后分布式追踪时代的性能问题定位 -- 方法级性能剖析

Sheng Wu 吴晟, Founding Engineer, Tetrate.io

Apache SkyWalking Founder & V.P. Apache Member

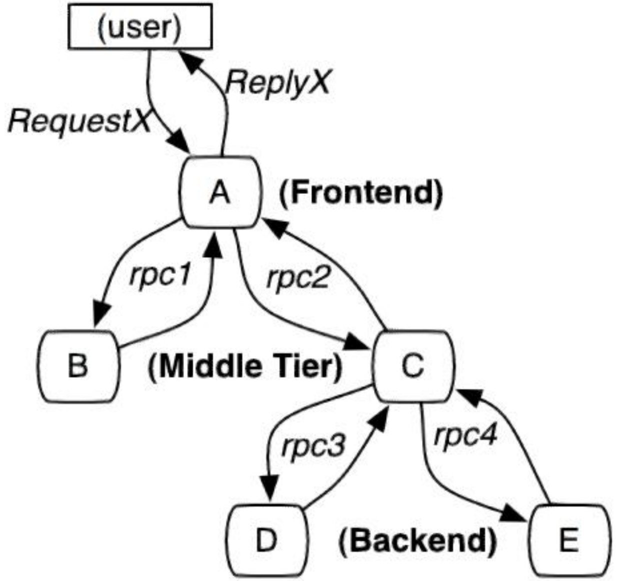
（微）服务化、云原生化、容器化

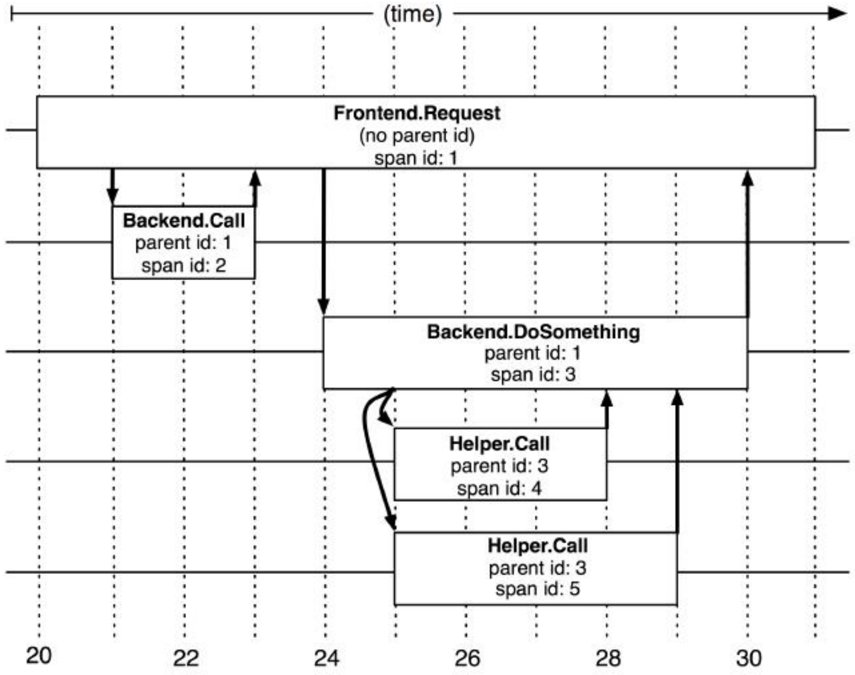
- 扩容和管理需求
- 分布式程序加剧
- 依赖复杂
- 问题诊断困难
- 传统手段失效，特别是传统日志

2010 Google Dapper

- Google搜索引擎的分布式追踪探索
- 一篇广为流传的论文 https://research.google/pubs/pub36356/
- 2012 Twitter实现Zipkin
- 2012 Naver开源Pinpoint
- 2015 个人SkyWalking开源
- 2017 SkyWalking加入Apache孵化器
- 2019 SkyWalking成为Apache顶级项目

分布式追踪简介 – 带上下文的日志

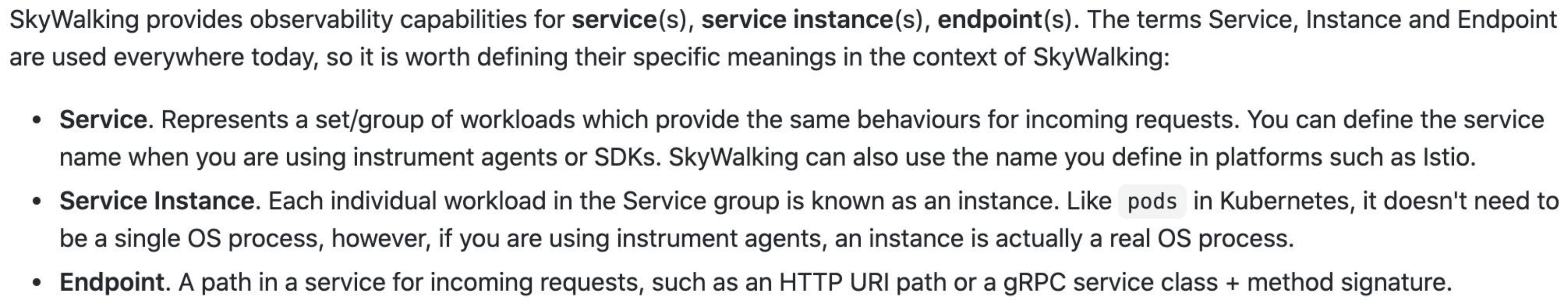




为什么SkyWalking和传统分布式追踪系统不同?

Distributed Tracing
V.S.
Distributed Tracing based APM

SkyWalking 原生概念



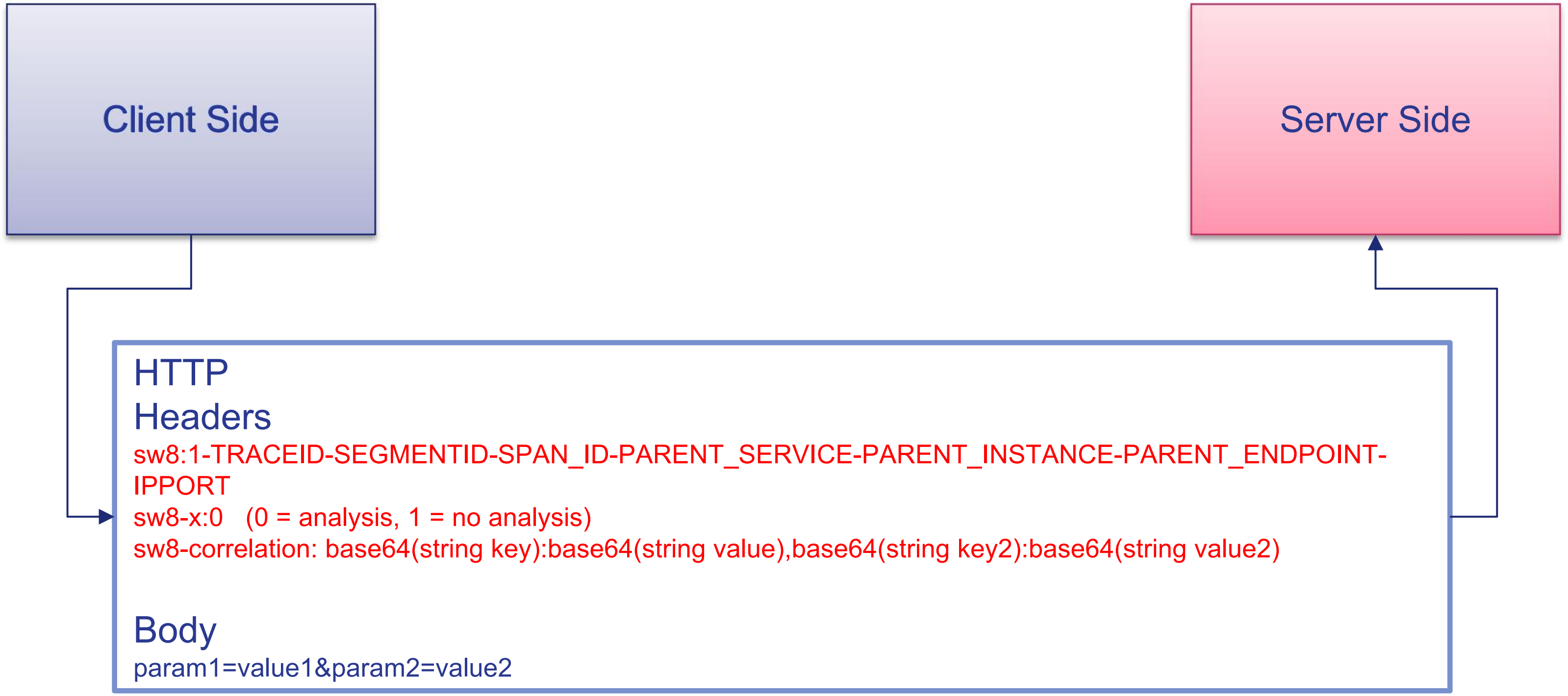
SkyWalking provides observability capabilities for **service**(s), **service instance**(s), **endpoint**(s). The terms Service, Instance and Endpoint are used everywhere today, so it is worth defining their specific meanings in the context of SkyWalking:

- **Service**. Represents a set/group of workloads which provide the same behaviours for incoming requests. You can define the service name when you are using instrument agents or SDKs. SkyWalking can also use the name you define in platforms such as Istio.
- **Service Instance**. Each individual workload in the Service group is known as an instance. Like `pods` in Kubernetes, it doesn't need to be a single OS process, however, if you are using instrument agents, an instance is actually a real OS process.
- **Endpoint**. A path in a service for incoming requests, such as an HTTP URI path or a gRPC service class + method signature.

