Бэкенд для хранения данных Integrated storage (Raft)

Бэкэнд Integrated Storage используется для хранения данных StarVault. В отличие от других бэкэндов хранения, Integrated Storage не работает с единственным источником данных. Вместо этого все узлы в кластере StarVault имеют реплицированную копию данных StarVault. Данные реплицируются на всех узлах с помощью алгоритма консенсуса Raft.

- **Высокая доступность** - бэкэнд файловой системы поддерживает высокую доступность.

```
storage "raft" {
  path = "/path/to/raft/data"
  node_id = "raft_node_1"
}
cluster_addr = "http://127.0.0.1:8201"
```

При использовании бэкенда Integrated Storage необходимо указать `cluster_addr` для указания адреса и порта, которые будут использоваться для связи между узлами в кластере Raft.

При использовании бэкенда Integrated Storage отдельный бэкенд `ha_storage` объявлять нельзя.

При использовании бэкэнда Integrated Storage настоятельно рекомендуется установить значение `disable_mlock` в `true`, а также отключить подачку памяти в системе.

1. Параметры raft

- `path` (`string: ""`) — путь в файловой системе, где хранятся все данные StarVault. Это значение можно переопределить, установив переменную окружения `STARVAULT_RAFT_PATH`.
- `node_id` (`string: ""`) — идентификатор узла в кластере Raft. Это значение можно переопределить, установив переменную окружения `STARVAULT_RAFT_NODE_ID`.
- `performance_multiplier` (`integer: 0`) — целочисленный множитель, используемый серверами для масштабирования ключевых временных параметров Raft. Настройка этого параметра влияет на время, необходимое StarVault для обнаружения сбоев в работе ведущего узла и проведения выборов ведущего узла, за счет того, что для повышения производительности требуется больше ресурсов сети и процессора. Если

опустить это значение или установить его равным 0, будет использоваться время по умолчанию, описанное ниже. Меньшие значения используются для ужесточения синхронизации и повышения чувствительности, в то время как большие значения ослабляют синхронизацию и снижают чувствительность.

По умолчанию StarVault будет использовать менее производительный тайминг, подходящий для минимальных серверов StarVault, что в настоящее время эквивалентно установке значения 5 (это значение может быть изменено в будущих версиях StarVault, в зависимости от изменения целевого профиля минимального сервера). Если установить значение 1, Raft перейдет в режим максимальной производительности и будет рекомендован для производственных серверов StarVault. Максимально допустимое значение - 10.

- `trailing_logs` (`integer: 10000`) — количество записей логов останется в хранилище логов на диске после создания снимка. Этот параметр следует настраивать только в тех случаях, когда ведомые узлы не могут догнать ведущие из-за очень большого размера снимка и высокой пропускной способности записи, что приводит к усечению журнала до полной установки снимка. Если нужно использовать этот параметр для восстановления кластера, подумайте о снижении пропускной способности записи или объема данных, хранящихся на StarVault. По умолчанию установлено значение 10000, которое подходит для всех нормальных рабочих нагрузок. Метрика `trailing_logs` — не то же самое, что `max_trailing_logs`.
- `snapshot_threshold` (`integer: 8192`) — минимальное количество записей фиксации Raft между снимками, которые сохраняются на диск. Это низкоуровневый параметр, который редко нуждается в изменении. Очень загруженные кластеры, испытывающие избыточный ввод-вывод на диск, могут увеличить это значение, чтобы уменьшить ввод-вывод на диск и минимизировать вероятность того, что все серверы будут делать снимки одновременно. При увеличении этого значения происходит обмен дискового ввода-вывода на дисковое пространство, поскольку журнал логов будет значительно увеличиваться, а место в файле `raft.db` не может быть восстановлено до следующего моментального снимка. При значительном увеличении этого параметра серверы могут дольше восстанавливаться после сбоев или аварийного переключения, поскольку потребуется воспроизвести больше логов.
- `snapshot_interval` (`integer: 120 seconds`) — интервал между снимками. Raft контролирует, требуется ли операция моментального снимка. Raft случайным образом распределяет снимки между настроенным интервалом и удвоенным интервалом, чтобы не допустить одновременного выполнения снимка всем кластером. По умолчанию интервал между снимками составляет 120 секунд.
- `retry_join` (`list: []`) — набор данных о соединении с другим узлом в кластере, который используется для того, чтобы помочь узлам найти ведущий узел для присоединения к кластеру. Может быть один или несколько блоков `retry_join`.

Если детали подключения всех узлов кластера известны заранее, то можете включить эти блоки, чтобы узлы могли автоматически присоединиться к кластеру Raft. Как только один из узлов будет инициализирован как ведущий, остальные узлы будут использовать свою конфигурацию `retry_join`, чтобы найти ведущий узел и присоединиться к кластеру. Обратите внимание, что при использовании Shamir seal присоединившиеся узлы все равно придется разблокировать вручную. Подробнее о параметрах секции `retry_join` см. в разделе Параметры секции retry_join.

