30.05.2023

Лучшие практики для создания надежных API

По мере масштабирования ваших API возрастает необходимость сделать их надежными и устойчивыми.

В этой статье рассматриваются передовые практики по созданию надежных API путем введения специального вида обратных прокси-серверов, называемых API-шлюзами.

Мы рассмотрим:

1. Проблемы с традиционными проектировками API
2. Что такое API-шлюзы
3. Как API-шлюзы улучшают API
4. Шаблоны и примеры использования API-шлюзов

Но сначала, что такое "надежные" API?

Что делает API надежным?

Как поставщик услуг, у вас могут быть соглашения об уровне обслуживания (SLA) с вашими клиентами, обычно выраженные в процентном соотношении времени работы сервиса — это гарантированное время, в течение которого сервис должен быть онлайн и работоспособен.

Время работы является узколобым представлением о надежности. Чтобы понять, что означает быть надежным, нужно рассмотреть факторы, влияющие на время работы. Когда вы понимаете эти факторы, вы будете в лучшей позиции для создания надежных сервисов.

Давайте рассмотрим эти факторы и вопросы, которые они вызывают:

1. Задержка: Насколько быстро ваше API отвечает на запросы?
2. Безопасность: Кто может получить доступ к вашему API? Является ли он безопасным?
3. Частота простоев: Как часто ваше API не работает?
4. Постоянство: Являются ли ваши конечные точки API постоянными? Нужно ли пользователям часто изменять свой код?
5. Мониторинг и отчетность: Можете ли вы отслеживать проблемы и сбои в вашем API? Осуществляете ли вы отчетность о них для ваших потребителей?

По мере перехода организаций к облачным нативным архитектурам, разработчическим командам становится сложно учитывать все эти факторы для каждого из их сервисов. И по мере масштабирования этих систем гораздо проще делегировать эти обязанности одной отдельной системе. Добро пожаловать в мир API-шлюзов!

API-шлюз, унифицированная точка входа

API-шлюз выступает в роли посредника между вашими клиентами и вашими API. Он принимает весь трафик (API-вызовы), подобно реверс-прокси, перенаправляет запросы к требуемым сервисам на вашем сервере и возвращает необходимые результаты.

API-шлюз может быть центральной точкой, которая обрабатывает аутентификацию, обеспечение безопасности, контроль трафика и мониторинг, оставляя разработчикам API возможность сосредоточиться на бизнес-потребностях и упрощении обеспечения надежности.

Существует множество открытых и управляемых предложений для API-шлюзов. В этой статье я буду использовать Apache APISIX.

В следующем разделе будут описаны некоторые bewше практики, которые помогут сделать ваши API надежными с использованием API-шлюзов.

Лучшие практики надежности при использовании API-шлюзов

Мы уделяем больше внимания паттернам, чем фактической реализации, так как она может варьироваться в зависимости от выбора вашего API-шлюза.

Я разделю эти паттерны на три категории:

Аутентификация и безопасность Мониторинг и наблюдаемость Управление версиями и безотказность

Давайте рассмотрим каждую категорию подробнее ниже.

Аутентификация и безопасность

Аутентификация пользователя

Аутентифицированные запросы с использованием API-шлюзов обеспечивают безопасность взаимодействия клиента с API. После аутентификации клиента ваш API-шлюз может использовать полученные данные о клиенте для тонкой настройки контроля.

APISIX обрабатывает аутентификацию непосредственно через плагины, такие как key-auth и jwt-auth. APISIX также поддерживает аутентификацию OAuth и системы контроля доступа на основе ролей, такие как wolf, с помощью плагинов openid-connect и wolf-rbac соответственно.

Ограничения

Имитирование (DoS-атаки) и непреднамеренные (клиенты, совершающие слишком много запросов) всплески трафика к вашим API могут привести к их отказу. Установка ограничения скорости (rate limiting) улучшит надежность ваших систем при обработке подобных сценариев.

Вы можете настроить ограничение скорости на своем API-шлюзе, и если количество запросов превышает заданный порог, API-шлюз может задержать или отклонить превышающие запросы.

