ВНЕДРЕНИЕ KEYSTONE FEDERATION В СУЩЕСТВУЮЩИЕ ОКРУЖЕНИЯ

IMPLEMENTING KEYSTONE FEDERATION IN EXISTING ENVIRONMENTS

УДК 004

Кучеренко Михаил Александрович, студент, МГТУ им. Н.Э. Баумана, Россия, г.Москва

Kucherenko Mikhail Alexandrovich, student, MSTU N.E. Bauman, Russia, Moscow, makucherenko@list.ru

Аннотация

Для более гибкой настройки системы аутентификации в облачной инфраструктуре виртуальных машин на основе OpenStack предусмотрен модуль аутентификации Keystone Federation. В данной работе рассматриваются основные принципы работы модуля и методов аутентификации, а также варианты окружений, в которых есть возможность его внедрения. Сделаны принципиальные выводы о том, как можно использовать данный модуль в наиболее распространенных вариантах окружения: с Apache или Nginx.

Summary

For more flexible configuration of the authentication system in the cloud infrastructure of virtual machines based on OpenStack, the Keystone Federation authentication module is provided. This paper discusses the basic principles of the module and authentication methods, as well as options for possible environments. Fundamental conclusions have been made on how this module can be used in the most common environments - with Apache or Nginx.

Ключевые слова: облако, аутентификация, Keystone Federation, OpenStack, Nginx, Apache.

Keywords: cloud, authentication, Keystone Federation, OpenStack, Nginx, Apache.

Для работы облачной инфраструктуры OpenStack требуется система, которая может управлять пользователями, проектами, доступами и идентифицировать пользователей и другие подсистемы.

Для этих целей был создан сервис аутентификации Keystone. В стандартной сборке - режиме по умолчанию, он сам исполняет все функции от выдачи токенов, до подтверждения доступов пользователя [1, с. 53]. Однако, данный сервис поддерживает работу с внешними поставщиками идентификации (identity providers) через свой модуль Федерация (Keystone Federation), что позволяет встроить его в уже существующие корпоративные решения, использующие SAML, OpenID Connect (OpenIDC), LDAP или другие [2, с. 1]. В таком режиме работы Keystone выступает, как поставщик услуг (service provider), исполняя только функции управления проектами и доступами, перекладывая задачи аутентификации и управления пользователями на внешние сервисы. У пользователей появляется возможность войти в систему с использованием указанных сервисов. Например, войти своим аккаунтом почты, социальной сети или внутренней корпоративной системы доступа.

Также существует режим Keystone-to-Keystone, но это крайне редкий вариант использования. Он позволяет соединить несколько сервисов Keystone из разных кластеров OpenStack в один, чтобы иметь единые доступы между всеми экземплярами систем. В таком случае один сервис выступает в роли поставщика идентификации (identity provider), а остальные в роли поставщика услуг (service provider) [3, с. 1].

Модуль Федерация реализует механизм единого входа (Single Sign-On), таким образом, когда пользователь в начале рабочего дня аутентифицируется в корпоративной системе доступов, назовем его А - ему больше не приходится повторно подтверждать свою личность при использовании OpenStack,

произойдет автоматическое обращение к сервису А и подтверждение личности пользователя через него. Это позволяет:

- Значительно повысить пользовательский опыт, за счет снижения затрат времени на побочные действия, такие как ввод логина и пароля.

- Повысить общий уровень безопасности системы, так как пользователю не придется хранить к каждому сервису отдельный пароль, а кроме обычной аутентификации по паролю можно подключить двухфакторную аутентификацию, например, с использованием аппаратных ключей безопасности или через одноразовые пароли (one time password, OTP).

- Объединить несколько разрозненных систем в единую инфраструктуру, в которой каждый компонент выполняет только свою определенную функцию, а значит, данные о пользователях будут более консистентными - при удалении доступа одному из пользователей его достаточно удалить только у внешнего поставщика идентификации.

- Дать компаниям возможность предоставлять услуги облачных решений внешним клиентам с удобной для них методом аутентификации за счет настройки нескольких модулей.

Из недостатков можно отметить только более высокую сложность настройки по сравнению с режимом по умолчанию, а также интеграцию "из коробки" только с Apache.

Стандартная установка OpenStack строго по документации, подразумевает установку и настройку Apache HTTP-сервера (httpd) [4, с. 1]. Пользователь, попадая на веб-интерфейс OpenStack - сервис Horizon, проходит через HTTPD-auth модуль, который в свою очередь делает запрос к внешнему поставщику идентификации, устанавливая у него личность пользователя. Если пользователь еще не заходил на данный сервис, то сможет аутентифицироваться, после чего происходит повторный запрос к Horizon, через HTTPD-auth. Если личность установлена - пользователь продолжает работу с OpenStack как обычно, ему выдается токен из Keystone.

Но что, если ваша инфраструктура уже построена на Nginx или другом HTTP-сервере, или вы вообще используете какой-то специфичный метод

аутентификации? В таком случае у вас, вероятно, нет подходящего HTTPD-auth модуля, который требуется для Keystone Federation.

В таком случае, потребуется провести некоторые специфичные операции по настройке и создать "прослойку", которая заменит HTTPD-auth модуль.