SkyWalking 原生概念

- Span
  - Entry/Local/Exit
  - Peer
  - Component
  - Layer
  - Tags
  - Logs
- Segment
  - Span Collection
- Segment Ref
  - Trace Id
  - Parent Segment Id
  - Parent Span Id
  - Parent service and instance
  - Parent endpoint

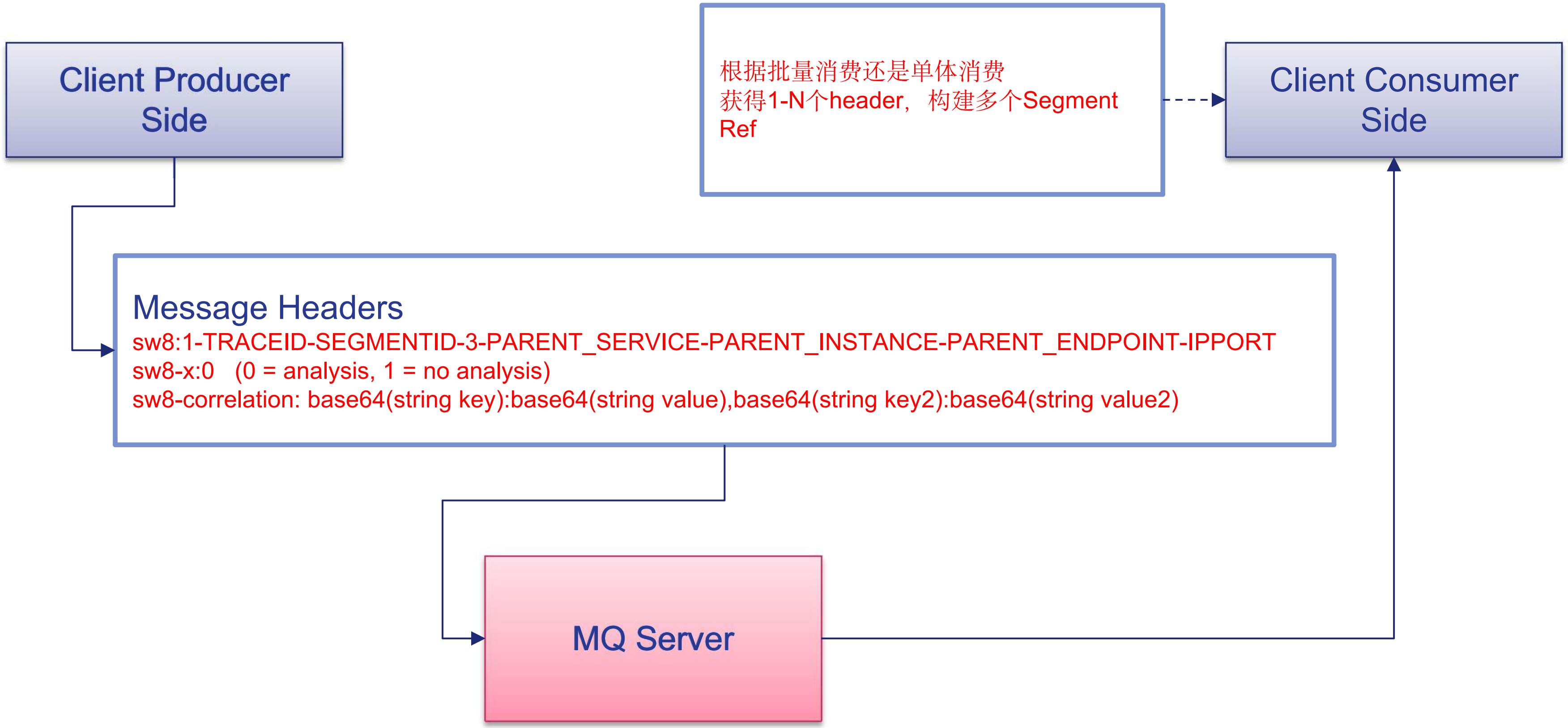
RPC Tracing



sw8:1-TRACEID-SEGMENTID-SPAN_ID-PARENT_SERVICE-PARENT_INSTANCE-PARENT_ENDPOINT-IPPORT
sw8-x:0   (0 = analysis, 1 = no analysis)
sw8-correlation: base64(string key):base64(string value),base64(string key2):base64(string value2)

param1=value1&param2=value2

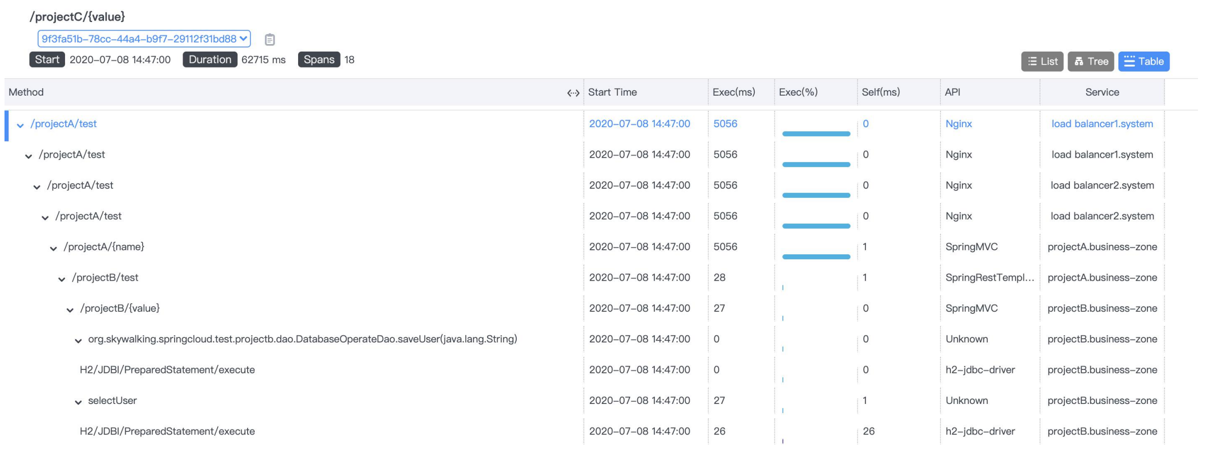
MQ Tracing



根据批量消费还是单体消费
获得1-N个header，构建多个Segment
Ref

Message Headers
sw8:1-TRACEID-SEGMENTID-3-PARENT_SERVICE-PARENT_INSTANCE-PARENT_ENDPOINT-IPPORT
sw8-x:0 (0 = analysis, 1 = no analysis)
sw8-correlation: base64(string key):base64(string value),base64(string key2):base64(string value2)

分布式调用栈



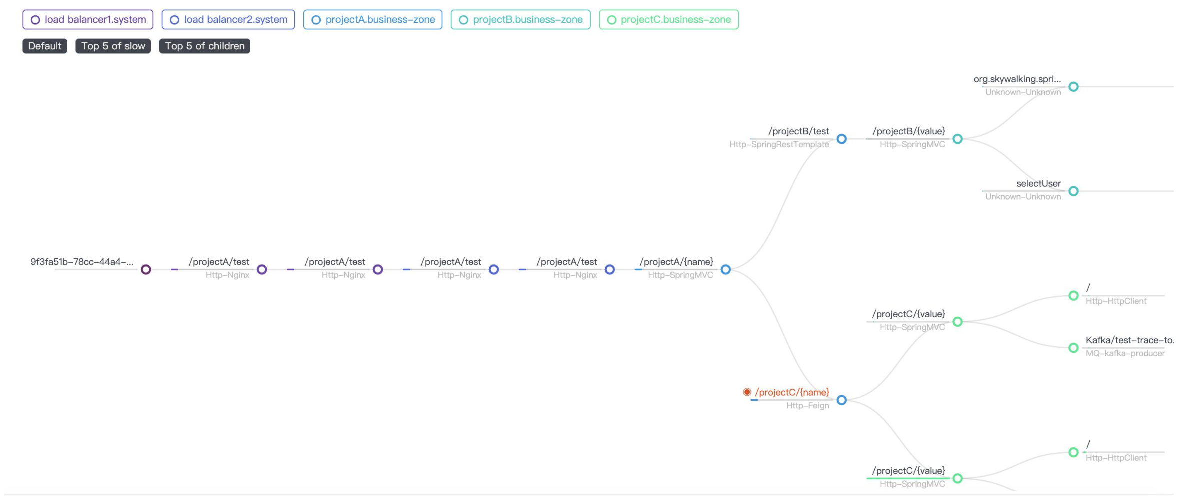
/projectC/{value}

9f3fa51b-78cc-44a4-b9f7-29112f31bd88

Start 2020-07-08 14:47:00 Duration 62715 ms Spans 18

List Tree Table

Method	Start Time	Exec(ms)	Exec(%)	Self(ms)	API	Service
/projectA/test	2020-07-08 14:47:00	5056		0	Nginx	load balancer1.system
/projectA/test	2020-07-08 14:47:00	5056		0	Nginx	load balancer1.system
/projectA/test	2020-07-08 14:47:00	5056		0	Nginx	load balancer2.system
/projectA/test	2020-07-08 14:47:00	5056		0	Nginx	load balancer2.system
/projectA/{name}	2020-07-08 14:47:00	5056		1	SpringMVC	projectA.business-zone
/projectB/test	2020-07-08 14:47:00	28		1	SpringRestTempl...	projectA.business-zone
/projectB/{value}	2020-07-08 14:47:00	27		0	SpringMVC	projectB.business-zone
org.skywalking.springcloud.test.projectb.dao.DatabaseOperateDao.saveUser(java.lang.String)	2020-07-08 14:47:00	0		0	Unknown	projectB.business-zone
H2/JDBI/PreparedStatement/execute	2020-07-08 14:47:00	0		0	h2-jdbc-driver	projectB.business-zone
selectUser	2020-07-08 14:47:00	27		1	Unknown	projectB.business-zone
H2/JDBI/PreparedStatement/execute	2020-07-08 14:47:00	26		26	h2-jdbc-driver	projectB.business-zone



Tags.