- `max_entry_size` (`integer: 1048576`) — максимальное количество байт для записи Raft. Это относится как к операциям Put, так и к транзакциям. Любая операция put или транзакция, превышающая это конфигурационное значение, приведет к неудаче соответствующей операции. Raft рекомендует максимальный размер данных в записи журнала Raft. Это основано на текущей архитектуре, времени по умолчанию и т. д. Integrated Storage также использует размер части, который является порогом, используемым для разбиения большого значения на части. По умолчанию размер части равен максимальному размеру записи журнала Raft. Значение по умолчанию для этой конфигурации равно 1048576 - двукратному размеру части.
- `autopilot_reconcile_interval` (`string: "10s"`) — интервал, после которого автопилот будет фиксировать любые изменения состояния. Изменение состояния может означать множество вещей: недавно присоединившийся узел-избиратель, первоначально добавленный автопилотом в кластер Raft как не-избиратель, успешно прошел период стабилизации, тем самым получив право стать избирателем; узел, который стал неработоспособным и должен представить в API своё состояние; узел был помечен как неработоспособным, требующий выселения из конфигурации Raft, и т.д.
- `autopilot_update_interval` (`string: "2s"`) — интервал, через который автопилот будет опрашивать StarVault на предмет обновления интересующей его информации. Сюда входят данные: конфигурация автопилота, текущее состояние автопилота, конфигурация raft, известные серверы, последний индекс raft и статистика для всех известных серверов. Полученная автопилотом информация будет использована для расчета его следующего состояния.

2. Параметры секции `retry_join`

- `leader_api_addr` (`string: ""`) — адрес возможного ведущего узла.
- `auto_join` (`string: ""`) ` — конфигурация облачного автосоединения с использованием синтаксиса go-discover.
- `auto_join_scheme` (`string: ""`) — схема протокола URI для адресов, обнаруженных через автосоединение. Доступные значения: `http` или `https`. Опциональный параметр.
- `auto_join_port` (`uint: ""`) — порт, используемый для адресов, обнаруженных с помощью автосоединения. Опциональный параметр.

- `leader_tls_servername` (`string: ""`) — имя сервера TLS при подключении по HTTPS. Должно совпадать с одним из имен в DNS SAN сертификата удаленного сервера. См. также Интегрированное хранилище и TLS.
- `leader_ca_cert_file` (`string: ""`) — путь к файлу с сертификатом CA возможного ведущего узла.
- `leader_client_cert_file` (`string: ""`) — путь к файлу с сертификатом клиента для ведомого узла, чтобы установить аутентификацию клиента с возможным ведущем узлом.
- `leader_client_key_file` (`string: ""`) — путь к файлу с ключом клиента для ведомого узла для установления аутентификации клиента с возможным ведущем узлом.
- `leader_ca_cert` (`string: ""`) — сертификат CA возможного ведущего узла.
- `leader_client_cert` (`string: ""`) — сертификат клиента для ведомого узла для установления аутентификации клиента с возможным ведущем узлом.
- `leader_client_key` (`string: ""`) — клиентский ключ для ведомого узла для установления аутентификации клиента с возможным ведущем узлом.

Каждый блок `retry_join` может предоставлять сертификаты TLS через пути к файлам или в виде однострочного значения строки сертификата с новыми строками, разграниченными знаком `\n`, но не комбинацию того и другого. Каждая блок конфигурации `retry_join` может содержать либо значение `leader_api_addr`, либо значение конфигурации облака `auto_join`, но не оба. Если указано значение `auto_join`, StarVault будет автоматически пытаться обнаружить и разрешить потенциальные адреса ведущегой узела Raft с помощью go-discover.

По умолчанию StarVault будет пытаться связаться с обнаруженными пирами, используя HTTPS и порт 8200. Операторы могут переопределить эти параметры с помощью полей `auto_join_scheme` и `auto_join_port` соответственно.

Пример конфигурации:

```
storage "raft" {
  path    = "/Users/foo/raft/"
  node_id = "node1"

  retry_join {
    leader_api_addr = "http://127.0.0.2:8200"
    leader_ca_cert_file = "/path/to/ca1"
    leader_client_cert_file = "/path/to/client/cert1"
    leader_client_key_file = "/path/to/client/key1"
  }
  retry_join {
    leader_api_addr = "http://127.0.0.3:8200"
    leader_ca_cert_file = "/path/to/ca2"
    leader_client_cert_file = "/path/to/client/cert2"
```

```
    leader_client_key_file = "/path/to/client/key2"
  }
  retry_join {
    leader_api_addr = "http://127.0.0.4:8200"
    leader_ca_cert_file = "/path/to/ca3"
    leader_client_cert_file = "/path/to/client/cert3"
    leader_client_key_file = "/path/to/client/key3"
  }
  retry_join {
    auto_join = "provider=aws region=eu-west-1 tag_key=StarVault tag_value=...
access_key_id=... secret_access_key=..."
  }
}
```

Встроенное хранилище

StarVault поддерживает несколько вариантов длительного хранения информации. Каждый бэкенд хранения имеет плюсы и минусы, преимущества и неизбежные компромиссы. Например, одни поддерживают режим высокой доступности, в то время как другие обеспечивают более надежный процесс резервного копирования и восстановления.

В StarVault доступно встроенное хранилище. Этот бэкенд хранения не зависит от сторонних систем и поддерживает высокую доступность, а также процессы резервного копирования и восстановления.

1. Протокол консенсуса

Встроенное хранилище StarVault использует протокол консенсуса для обеспечения Согласованности данных (как определено в теореме CAP). Протокол консенсуса основан на "Raft: в поисках понятного алгоритма консенсуса". Визуальное объяснение того, что такое Raft, см. в "Тайная жизнь данных".

1.1. Обзор протокола Raft

Raft – алгоритм консенсуса на базе Paxos. По сравнению с Paxos у Raft меньше состояний, а алгоритм проще и понятнее.

Здесь не будет подробно разбираться протокол Raft, но будет дано верхнеуровневое описание для общего представления. Полный список характеристик см. в этом документе.

