Механизм секретов PKI

Этот механизм может использовать внешние сертификаты X.509 в рамках TLS или для проверки подписи. Проверка подписей с использованием сертификатов X.509, использующих SHA-1, не поддерживается без обходных решений. См. FAQ по устареванию для получения дополнительной информации.

Механизм секретов PKI генерирует динамические сертификаты X.509. С помощью этого механизма секретов службы могут получать сертификаты, не проходя обычный процесс генерации закрытого ключа и Certificate Signing Request (CSR) вручную, отправки в центр сертификации и ожидания завершения процесса проверки и подписания. Встроенные в StarVault механизмы аутентификации и авторизации обеспечивают функциональность проверки.

Благодаря тому, что TTL относительно коротки, реже возникает необходимость в отзывах, что сокращает срок действия CRL и помогает механизму секретов масштабироваться для больших рабочих нагрузок. Это, в свою очередь, позволяет каждому экземпляру работающего приложения иметь уникальный сертификат, что исключает совместное использование сертификатов и сопутствующие проблемы, связанные с их отзывом и переносом.

Кроме того, позволяя отказаться от отзыва, этот механизм секретов позволяет использовать сертификаты. Сертификаты могут быть получены и сохранены в памяти при запуске приложения и отброшены при его завершении, без записи на диск.

1. Оглавление

Документация по PKI Secrets Engine разделена на следующие части:

- **Обзор** - данный документ.
- **Настройка и использование** - краткое описание настройки и использования PKI Secrets Engine для выпуска сертификатов.
- **Быстрый старт - Настройка Root CA** - краткое руководство по настройке root центра сертификации.
- **Быстрый старт - Настройка промежуточного ЦС** - краткое руководство по настройке промежуточного ЦС.
- **Советы** - список полезных советов, которые следует учитывать при использовании и эксплуатации PKI Secrets Engine.

- **Устранение неполадок ACME** - список советов по устранению неполадок с выпуском ACME и StarVault PKI.
- **Примитивы ротации** - документ, описывающий различные типы сертификатов, применяемые при ротации.

2. API

Движок PKI secrets имеет полный HTTP API. Более подробную информацию см. в разделе API движка PKI secrets.

Секреты движка RabbitMQ

Механизм секретов RabbitMQ динамически генерирует учетные данные пользователей на основе настроенных разрешений и виртуальных хостов. Это означает, что службам, которым необходим доступ к виртуальному хосту, больше не нужно жестко прописывать учетные данные.

Поскольку каждая служба обращается к очереди сообщений с уникальными учетными данными, аудит значительно упрощается при обнаружении сомнительного доступа к данным. Легко отслеживайте проблемы вплоть до конкретного экземпляра службы по имени пользователя RabbitMQ.

StarVault использует как собственную внутреннюю систему отзыва, так и функцию удаления пользователей RabbitMQ при создании пользователей RabbitMQ, чтобы гарантировать, что пользователи станут недействительными в течение разумного срока действия аренды.

1. Установка

Большинство секретных модулей необходимо настроить заранее, прежде чем они смогут выполнять свои функции. Эти действия обычно выполняет оператор или инструмент управления конфигурацией.

1. Включите механизм секретов RabbitMQ:

   ```bash
   $ starvault secrets enable rabbitmq
   Success! Enabled the rabbitmq secrets engine at: rabbitmq/
   ```

   По умолчанию движок секретов будет монтироваться по имени движка. Чтобы включить движок секретов по другому пути, используйте аргумент -path.

2. Настройте учетные данные, которые StarVault использует для связи с RabbitMQ для генерации учетных данных:

   ```bash
   $ starvault write rabbitmq/config/connection \
    connection_uri="http://localhost:15672" \
    username="admin" \
    password="password"
   Success! Data written to: rabbitmq/config/connection
   ```

   Пользователь StarVault должен обладать правами администратора для управления другими пользователями.