Endpoint:	H2/JDBI/PreparedStatement/execute
Span Type:	Exit
Component:	h2-jdbc-driver
Peer:	localhost:-1
Error:	false
db.type:	sql
db.instance:	test
db.statement:	INSERT INTO user(name) VALUES(?)

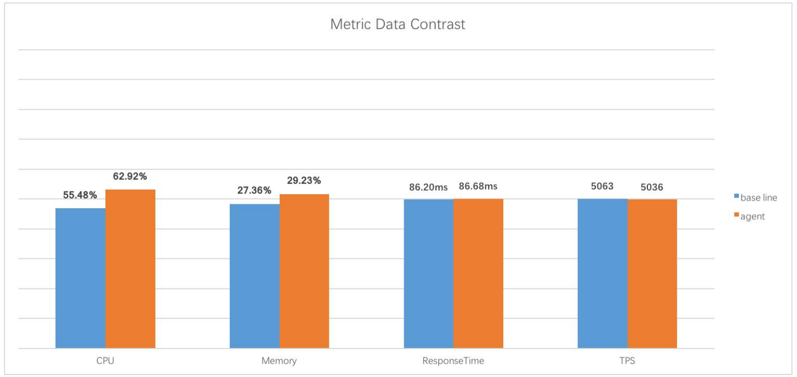
分布式追踪的两面性

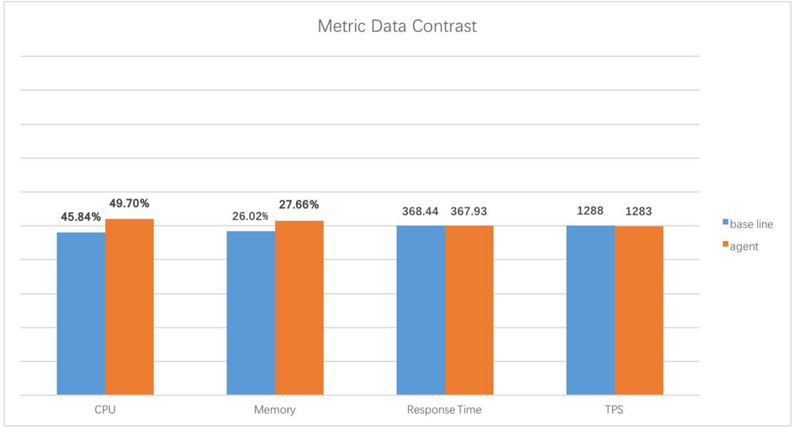
- 有效的移除分布式造成的信息缺失
    - 如RPC故障
    - 路由以及熔断情况
    - 分布式访问参数
- 重新加强日志的能力，trace和log可以集成使用

- Trace流量可能高达数百T每天
    - 追踪探针的额外负载
    - 后端分析能力
    - 后端存储能力
    - 数据时间有效性（TTL）
- Trace只是一个有上下文的日志么?
- 通过Trace还能得到什么?

分布式追踪消耗 – Span Overload

- 对于100ms / 1000+QPS 级别的应用访问，大概造成10%的性能损耗（< 20个span）
- 方法级追踪 -> 分布式追踪
- 日志 -> 分布式追踪
- 100%采样 （探针 / 后端）
- 直接上报 / MQ / Local Collector





Paper, STAM: Enhancing Topology Auto Detection For A Highly Distributed and Large-Scale Application System

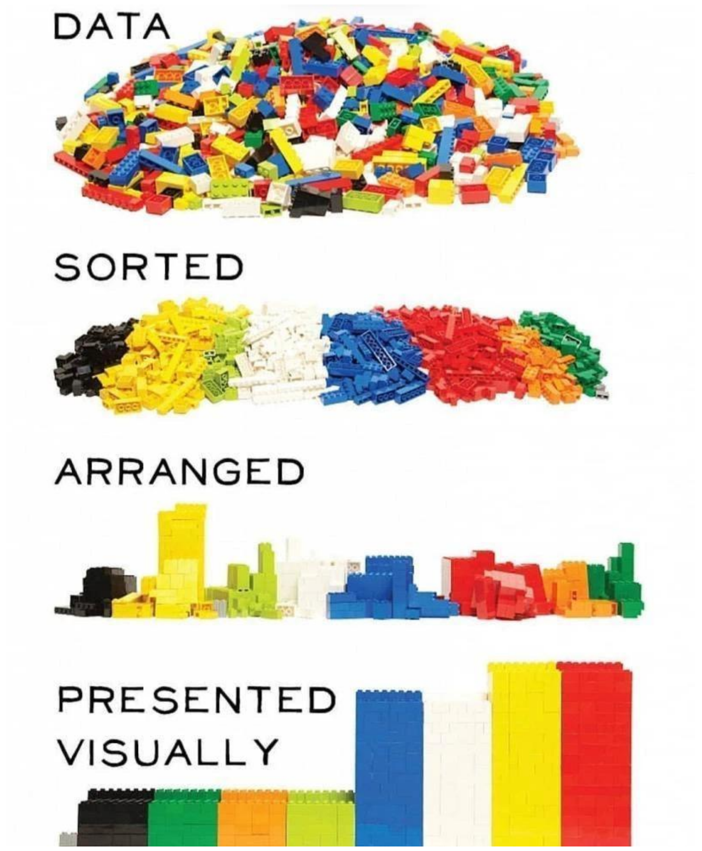
- SkyWalking特有的拓扑分析方法
- Peer别名机制

- https://wu-sheng.github.io/STAM/
- 中文翻译 https://wu-sheng.github.io/STAM/README-cn

