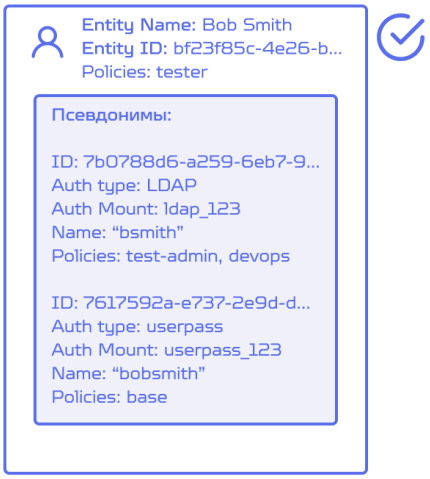
Идентификация

Данная статья содержит концептуальную информацию об идентификационных данных, а также обзор различных терминов и их концепций. Идея идентификации заключается в том, чтобы поддерживать клиентов, которые распознаются StarVault. Таким образом, StarVault предоставляет решение для управления идентификационными данными с помощью механизма **Identity secrets engine**. Для получения дополнительной информации о механизме идентификации секретов и о том, как он используется, обратитесь к документации по механизму идентификации секретов.

1. Сущности и псевдонимы

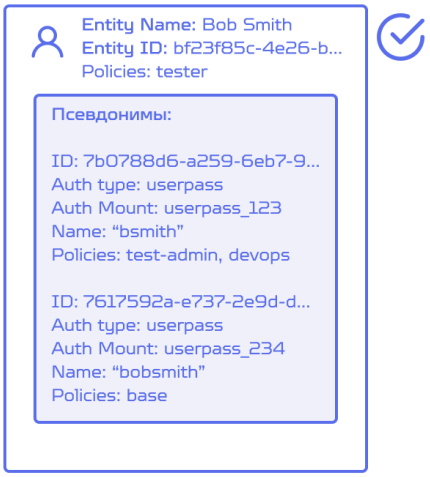
У каждого пользователя может быть несколько учетных записей у различных поставщиков удостоверений, и StarVault поддерживает многих из этих поставщиков для аутентификации. StarVault Identity может привязывать аутентификацию с помощью различных методов аутентификации к единому представлению. Это представление объединенного идентификатора называется сущностью. Соответствующие учетные записи у поставщиков аутентификации могут быть сопоставлены как псевдонимы. Таким образом, каждая сущность состоит из нуля или более псевдонимов. Сущность не может иметь более одного псевдонима для определенного сервера аутентификации.

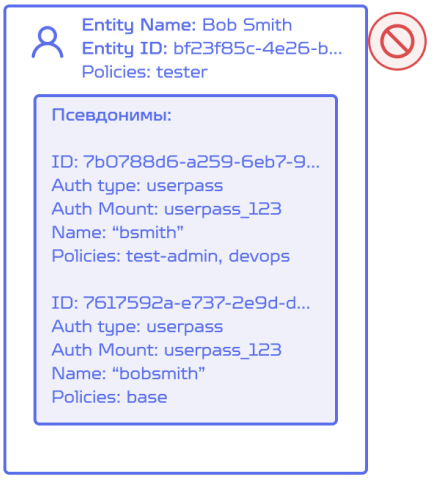
> Например, пользователь с учетными записями как в userpass, так и в LDAP может быть сопоставлен с одной сущностью в StarVault с двумя псевдонимами, один из которых имеет тип userpass, а другой - тип LDAP.



Однако, если оба псевдонима созданы при одном и том же подключении для аутентификации, например, при подключении на LDAP, оба псевдонима не могут быть

сопоставлены одной и той же сущности. Псевдонимы могут иметь один и тот же тип аутентификации, при условии, что авторские настройки разные, и при этом они могут быть связаны с одной и той же сущностью. На рисунках ниже показаны как допустимые, так и недопустимые сценарии.





Когда клиент проходит аутентификацию с помощью любого сервера с учетными данными (кроме сервера с токенами), StarVault создает новую сущность. К ней присваивается новый псевдоним, если соответствующая сущность еще не существует. Идентификатор сущности будет привязан к аутентифицированному токену. Когда используются такие токены, их идентификаторы сущностей регистрируются в журнале аудита, что позволяет отслеживать действия, выполняемые конкретными пользователями.