3. Настройте роль, которая сопоставляет имя в StarVault с разрешениями виртуального хоста:

```
$ starvault write rabbitmq/roles/my-role \
 vhosts='{"/":{"write": ".*", "read": ".*"}}'
Success! Data written to: rabbitmq/roles/my-role
```

Записывая путь `roles/my-role`, мы определяем роль `my-role`. Эта роль будет создана путём оценки заданных выражений `vhosts`, `vhost_topics` и `tags`. По умолчанию роли не назначены ни теги, ни виртуальные хосты, ни разрешения на доступ к топикам. Если разрешения на доступ к топикам не определены и используется бэкэнд авторизации по умолчанию, публикация в topic exchange или подписка на топик всегда разрешены.

2. Использование

После настройки механизма секретов и получения пользователем/машиной токена StarVault с необходимыми разрешениями он может генерировать учетные данные.

1. Создайте новые учетные данные, прочитав из конечной точки `/creds` имя роли:

```
$ starvault read rabbitmq/creds/my-role
```

Key	Value
`lease_id`	rabbitmq/creds/my-role/l39Hu8XX0mbof4wiK5bKMn9
`lease_duration`	768h
`lease_renewable`	true
`password`	3yNDBikgQvrkx2VA2zhq5IdSM7Iwk1rYMYJr
`username`	root-39669250-3894-8032-c420-3d58483ebfc4

Используя списки управления доступом (ACL), можно ограничить использование движка секретов rabbitmq, так что доверенные операторы смогут управлять определениями ролей, а пользователи и приложения будут ограничены в учетных данных, которые им разрешено читать.

3. API

Движок секретов RabbitMQ имеет полноценный HTTP API. Подробнее см. `в API движка секретов RabbitMQ`.

Механизм секретов SSH

1. Общая информация

Механизм секретов StarVault SSH обеспечивает безопасную аутентификацию и авторизацию для доступа к машинам по протоколу SSH. Механизм секретов StarVault SSH помогает управлять доступом к инфраструктуре машины, предоставляя несколько способов выдачи учетных данных SSH.

Механизм секретов StarVault SSH поддерживает следующие режимы. Каждый режим отдельно документирован на своей странице:

- Подписанные сертификаты SSH
- Одноразовые пароли SSH

Все руководства предполагают базовое знакомство с протоколом SSH.

2. Подписанные сертификаты

Подписанные SSH-сертификаты являются самыми простыми и эффективными с точки зрения сложности настройки и независимости от платформы. Используя мощные возможности центра сертификации StarVault и функциональность, встроенную в OpenSSH, клиенты могут подключаться к целевым хостам по SSH, используя локальные SSH-ключи.

В этом разделе термин "клиент" относится к лицу или компьютеру, выполняющему операцию SSH. "Хост" относится к целевому компьютеру. Если вас это смущает, замените "клиент" на "пользователь".

На этой странице будет приведен краткий обзор механизма управления секретами. Для получения подробной документации по каждому пути воспользуйтесь `starvault path-help` после установки механизма секретов.

2.1. Подписание ключей клиента

Прежде чем клиент сможет запросить подписание своего SSH-ключа, необходимо настроить механизм секретов SSH StarVault. Обычно эти действия выполняет администратор StarVault или команда безопасности. Также можно автоматизировать действия с помощью таких инструментов управления конфигурацией, как Chef, Puppet, Ansible или Salt.

2.1.1. Подписание ключей и настройка ролей

Следующие шаги выполняются заранее администратором хранилища, командой безопасности или средствами управления конфигурацией.

---

1. Смонтируйте механизм секретов. Как и все механизмы секретов в StarVault, механизм секретов SSH должен быть смонтирован перед использованием.

   ```bash
   $ starvault secrets enable -path=ssh-client-signer ssh
   Successfully mounted 'ssh' at 'ssh-client-signer'!
   ```

   Это активирует механизм секретов SSH по пути "ssh-client-signer". Можно подключать один и тот же механизм секретов несколько раз, используя разные аргументы `-path`. Имя "ssh-client-signer" не является специальным - оно может быть любым, но в данной документации будет принято "ssh-client-signer".