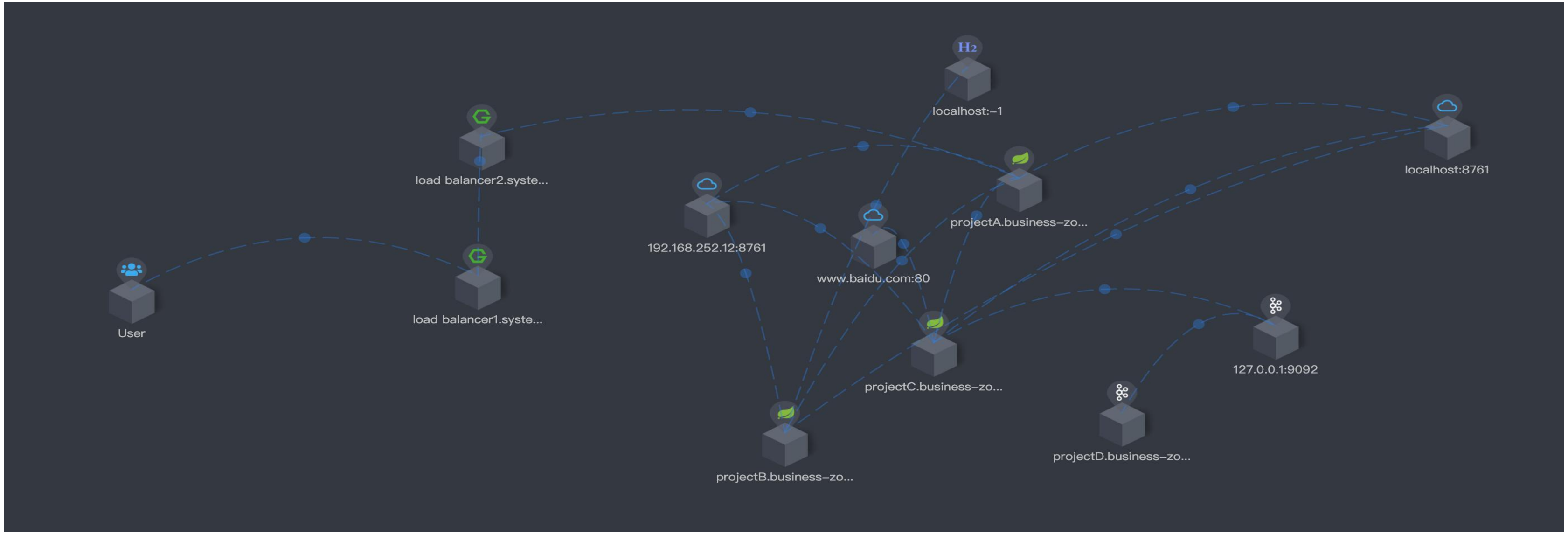
如何才能正确的定位问题

分布式追踪提供了数据

APM工具提供
聚合和
数据可视化

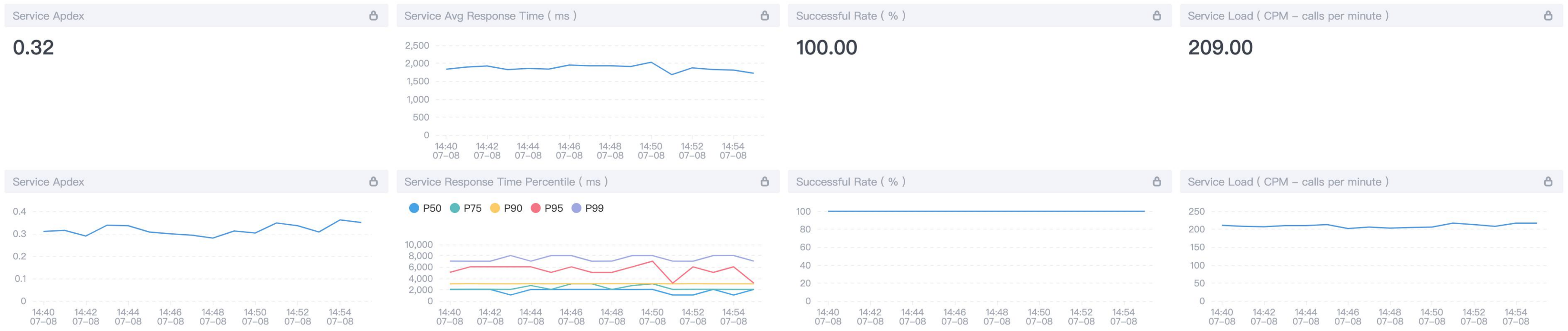


拓扑图的价值



- 理解系统
- 理论架构 -> 实际架构
- 领导需要
- 提供下钻点

监控的起点，metrics和告警

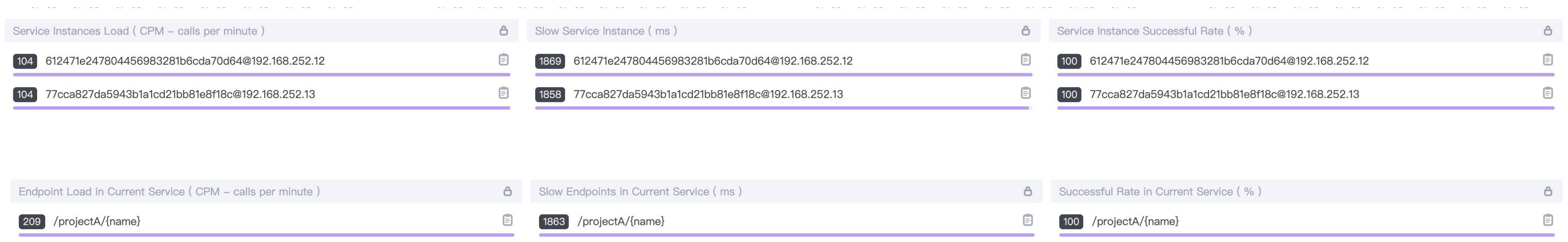

Service Apdex
0.32
Service Avg Response Time (ms)
Successful Rate (%)
100.00
Service Load (CPM – calls per minute)
209.00
Service Apdex
Service Response Time Percentile (ms)
P50 P75 P90 P95 P99
Successful Rate (%)
Service Load (CPM – calls per minute)

Entity name

Define the relation between scope and entity name.

- **Service**: Service name
- **Instance**: {Instance name} of {Service name}
- **Endpoint**: {Endpoint name} in {Service name}
- **Database**: Database service name
- **Service Relation**: {Source service name} to {Dest service name}
- **Instance Relation**: {Source instance name} of {Source service name} to {Dest instance name} of {Dest service name}
- **Endpoint Relation**: {Source endpoint name} in {Source Service name} to {Dest endpoint name} in {Dest service name}

下钻检查 – 实例 or Endpoint



Service Instances Load (CPM – calls per minute)

104	612471e247804456983281b6cda70d64@192.168.252.12
104	77cca827da5943b1a1cd21bb81e8f18c@192.168.252.13

Slow Service Instance (ms)

1869	612471e247804456983281b6cda70d64@192.168.252.12
1858	77cca827da5943b1a1cd21bb81e8f18c@192.168.252.13

Service Instance Successful Rate (%)

100	612471e247804456983281b6cda70d64@192.168.252.12
100	77cca827da5943b1a1cd21bb81e8f18c@192.168.252.13

Endpoint Load in Current Service (CPM – calls per minute)

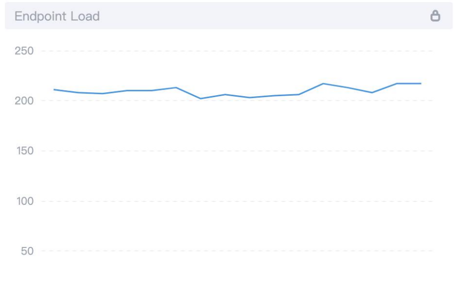
209	/projectA/{name}

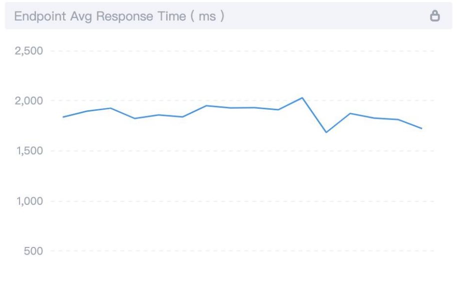
Slow Endpoints in Current Service (ms)

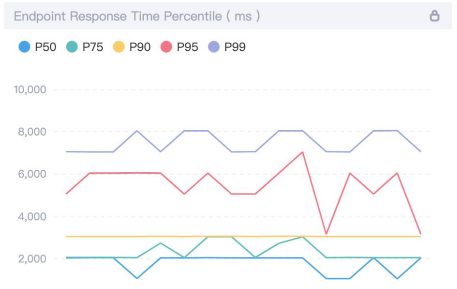
1863	/projectA/{name}

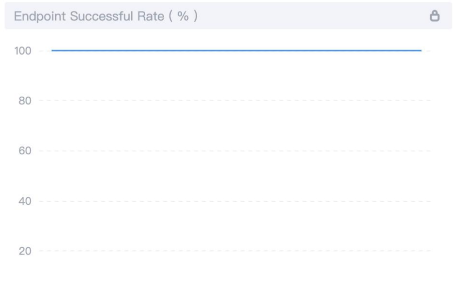
Successful Rate in Current Service (%)