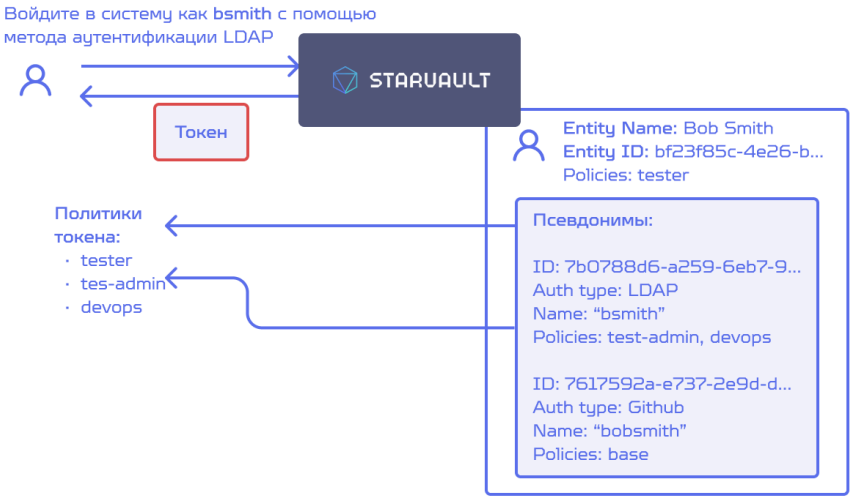
Сущность StarVault используется для подсчета количества клиентов StarVault. Чтобы узнать больше о количестве клиентов, обратитесь к документации по подсчету клиентов.

2. Управление сущностями

Сущности в StarVault автоматически не извлекают идентификационную информацию из любого места. Операторы должны управлять ею в явном виде. Таким образом, это позволяет гибко управлять количеством сущностей, синхронизируемых со StarVault. В некотором смысле, StarVault будет служить кэшем идентификационных данных, а не источником идентификационных данных.

3. Политики сущностей

Политики хранилища могут быть назначены сущностям, которые будут предоставлять дополнительные разрешения для токена поверх существующих политик для токена. Если токен, представленный в запросе API, содержит идентификатор сущности и если для неё установлен набор политик, то токен также будет способен выполнять действия, разрешенные политиками для сущности.



Это изменение схемы с точки зрения того, когда будут оцениваться политики токена. До появления идентификации, имена политик в токене были неизменяемыми (но не содержимое этих политик). С политиками сущностей, наряду с неизменяемым набором имен политик в токене, оценка политик, применимых к токену, через его идентификатор будет выполняться во время запроса. Это добавляет гибкость в управлении поведением уже выпущенных токенов.

Важно отметить, что политики для сущности являются лишь средством предоставления дополнительных возможностей, а не заменой политик для токена. Чтобы узнать полный набор возможностей токена с соответствующим идентификатором сущности, следует учитывать политики для токена.

Будьте осторожны при предоставлении разрешений конечным точкам идентификации, не предназначенным только для чтения.

- Если пользователь может изменять сущность, он может предоставить ему дополнительные привилегии с помощью политик.
- Если пользователь может изменять псевдоним, под которым он может входить в систему, он может привязать его к сущности с более высокими привилегиями.
- Если пользователь может изменить членство в группе, он может добавить свою сущность в группу с более высокими привилегиями.

4. Монтирование привязанных псевдонимом

StarVault поддерживает несколько серверных систем аутентификации. Также позволяет использовать один и тот же тип серверной системы аутентификации на разных путях подключения. Псевдоним пользователя будет уникальным в пределах подключенных серверных систем. Но хранилище идентификаторов должно однозначно различать конфликтующие имена псевдонимов для разных подключений поставщиков идентификационных данных. Следовательно, имя псевдонима в сочетании с атрибутами для установки серверной части аутентификации служат уникальным идентификатором псевдонима.

В таблице ниже показано, какую информацию использует каждый из поддерживаемых методов аутентификации для формирования псевдонима. Это идентифицирующая информация, которая используется для сопоставления или создания сущности. Если никакие сущности явно не созданы или не объединены, то для каждого объекта в правой части таблицы будет неявно создана одна сущность, когда он используется для аутентификации в определенной точке подключения auth.