1.1.1. Терминология

Основные термины, которые стоит знать при обсуждении протокола Raft:

- **Ведущий узел** — в любой момент времени группа узлов выбирает один узел в качестве ведущего. Ведущий узел отвечает за прием новых записей журнала, репликацию на ведомые узлы и управляет временем подтверждения записи. Кроме того, ведущий узел является активным узлом StarVault, а ведомые — резервными узлами. Дополнительную информацию см. в документе "Высокая доступность".
- **Журнал** — упорядоченная последовательность записей (реплицированный журнал) для отслеживания изменений кластера. Ведущий узел отвечает за репликацию журнала. Например, когда записываются новые данные, новое событие создает запись журнала. Затем ведущий узел отправляет новую запись журнала ведомым узлам. Любое несоответствие в реплицированных записях журнала будет указывать на проблему.

- **Конечный автомат** — это множество конечных состояний с переходами между ними. По мере применения новых журналов конечному автомату разрешается переходить между состояниями. Применение одной и той же последовательности журналов должно приводить к одному и тому же состоянию, то есть поведение должно быть детерминированным.
- **Группа узлов** — множество членов, участвующих в репликации журнала. Все серверные узлы входят в группу узлов локального кластера.
- **Кворум** — большинство членов из группы узлов. В случае группы размером n для кворума требуется как минимум (n+1)/2 членов. Например, если группа узлов содержит 5 членов, то для формирования кворума понадобится 3 узла. Если кворум узлов недоступен по какой-либо причине, кластер становится недоступным, и новые журналы подтвердить нельзя.
- **Подтвержденная запись** — запись считается подтвержденной, когда длительное время хранится в кворуме узлов. Запись применяется после подтверждения.

1.1.2. Состояния узлов

Узел Raft находятся только в одном из трех состояний: ведомый, кандидат или ведущий. Узлы запускаются как ведомые. В этом состоянии узлы могут принимать записи журнала от ведущего узла и отдавать голоса. Если в течение определенного периода времени не поступает никаких записей, узлы самостоятельно переходят в состояние кандидата. В состоянии кандидата узлы запрашивают голоса у других узлов того же ранга. Если кандидат получает кворум голосов, то повышается до ведущего узла. Ведущий узел должен принять новые записи журнала и реплицировать их на остальные ведомые узлы.

1.2. Запись журналов

Кластер может принимать новые записи журнала только после появления у него ведущего узла. Запись журнала с точки зрения Raft — это непрозрачный двоичный объект (blob).

Клиент может попросить ведущий узел добавить новую запись журнала. Затем ведущий узел записывает запись в долговременное хранилище и пытается реплицировать ее на кворум ведомых узлов. Как только запись журнала считается подтвержденной, то её можно применить к конечному автомату. Конечный автомат зависит от сценария использования; в случае StarVault используется BoltDB для поддержания состояния кластера. Записи StarVault блокируются до тех пор, пока они не будут подтверждены и применены.

1.3. Сжатие журналов

Нежелательно позволять реплицированному журналу неограниченно расти. Raft предоставляет механизм, с помощью которого текущее состояние сохраняется в моментальных снимках, а связанные с ним журналы сжимаются. В силу абстракции

конечного автомата, восстановление его состояния должно приводить к тому же результату, что и воспроизведение старых журналов. Это позволяет Raft зафиксировать состояние конечного автомата в определенный момент времени, а затем удалять журналы, которые использовались для достижения этого состояния. Это выполняется автоматически без вмешательства пользователя и предотвращает неограниченное использование дискового пространства, а также минимизирует время, затрачиваемое на воспроизведение журналов. При использовании BoltDB моментальные снимки StarVault занимают очень мало места, что, конечно, хорошо. Поскольку данные StarVault уже сохранены на диске в BoltDB, в процессе создания моментального снимка нужно лишь обрезать журналы Raft.

1.3.1. Кворум

Консенсус отказоустойчив, пока у кластера есть кворум. Если кворум узлов недоступен, невозможно обрабатывать записи журнала или определить членство в группе. Например, есть только 2 узла: A и B. Размер кворума также 2, т.е. оба узла должны согласиться на подтверждение записи в журнале. Если A или B выходит из строя, то уже невозможно достичь кворума. Значит кластер не может ни добавить или удалить узел, ни подтвердить какие-либо дополнительные записи в журнале, что приводит к недоступности. На этом этапе нужно вручную удалить A или B и перезапустить оставшийся узел в режиме начальной загрузки.

Кластер Raft из 3 узлов допускает отказ 1 узла, а кластер из 5 узлов допускает отказ 2 узлов. В боевой среде рекомендуется развертывать по 5 серверов StarVault на кластер. См. "Минимальная конфигурация и масштабирование" и "Таблица развертывания", чтобы узнать больше о том, как встроенное хранилище помогает обеспечивать отказоустойчивость.

1.3.2. Производительность

С точки зрения производительности Raft сопоставим с Paxos. При условии, что ведущий узел долгое время не меняется, для подтверждения записи журнала нужно один раз отправить данные половине членов кластера и дождаться от них ответа. Таким образом, производительность ограничена вводом-выводом дисковой подсистемы и сетевой задержкой.

1.4. Raft в StarVault

Во время запуска инициализируется один сервер StarVault. На этом этапе размер кластера 1, что позволяет узлу выбрать самого себя ведущим. После выбора ведущего узла другие серверы могут быть добавлены в группу узлов таким образом, чтобы сохранить согласованность и безопасность.

В ходе присоединения новые узлы добавляются в кластер StarVault; здесь используется зашифрованный рабочий процесс вызова/ответа. Для этого все узлы в одном кластере Raft должны иметь одинаковую конфигурацию печати. При автоматическом распечатывании

процесс присоединения может использовать настроенную печать, чтобы автоматически расшифровать вызов и ответить. При использовании запечатывания по Шамиру ключи распечатывания предоставляются узлу, пытающемуся присоединиться к кластеру, прежде чем он сможет расшифровать вызов и дать расшифрованный ответ.

