роль базы данных, шаблон агента StarVault будет извлекать новый секрет по мере его изменения в StarVault. Он делает это, проверяя время жизни секрета (TTL).

7.5. Сертификаты

В StarVault сертификаты могут быть созданы с использованием функций `pkiCert` или `секретных` шаблонов, хотя рекомендуется использовать `pkiCert`, чтобы избежать ненужной генерации сертификатов при каждом перезапуске или повторной аутентификации агента.

Рендеринг с использованием функции шаблона `pkiCert`

Если сертификат визуализируется с использованием функции шаблона pkiCert, шаблон StarVault Agent будет иметь следующие варианты извлечения и повторной визуализации сертификатов:

- Извлекает новый сертификат при запуске агента, если ранее не было предоставлено ни одного сертификата или истек срок действия текущего предоставленного сертификата.
- При повторной аутентификации агента с автоматической аутентификацией, например, из-за истечения срока действия токена, пропустит извлечение, если только текущий предоставленный сертификат не истек.

Рендеринг с использованием `секретной` функции шаблона

Если сертификат визуализируется с использованием функции `секретного` шаблона, шаблон StarVault Agent будет иметь следующие поведения при извлечении и повторной визуализации сертификатов:

- Извлекает новый сертификат при запуске агента, даже если ранее предоставленные сертификаты все еще действительны.
- Если `generate_lease` не установлен или установлен на `false`, он использует поле `validTo` сертификата для определения интервала повторного извлечения.
- Если `generate_lease` установлен на `true`, применяются невозобновляемые, арендованные секретные правила.
- При повторной аутентификации автоматической аутентификации агента, например, из-за истечения срока действия токена, он извлекает и повторно отображает новый сертификат, даже если существующий сертификат действителен.

8. Пример конфигурации шаблона

```
# Другие блоки конфигурации агента StarVault
# ...

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

template {
  source      = "/tmp/agent/template.ctmpl"
  destination = "/tmp/agent/render.txt"
}

template {
  contents     = "{{ with secret \"secret/my-secret\" }}{{ .Data.data.foo }}{{
end }}"
  destination  = "/tmp/agent/render-content.txt"
}
```

Ниже демонстрируется, как выглядят шаблоны при использовании `env_template` с режимом Process Supervisor.

```
# Другие блоки конфигурации агента StarVault
# ...

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

env_template "MY_ENV_VAR" {
  contents = "{{ with secret \"secret/my-secret\" }}{{ .Data.data.foo }}{{ end
}}"
}

env_template "ENV_VAR_FROM_FILE" {
  source = "/tmp/agent/template.ctmpl"
}
```