2. Настройте StarVault на CA для подписи клиентских ключей с помощью конечной точки `/config/ca`. Если у вас нет внутреннего CA, StarVault может сгенерировать пару ключей для вас.

   ```bash
   $ starvault write ssh-client-signer/config/ca generate_signing_key=true
   ```

   Вывод:

Key	Value
public_key	ssh-rsa AAAAB3NzaC1yc2EA…

   Если у вас уже есть пара ключей, укажите части открытого и закрытого ключей в составе полезной нагрузки:

   ```bash
   $ starvault write ssh-client-signer/config/ca \
       private_key="..." \
       public_key="..."
   ```

   Независимо от того, был ли он сгенерирован или загружен, открытый ключ клиента доступен через API в конечной точке `/public_key` или CLI (см. следующий шаг).

3. Добавьте открытый ключ во все конфигурации SSH целевого узла. Этот процесс можно выполнить вручную или автоматизировать с помощью инструмента управления конфигурацией. Открытый ключ доступен через API и не требует аутентификации.

```bash
$ curl -o /etc/ssh/trusted-user-ca-keys.pem http://127.0.0.1:8200/v1/ssh-client-signer/public_key
```

```bash
$ starvault read -field=public_key ssh-client-signer/config/ca > /etc/ssh/trusted-user-ca-keys.pem
```

Добавьте путь, где хранится содержимое открытого ключа, в файл конфигурации SSH в качестве параметра `TrustedUserCAKeys`.

```bash
# /etc/ssh/sshd_config
# ...
TrustedUserCAKeys /etc/ssh/trusted-user-ca-keys.pem
```

Перезапустите службу SSH, чтобы получить изменения.

4. Создайте именованную роль StarVault для подписания клиентских ключей.

   Из-за того, что некоторые функции сертификата SSH реализованы, опции передаются в виде карты. Следующий пример добавляет расширение `permit-pty` к сертификату и позволяет пользователю указать собственные значения для `permit-pty` и `permit-port-forwarding` при запросе сертификата.

```bash
$ starvault write ssh-client-signer/roles/my-role -<<"EOH"
{
  "allow_user_certificates": true,
  "allowed_users": "*",
  "allowed_extensions": "permit-pty,permit-port-forwarding",
  "default_extensions":
  {
    "permit-pty": ""
  },
  "key_type": "ca",
  "default_user": "ubuntu",
  "ttl": "30m0s"
}
EOH
```

2.1.2. Клиентская аутентификация SSH

Следующие шаги выполняются клиентом (пользователем), который хочет аутентифицироваться на машинах, управляемых StarVault. Эти команды обычно выполняются с локальной рабочей станции клиента.

1. Найдите или сгенерируйте открытый ключ SSH. Обычно это `~/.ssh/id_rsa.pub`. Если у вас нет пары ключей SSH, сгенерируйте ее:

```bash
$ ssh-keygen -t rsa -C "user@example.com"
```

2. Попросите StarVault подписать ваш открытый ключ. Этот файл обычно заканчивается .pub, а его содержимое начинается с `ssh-rsa ….` .

```bash
$ starvault write ssh-client-signer/sign/my-role \
    public_key=@$HOME/.ssh/id_rsa.pub \
    valid_principals=ubuntu
```

Вывод:

Key	Value
---	----
serial_number	c73f26d2340276aa
signed_key	ssh-rsa-cert-v01@openssh.com AAAAHHNzaC1…

Результат будет содержать серийный и подписанный ключ. Этот подписанный ключ является еще одним открытым ключом.