100	/projectA/{name}

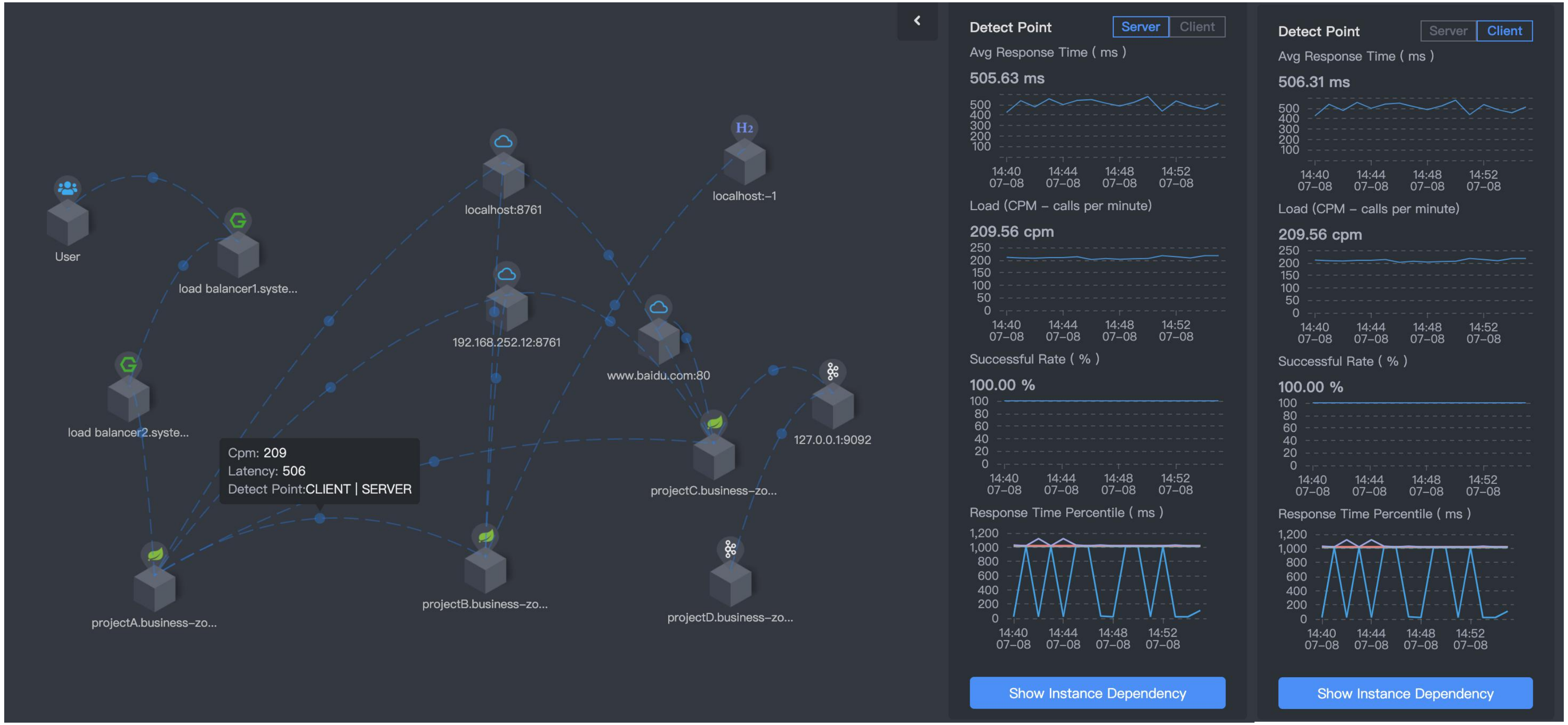








下钻检查 – 分布式依赖维度

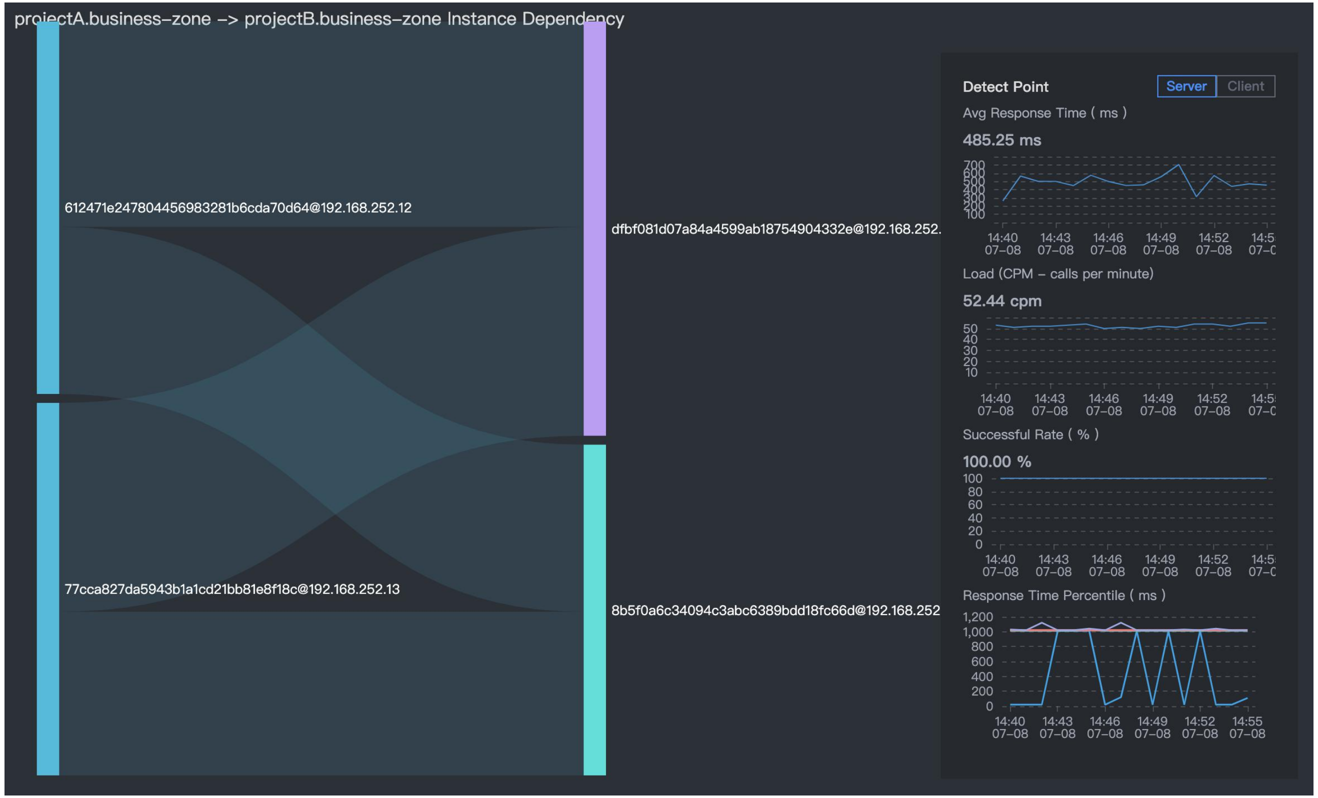


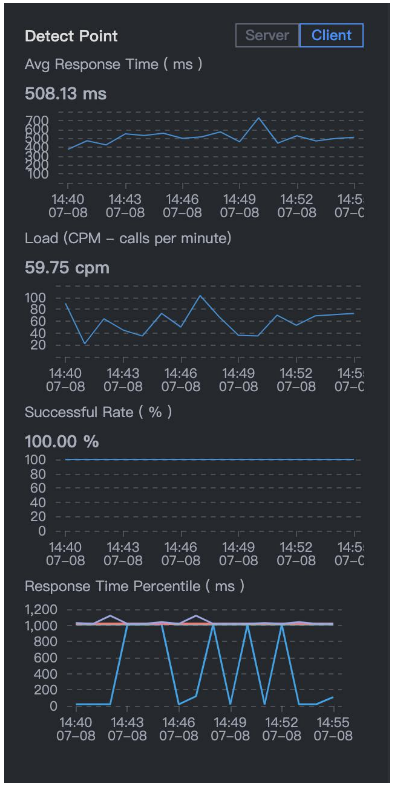
Cpm: 209
Latency: 506
Detect Point:CLIENT | SERVER

Detect Point: Server
Avg Response Time (ms)
505.63 ms
Load (CPM – calls per minute)
209.56 cpm
Successful Rate (%)
100.00 %
Response Time Percentile (ms)

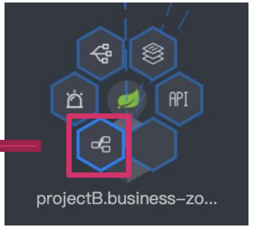
Detect Point: Client
Avg Response Time (ms)
506.31 ms
Load (CPM – calls per minute)
209.56 cpm
Successful Rate (%)
100.00 %
Response Time Percentile (ms)

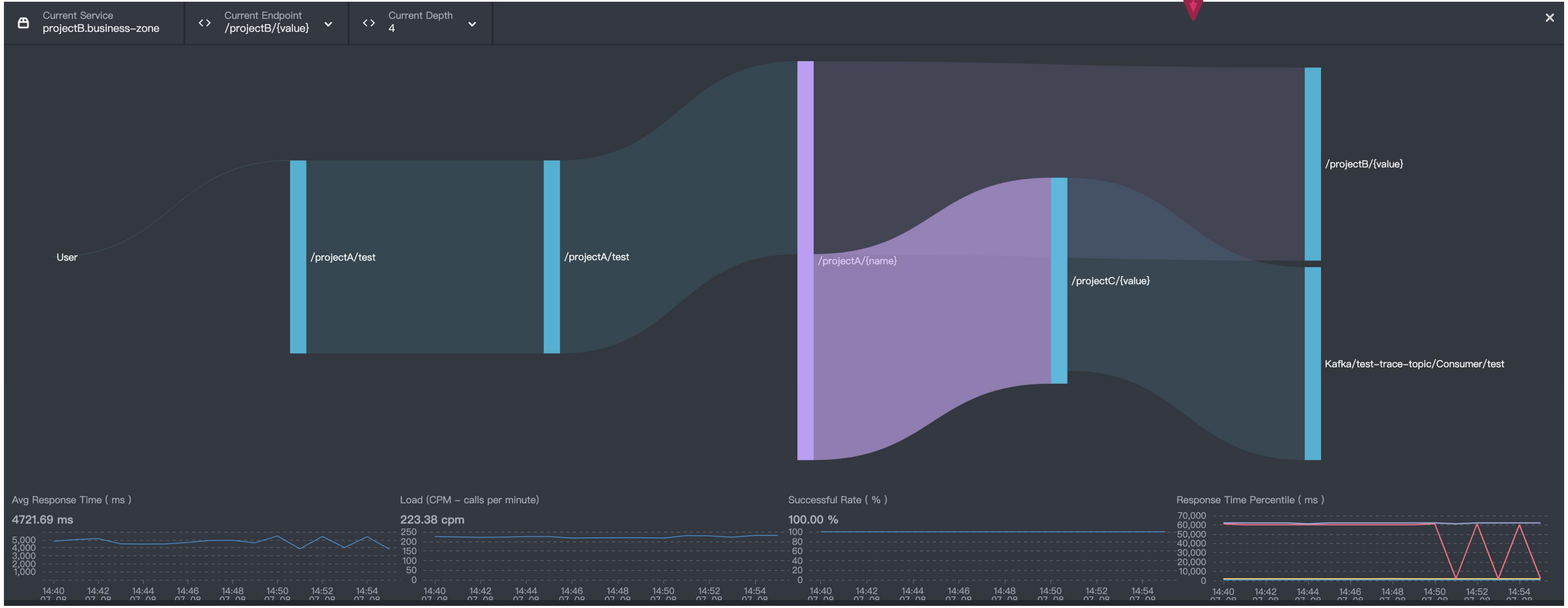
下钻检查 – 实例级别依赖故障（限流、网络）


projectA.business–zone –> projectB.business–zone Instance Dependency
612471e247804456983281b6cda70d64@192.168.252.12
77cca827da5943b1a1cd21bb81e8f18c@192.168.252.13
dfbf081d07a84a4599ab18754904332e@192.168.252.
8b5f0a6c34094c3abc6389bdd18fc66d@192.168.252
Detect Point
Server
Client
Avg Response Time (ms)
485.25 ms
Load (CPM – calls per minute)
52.44 cpm
Successful Rate (%)
100.00 %
Response Time Percentile (ms)