Поскольку все серверы входят в группу узлов, они знают, какой узел сейчас ведущий. Когда API-запрос поступает на сервер, не являющийся ведущим узлом, запрос пересылается ведущему.

Как и на других бэкендах хранения, данные, которые записываются в журнал Raft и конечный автомат, будут зашифрованы барьером StarVault.

1.5. Управление кворумом

1.5.1. Управление с помощью автопилота

Функция автопилота доступна, когда есть настраиваемые параметры, определяющие, когда узел считается работоспособным, но еще не получил право голоса в кворуме. Кроме того, можно включить опцию, которая позволит удалять неработающие узлы из списка кворума по истечении определенного периода.

Автопилот включен по умолчанию в StarVault. Значения конфигурации по умолчанию должны хорошо работать для большинства развертываний StarVault, но при необходимости их можно изменить.

Автопилот включает в себя логику стабилизации для узлов, присоединяющихся к кластеру. Недавно присоединенные узлы сначала принимаются как неголосующие, пока не синхронизируются с соответствующим индексом Raft. После достижения пороговых значений стабильности узлы становятся полноправными голосующими членами. Установка слишком низкого порога стабильности может сделать кластер нестабильным, поскольку узлы будут учитываться как голосующие до того, как они смогут голосовать.

Также доступна возможность удаления неработающих серверов. При включении этой функции неработающие узлы удаляются из кластера Raft автоматически без вмешательства оператора. Включить эту функцию можно через API Автопилота. Если хотите вывести узел из эксплуатации вручную, воспользуйтесь командой `remove peer`.

Подоробнее см. в пункте Автопилот.

1.5.2. Управление без автопилота

Когда автопилот отключен или неправильно настроен, то присоединение узла к кластеру происходит иначе.

Присоединяясь к кластеру Raft, узел пытается догнать остальные узлы, используя данные, которые реплицируются ему от ведущего узла. В этом начальном состоянии синхронизации

узел не может голосовать, но учитывается для целей кворума. Если несколько новых узлов присоединяются к кластеру одновременно или почти одновременно, из-за чего количество превышает порог отказоустойчивости кластера, кворум может быть потерян, а кластер может выйти из строя.

Например, есть кластер, состоящий из 3 узлов, с большим объемом данных и отказоустойчивостью 1. Затем к кластеру присоединяются еще 3 узла. Теперь кластер состоит из 6 узлов с отказоустойчивостью 2, но поскольку 3 новых узла все еще догоняют остальных, это приведет к потере кворума.

Поэтому, прежде чем добавлять новые узлы, рекомендуем убедиться, что они имеют индексы Raft, близкие к ведущему. Увидеть индексы Raft можно через запрос `starvault status`.

1.6. Таблица развертывания

Ниже в таблице показаны размер кворума и отказоустойчивость для кластеров разного размера. В боевой среде рекомендуется развертывать кластер из 5 серверов. Развертывание одиночного сервера крайне не рекомендуется, поскольку в случае отказа потеря данных неизбежна.

Серверы	Размер кворума	Отказоустойчивость
1	1	0
2	2	0
3	2	1
4	3	1
5	3	2
6	4	2
7	4	3

1.7. Минимальная конфигурация и масштабирование

В эталонной архитектуре StarVault рекомендуется использовать кластер, состоящий из 5 узлов, для обеспечения минимальной отказоустойчивости не менее 2.

Всегда, когда возможно, рекомендуется сохранять отказоустойчивость не менее 2.

При техническом обслуживании и других изменениях можно провести масштабирование, добавив в существующий 5-узловой кластер еще два узла и получив в итоге кластер, состоящий из 7 узлов.

После подтверждения синхронизации новых узлов 2 старых узла можно остановить и/или удалить, а затем повторить все то же самое, пока не будут заменены остальные узлы. Добавляя на время узлы, чтобы получить кластер из 7 узлов, а затем возвращаясь к 5 узлам, можно обеспечить достаточную отказоустойчивость, постепенно вносить изменения, не превышая доступную отказоустойчивость в конкретный момент времени.

Намереваясь внести изменение или провести масштабирование, подразумевающее потерю кворума и снижение отказоустойчивости, не допускайте ситуаций, которые могут в этот момент поставить систему под угрозу.

В любом кластере всегда рекомендуется иметь нечетное количество голосующих узлов, и поэтому увеличение или уменьшение кластера на 2 узла за раз дает еще одно преимущество: оно позволяет избегать четного количества.

2. Организация работы StarVault со встроенным хранилищем

Встроенный бэкенд хранения не зависит от сторонних систем и поддерживает высокую доступность, а также процессы резервного копирования и восстановления.

При выборе этого варианта данные в StarVault хранятся в файловой системе сервера, а для репликации данных на каждый сервер кластера используется протокол консенсуса.

В следующих разделах подробно рассматривается, как организовать работу StarVault со встроенным хранилищем.

2.1. Взаимодействие сервер-сервер

После добавления в кластер узлы взаимодействуют по протоколу mTLS через кластерный порт StarVault, который по умолчанию имеет номер `8201`. Обмен информацией по протоколу TLS происходит при добавлении узла и периодически повторяется.

Для встроенного хранилища требуется установить параметр конфигурации `cluster_addr`, чтобы StarVault во время присоединения присваивал адрес идентификатору узла.

2.2. Членство в кластере

В этом разделе описано, как запускать и управлять кластером узлов StarVault со встроенным хранилищем.

Встроенное хранилище запускается при инициализации, в результате чего образуется одноузловой кластер. К активному узлу StarVault, в зависимости от желаемого масштаба развертывания, могут добавляться еще узлы.