Подробно рассмотрим принцип работы системы в таком виде, в качестве HTTP-сервера возьмем самую популярную замену Apache - Nginx.

Основные API-сервисы OpenStack: Keystone-API, Nova-API, Neutron-API, Glance-API, Cinder-API, Heat-API и Horizon, с помощью systemd-unit файлов настроим как отдельные WSGI-приложения с помощью uWSGI. К ним будем получать доступ через Nginx, используя опцию proxy_pass в отдельных секциях server для каждого сервиса. Так мы ограничим доступ единым ACL (опции access и deny) и обезопасим API от тяжелых запросов (client_max_body_size) [5, с. 1]. Остальные настройки проводим как в документации - создаем БД, пользователя в OpenStack, и так далее.

Отдельно собираем WSGI-приложение "прослойку", оно должно уметь обрабатывать требуемые нам методы аутентификации от внешних проводников идентификации. Его целесообразно реализовать наследованием от классов библиотеки Flask. Данное приложение должно уметь обрабатывать 3 варианта запросов, описывающих 3 этапа аутентификации через внешних проводников идентификации. Рассмотрим поэтапно:

1. Пользователь сделал запрос без идентификационных данных, его необходимо перенаправить во внешний проводник идентификации для аутентификации и получения токена.

2. Пользователь сделал запрос с токеном, необходимо сделать подзапрос и проверить этот токен у внешнего проводника - действительно ли он выдавал такой токен. Если подзапрос успешен - устанавливаем пользователю сессию (session_id в cookie), чтобы больше не повторять данную процедуру на время действия токена.

3. Пользователь сделал запрос и мы уже можем идентифицировать его с помощью сессии (верная session_id в cookie), считаем что пользователь уже аутентифицирован и предоставляем ему доступ к системе.

Рассмотрим весь процесс подробно. Клиент делает запрос к Horizon, через Nginx. И если еще не аутентифицирован - ему вернется HTML-форма аутентификации. По нажатию кнопки на ней, происходит запрос к конечной точке /v3/auth/OS-FEDERATION/websso/METHOD. Если бы это был Apache, запрос попал бы в настроенный заранее HTTPD-auth модуль. Но в нашей схеме это будет приложение-прослойка. Запрос к нему без параметров, как было описано выше, приведет к ответу 302 redirect на внешний проводник идентификации. Проведя аутентификацию там, пользователь вернется на эту же конченую точку, но уже с токеном. Приложение проверяет точность токена и подтверждает личность пользователя, возвращает 200, после чего запрос направляется к Keystone-API, где вызывается auto-submit форма и пользователю выдается Keystone токен. Далее можно продолжать использование OpenStack как обычно.

Keystone Federation позволяет более гибко настроить облачную инфраструктуру компании и интегрировать её в любое существующее окружение, либо предоставить пользователям выбор из нескольких методов аутентификации. Однако, настройка в окружении с Nginx или специфичными методами аутентификации потребует дополнительных настроек и реализации приложения-прослойки.

Литература

1. Маркелов А. А. OpenStack. Практическое знакомство с облачной операционной системой / 4-е изд., доп. и исправ. - М.: ДМК Пресс, 2018. - 306 с.: ил.
2. Identity service overview [Электронный ресурс] // OpenStack Docs : Identity service overview. URL: https://docs.openstack.org/keystone/latest/install/get-started-rdo.html (дата обращения: 27.07.2020)
3. Introduction to Keystone Federation [Электронный ресурс] // OpenStack Docs : Introduction to Keystone Federation. URL: https://docs.openstack.org/keystone/latest/admin/federation/introduction.html (дата обращения: 27.07.2020)

4. Configuring Keystone for Federation [Электронный ресурс] // OpenStack Docs : Configuring Keystone for Federation. URL: https://docs.openstack.org/keystone/latest/admin/federation/configure_federation.html (дата обращения: 27.07.2020)
5. Модуль ngx_http_proxy_module [Электронный ресурс] // Nginx : Документация. URL: https://nginx.org/ru/docs/http/ngx_http_proxy_module.html (дата обращения: 27.07.2020)

Literature

1. Markelov A. A. OpenStack. Practical introduction to the cloud operating system / 4th ed., Add. and corrected. - M .: DMK Press, 2018 .-- 306 p.: Ill.
2. Identity service overview [Electronic resource] // OpenStack Docs: Identity service overview. URL: https://docs.openstack.org/keystone/latest/install/get-started-rdo.html (date accessed: 07/27/2020)
3. Introduction to Keystone Federation [Electronic resource] // OpenStack Docs: Introduction to Keystone Federation. URL: https://docs.openstack.org/keystone/latest/admin/federation/introduction.html (date accessed: 07/27/2020)
4. Configuring Keystone for Federation [Electronic resource] // OpenStack Docs: Configuring Keystone for Federation. URL: https://docs.openstack.org/keystone/latest/admin/federation/configure_federation.html (date accessed: 07/27/2020)
5. Module ngx_http_proxy_module [Electronic resource] // Nginx: Documentation. URL: https://nginx.org/ru/docs/http/ngx_http_proxy_module.html (date accessed: 07/27/2020)