Метод аутентификации	**Имя, передаваемое методом аутентификации**
AppRole	Role ID
Cloud Foundry	App ID
Google Cloud	Настраивается с помощью `iam_alias` на один из следующих параметров: Role ID (по умолчанию), Service account unique ID
JWT/OIDC	Настраивается с помощью `user_claim` к одному из представленных пунктов формулы (без значения по умолчанию)
Kerberos	Username
Kubernetes	Настраивается с помощью `alias_name_source` на один из следующих параметров: UID учетной записи службы (по умолчанию), имя учетной записи службы

Метод аутентификации	Имя, передаваемое методом аутентификации
LDAP	Username
RADIUS	Username
TLS Certificate	Subject CommonName
Token	`entity_alias`, если указано
Username (userpass)	Username

5. Методы локальной аутентификации

Все методы аутентификации будут генерировать сущность по умолчанию при выдаче токена, за исключением хранилища токенов. Это применимо как для подключений, которые совместно используются кластерами, так и для подключений локальной аутентификации кластера (с использованием `local=true`), когда используется репликация хранилища. Если целью обозначения метода аутентификации как `local` было соответствие руководящим принципам GDPR, то необходимо соблюдать осторожность, чтобы не указывать данные, относящиеся к локальному подключению для аутентификации, или псевдонимы для локального подключения для аутентификации в метаданных связанной сущности.

6. Неявные сущности

Операторы могут заранее создавать сущности для всех пользователей с авторизацией и назначать им политики, чтобы при входе пользователей в систему желаемые возможности токенов через сущности были уже назначены. Но если этого не сделать, то после успешного входа пользователя в систему с любого из серверов аутентификации StarVault создаст новую сущность и назначит псевдоним для успешного входа в систему.

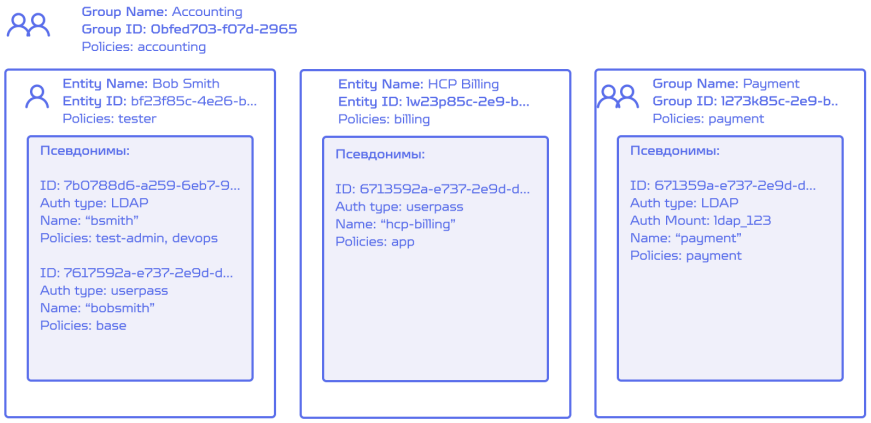
Обратите внимание, что токены, созданные с использованием серверной части проверки подлинности токенов, обычно не содержат никакой связанной идентификационной информации. Существующая или новая неявная сущность может быть назначена с помощью параметра `entity_alias` при создании токена с использованием роли токена с настроенным списком `allowed_entity_aliases`.

7. Аудит идентификационных данных

Если токен, используемый для выполнения вызовов API, имеет связанный идентификатор сущности, он также будет записан в журнал аудита. Это позволяет отслеживать действия, выполненные конкретными пользователями.

8. Идентификационные группы

StarVault identity поддерживает группы. Группа может содержать несколько сущностей в качестве своих членов. У группы также могут быть подгруппы. Политики, установленные для группы, предоставляются всем членам группы. Во время запроса, когда идентификатор сущности токена проверяется на предмет политик, к которым у него есть доступ, политики, унаследованные из-за членства в группах, предоставляются вместе с политиками для самой сущности.



9. Групповые иерархические разрешения

Сущности могут быть прямыми членами групп, и в этом случае они наследуют политики групп, к которым они принадлежат. Сущности также могут быть косвенными членами групп.

> Например, если группа А содержит группу Б качестве подгруппы, то члены группы Б являются косвенными членами группы А. Следовательно, члены группы Б будут иметь доступ к политикам как для группы А, так и для группы Б.

10. Внешние vs внутренние группы

По умолчанию группы, созданные в identity store, называются **внутренними группами**. Управление членством в этих группах должно выполняться вручную.

