

# 13、Skywalking的埋点数据发送-基于kafka的默认发送机制

 rock\_fish 关注2021.06.23 14:02:50 字数 592 阅读 229

## TraceSegment 基于Kafka的发送

基于kafka的数据推送和消费中，有这么一个疑问萦绕：OAPServer的数据聚合是怎么实现的？

- 如果应用A的实例AA的数据都提交在同一个OAPServer实例中，那么在这个实例中就可以进行数据聚合
- 如果应用A的实例AA的数据提交在多个不同一个OAPServer实例中，那么每个实例中聚合之后，还要再把每个实例聚合后的数据再进行一次聚合。

其大致的处理流程是这样的：采集->发送kafka->OAP消费kafka->聚合/存储

这里先看采集和发送的处理，OAP的处理在OAP端流程梳理的章节看

## 消息发送的分区策略

KafkaTraceSegmentServiceClient 中梳理一下2个知识点：

- 1.看发送的消息是否有key 和分区信息
- 2.看分区算法

### 1. 看消息构造

在 KafkaTraceSegmentServiceClient#consume 中可以看到构造的 ProducerRecord 中 指定了topic和key

```
1 public void consume(final List<TraceSegment> data) {
2     data.forEach(traceSegment -> {
3         SegmentObject upstreamSegment = traceSegment.transform();
4         ProducerRecord<String, Bytes> record = new ProducerRecord<>(
5             topic,//指定了topic
6             upstreamSegment.getTraceSegmentId(),//指定了key
7             Bytes.wrap(upstreamSegment.toByteArray())
8         );
9         ...省略部分代码
10    });
11 }
```

### 2. 看分区算法

默认情况下未指定Partitioner，则使用Kafka默认的DefaultPartitioner的分区算法  
从源码可知，指定了key，则会按照key做hash算出分区，key不变的情况下，所算出的分区是一致的

```
1 public int partition(String topic, Object key, byte[] keyBytes, Object value, byte[] value
2     if (keyBytes == null) {
3         return stickyPartitionCache.partition(topic, cluster);
4     }
5     List<PartitionInfo> partitions = cluster.partitionsForTopic(topic);
6     int numPartitions = partitions.size();
7     // hash the keyBytes to choose a partition
8     return Utils.toPositive(Utils.murmur2(keyBytes)) % numPartitions;
9 }
```

原画学习课程

 rock\_fish 总资产6关注

21、Skywalking的埋点-Agent动态采样控制  
阅读 219

22、skywalking的Trace数据协议  
阅读 12

skywalking的日常维护1:  
com.netflix.zuul.exception.ZuulExcept  
阅读 41

推荐阅读

Kafka的设计原理介绍  
阅读 354

List 去除重复数据的五种方式  
阅读 240

iOS 动态方法解析、消息转发实现原理  
阅读 159

RocketMQ保证顺序消费demo  
阅读 525

多线程消费顺序-01  
阅读 293

大数据教程





根据1, 2可知, 在使用默认的分发策略的情况下, 相同的TraceSegmentId 消息提交到同一个kafka分区中。

#### 4.TraceSegmentId 的生命周期

TraceSegment的构造中给traceSegmentId进行了赋值。

```
1 public TraceSegment() {
2     this.traceSegmentId = GlobalIdGenerator.generate();
3     this.spans = new LinkedList<>();
4     this.relatedGlobalTraceId = new NewDistributedTraceId();
5     this.createTime = System.currentTimeMillis();
6 }
```

一个TraceSegment有唯一的一个TraceSegmentId, 接下来就需要梳理清楚是不是一次调用在一个进程内就有一个TraceSegment呢? 先给出答案进程创建第一个EntrySpan的时候会先创建一个TraceSegment, 其内部承载了一个span集合。

梳理源码的入口是这样的: 对于skywalking来说, 进程的入口点是创建一个EntrySpan,所以我们从 `ContextManager#createEntrySpan` 源码中跟踪下去就能寻找到答案:

```
1 public static AbstractSpan createEntrySpan(String operationName, ContextCarrier carrier) {
2     AbstractSpan span;
3     AbstractTracerContext context;
4     operationName = StringUtil.cut(operationName, OPERATION_NAME_THRESHOLD);
5     ...
6     context = getOrCreate(operationName, false); // 这里有创建Context的机会
7     span = context.createEntrySpan(operationName);
8     ...
9     return span;
10 }
```

`context = getOrCreate(operationName, false)` 这个方法中, 会有创建TraceContext的逻辑:

```
1 context = EXTEND_SERVICE.createTraceContext(operationName, forceSampling);
```

继续跟入代码, 可以看到new TracingContext的操作

```
1 context = new TracingContext(operationName, spanLimitWatcher);
```

TracingContext 中有个成员 `private TraceSegment segment`, 在TracingContext构造方法中可以看到TraceSegment的创建过程

```
1 TracingContext(String firstOPName, SpanLimitWatcher spanLimitWatcher) {
2     this.segment = new TraceSegment();
3     ...
4 }
```



0人点赞 >

监控



更多精彩内容, 就在简书APP

"小礼物"关注我"

赞赏支持

还没有人赞赏，支持一下

rock\_fish

总资产6 共写了9.4W字 获得82个赞 共27个粉丝

关注

新款雷克萨斯ES上市售价10.98万元起



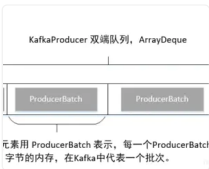
推荐阅读

[Kafka] Producer发送消息机制解析

1. 前言 Sync Producer：低延迟，低吞吐率，无数据丢失 Async Producer：高延迟，高吞吐...

LZhan

阅读 797 评论 0 赞 2

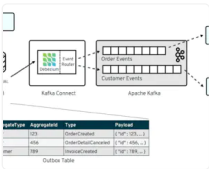


通过Kafka分布式事务实现微服务数据交换与发件箱模式

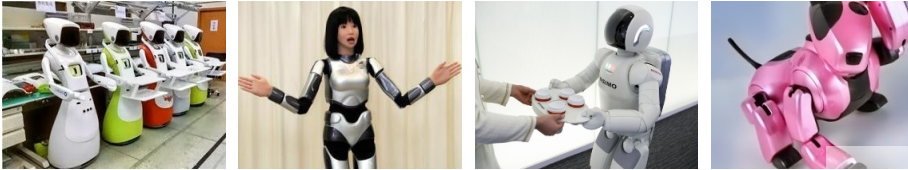
作为其业务逻辑的一部分，微服务通常不仅需要更新自己的本地数据存储，而且还需要向其他服务通知发生的数据更改。发件箱模...

水岩

阅读 977 评论 0 赞 0



智能机器人报价，货到付款！

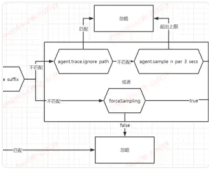


12、Skywalking的埋点-TraceSegment数据的创建、收集和发...

采样创建TraceSegment 默认情况下，每个请求都应该生成一条完整的 Trace。但在面对海量请求时这就会给...

rock\_fish

阅读 121 评论 0 赞 0



2018-07-18

董多娇第226天坚持分享，焦点相信，每个人在每一刻都会为自己做出一个决定与选择，是他们当时认为最合适自己的，所以任...

良知良能良知良能

阅读 1,271 评论 1 赞 1

初识jQuery之jQuery设计思想（一）

写下你的评论...

评论0 赞