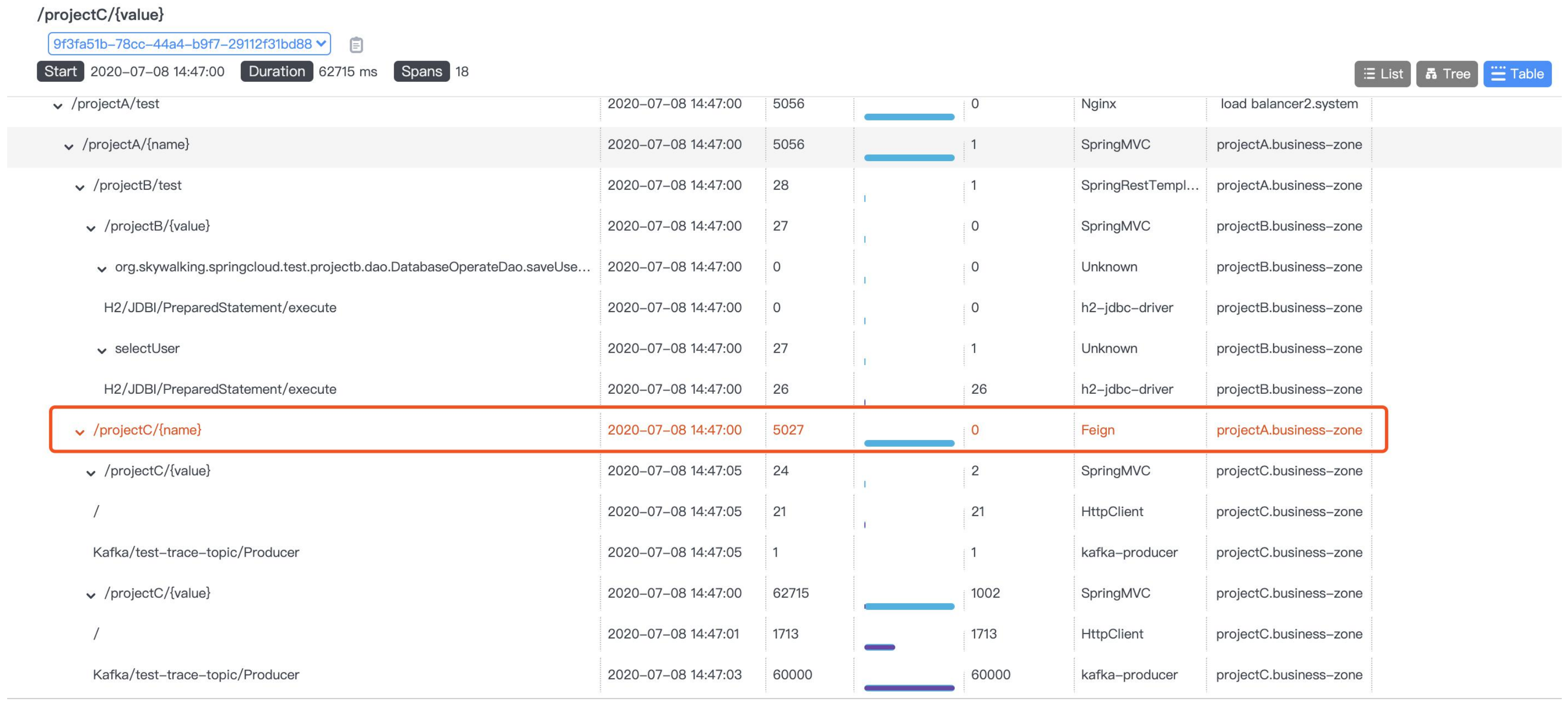


业务服务上下游依赖


API
projectB.business-zo...


Current Service
projectB.business-zone
Current Endpoint
/projectB/{value}
Current Depth
4
User
/projectA/test
/projectA/test
/projectA/{name}
/projectC/{value}
/projectB/{value}
Kafka/test-trace-topic/Consumer/test
Avg Response Time (ms)
4721.69 ms
Load (CPM – calls per minute)
223.38 cpm
Successful Rate (%)
100.00 %
Response Time Percentile (ms)