Группа также может быть создана как **внешняя группа**. В этом случае членство сущности в группе управляется полуавтоматически. Внешняя группа служит для сопоставления с группой, которая находится за пределами хранилища идентификаторов. Внешние группы могут иметь один (и только один) псевдоним. Этот псевдоним должен соответствовать понятию группы, которая находится за пределами хранилища идентификаторов.

Например, группы в LDAP и команды в userpass. Имя пользователя в LDAP, принадлежащее к группе в LDAP, может автоматически получить идентификатор сущности в качестве члена группы в StarVault при входе в систему и обновлении токена. Это работает, только если группа в StarVault является внешней группой и имеет псевдоним, соответствующий группе в LDAP. Если пользователь удаляется из группы в LDAP, это изменение отражается в StarVault только при последующем входе в систему или обновлении.

Для получения информации о механизме идентификации секретов обратитесь к механизму идентификации секретов.

2025 orionsoft. Все права защищены.

Аренда, обновление и отзыв

С каждым токеном аутентификации типа `dynamic secret` и `service` StarVault создает метаданные, содержащие такую информацию, как срок действия, возможность продления и т.д. StarVault обещает, что данные будут действительны в течение заданного срока, или срока службы (TTL). Как только срок аренды истекает, StarVault может автоматически отозвать данные, и пользователь секрета больше не может быть уверен в том, что они действительны.

Выгода очевидна: пользователям секрета необходимо регулярно обращаться в StarVault, чтобы либо продлить аренду (если это разрешено), либо запросить замену секрета. Это повышает ценность журналов аудита StarVault, а также значительно упрощает передачу ключей.

Для всех динамических секретов в StarVault требуется аренда. Даже если предполагается, что данные будут действительны вечно, требуется аренда, чтобы заставить пользователя регулярно регистрироваться.

Помимо продления, аренда может быть отозвана. При аннулировании договора аренды этот секрет немедленно становится недействительным и предотвращает любые дальнейшие продления.

Аннулирование может быть произведено вручную через API, с помощью команды `starvault lease revoke`, пользовательского интерфейса (UI) на вкладке Access или автоматически с помощью StarVault. Когда срок действия договора аренды истечет, StarVault автоматически аннулирует этот договор аренды. Когда токен аннулируется, StarVault аннулирует все договоры аренды, которые были созданы с использованием этого токена.

Серверная часть Key/Value, которая хранит произвольные секреты, не выдает договоры аренды, хотя иногда возвращает информацию о продолжительности аренды; дополнительную информацию смотрите в документации.

1. Идентификаторы аренды

При чтении динамического файла секретов, например, с использованием команды `starvault read`, StarVault всегда возвращает `lease_id`. Этот идентификатор используется в таких командах, как продление срока аренды хранилища и отмена аренды хранилища, для управления арендой файла секретов.

2. Сроки аренды и ее продление

Вместе с идентификатором аренды можно прочитать срок аренды. Срок действия аренды - это время в секундах, в течение которого действует аренда. Пользователь должен продлить срок аренды секрета в течение этого времени.

При продлении срока аренды пользователь может запросить определенное количество времени, в течение которого аренда будет действительна. Это время называется приращением (`increment`).

Время приращения - это не дополнительное время, прибавляемое к сроку аренды, а новый срок от текущего момента времени.

Например, `starvault lease renew -increment=3600 my-lease-id` запросит, чтобы срок действия аренды был изменен на 1 час (3600 секунд).

Ориентация приращения на текущее время, а не на момент окончания срока аренды, упрощает пользователям сокращение срока аренды, если им на самом деле не нужны учетные данные на весь возможный период аренды, что позволяет быстрее истечь сроку действия этих учетных данных и раньше очистить ресурсы.

Запрашиваемое увеличение является исключительно рекомендуемым. Серверная часть, ответственная за секрет, может полностью проигнорировать его. Для большинства секретов серверная часть делает все возможное, чтобы соблюдать это увеличение, но часто ограничивает его, чтобы обеспечить периодическое обновление.

В результате следует тщательно изучить возвращаемое значение продления аренды, чтобы определить, что представляет собой новая аренда.

Чтобы реализовать логику обновления токена в коде вашего приложения, обратитесь к примеру кода в документе Authentication doc.

