**Показатели OpenTelemetry**

Показатели OpenTelemetry, доступные в gRPC

Обзор

gRPC поддерживает плагин OpenTelemetry, предоставляющий показатели, которые могут вам помочь.

* Устранение неполадок в вашей системе
* Повторяйте шаги по повышению производительности системы
* Настройте непрерывный мониторинг и оповещение.

Предыстория

OpenTelemetry — это платформа для создания и управления данными телеметрии. Ранее gRPC предоставлял поддержку телеметрии через OpenCensus, которая была прекращена в пользу OpenTelemetry.

Инструменты

Плагин gRPC OpenTelemetry принимает MeterProvider и использует OpenTelemetry API для создания Meter, который идентифицирует используемую библиотеку gRPC, например, grpc-c++ версии 1.57.1. С помощью этого Meter создаются следующие инструменты. Пользователи должны использовать OpenTelemetry SDK для настройки представлений, экспортируемых OpenTelemetry.

Все больше и больше компонентов gRPC оснащаются средствами мониторинга. В настоящее время мы оснастили следующими компонентами:

* Для каждого вызова (стабильно, включено по умолчанию): отслеживайте сами RPC (например, задержку).
  + Клиент за звонок: Наблюдайте за звонком клиента
  + Попытки для каждого клиента: отслеживайте попытки для каждого клиентского вызова, поскольку вызов может иметь несколько попыток из-за повторных вызовов или хеджирования.
  + Server : Observe a call received at the server.
* LB Policy : Observe various load-balancing policies
  + Weighted Round Robin (experimental)
  + Pick-First (experimental)
* XdsClient (experimental)

**NOTE** Some instruments are off by default and need to be explicitly enabled from the gRPC OpenTelemetry plugin API. Experimental metrics are always off by default. (Reference C++ API)

Per-Call Metrics

Client Per-Call Instruments

NameTypeUnitLabels (required)Descriptiongrpc.client.call.durationHistogramsgrpc.method, grpc.target , grpc.statusThis metric aims to measure the end-to-end time the gRPC library takes to complete an RPC from the application’s perspective.

Refer A66: OpenTelemetry Metrics for details.

Client Per-Attempt Instruments

NameTypeUnitLabels (disposition)Descriptiongrpc.client.attempt.  
startedCounter{attempt}grpc.method (required), grpc.target (required)The total number of RPC attempts started, including those that have not completed.grpc.client.attempt.  
durationHistogramsgrpc.method (required), grpc.target (required), grpc.status (required), grpc.lb.locality (optional)End-to-end time taken to complete an RPC attempt including the time it takes to pick a subchannel.grpc.client.attempt.  
sent\_total\_compressed\_message\_sizeHistogramBygrpc.method (required), grpc.target (required), grpc.status (required), grpc.lb.locality (optional)Total bytes (compressed but not encrypted) sent across all request messages (metadata excluded) per RPC attempt; does not include grpc or transport framing bytes.grpc.client.attempt.  
rcvd\_total\_compressed\_message\_sizeHistogramBygrpc.method (required), grpc.target (required), grpc.status (required), grpc.lb.locality (optional)Total bytes (compressed but not encrypted) received across all response messages (metadata excluded) per RPC attempt; does not include grpc or transport framing bytes.

Refer A66: OpenTelemetry Metrics for details.

Server Instruments

NameTypeUnitLabels (required)Descriptiongrpc.server.call.  
startedCounter{call}grpc.methodThe total number of RPCs started, including those that have not completed.grpc.server.call.  
sent\_total\_compressed\_message\_sizeHistogramBygrpc.method, grpc.statusTotal bytes (compressed but not encrypted) sent across all response messages (metadata excluded) per RPC; does not include grpc or transport framing bytes.grpc.server.call.  
rcvd\_total\_compressed\_message\_sizeHistogramBygrpc.method, grpc.statusTotal bytes (compressed but not encrypted) received across all request messages (metadata excluded) per RPC; does not include grpc or transport framing bytes.grpc.server.call.  
durationHistogramsgrpc.method, grpc.statusThis metric aims to measure the end2end time an RPC takes from the server transport’s (HTTP2/ inproc) perspective.

Refer A66: OpenTelemetry Metrics for details.

LB Policy Instruments

Weighted Round Robin LB Policy Instruments

NameTypeUnitLabels (disposition)Descriptiongrpc.lb.wrr.  
rr\_fallbackCounter{update}grpc.target (required), grpc.lb.locality (optional)EXPERIMENTAL: Number of scheduler updates in which there were not enough endpoints with valid weight, which caused the WRR policy to fall back to RR behavior.grpc.lb.wrr.  
endpoint\_weight\_not\_yet\_usableCounter{endpoint}grpc.target (required), grpc.lb.locality (optional)EXPERIMENTAL: Number of endpoints from each scheduler update that don’t yet have usable weight information (i.e., either the load report has not yet been received, or it is within the blackout period).grpc.lb.wrr.  
endpoint\_weight\_staleCounter{endpoint}grpc.target (required), grpc.lb.locality (optional)EXPERIMENTAL: Number of endpoints from each scheduler update whose latest weight is older than the expiration period.grpc.lb.wrr.  
endpoint\_weightsHistogram{weight}grpc.target (required), grpc.lb.locality (optional)EXPERIMENTAL: Weight of an endpoint recorded every scheduler update.