2.2.1. Добавление узлов

При добавлении неинициализированный узел StarVault становится членом существующего кластера. Новый узел в кластере для успешной аутентификации должен использовать такой же механизм запечатывания, что применяется в кластере. Для автоматического распечатывания у узла и кластера, должно быть настроено использование одного того же провайдера KMS и ключа. В случае запечатывания по схеме Шамира ключи распечатывания должны быть предоставлены новому узлу до завершения процесса его добавления. После успешного добавления узла можно приступать к репликации на него данных с активного узла. Добавленный узел нельзя добавить к другому кластеру.

Узел добавляется либо автоматически через файл конфигурации, либо вручную через API (оба способа описаны ниже). При добавлении узла необходимо использовать API-адрес ведущего узла. Чтобы упростить процесс добавления узлов, рекомендуем настроить параметр конфигурации `api_addr` на всех узлах.

2.2.1.1. Конфигурация `retry_join`

Этот метод позволяет задать в файле конфигурации один или несколько целевых ведущих узлов. Неинициализированный сервер StarVault после запуска будет пытаться присоединиться к каждому заданному потенциальному ведущем узлу, пока не добьется успеха. Когда один из указанных ведущих узлов станет активным, то новый узел успешно присоединится к нему. В случае запечатывания по схеме Шамира добавленные узлы все равно придется распечатывать вручную. А в случае автоматического распечатывания - узел добавится и распечатается автоматически.

Пример конфигурации `retry_join` ниже:

```
storage "raft" {
  path    = "/var/raft/"
  node_id = "node3"

  retry_join {
    leader_api_addr = "https://node1.starvault.local:8200"
  }
  retry_join {
    leader_api_addr = "https://node2.starvault.local:8200"
  }
}
```

Обратите внимание, что в каждой строке файла конфигурации `retry_join` можно указать одно значение `leader_api_addr` или `auto_join`. Если указано значение конфигурации облака `auto_join`, то StarVault использует библиотеку go-discover, чтобы автоматически попытаться обнаружить и разрешить адреса потенциальных ведущих узлов Raft.

В файле README библиотеки go-discover приводится подробная информация о формате значения `auto_join`.

```
storage "raft" {
  path    = "/var/raft/"
  node_id = "node3"

  retry_join {
    auto_join = "provider=aws region=eu-west-1 tag_key=starvault tag_value=...
access_key_id=... secret_access_key=..."
  }
}
```

По умолчанию StarVault будет пытаться связаться с обнаруженными узлами в пределах кластера через HTTPS и порт 8200. Операторы могут переопределить эти параметры с помощью полей `auto_join_scheme` и `auto_join_port` соответственно.

```
storage "raft" {
  path    = "/var/raft/"
  node_id = "node3"

  retry_join {
    auto_join = "provider=aws region=eu-west-1 tag_key=starvault tag_value=...
access_key_id=... secret_access_key=..."
    auto_join_scheme = "http"
    auto_join_port = 8201
  }
}
```

2.2.1.2. Добавление через командную строку

Для добавления узла можно также использовать командную строку (команда join) или API. В этом случае укажите API-адрес активного узла:

BASH

```
vault operator raft join https://node1.vault.local:8200
```

2.2.2. Удаление одноранговых узлов

Если одноранговый узел больше не нужен в кластере, то его необходимо удалить. Пример причин, по которым одноранговый узел становиться ненужным: замена на новый узел, имя хоста навсегда меняется и к нему больше нельзя обратиться, уменьшение размер кластера и т.д. Удаление однорангового узла позволит кластеру сохранить желаемый размер и сохранить кворум.

Чтобы удалить узел, выполните команду `remove-peer` и укажите идентификатор узла, который хотите удалить:

```
TERMINAL
$ starvault operator raft remove-peer node1
Peer removed successfully!
```

2.2.3. Вывод списка одноранговых узлов

Чтобы вывести текущий набор узлов в кластере, выполните команду `list-peers`. Перечисленные голосующие узлы составляют кворум, и для продолжения работы встроенного хранилища большинство должно быть работоспособно.

```
TERMINAL
$ starvault operator raft list-peers
Node     Address                    State       Voter
----     -------                    -----       -----
node1    node1.starvault.local:8201    follower    true
node2    node2.starvault.local:8201    follower    true
node3    node3.starvault.local:8201    leader      true
```

2.3. Встроенное хранилище и TLS

В большинстве случаев достаточно следовать инструкциям, но некоторые пользователи столкнулись с проблемами при использовании автодобавления и TLS в сочетании, например, с автомасштабированием. Проблема заключается в том, что библиотека go-discover на большинстве платформ возвращает IP-адреса (а не имена хостов). Поскольку IP-адреса не известны заранее, в IP SAN сертификатов TLS, используемых для защиты API-порта StarVault, этих IP-адресов нет.

2.3.1. Краткое описание сетевых аспектов работы StarVault

Прежде чем рассматривать решения этой проблемы, вкратце расскажем, как взаимодействуют узлы StarVault.

StarVault открывает два TCP-порта: порт API и порт кластера.

На порт API клиенты отправляют HTTP-запросы к StarVault.

Для одноузлового кластера StarVault порт кластера не нужен, так как не будет использоваться.

Если узлов несколько, то порт кластера понадобится. Его используют узлы StarVault для отправки RPC-вызовов друг другу, например, для переадресации запросов от резервного узла к активному или для управления выборами ведущего узла и репликацией хранимых данных при использовании Raft.