3. Аннулирование на основе префикса

В дополнение к отмене одного файла секретов, операторы с надлежащим контролем доступа могут отозвать несколько секретов на основе префикса идентификатора аренды.

Идентификаторы аренды структурированы таким образом, что их префиксом всегда является путь, по которому был запрошен секрет. Это позволяет отзывать деревья секретов.

Для получения дополнительной информации о команде аннулирования, пожалуйста, ознакомьтесь с документацией по команде отзыва аренды.

Это очень полезно, если происходит вторжение в конкретную систему: все секреты определенного серверного интерфейса или определенной настроенной серверной части могут быть быстро и легко отозваны.

2025 orionsoft. Все права защищены.

Провайдер OIDC

В документе представлена концептуальная информация о функции поставщика идентификационных данных StarVault OpenID Connect (OIDC). Эта функция позволяет клиентским приложениям, использующим протокол OIDC, использовать источник идентификации StarVault и широкий спектр методов аутентификации при проверке подлинности конечных пользователей.

1. Параметры конфигурации

В следующих разделах документа приводятся подробности реализации каждого ресурса, позволяющего конфигурировать StarVault в качестве поставщика идентификационных данных OIDC.

1.1. Провайдеры OIDC

Каждое пространство имен StarVault будет содержать встроенный ресурс провайдера с именем default. Провайдер по умолчанию позволит всем клиентским приложениям в пространстве имен использовать его для потоков OIDC. Провайдер по умолчанию можно изменять, но нельзя удалять.

Кроме того, пространство имен StarVault может содержать несколько ресурсов провайдеров. Каждый настроенный провайдер будет публиковать API, перечисленные в разделе потоков OIDC. API будут обслуживаться через маршрутизацию на основе путей бэкенда по адресу прослушивания StarVault.

Провайдер имеет следующие параметры конфигурации:

- **URL-адрес эмитента**: используется в утверждении идентификационных токенов `iss`.
- **Разрешенные идентификаторы клиентов**: ограничивает, какие клиенты могут получить доступ к провайдеру.
- **Поддерживаемые области видимости**: ограничивает, какая идентификационная информация доступна в виде утверждений.

Параметр URL эмитента необходим для проверки ID-токенов клиентами. Если параметр URL не указан явно, по умолчанию будет использоваться URL с api_addr StarVault в качестве компонента `scheme://host:port` и `/v1/identity/oidc/provider/:name` в качестве компонента path. Это означает, что токены, выпущенные провайдером в указанном кластере StarVault, должны быть проверены в этом же кластере. Если URL-адрес эмитента указан

явно, он должен указывать на экземпляр StarVault, доступный клиентам по сети для проверки ID-токенов.

Параметр "Разрешенные идентификаторы клиентов" использует список идентификаторов клиентов, которые были сгенерированы StarVault в процессе регистрации клиентов. По умолчанию все клиенты будут запрещены. Указание `*` в качестве значения параметра позволит всем клиентам использовать провайдера.

Параметр scopes использует список ссылок на именованные области видимости ресурсов. Предоставленные значения можно обнаружить по ключу `scopes_supported` в документе обнаружения OIDC провайдера. По умолчанию провайдеру доступен диапазон `openid`. Более подробную информацию о диапазоне `openid` см. в разделе диапазонов ниже.

1.2. Области видимости

Провайдеры могут ссылаться на ресурсы scope через параметр scopes_supported, чтобы сделать конкретную идентификационную информацию доступной в виде утверждений.

У диапазона будут следующие параметры конфигурации:

- **Описание**: информация об идентификации, захваченная областью действия
- **Шаблон**: сопоставляет отдельные утверждения с информацией об идентификации StarVault

Параметр шаблона использует преимущества шаблонизации на основе JSON, используемой токенами идентификации для сопоставления утверждений. Это означает, что параметр принимает JSON-строку произвольной структуры, в которой значения могут быть заменены на конкретную идентификационную информацию. Параметры шаблона, которые не присутствуют для идентификатора StarVault, будут пропущены в результирующих формулах без ошибки.

Далее приведены примеры шаблона JSON для области видимости:

1. Используйте следующий шаблон, если необходимо привести в соответствие параметры только для указанного монтирования:

```JSON
{
    "username": {{identity.entity.aliases.$MOUNT_ACCESSOR.name}},
    "contact": {
        "email": {{identity.entity.metadata.email}},
        "phone_number": {{identity.entity.metadata.phone_number}}
    },
    "groups": {{identity.entity.groups.names}}
}
```

Если параметр `$MOUNT_ACCESSOR` не указан, то параметры LDAP не отразятся.