明细级别探查 – Error Span



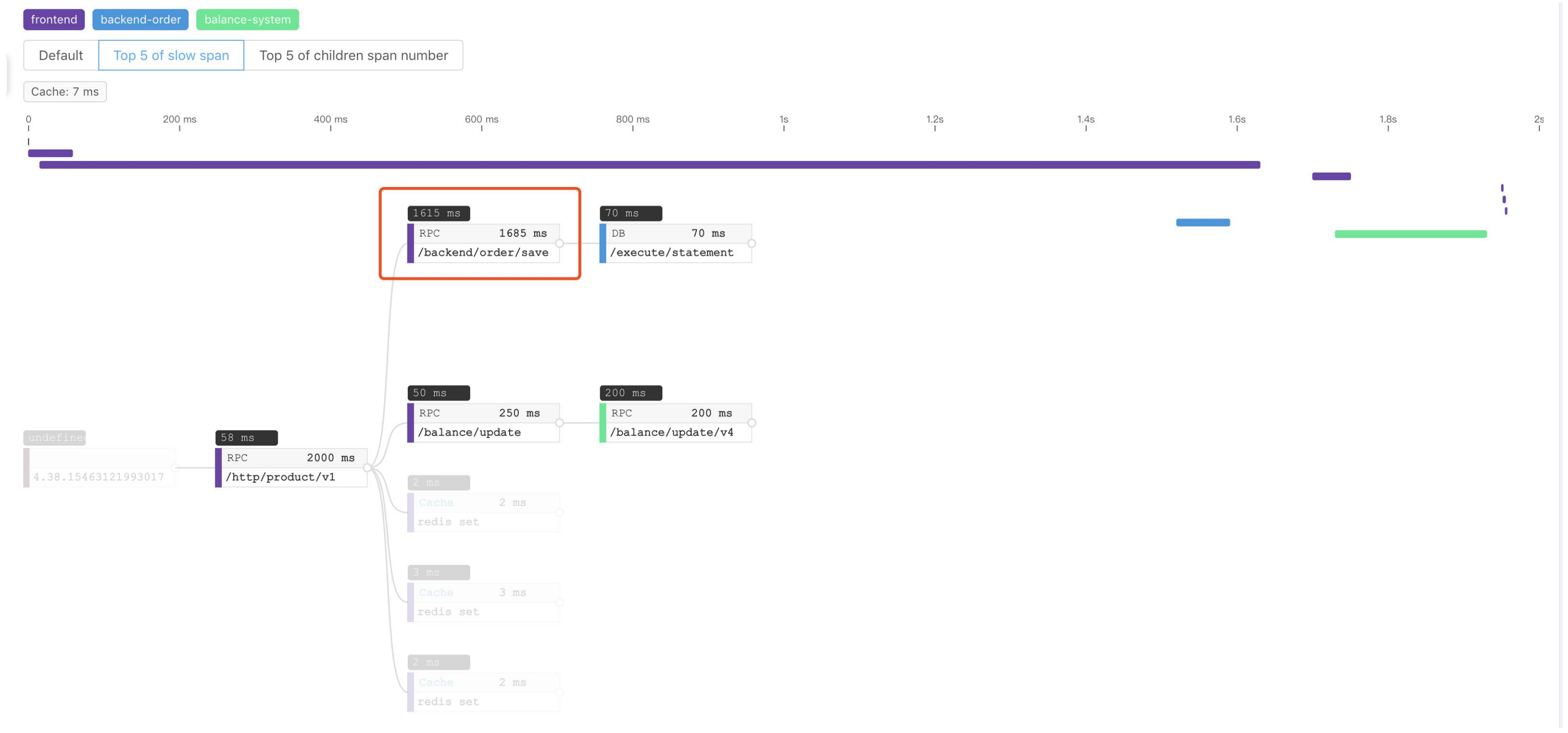
/projectC/{value}

9f3fa51b–78cc–44a4–b9f7–29112f31bd88

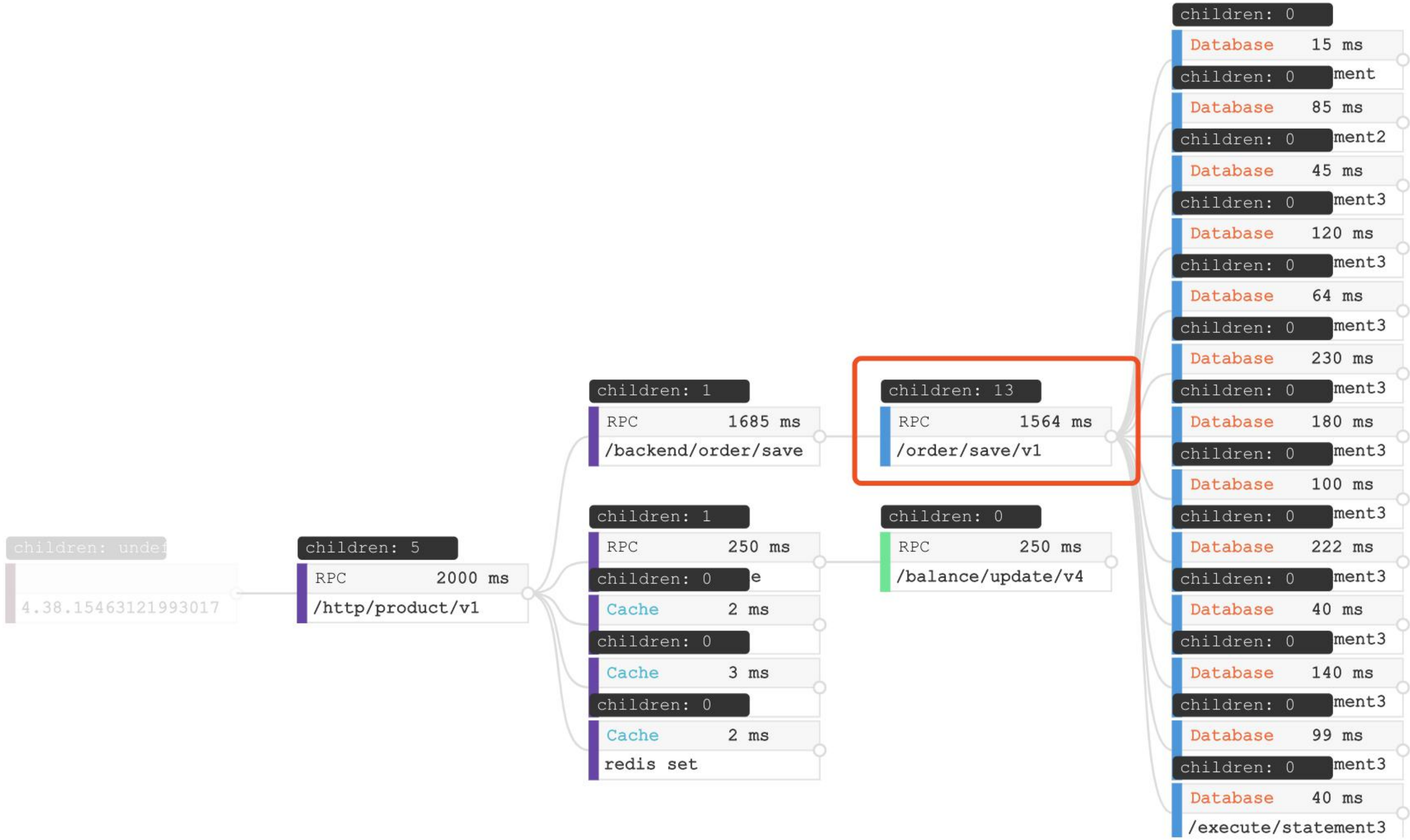
Start 2020–07–08 14:47:00 Duration 62715 ms Spans 18

/projectA/test	2020–07–08 14:47:00	5056	0	Nginx	load balancer2.system
/projectA/{name}	2020–07–08 14:47:00	5056	1	SpringMVC	projectA.business–zone
/projectB/test	2020–07–08 14:47:00	28	1	SpringRestTempl…	projectA.business–zone
/projectB/{value}	2020–07–08 14:47:00	27	0	SpringMVC	projectB.business–zone
org.skywalking.springcloud.test.projectb.dao.DatabaseOperateDao.saveUse…	2020–07–08 14:47:00	0	0	Unknown	projectB.business–zone
H2/JDBI/PreparedStatement/execute	2020–07–08 14:47:00	0	0	h2–jdbc–driver	projectB.business–zone
selectUser	2020–07–08 14:47:00	27	1	Unknown	projectB.business–zone
H2/JDBI/PreparedStatement/execute	2020–07–08 14:47:00	26	26	h2–jdbc–driver	projectB.business–zone
/projectC/{name}	2020–07–08 14:47:00	5027	0	Feign	projectA.business–zone
/projectC/{value}	2020–07–08 14:47:05	24	2	SpringMVC	projectC.business–zone
/	2020–07–08 14:47:05	21	21	HttpClient	projectC.business–zone
Kafka/test–trace–topic/Producer	2020–07–08 14:47:05	1	1	kafka–producer	projectC.business–zone
/projectC/{value}	2020–07–08 14:47:00	62715	1002	SpringMVC	projectC.business–zone
/	2020–07–08 14:47:01	1713	1713	HttpClient	projectC.business–zone
Kafka/test–trace–topic/Producer	2020–07–08 14:47:03	60000	60000	kafka–producer	projectC.business–zone

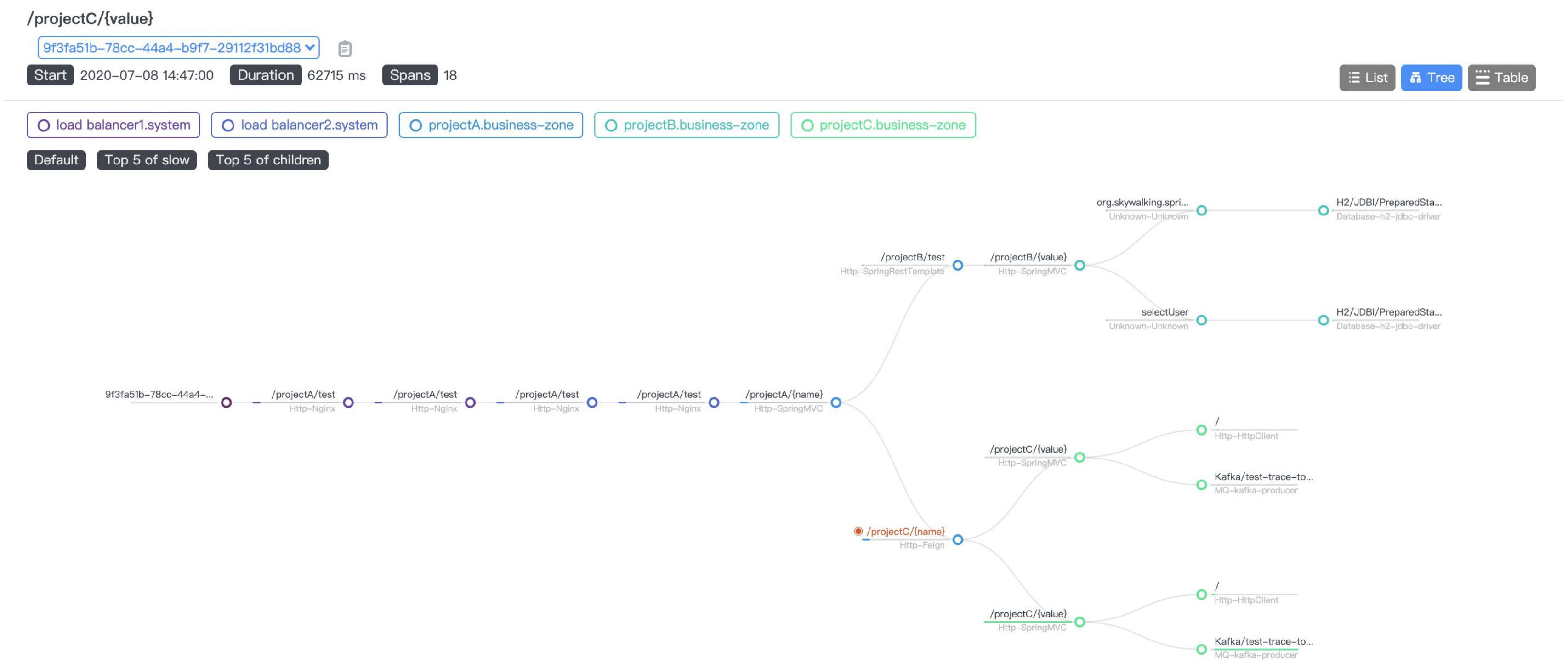
明细级别探查 – Slow Span



明细级别探查 – Access Overload



明细级别探查 – Dependency Depth


/projectC/{value}

9f3fa51b-78cc-44a4-b9f7-29112f31bd88

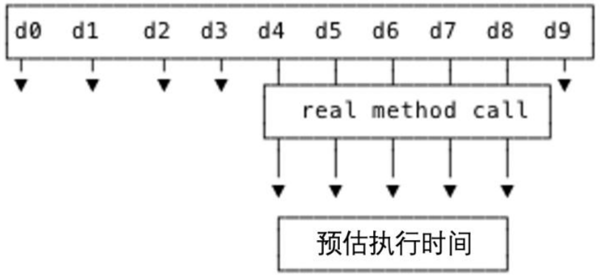
Start 2020-07-08 14:47:00 Duration 62715 ms Spans 18

List
Tree
Table
load balancer1.system
load balancer2.system
projectA.business-zone
projectB.business-zone
projectC.business-zone
Default
Top 5 of slow
Top 5 of children
9f3fa51b-78cc-44a4-...
/projectA/test Http-Nginx
/projectA/test Http-Nginx
/projectA/test Http-Nginx
/projectA/test Http-Nginx
/projectA/{name} Http-SpringMVC
/projectB/test Http-SpringRestTemplate
/projectB/{value} Http-SpringMVC
org.skywalking.spri... Unknown-Unknown
H2/JDBI/PreparedSta... Database-h2-jdbc-driver
selectUser Unknown-Unknown
H2/JDBI/PreparedSta... Database-h2-jdbc-driver
/projectC/{name} Http-Feign
/projectC/{value} Http-SpringMVC
/ Http-HttpClient
Kafka/test-trace-to... MQ-kafka-producer
/projectC/{value} Http-SpringMVC
/ Http-HttpClient
Kafka/test-trace-to... MQ-kafka-producer

如果上述方法依赖失效?