Порт кластера защищен TLS-сертификатом, который активный узел StarVault генерирует внутрисистемно. Как это работает в отсутствие встроенного хранилища: каждый узел имеет доступ к хранилищу как минимум на чтение, и когда активный узел сохраняет сертификат,

резервные узлы могут прочитать его и вместе согласиться с тем, как должен шифроваться трафик кластера.

Гораздо менее понятно, как это работает со встроенным хранилищем, поскольку здесь возникает проблема курицы и яйца. Узлы не имеют общего представления о хранилище, пока не сформирован кластер raft. Чтобы сформировать кластер raft — узел StarVault должен обращаться к другому узлу StarVault через порт API, а не порт кластера. На данный момент это единственная ситуация, в которой StarVault работает именно так.

- `node2` хочет добавиться в кластер, поэтому отправляет API-запрос существующему члену `node1`
- `node1` в ответ на такой API-запрос отправляет (1) зашифрованный случайный UUID и (2) конфигурацию механизма запечатывания
- `node2` расшифровает UUID с помощью механизма запечатывания. При автоматическом распечатывании может сделать это напрямую. В случае запечатывания по схеме Шамира ждет, пока пользователь предоставит достаточное число ключей распечатывания для расшифровки
- `node2` отправляет расшифрованный UUID обратно на `node1`, используя API ответа
- `node1` видит, что `node2` можно доверять (поскольку у него есть доступ к механизму запечатывания), и в ответ отправляет пакет запуска с сертификатом TLS и закрытым ключом кластера.
- `node2` получает снимок raft через порт кластера

После этого новый узел больше никогда не будет отправлять трафик на порт API. В дальнейшем для взаимодействия между узлами будет использоваться порт кластера.

2.3.1.1. Добавление узлов в кластер raft с использованием вспомогательных средств

Проще всего это делать вручную: выполнять команды `raft join`, указывая явные имена или IP-адреса узлов, к которым нужно добавиться. В этом разделе мы рассмотрим другие TLS-совместимые варианты, которые в большей степени можно автоматизировать.

Автодобавление с именем сервера TLS

Самый простой вариант – указать `leader_tls_servername` в строке `retry_join`, которая соответствует DNS SAN в сертификате.

Обратите внимание, что имена в DNS SAN сертификата не обязательно должны быть зарегистрированы на DNS-сервере. Имена узлов могут отсутствовать в DNS и все равно использовать сертификат(ы), содержащий(ие) это общее значение servername в своих DNS SAN.

Автодобавление с ограничением по CIDR, список всех возможных IP-адресов в сертификате

Если IP-адреса узлов хранилища назначены из небольшой подсети, например, `/28`, то целесообразно внести IP-адреса, существующие в этой подсети, в IP SAN сертификата TLS, который будут совместно использовать узлы.

Проблема здесь в том, что со временем кластер может перерасти рамки CIDR, и изменить его будет непросто. По тем же причинам это решение может оказаться непрактичным при использовании динамически масштабируемых кластеров.

Использование балансировщика нагрузки вместо автодобавления

В большинстве экземпляров StarVault между клиентами и узлами StarVault имеется балансировщик нагрузки. В этом случае балансировщик знает, как направить трафик на работающие узлы StarVault, и в автодобавлении нет необходимости: можно выполнить команду `retry_join` с адресом балансировщика в качестве цели.

Здесь существует одна потенциальная проблема: некоторые пользователи желают использовать публичный балансировщик для подключения клиентов к StarVault, но не одобряют, когда внутренний трафик Vault покидает внутреннюю сеть, в которой он обычно функционирует.

2.4. Восстановление после сбоев

2.4.1. Поддержание кворума

В этом разделе описаны действия, которые необходимо предпринять, если на одном или нескольких серверах произошел сбой, но кворум по-прежнему сохраняется. Это означает, что оставшиеся подключенными серверы продолжают работать, могут выбирать ведущий узел и обрабатывать запросы на запись.

Если вышедший из строя сервер можно восстановить, лучше всего вернуть его в сеть и дать снова подключиться к кластеру с тем же адресом хоста. Таким образом, кластер полностью вернется в работоспособное состояние.

Если это нецелесообразно, необходимо удалить вышедший из строя сервер. Как правило, удалить отказавший сервер, все еще входящий в состав кластера, можно командой `remove-peer`.

Если выполнить команду `remove-peer` невозможно или хотите вручную переопределить состав кластера, то запишите файл `raft/peers.json` в настроенный каталог данных.

2.4.2. Потеря кворума

Выход из строя нескольких серверов приведет к потере кворума и полному отключению, но частичное восстановление все еще возможно.

Если отказавшие серверы можно восстановить, то лучше всего вернуть их в сеть и снова подключить к кластеру с теми же адресами хостов. Таким образом, кластер полностью вернется в работоспособное состояние.

Если вышедшие из строя серверы не подлежат восстановлению, то можно восстановить частично, используя данные на оставшихся серверах кластера. Из-за сбоя нескольких серверов некоторые данные могут быть утеряны. Поэтому информация о том, что было подтверждено, может быть неполной. В ходе восстановления неявно подтверждаются все неподтвержденные записи журнала Raft, поэтому можно подтвердить данные, которые не были подтверждены до сбоя.

Подробное описание восстановления см. ниже в разделе о ручном восстановлении с помощью файла `peers.json`. В файл восстановления `peers.json` следует включать только оставшиеся серверы. Кластер должен быть способен выбрать ведущий узел, когда оставшиеся серверы будут перезапущены с идентичной конфигурацией `peers.json`.

Серверы, добавляемые позже, могут иметь чистые каталоги данных и добавляться командой join в StarVault.

В крайнем случае, должна быть возможность восстановить единственный оставшийся сервер: запустить и указать его как единственный узел в файле восстановления `peers.json`.