2. Используйте следующий шаблон для привязки к последней сессии:

```JSON
{
    "username": {{identity.entity.aliases.latest.name}},
    "contact": {
        "email": {{identity.entity.metadata.email}},
        "phone_number": {{identity.entity.metadata.phone_number}}
    },
    "groups": {{identity.entity.groups.names}}
}
```

Полный список параметров шаблона приведен в следующей таблице:

Имя параметра	**Описание**
`identity.entity.id`	Идентификатор сущности
`identity.entity.name`	Имя сущности
`identity.entity.groups.ids`	Идентификаторы групп, членом которых является сущность
`identity.entity.groups.names`	Имена групп, членом которых является сущность
`identity.entity.metadata`	Метаданные, связанные с сущностью
`identity.entity.metadata.<metadata key>`	Метаданные, связанные с сущностью для указанного ключа
`identity.entity.aliases.<mount accessor>` или `identity.entity.aliases.latest`	Метаданные, связанные с псевдонимом для указанного монтирования или связанные с последним alias из сессии
`identity.entity.aliases.<mount accessor>.id` или `identity.entity.aliases.latest.id`	Идентификатор псевдонима сущности для указанного монтирования или связанного с последним alias из сессии
`identity.entity.aliases.<mount accessor>.name` или `identity.entity.aliases.latest.name`	Имя псевдонима сущности для указанного монтирования или связанное с последним alias из сессии
`identity.entity.aliases.<mount accessor>.metadata.<metadata key>` или `identity.entity.aliases.latest.metadata.<metadata key>`	Метаданные, связанные с псевдонимом для указанного монтирования и ключом метаданных или связанные с последним alias из сессии и ключом метаданных
`identity.entity.aliases.<mount accessor>.custom_metadata` или `identity.entity.aliases.latest.custom_metadata`	Пользовательские метаданные, связанные с псевдонимом для указанного монтирования или связанные с последним alias из сессии

Имя параметра	Описание
`identity.entity.aliases.<mount accessor>.custom_metadata.<custom_metadata key>` или `identity.entity.aliases.latest.custom_metadata.<custom_metadata key>`	Пользовательские метаданные, связанные с псевдонимом для указанного монтирования и пользовательским ключом метаданных или связанные с последним alias из сессии и пользовательским ключом
`time.now`	Текущее время в виде интегральных секунд с начала Эпохи
`time.now.plus.<duration>`	Текущее время плюс строка формата длительности
`time.now.minus.<duration>`	Текущее время минус строка формата длительности

Для отдельного провайдера может быть доступно несколько именованных диапазонов. Обратите внимание, что ключи верхнего уровня в шаблоне JSON могут конфликтовать с ключами другого диапазона. Когда диапазоны становятся доступными для провайдера, их шаблоны проверяются на наличие конфликтов верхнего уровня. При обнаружении конфликтов оператору StarVault будет выдано предупреждение. Это может привести к ошибке, если диапазоны запрашиваются в запросе аутентификации OIDC.

Область openid - это уникальная область действия, которая не может быть изменена или удалена. Эта область будет существовать в StarVault и поддерживаться каждым провайдером по умолчанию. Область openid представляет собой минимальный набор утверждений, требуемых спецификацией OIDC для включения в идентификационные токены. Поэтому шаблоны не могут содержать ключи верхнего уровня, которые перезаписывают формулы, созданные в области openid.

Ниже определено сопоставление ключей и значений формул для области openid:

- `iss` — сконфигурированный эмитент провайдера
- `sub` — уникальный идентификатор сущности пользователя хранилища
- `aud` — идентификатор клиента
- `iat` — время выпуска токена
- `exp` — время выпуска токена + ID-токен TTL

1.3. Клиентские приложения

Ресурс клиента представляет приложение, которое делегирует аутентификацию конечного пользователя на StarVault с помощью протокола OIDC. Информация, предоставленная клиентским ресурсом, может быть использована для настройки доверенной стороны OIDC.