Необязательно

Чтобы настроить параметры подписи, используйте полезную нагрузку в формате JSON:

```bash
$ starvault write ssh-client-signer/sign/my-role -<<"EOH"
{
  "public_key": "ssh-rsa AAA...",
  "valid_principals": "my-user",
  "extensions": {
    "permit-pty": "",
    "permit-port-forwarding": ""
  }
}
EOH
```

3. Сохраните полученный подписанный открытый ключ на диске. При необходимости ограничьте права доступа.

```bash
$ starvault write -field=signed_key ssh-client-signer/sign/my-role \
    public_key=@$HOME/.ssh/id_rsa.pub \
    valid_principals=ubuntu > signed-cert.pub
```

Если вы сохраняете сертификат непосредственно рядом с парой ключей SSH, добавьте к его имени суффикс `-cert.pub` (`~/.ssh/id_rsa-cert.pub`). При такой схеме

именования OpenSSH будет автоматически использовать его при аутентификации.

4. (Необязательно) Просмотр включенных расширений, принципалов и метаданных подписанного ключа.

```
$ ssh-keygen -Lf ~/.ssh/signed-cert.pub
```

5. Выполните SSH на хост-машине, используя подписанный ключ. Вы должны предоставить как подписанный открытый ключ из StarVault, так и соответствующий закрытый ключ в качестве проверки подлинности для вызова SSH.

```
$ ssh -i signed-cert.pub -i ~/.ssh/id_rsa username@10.0.23.5
```

2.2. Подписание ключа хоста

Для дополнительного уровня безопасности мы рекомендуем включить подпись ключей хоста. Она используется вместе с подписью клиентских ключей для обеспечения дополнительного уровня целостности. Если эта функция включена, агент SSH будет проверять, что целевой хост является действительным и доверенным, прежде чем попытаться выполнить SSH. Это снизит вероятность того, что пользователь случайно подключится по SSH к неуправляемой или вредоносной машине.

2.2.1. Конфигурация ключа подписи

---

1. Смонтируйте механизм секретов. Для наибольшей безопасности монтируйте его по другому пути, чем клиентский подписывающий сервер.

```
$ starvault secrets enable -path=ssh-host-signer ssh
Successfully mounted 'ssh' at 'ssh-host-signer'!
```

2. Настройте StarVault на CA для подписания ключей хоста с помощью конечной точки `/config/ca`. Если у вас нет внутреннего CA, StarVault может сгенерировать пару ключей для вас.

```
$ starvault write ssh-host-signer/config/ca generate_signing_key=true
```

Вывод:

Key	Value
public_key	ssh-rsa AAAAB3NzaC1yc2EA…

Если у вас уже есть пара ключей, укажите части открытого и закрытого ключей в составе полезной нагрузки:

```
$ starvault write ssh-host-signer/config/ca \
    private_key="..." \
    public_key="..."
```

Независимо от того, сгенерирован он или загружен, открытый ключ подписывающего хоста доступен через API в конечной точке `/public_key`.

3. Увеличение параметра TTL сертификата ключа хоста.

```
$ starvault secrets tune -max-lease-ttl=87600h ssh-host-signer
```

4. Создайте роль для подписи ключей хоста. Обязательно заполните список разрешенных доменов, установите `allow_bare_domains` или оба параметра.

```
$ starvault write ssh-host-signer/roles/hostrole \
    key_type=ca \
    ttl=87600h \
    allow_host_certificates=true \
    allowed_domains="localdomain,example.com" \
    allow_subdomains=true
```

5. (Необязательно) Сгенерируйте ключ для сервера

```
sudo ssh-keygen -t rsa -b 4096 -f /etc/ssh/ssh_host_rsa_key -N ""
```

6. Подпишите открытый ключ SSH хоста.

```
$ starvault write ssh-host-signer/sign/hostrole \
    cert_type=host \
    public_key=@/etc/ssh/ssh_host_rsa_key.pub
```

Вывод:

Key	Value
serial_number	3746eb17371540d9
signed_key	ssh-rsa-cert-v01@openssh.com AAAAHHNzaC1y…

7. Установите полученный подписанный сертификат в качестве `HostCertificate` в конфигурации SSH на хост-машине.

```bash
$ starvault write -field=signed_key ssh-host-signer/sign/hostrole \
    cert_type=host \
    public_key=@/etc/ssh/ssh_host_rsa_key.pub > /etc/ssh/ssh_host_rsa_key-cert.pub
```

