**Конфигурация службы**

Как владельцы служб могут использовать конфигурацию службы для управления поведением клиентов.

Обзор

Конфигурация сервиса определяет, как клиенты gRPC должны вести себя при взаимодействии с сервером gRPC. Владельцы сервисов могут предоставить конфигурацию сервиса с ожидаемым поведением всех клиентов сервиса. Настройки в конфигурации сервиса всегда применяются к конкретной целевой строке (например, «api.myapp.com»), а не глобально.

Поведение, контролируемое Конфигурацией Службы

Настройки в конфигурации сервиса влияют на балансировку нагрузки на стороне клиента, поведение вызовов и проверку работоспособности.

На этой странице описаны параметры конфигурации сервиса, но полная структура данных конфигурации сервиса описана в определении protobuf.

Балансировка нагрузки

Сервис может состоять из нескольких серверов, и конфигурация балансировки нагрузки определяет, как вызовы от клиентов должны распределяться между этими серверами. По умолчанию используется политика балансировки нагрузки pick\_first, но в конфигурации сервиса можно указать другую политику. Например, если указать политику round\_robin, клиенты будут переключаться между серверами вместо того, чтобы постоянно использовать первый сервер.

Поведение при вызове

RPC могут быть сконфигурированы многими способами:

* Если включена функция wait-for-ready, то при невозможности подключения клиента к серверной части RPC-запрос будет отложен, а не сразу прерван.
* Можно указать время ожидания вызова, обозначив максимальное время, которое клиент должен ожидать, прежде чем отказаться от RPC.
* Один из:
  + Политика повторных попыток (максимальное количество попыток, настройки задержки, коды состояния, допускающие повторную попытку)
  + Хеджирование политики (максимальное количество попыток, задержка, некритичные коды состояния)

Примечание

Эти настройки поведения при вызове могут быть ограничены отдельной услугой или методом.

Политики повторных попыток и хеджирования можно дополнительно настроить, задав *политику ограничения повторных попыток*, но она будет применяться ко всем сервисам и методам.

Проверка работоспособности

Клиент может быть настроен на выполнение проверки работоспособности с указанием имени проверки работоспособности. В этом случае клиент будет использовать стандартную службу проверки работоспособности gRPC.

Получение конфигурации службы

A service config can be provided to a client either via name resolution or programatically by the client application.

Name Resolution

Механизм разрешения имён gRPC позволяет подключать реализации резолверов имён. Эти реализации возвращают адреса, связанные с именем, а также связанную с ним конфигурацию сервиса. Это механизм, который владельцы сервисов могут использовать для распространения конфигурации своих сервисов среди множества клиентов gRPC.

* Преобразователь имён xDS преобразует конфигурацию xDS, полученную из плоскости управления, в соответствующую конфигурацию службы.
* Стандартный преобразователь DNS-имен в реализации Go поддерживает конфигурации сервисов, хранящиеся в виде записей TXT на сервере имён.

Примечание

Несмотря на то, что структура конфигурации сервиса задокументирована с помощью определения протокола, внутреннее представление в клиенте — это JSON. Реализации преобразователей имён могут хранить информацию о конфигурации сервиса любым удобным для них способом, если они предоставляют её в формате JSON во время разрешения имён.

Программно

Клиентский API gRPC позволяет указать конфигурацию службы в формате JSON. Это используется для предоставления конфигурации службы по умолчанию, которая будет использоваться в ситуациях, когда преобразователь имён не предоставляет конфигурацию службы. Это также может быть полезно в некоторых ситуациях при тестировании.

Пример Конфигурации службы

Приведенный ниже пример выполняет следующее:

* Включает round\_robin политику балансировки нагрузки.
* Устанавливает время ожидания вызова по умолчанию, равное 1 секунде, которое применяется ко всем методам во всех сервисах.
* Увеличивает время ожидания до 2 секунд для метода bar в сервисе foo и для всех методов в сервисе baz

{

**"loadBalancingConfig"**: [ { **"round\_robin"**: {} } ],

**"methodConfig"**: [

{

**"name"**: [{}],

**"timeout"**: "1s"

},

{

**"name"**: [

{ **"service"**: "foo", **"method"**: "bar" },

{ **"service"**: "baz" }

],

**"timeout"**: "2s"

}

]

}

Последнее изменение 29 февраля 2024 г.: используйте абсолютные пути вместо абсолютных URL (#1268) (4f733b4)