Клиент имеет следующие параметры конфигурации:

- **Перенаправление URIs**: ограничивает допустимые URI перенаправления в запросе аутентификации
- **Задания**: определяет, кто может аутентифицироваться на клиенте
- **Ключ**: используется для подписи ID-токенов
- **ID токен TTL**: определяет время жизни ID-токенов
- **Токен доступа TTL**: определяет время жизни токенов доступа
- **Тип клиента**: определяет способность клиента поддерживать конфиденциальность учетных данных

Параметр `key` опционален. Ключ будет использоваться для подписи ID-токенов клиента. Параметр не может быть изменен после создания. Если параметр не указан, по умолчанию используется встроенный ключ по умолчанию.

Параметр `client_id` генерируется и возвращается после успешной регистрации клиента. `client_id` уникально идентифицирует клиента. Значение `client_id` — строка с 32 случайными символами из набора символов base62.

Один из URI перенаправления клиента должен точно соответствовать параметру `redirect_uri`, используемому в запросе аутентификации, инициированном клиентом.

1.3.1. Типы клиентов

Ресурс клиента имеет параметр `client_type`, который определяет тип клиента OAuth 2.0 в зависимости от способности сохранять конфиденциальность учетных данных. В следующих разделах подробно описаны различия между конфиденциальными и публичными клиентами в StarVault.

1.3.2. Конфиденциальные клиенты

Конфиденциальные клиенты способны сохранять конфиденциальность своих учетных данных. У конфиденциальных клиентов есть секрет клиента (`client_secret`). `client_secret` имеет префикс `hvo_secret`, за которым следуют 64 случайных символа из набора символов base62.

Конфиденциальные клиенты могут использовать ключ доказательства для обмена кодами (PKCE) во время потока кода авторизации.

Конфиденциальные клиенты должны аутентифицироваться на конечной точке токена с помощью метода аутентификации `client_secret_basic` или `client_secret_post` .

1.3.3. Публичные клиенты

Публичные клиенты не способны сохранять конфиденциальность своих учетных данных. Поэтому у публичных клиентов нет `client_secret`.

Публичные клиенты должны использовать ключ доказательства для обмена кодами (PKCE) во время потока кода авторизации.

Публичные клиенты используют метод аутентификации `none`.

1.4. Назначения

Клиенты ссылаются на ресурсы назначений с помощью параметра `assignments`. Этот параметр ограничивает набор пользователей StarVault, которым разрешена аутентификация. Назначения связанного клиента проверяются во время запроса на аутентификацию, гарантируя, что идентификатор StarVault, связанный с запросом, является членом сущностей или групп назначения.

Каждое пространство имен StarVault содержит встроенный ресурс назначения с именем `allow_all`. Назначение `allow_all` позволяет всем сущностям StarVault проходить аутентификацию через клиента. Назначение `allow_all` не может быть изменено или удалено.

1.5. Ключи

Клиенты обращаются к ключевым ресурсам через параметр `key`. Этот параметр задает ключ, который будет использоваться для подписи ID-токенов клиента. В настоящее время ключ, на который ссылается клиент, не может быть изменен.

Каждое пространство имен StarVault будет содержать встроенный ключевой ресурс с именем `default`. Ключ по умолчанию может быть изменен, но не удален. Клиенты, не указавшие параметр ключа во время создания, будут использовать ключ по умолчанию.

Ключ по умолчанию (`default`) будет иметь следующую конфигурацию:

- `algorithm` — `RS256`
- `allowed_client_ids` — `*`
- `rotation_period` — `24h`
- `verification_ttl` — `24h`

2. Поток OIDC

В настоящее время функция StarVault OIDC Provider поддерживает только поток кодов авторизации.

В следующих разделах приведены подробности реализации OIDC-совместимых API, предоставляемых поставщиками StarVault OIDC.

Провайдеры StarVault OIDC позволяют зарегистрированным клиентам аутентифицировать и получать идентификационную информацию (или "утверждения") для своих конечных пользователей. Для этого провайдеры предоставляют API и поведение, необходимые для удовлетворения спецификации OIDC для потока кода авторизации. Все клиенты рассматриваются как первые лица. Это означает, что конечные пользователи не должны предоставлять провайдеру согласие, как указано в разделе 3.1.2.4 спецификации OIDC. Провайдер будет предоставлять информацию клиентам до тех пор, пока конечный пользователь имеет ACL-доступ к провайдеру и его личность была авторизована с помощью назначения.