Установите права доступа к сертификату на `0640`:

```bash
$ chmod 0640 /etc/ssh/ssh_host_rsa_key-cert.pub
```

Добавьте ключ хоста и сертификат хоста в файл конфигурации SSH.

```bash
# /etc/ssh/sshd_config
# ...

# For client keys
TrustedUserCAKeys /etc/ssh/trusted-user-ca-keys.pem

# For host keys
HostKey /etc/ssh/ssh_host_rsa_key
HostCertificate /etc/ssh/ssh_host_rsa_key-cert.pub
```

Перезапустите службу SSH, чтобы получить изменения.

2.2.2. Проверка хоста на стороне клиента

1. Получение открытого ключа центра сертификации подписи хоста для проверки подписи хоста целевых машин.

```bash
$ curl http://127.0.0.1:8200/v1/ssh-host-signer/public_key
```

```bash
starvault read -field=public_key ssh-host-signer/config/ca
```

2. Добавьте полученный открытый ключ в файл `known_hosts` с полномочиями.

```bash
# ~/.ssh/known_hosts
@cert-authority *.example.com ssh-rsa AAAAB3NzaC1yc2EAAA...
```

3. Зайдите по SSH на целевые машины с использованием полного доменного имени (например, test.example.com). Теперь не будет выскакивать предупреждение о неизвестном хосте, т.к. сертификат подписан доверенным CA. ---

2.3. Устранение неполадок

При первоначальной настройке этого типа подписи ключа включите подробное ведение журнала по SSH, чтобы помочь аннотировать любые ошибки в журнале:

```bash
# /etc/ssh/sshd_config
# ...
LogLevel VERBOSE
```

Перезапустите SSH после внесения этих изменений.

По умолчанию SSH ведет журнал в `/var/log/auth.log`, но это касается и многих других вещей. Чтобы извлечь только журналы SSH, выполните следующие действия:

```bash
$ tail -f /var/log/auth.log | grep --line-buffered "sshd"
```

Если вам не удается установить соединение с хостом, журналы сервера SSH могут подсказать и помочь.

2.3.1. Имя не указано в списке участников

Если в журнале `auth.log` отображаются следующие сообщения:

```bash
# /var/log/auth.log
key_cert_check_authority: invalid certificate
Certificate invalid: name is not a listed principal
```

Сертификат не разрешает использовать имя пользователя в качестве основного имени для аутентификации в системе. Скорее всего, это связано с ошибкой OpenSSH (см. известные проблемы для получения дополнительной информации). Эта ошибка не учитывает значение параметра `allowed_users`, равное "*". Вот способы обойти эту проблему:

---

1. Установите `default_user` в роли. Если вы всегда проходите аутентификацию под одним и тем же пользователем, установите в роли `default_user` имя пользователя, под которым вы подключаетесь к целевой машине по SSH:

   ```bash
   $ starvault write ssh/roles/my-role -<<"EOH"
   {
     "default_user": "YOUR_USER",
     // ...
   }
   EOH
   ```

2. Установите значение `valid_principals` во время подписания. В ситуациях, когда несколько пользователей могут проходить аутентификацию в SSH через StarVault, убедитесь, что список действительных принципов при подписании ключа включал текущее имя пользователя:

```
$ starvault write ssh-client-signer/sign/my-role -<<"EOH"
{
  "valid_principals": "my-user"
  // ...
}
EOH
```

2.3.2. Нет подсказки после входа в систему

Если после аутентификации на хост-машине вы не видите подсказки, возможно, в подписанном сертификате отсутствует расширение `permit-pty`. Существует два способа добавить это расширение в подписанный сертификат:

- В рамках создания роли

```
$ starvault write ssh-client-signer/roles/my-role -<<"EOH"
{
  "default_extensions": {
    "permit-pty": ""
  }
  // ...
}
EOH
```

- В рамках самой операции подписания:

```
$ starvault write ssh-client-signer/sign/my-role -<<"EOH"
{
  "extensions": {
    "permit-pty": ""
  }
  // ...
}
EOH
```