Trace的局限性

基于线程快照的执行时间估算



- org.test.code.classA.methodA(ClassA.java, line:24) T2-T1 ms
- org.test.code.classA.methodB(ClassA.java, line:24) T2-T1 ms
- org.test.code.classC.methodB(ClassA.java, line:33) 0 ms

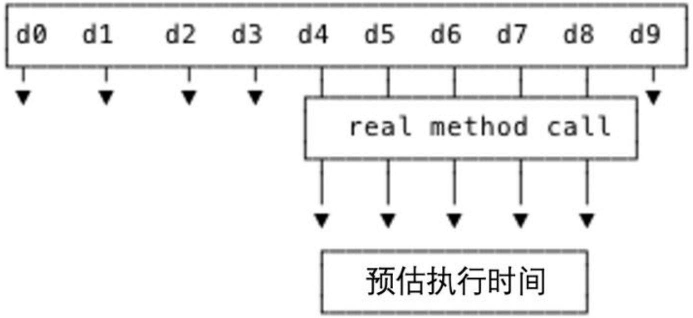
T1 Snapshot

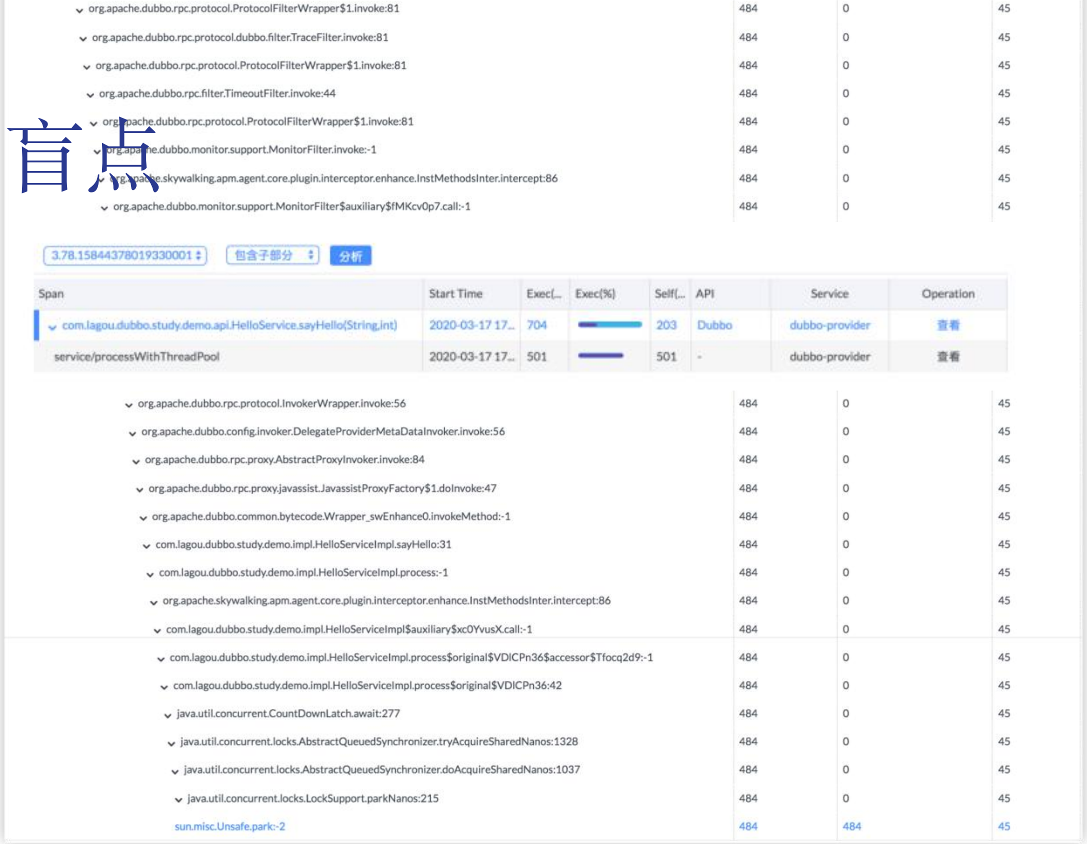
- org.test.code.classA.methodA(ClassA.java, line:24)
    - org.test.code.classA.methodB(ClassA.java, line:24)
        - org.test.code.classC.methodB(ClassA.java, line:33)

T2 Snapshot

- org.test.code.classA.methodA(ClassA.java, line:24)
    - org.test.code.classA.methodB(ClassA.java, line:35)
        - org.test.code.classD.methodB(ClassA.java, line:42)

性能剖析 – Trace的盲点





org.apache.dubbo.rpc.protocol.ProtocolFilterWrapper$1.invoke:81	484	0	45
org.apache.dubbo.rpc.protocol.dubbo.filter.TraceFilter.invoke:81	484	0	45
org.apache.dubbo.rpc.protocol.ProtocolFilterWrapper$1.invoke:81	484	0	45
org.apache.dubbo.rpc.filter.TimeoutFilter.invoke:44	484	0	45
org.apache.dubbo.rpc.protocol.ProtocolFilterWrapper$1.invoke:81	484	0	45
org.apache.dubbo.monitor.support.MonitorFilter.invoke:-1	484	0	45
org.apache.skywalking.apm.agent.core.plugin.interceptor.enhance.InstMethodsInter.intercept:86	484	0	45
org.apache.dubbo.monitor.support.MonitorFilter$auxiliary$fMKcv0p7.call:-1	484	0	45

3.78.15844378019330001 包含子部分 分析

Span	Start Time	Exec(...	Exec(%)	Self(...	API	Service	Operation
com.lagou.dubbo.study.demo.api.HelloService.sayHello(String,int)	2020-03-17 17...	704		203	Dubbo	dubbo-provider	查看
service/processWithThreadPool	2020-03-17 17...	501		501	-	dubbo-provider	查看

org.apache.dubbo.rpc.protocol.InvokerWrapper.invoke:56	484	0	45
org.apache.dubbo.config.invoker.DelegateProviderMetaDataInvoker.invoke:56	484	0	45
org.apache.dubbo.rpc.proxy.AbstractProxyInvoker.invoke:84	484	0	45
org.apache.dubbo.rpc.proxy.javassist.JavassistProxyFactory$1.doInvoke:47	484	0	45
org.apache.dubbo.common.bytecode.Wrapper_swEnhance0.invokeMethod:-1	484	0	45
com.lagou.dubbo.study.demo.impl.HelloServiceImpl.sayHello:31	484	0	45
com.lagou.dubbo.study.demo.impl.HelloServiceImpl.process:-1	484	0	45
org.apache.skywalking.apm.agent.core.plugin.interceptor.enhance.InstMethodsInter.intercept:86	484	0	45
com.lagou.dubbo.study.demo.impl.HelloServiceImpl$auxiliary$xc0YvusX.call:-1	484	0	45
com.lagou.dubbo.study.demo.impl.HelloServiceImpl.process$original$VDICPn36$accessor$Tfocq2d9:-1	484	0	45
com.lagou.dubbo.study.demo.impl.HelloServiceImpl.process$original$VDICPn36:42	484	0	45
java.util.concurrent.CountDownLatch.await:277	484	0	45
java.util.concurrent.locks.AbstractQueuedSynchronizer.tryAcquireSharedNanos:1328	484	0	45
java.util.concurrent.locks.AbstractQueuedSynchronizer.doAcquireSharedNanos:1037	484	0	45
java.util.concurrent.locks.LockSupport.parkNanos:215	484	0	45
sun.misc.Unsafe.park:-2	484	484	45

- InfoQ文章 https://www.infoq.cn/article/CWUOl1JA0EyXw0CxQ4Zm
- TheNewStack（英文）https://thenewstack.io/apache-skywalking-use-profiling-to-fix-the-blind-spot-of-distributed-tracing/

Open Source Community

- 基于Trace的诊断分析, http://skywalking.apache.org/blog/2019-01-01-Understand-Trace.html
- 多语言
  - Java, https://github.com/apache/skywalking
  - Nginx Lua, https://github.com/apache/skywalking-nginx-lua
  - Python, https://github.com/apache/skywalking-python
  - PHP, https://github.com/SkyAPM/SkyAPM-php-sdk
  - .Net, https://github.com/SkyAPM/SkyAPM-dotnet
  - NodeJs, https://github.com/SkyAPM/SkyAPM-nodejs | https://github.com/apache/skywalking-nodejs
  - Golang, https://github.com/SkyAPM/go2sky
- 可以选择任何你喜欢的语言

Apache Local Community

Cloud Native Community