Провайдеры StarVault OIDC внедряют ключ доказательства для обмена кодами (PKCE) для защиты от атак перехвата кода авторизации. PKCE требуется для публичных типов клиентов (`public`) и необязателен для конфиденциальных типов клиентов (`confidential`).

2.1. Конфигурация OpenID

Каждый провайдер предлагает неаутентифицированную конечную точку, которая облегчает обнаружение OIDC. Все необходимые метаданные, перечисленные в OpenID Provider Metadata, включены в документ обнаружения. Кроме того, включены рекомендуемые метаданные `userinfo_endpoint` и `scopes_supported` .

2.2. Ключи

Каждый провайдер предлагает неаутентифицированную конечную точку, которая предоставляет публичную часть ключей, используемых для подписания ID-токенов. Ключи публикуются в формате JSON Web Key Set (JWKS). Набор ключей для отдельного провайдера содержит ключи, на которые ссылаются все клиенты через параметр конфигурации `allowed_client_ids` . Заголовок `Cache-Control` устанавливается на основе ответов, позволяя клиентам обновлять свои ключи при ротации. Максимальный срок действия заголовка (`max-age`) устанавливается на основе самого раннего времени ротации любого из ключей в наборе ключей.

2.3. Конечная точка авторизации

Каждый провайдер предлагает аутентифицированную конечную точку авторизации. Конечная точка авторизации для каждого провайдера добавляется в политику StarVault по умолчанию с помощью пути `identity/oidc/provider/+/authorize` . Конечная точка

включает в себя все необходимые параметры запроса аутентификации в качестве входных данных.

Конечная точка проверяет клиентские запросы и убеждается, что все необходимые параметры присутствуют и действительны. `redirect_uri` запроса сверяется с `redirect_uris` клиента. Запрашивающая сущность StarVault проверяется на соответствие назначениям клиента (`assignments`). Для недействительных запросов возвращается соответствующий код ошибки.

При успешной проверке запроса генерируется код авторизации. Код авторизации — одноразовый и кэшируется со временем жизни около 5 минут, что снижает риск утечки. Ответ, содержащий исходное состояние `state` , представленное клиентом, и `code` , будет возвращен в пользовательский интерфейс StarVault, инициировавший запрос. StarVault выдаст HTTP 302 редирект на `redirect_uri` запроса, который включает `code` и `state` в качестве параметров запроса.

2.4. Конечная точка токенов

Каждый провайдер предлагает конечную точку токена. Конечная точка может быть неаутентифицированной в StarVault, но аутентифицируется с помощью запроса `client_secret` , как описано в разделе "Аутентификация клиента". Конечная точка получает все необходимые параметры запроса токена в качестве входных данных. Конечная точка проверяет клиентские запросы и обменивается кодом авторизации на ID-токен и токен доступа. Кэш кодов авторизации будет сверен с кодом, представленным в процессе обмена. Для всех недействительных запросов возвращаются соответствующие коды ошибок.

ID-токен генерируется и возвращается после успешной аутентификации клиента и проверки запроса. ID-токен будет содержать комбинацию обязательных и настраиваемых требований. Требуемые утверждения перечислены в разделе Области видимости выше для области видимости `openid` . Настраиваемые утверждения заполняются шаблонами, связанными с областями, указанными в запросе аутентификации, который сгенерировал код авторизации.

Также генерируется маркер доступа, который возвращается после успешной аутентификации клиента и проверки запроса. Токен доступа представляет собой пакетный токен StarVault с политикой, предоставляющей только доступ на чтение к конечной точке userinfo выдающего провайдера. Токен доступа также имеет TTL, определяемый `access_token_ttl` запрашивающего клиента.

2.5. Конечная точка UserInfo

Каждый провайдер предоставляет аутентифицированную конечную точку userinfo. Конечная точка принимает токен доступа, полученный от конечной точки token, в качестве маркера

предъявителя. Ответ userinfo представляет собой объект JSON с типом содержимого `application/json`. Объект JSON содержит утверждения для сущности StarVault, связанной с маркером доступа. Возвращаемые утверждения определяются диапазонами, запрошенными в запросе аутентификации, в результате которого был получен маркер доступа. `sub` всегда возвращается как идентификатор сущности в ответе userinfo.

2025 orionsoft. Все права защищены.