2.3.3. Нет переадресации портов

Если переадресация портов с гостевого компьютера на хост не работает, в подписанном сертификате может отсутствовать расширение `permit-port-forwarding`. Добавьте расширение в процессе создания или подписания роли, чтобы включить переадресацию портов. Примеры см. в разделе Отсутствие запроса после входа в систему.

```
{
  "default_extensions": {
    "permit-port-forwarding": ""
```

```
  }
}
```

2.4. Нет переадресации x11

Если переадресация X11 с гостевого компьютера на хост не работает, возможно, в подписанном сертификате отсутствует расширение `permit-X11-forwarding`. Добавьте расширение в процессе создания или подписания роли, чтобы включить переадресацию X11. Примеры см. в разделе Отсутствие запроса после входа в систему.

BASH

```
{
  "default_extensions": {
    "permit-X11-forwarding": ""
  }
}
```

2.4.1. Нет переадресации агентов

Если переадресация агентов с гостевого компьютера на хост не работает, в подписанном сертификате может отсутствовать расширение `permit-agent-forwarding`. Добавьте расширение в процессе создания или подписания роли, чтобы включить переадресацию агентов. Примеры см. в разделе Отсутствие запроса после входа в систему.

BASH

```
{
  "default_extensions": {
    "permit-agent-forwarding": ""
  }
}
```

2.4.2. Основные комментарии

Для сохранения атрибутов комментариев в ключах необходимы дополнительные шаги, которые следует учитывать, если они требуются. К закрытому и открытому ключу могут быть применены комментарии, например, при использовании `ssh-keygen` с параметром `-C` - аналогично:

BASH

```
ssh-keygen -C "...Comments" -N "" -t rsa -b 4096 -f host-ca
```

Адаптированные значения ключей, содержащие комментарии, должны быть предоставлены вместе с параметрами, связанными с ключом, в соответствии с шагами StarVault CLI и API, показанными ниже.

```
# Using CLI:
starvault secrets enable -path=hosts-ca ssh
```

```
# Create / update keypair in StarVault
starvault write ssh-client-signer/config/ca \
  generate_signing_key=false \
  private_key=@~/.ssh/id_rsa \
  public_key=@~/.ssh/id_rsa.pub
```

```
# Using API:
curl -X POST -H "X-Vault-Token: ..." -d '{"type":"ssh"}'
http://127.0.0.1:8200/v1/sys/mounts/hosts-ca
KEY_PRI=$(cat ~/.ssh/id_rsa | sed -z 's/\n/\\n/g')
KEY_PUB=$(cat ~/.ssh/id_rsa.pub | sed -z 's/\n/\\n/g')
tee payload.json <<EOF
{
  "generate_signing_key" : false,
  "private_key"          : "${KEY_PRI}",
  "public_key"           : "${KEY_PUB}"
}
EOF
# Create / update keypair in StarVault
curl -X POST -H "X-Vault-Token: ..." -d @payload.json
http://127.0.0.1:8200/v1/hosts-ca/config/ca
```

НЕ добавляйте пароль к закрытому ключу, так как StarVault не сможет его расшифровать. Удалите пару ключей и файл `payload.json` с хоста сразу после подтверждения успешной загрузки.

2.4.3. Известные проблемы

- В системах, поддерживающих SELinux, может потребоваться изменить связанные типы, чтобы демон SSH мог их читать. Например, настройте подписанный сертификат хоста на тип `sshd_key_t`.
- В некоторых версиях SSH вы можете получить следующую ошибку:

  ```
  no separate private key for certificate
  ```

  Эта ошибка появилась в OpenSSH версии 7.2 и была исправлена в версии 7.5. Подробнее см. ошибку OpenSSH 2617.
- В некоторых версиях SSH вы можете получить следующую ошибку на целевом хосте:

  ```
  userauth_pubkey: certificate signature algorithm ssh-rsa: signature
  algorithm not supported [preauth]
  ```

  Исправление заключается в добавлении следующей строки в /etc/ssh/sshd_config

```
CASignatureAlgorithms ^ssh-rsa
```

Алгоритм ssh-rsa больше не поддерживается в OpenSSH 8.2

---

3. Одноразовые пароли

Тип механизма секретов SSH One-Time SSH Password (одноразовый пароль) (OTP) позволяет серверу StarVault выдавать одноразовый пароль каждый раз, когда клиент хочет подключиться к удаленному узлу по SSH, используя вспомогательную команду на удаленном узле для выполнения проверки.