Refer A78: gRPC OTel Metrics for WRR, Pick First, and XdsClient for details.

Pick First LB Policy Instruments

NameTypeUnitLabels (required)Descriptiongrpc.lb.pick\_first.  
disconnectionsCounter{disconnection}grpc.targetEXPERIMENTAL: Number of times the selected subchannel becomes disconnected.grpc.lb.pick\_first.  
connection\_attempts\_succeededCounter{attempt}grpc.targetEXPERIMENTAL: Number of successful connection attempts.grpc.lb.pick\_first.  
connection\_attempts\_failedCounter{attempt}grpc.targetEXPERIMENTAL: Number of failed connection attempts.

Refer A78: gRPC OTel Metrics for WRR, Pick First, and XdsClient for details.

XdsClient Instruments

NameTypeUnitLabels (required)Descriptiongrpc.xds\_client.  
connectedGauge{bool}grpc.target, grpc.xds.serverEXPERIMENTAL: Whether or not the xDS client currently has a working ADS stream to the xDS server.grpc.xds\_client.  
server\_failureCounter{failure}grpc.target, grpc.xds.serverEXPERIMENTAL: A counter of xDS servers going from healthy to unhealthy.grpc.xds\_client.  
resource\_updates\_validCounter{resource}grpc.target, grpc.xds.server, grpc.xds.resource\_typeEXPERIMENTAL: A counter of resources received that were considered valid, even if unchanged.grpc.xds\_client.  
resource\_updates\_invalidCounter{resource}grpc.target, grpc.xds.server, grpc.xds.resource\_typeEXPERIMENTAL: A counter of resources received that were considered invalid.grpc.xds\_client.  
resourcesGauge{resource}grpc.target, grpc.xds.authority, grpc.xds.cache\_state, grpc.xds.resource\_typeEXPERIMENTAL: Number of xDS resources.

Refer A78: gRPC OTel Metrics for WRR, Pick First, and XdsClient for details.

Labels/Attributes

With a recorded measurement for an instrument, gRPC might provide some additional information as attributes or labels. For example, grpc.client.attempt.started has the labels grpc.method and grpc.target along with each measurement that tell us the method and the target associated with the RPC attempt being observed.

**NOTE** Some attributes are marked as optional on the instruments. These need to be explicitly enabled from the gRPC OpenTelemetry Plugin API. (Reference C++ API)

NameDescriptiongrpc.methodFull gRPC method name, including package, service and method, e.g. “google.bigtable.v2.Bigtable/CheckAndMutateRow”.grpc.statusgRPC server status code received, e.g. “OK”, “CANCELLED”, “DEADLINE\_EXCEEDED”.grpc.цельКанонический целевой URI, используемый при создании канала gRPC, например «dns:///pubsub.googleapis.com:443», «xds:///helloworld-gke:8000».grpc.lb.населенный пунктНаселенный пункт, в который отправляется трафик.grpc.xds.серверДля клиентов указывает цель канала gRPC, в котором используется XdsClient. Для серверов будет использоваться строка «#сервер».grpc.xds.authorityУполномоченный xDS. Значение будет «#old» для старых имен ресурсов, не использующих xdstp.grpc.xds.cache\_stateУказывает состояние кэша ресурса xDS («запрашивается», «не существует», «подтверждено», «не подтверждено», «не подтверждено, но кэшировано»).grpc.xds.resource\_typeТип ресурса xDS, например «envoy.config.listener.v3.Listener».

Вопросы и ответы

Вопрос: Как мне получить пропускную способность или QPS (количество запросов в секунду)?

Используйте агрегацию подсчёта для показателей гистограммы задержки — grpc.client.attempt.duration / grpc.client.call.duration (для клиентов) или grpc.server.call.duration (для серверов).

В. Как мне получить частоту ошибок для RPC?

Количество ошибок можно подсчитать с помощью значения фильтра grpc.status != OK в гистограмме задержек grpc.client.attempt.duration / grpc.client.call.duration (для клиентов) или grpc.server.call.duration (для серверов).

Языковые примеры

ЯзыкПримерC++Пример C ++ВпередПерейти к ПримеруJavaПример JavaПитонПример Python

Дополнительные ресурсы

* A66: Показатели OpenTelemetry
* A78: показатели gRPC OTel для WRR, Pick First и XdsClient
* A79: Архитектура без учета показателей для каждого вызова

Последнее изменение 24 января 2025 года: OTel Metrics: обновление ссылки на видео на YouTube (#1406) (e4f45b9)