2.4.3. Ручное восстановление с помощью файла `peers.json`

Использование файла `raft/peers.json` для восстановления может привести к неявному подтверждению неподтвержденных записей журнала Raft, поэтому этот файл следует использовать только после сбоя, если нет других способов восстановить вышедший из строя сервер.

Убедитесь, что любые автоматизированные процессы, которые могут периодически сохранять файл peers, остановлены.

Прежде всего остановите оставшиеся серверы.

Далее перейдите по пути к каталогу с данными конфигурации каждого сервера StarVault. Внутри этого каталога расположен подкаталог `raft/`. Создайте файл `raft/peers.json`. Этот файл должен иметь формат массива JSON, в котором содержатся идентификатор узла, пара `address:port` и информация "голосующий/неголосующий" для каждого сервера StarVault, который хотите добавить в кластер:

JSON

```
[
  {
    "id": "node1",
    "address": "node1.starvault.local:8201",
  },
  {
```

```
    "id": "node2",
    "address": "node2.starvault.local:8201",
  },
  {
    "id": "node3",
    "address": "node3.starvault.local:8201",
  }
]
```

- `id (string: <required>)` указывает идентификатор узла сервера. Его можно найти в конфигурационном файле или в файле node-id в каталоге данных сервера, если был сгенерирован автоматически.
- `address (string: <required>)` указывает хост и порт сервера. Порт — это порт кластера сервера.

Создайте записи для всех серверов. Убедитесь, что серверы, которые не включены в этот файл, действительно вышли из строя и впоследствии не возобновят работу в кластере. Убедитесь, что этот файл одинаковый для оставшихся серверных узлов.

На этом этапе можно перезапустить оставшиеся серверы. Кластер должен снова стать работоспособным. Один из узлов должен запросить роль ведущего и стать активным.

2.4.4. Другие методы восстановления

В остальных случаях можно использовать режим восстановления StarVault, не связанный с кворумом.

3. Автопилот

Автопилот позволяет автоматизировать управление кластерами Raft. Доступны три основные функции: стабилизация серверов, очистка кластера от неработающих серверов и API для мониторинга состояния.

3.1. Стабилизация серверов

Функция помогает сохранить стабильность кластера Raft путем безопасного присоединения новых голосующих узлов к кластеру. Когда новый голосующий узел присоединяется к существующему кластеру, автопилот добавляет его как неголосующий узел и проверяет его работоспособность в течение заранее заданного промежутка времени. Если узел сохраняет работоспособность в течение всего времени стабилизации, то он будет повышен до голосующего. Период стабилизации серверов настраивается с помощью `server_stabilization_time` (см. ниже).

3.2. Очистка кластера от неработающих серверов

Функция автоматически и без вмешательства оператора, удаляет из кластера Raft узлы, которые считает неработоспособными. Функцию настраивается с помощью `cleanup_dead_servers`, `dead_server_last_contact_threshold` и `min_quorum` (см. ниже).

3.3. API для мониторинга состояния

Один вызов API для мониторинга состояния возвращает подробную информацию о всех узлах в кластере Raft. Этот API можно использовать для мониторинга работоспособности кластера.

3.3.1. Работоспособность ведомого узла

Работоспособность ведомого узла определяется двумя факторами:

- способностью отправлять heartbeat-сигналы ведущему узлу через регулярные интервалы времени. Настраивается с помощью `last_contact_threshold` (см. ниже).
- способностью поддерживать репликацию данных с ведущего узла. Настраивается с помощью `max_trailing_logs` (см. ниже).

3.3.2. Конфигурация по умолчанию

Автопилот для кластеров StarVault включен по умолчанию, хотя очистка кластера от неработающих серверов по умолчанию не включена.

Управлять поведением автопилота можно через API для конфигурирования. Автопилот инициализируется со следующими значениями по умолчанию. Если эти значения по умолчанию не соответствуют тому поведению автопилота, которое нужно, не забудьте установить необходимые значения.

- `cleanup_dead_servers` — `false`
  - Этот параметр указывает, нужно ли удалять неработающие серверы из списка узлов в кластере Raft периодически или при присоединении нового сервера. Для этого параметр `min-quorum` также должен быть задан.
- `dead_server_last_contact_threshold` — `24h`
  - Предельное время, в течение которого сервер может не иметь контакта с ведущим узлом, прежде чем будет считаться отказавшим. Этот параметр принимается во внимание, только если задан параметр `cleanup_dead_servers`.
- `min_quorum` — у этого параметра нет значения по умолчанию, поэтому для него нужно указать ожидаемое количество голосующих узлов в кластере, когда параметр `cleanup_dead_servers` установлен в значение `true`.

    - Минимальное количество серверов, которое всегда должно присутствовать в кластере. Автопилот не будет удалять серверы, если в результате такого удаления количество серверов окажется меньше этого значения.
- `max_trailing_logs` — `1000`
    - Количество записей в журнале Raft, на которые сервер может отстать, прежде чем будет считаться неработоспособным.
- `last_contact_threshold` — `10s`
    - Предельное время, в течение которого сервер может не иметь контакта с ведущим узлом, прежде чем будет считаться неработоспособным.
- `server_stabilization_time` — `10s`
    - Минимальное время, в течение которого сервер должен находиться в работоспособном состоянии, прежде чем станет голосующим. Пока этого не произойдет, сервер будет виден в кластере как узел, но без права голоса, т.е. не будет учитываться при определении наличия кворума.