Аутентифицированный клиент запрашивает учетные данные у сервера StarVault и, если он авторизован, ему выдается OTP. Когда клиент устанавливает SSH-соединение с нужным удаленным узлом, OTP, использованный при SSH-аутентификации, получает Vault helper, который затем проверяет OTP на сервере StarVault. Затем сервер StarVault удаляет этот OTP, гарантируя, что он может быть использован только один раз.

Поскольку к серверу StarVault обращаются во время установления SSH-соединения, каждая попытка входа в систему и соответствующая информация об аренде StarVault записываются в журнал секретов аудита.

На этой странице представлен быстрый старт для этого механизма секретов. Для получения подробной документации по каждому пути используйте `starvault path-help` после установки механизма секретов.

3.1. Недостатки

Основное опасение при использовании механизма секретов OTP вызывает соединение удаленного узла с StarVault; если его взломать, злоумышленник может подделать сервер StarVault, возвращающий успешный запрос. Этот риск можно снизить, используя TLS для соединения с StarVault и проверяя действительность сертификата; будущие усовершенствования этого механизма секретов могут позволить обеспечить дополнительную безопасность в дополнение к той, что обеспечивает TLS.

3.2. Монтирование механизма секретов

BASH

```
$ starvault secrets enable ssh
Successfully mounted 'ssh' at 'ssh'!
```

3.3. Создание роли

Создайте роль с параметром `key_type`, установленным на `otp`. На всех машинах, представленных в списке CIDR роли, должен быть правильно установлен и настроен helper.

```
$ starvault write ssh/roles/otp_key_role \
    key_type=otp \
    default_user=username \
    cidr_list=x.x.x.x/y,m.m.m.m/n
Success! Data written to: ssh/roles/otp_key_role
```

3.4. Создание учетной записи

Создайте учетную запись OTP для IP-адреса удаленного узла, принадлежащего к роли `otp_key_role`.

```
$ starvault write ssh/creds/otp_key_role ip=x.x.x.x
```

Вывод:

Key	Value
lease_id	ssh/creds/otp_key_role/73bbf513-9606-4bec-816c-5a2f009765a5
lease_duration	600
lease_renewable	false
port	22
username	username
ip	x.x.x.x
key	2f7e25a2-24c9-4b7b-0d35-27d5e5203a5c
key_type	otp

3.5. Установите сеанс SSH

```
$ ssh username@x.x.x.x
Password: <Enter OTP>
username@x.x.x.x:~$
```

3.6. Автоматизируйте

С помощью одной команды CLI можно создать новый OTP и вызвать SSH с правильными параметрами для подключения к хосту.

```bash
$ starvault ssh -role otp_key_role -mode otp username@x.x.x.x
OTP for the session is `b4d47e1b-4879-5f4e-ce5c-7988d7986f37`
[Note: Install `sshpass` to automate typing in OTP]
Password: <Enter OTP>
```

OTP будет введен автоматически с помощью `sshpass`, если он установлен.

```bash
$ starvault ssh -role otp_key_role -mode otp -strict-host-key-checking=no
username@x.x.x.x
username@<IP of remote host>:~$
```

`sshpass` не может обрабатывать проверку ключей хоста. Проверку ключей хоста можно отключить, задав `-strict-host-key-checking=no`.

4. API

Механизм секретов SSH имеет полноценный HTTP API. Более подробную информацию можно найти в разделе API движка SSH secrets.