С помощью APISIX вы можете использовать любой из трех плагинов для настройки ограничений скорости на основе количества запросов, количества параллельных запросов от клиента и подсчета (limit-req, limit-conn, limit-count).

Мониторинг

Надежность вашего API и настройка мониторинга тесно связаны друг с другом. Вы можете отслеживать показатели надежности, настроив мониторинг на вашем API-шлюзе.

Журналы и трассировки API предоставляют подробную информацию о вызове API. Эта информация поможет вам узнать о сбое или ошибке вашего API как можно скорее. Тихие сбои приводят к неисправленным ошибкам, которые могут вызвать проблемы в будущем.

При настройке вы также сможете прогнозировать и предвидеть трафик в будущем, что поможет вам обеспечить надежное масштабирование.

APISIX имеет плагины, которые интегрируются с платформами/спецификациями для журналирования (Apache SkyWalking, RocketMQ), метрик (Prometheus, Datadog) и трассировки (OpenTelemetry, Zipkin). Больше информации об обеспечении наблюдаемости API с помощью плагинов APISIX можно найти на их сайте.

Контроль версий и нулевое время простоя

Canary

При переходе на новые версии вашего API необходимо обеспечить непрерывность обслуживания. Клиенты должны иметь возможность отправлять запросы к вашему API и получать правильные ответы.

С помощью API-шлюза вы можете настроить поэтапное внедрение (canary releases). Это позволит вам гарантировать функциональность вашего API во время перехода, а также откатиться к предыдущей версии в случае проблем.

Изначально API-шлюз будет направлять весь трафик к предыдущей версии вашего API.

Когда у вас появляется новая версия, вы можете настроить API-шлюз для перенаправления части трафика на эту новую версию. Вы можете постепенно увеличивать процент трафика к вашему новому сервису и проверять, что все работает так, как ожидается.

В конце концов, вы можете перенаправить весь трафик на ваш новый API.

APISIX использует плагин traffic-split, который позволяет вам контролировать трафик к вашим сервисам. Вы можете использовать его для настройки поэтапного выпуска (canary releases) или для настройки пользовательской конфигурации выпуска.

Разрыв цепи

Когда один из ваших верхних сервисов недоступен или испытывает высокую задержку, его необходимо отключить от системы. В противном случае клиент будет продолжать повторные запросы, что приведет к истощению ресурсов. Эта неисправность может распространиться на другие сервисы в системе и привести к их сбою.

Подобно тому, как электрические автоматические выключатели изолируют неисправные компоненты от цепи, API-шлюзы имеют функцию автоматического выключателя, который отключает неисправные сервисы, поддерживая систему в здоровом состоянии. Трафик к этим сервисам перенаправляется или задерживается до восстановления их работоспособности.

APISIX имеет плагин api-breaker, который реализует этот шаблон.

Редиректы

При обновлении API их конечные точки могут измениться. Традиционно это означало, что клиентское приложение должно отправлять запросы на /новую-конечную-точку-API вместо /старой-конечной-точки-API, что требовало ручного изменения каждого вызова этой конечной точки API клиентами.

Если это неожиданно, это может привести к ошибкам в клиентских приложениях.

С помощью API-шлюза вы можете предоставить абстракцию и перенаправлять запросы на /новую-конечную-точку-API без изменения запросов со стороны клиентов. С правильными кодами состояния и сообщениями о перенаправлении, вы можете постепенно отказаться от использования /старой-конечной-точки-API, не вызывая перерыва в работе для ваших клиентов.

С помощью APISIX вы можете использовать плагин redirect для настройки перенаправлений.

Заключение

Когда надежность становится первостепенной задачей, становится очевидным, что API-шлюзы необходимы, поскольку все больше организаций разбивают свои монолиты на микросервисы и переходят к облачным архитектурам.

Однако это не означает, что API-шлюзы подходят для всех. В зависимости от размера и использования вашего API, API-шлюз может быть избыточным, и вы можете обойтись использованием обратного прокси с основными возможностями маршрутизации и балансировки нагрузки.

Указанные здесь примеры использования лишь кратко затрагивают возможности API-шлюзов. Вы можете узнать больше о API-шлюзах и Apache APISIX на сайте apisix.apache.org.