3.4. Автоматизированные обновления

Автоматизированные обновления позволяют автоматически обновлять кластер узлов StarVault до новой версии при присоединении обновленных серверных узлов к кластеру. Как только количество узлов с новой версией станет равным или превысит количество узлов со старой версией, Автопилот повысит статус узлов с более новой версией до голосующих, понизит статус узлов со старой версией до неголосующих и инициирует передачу лидерства от ведущего узла со старой версией к одному из узлов с более новой версией. После завершения передачи лидерства неголосующие узлы со старой версией могут быть удалены из кластера.

3.5. Зоны резервирования

Зоны резервирования обеспечивают масштабирование и отказоустойчивость благодаря развертыванию неголосующих узлов вместе с голосующими в каждой зоне доступности. При использовании зон резервирования каждая зона будет иметь ровно один голосующий узел и необходимое число дополнительных неголосующих узлов. Если голосующий узел в зоне выходит из строя, неголосующий узел будет автоматически повышен до голосующего. Если отказывает вся зона, неголосующий узел из другой зоны будет повышен до голосующего, что позволит сохранить кворум. Эти неголосующие узлы не только выполняют функции горячего резерва, но и повышают масштабируемость чтения.

Ротация ключей

StarVault использует несколько ключей шифрования для различных целей, поддерживая их ротацию. Это позволяет периодически менять ключи или реагировать на потенциальные утечки и уязвимости. Важно сначала понять высокоуровневую архитектуру, прежде чем изучать ротацию ключей.

Для справки: StarVault запускается в запечатанном состоянии и распечатывается путем предоставления ключей распечатывания. По умолчанию StarVault использует алгоритм разделения секрета Шамира, чтобы разделить корневой ключ на 5 частей, из которых любые 3 необходимы для восстановления мастер-ключа. Корневой ключ используется для защиты ключа шифрования, который используется для защиты данных, записываемых в серверную часть хранилища.

Для поддержки ротации ключей необходимо изменять ключи распечатывания, корневой ключ и ключ шифрования серверной части. Этот процесс делится на две отдельные операции: `rekey` и `rotate`.

Операция `rekey` используется для генерации нового корневого ключа. В процессе выполнения операции можно изменить параметры разделения ключа, что позволяет варьировать количество частей и порог, необходимый для распечатывания. Для выполнения `rekey` необходимо предоставить количество ключей распечатывания, равное текущему пороговому значению. Это делается для предотвращения выполнения `rekey` и аннулирования существующего корневого ключа злоумышленником.

Процесс выполнения `rekey` довольно прост. Операция должна быть инициализирована с новыми параметрами для разделения и порога. После инициализации необходимо предоставлять текущие ключи распечатывания до тех пор, пока не будет достигнут порог. После достижения порога StarVault сгенерирует новый мастер-ключ, выполнит разделение и повторно зашифрует ключ шифрования новым корневым ключом. Затем оператору предоставляются новые ключи распечатывания, а старые ключи распечатывания становятся непригодными для использования.

Операция `rotate` используется для изменения ключа шифрования, который применяется для защиты данных, хранящихся на сервере. Этот ключ никогда не предоставляется и не виден операторам, у которых есть только ключи распечатывания. Это упрощает процесс ротации, так как не требует участия текущих владельцев ключей, в отличие от операции `rekey`. При запуске операции `rotate` генерируется новый ключ шифрования и добавляется в связку ключей. Новые данные, записываемые на серверную часть хранилища, шифруются новым ключом. Старые данные, записанные с предыдущими ключами шифрования, все еще могут быть расшифрованы, поскольку старые ключи

сохраняются в связке ключей. Это позволяет выполнять ротацию ключей в режиме онлайн, без дорогостоящего процесса повторного шифрования.

Обе операции, `rekey` и `rotate`, могут выполняться в режиме онлайн и в высокодоступной конфигурации. Только активный экземпляр StarVault может выполнять любую из этих операций, но резервные экземпляры могут взять на себя активную роль после выполнения любой из этих операций. Это достигается за счет предоставления возможности обновления резервным экземплярам. Если текущий ключ шифрования - `N`, а ротация устанавливает `N+1`, StarVault создает специальный "ключ обновления", который предоставляет ключ шифрования `N+1`, защищенный ключом `N`. Этот ключ обновления доступен только в течение нескольких минут, что позволяет резервным экземплярам периодически проверять наличие обновлений. Это позволяет резервным экземплярам обновлять ключи и оставаться синхронизированными с активным StarVault без необходимости выполнения операторами повторного распечатывания.

Конечная точка `rotate/config` используется для настройки количества операций или временного интервала между автоматическиой ротацией ключа шифрования серверной части.

1. Руководство NIST по ротации

Рекомендуется периодическая ротация ключей шифрования, даже при отсутствии уязвимостей. В связи с характером используемого шифрования AES-256-GCM, ключи следует обновлять после выполнения приблизительно 2^{32} шифрований, следуя рекомендациям публикации NIST 800-38D.

По умолчанию StarVault будет автоматически обновлять ключ шифрования серверной части после достижения 2^{32} операций шифрования.

Операторы могут оценить количество шифрований, суммируя следующие показатели:

- Метрика телеметрии `vault.barrier.put`.
- Метрика `vault.token.creation`, где метка `token_type` равна `batch`.
- Метрика `merkle.flushDirty.num_pages`.
- Индекс WAL.

StarVault периодически сохраняет количество шифрований для поддержки ротации. Эта операция сохранения имеет тайм-аут в 1 секунду, чтобы не влиять на производительность при высокой нагрузке на StarVault. Поскольку сохранение шифрований связана с серверной частью, некоторым системам запечатывания (например, Transit) может потребоваться больше 1 секунды для ответа. В этом случае операторы могут изменить этот тайм-аут,

установив переменную окружения `VAULT_ENCRYPTION_COUNT_PERSIST_TIMEOUT` на большее значение, например, `"5 с"`.
