



Agenda

- Distributed Tracing
- Zipkin
- Spring Cloud Sleuth Stream
- Roadmap
- Demo

Pivotal™

© Copyright 2015 Pivotal. All rights reserved.

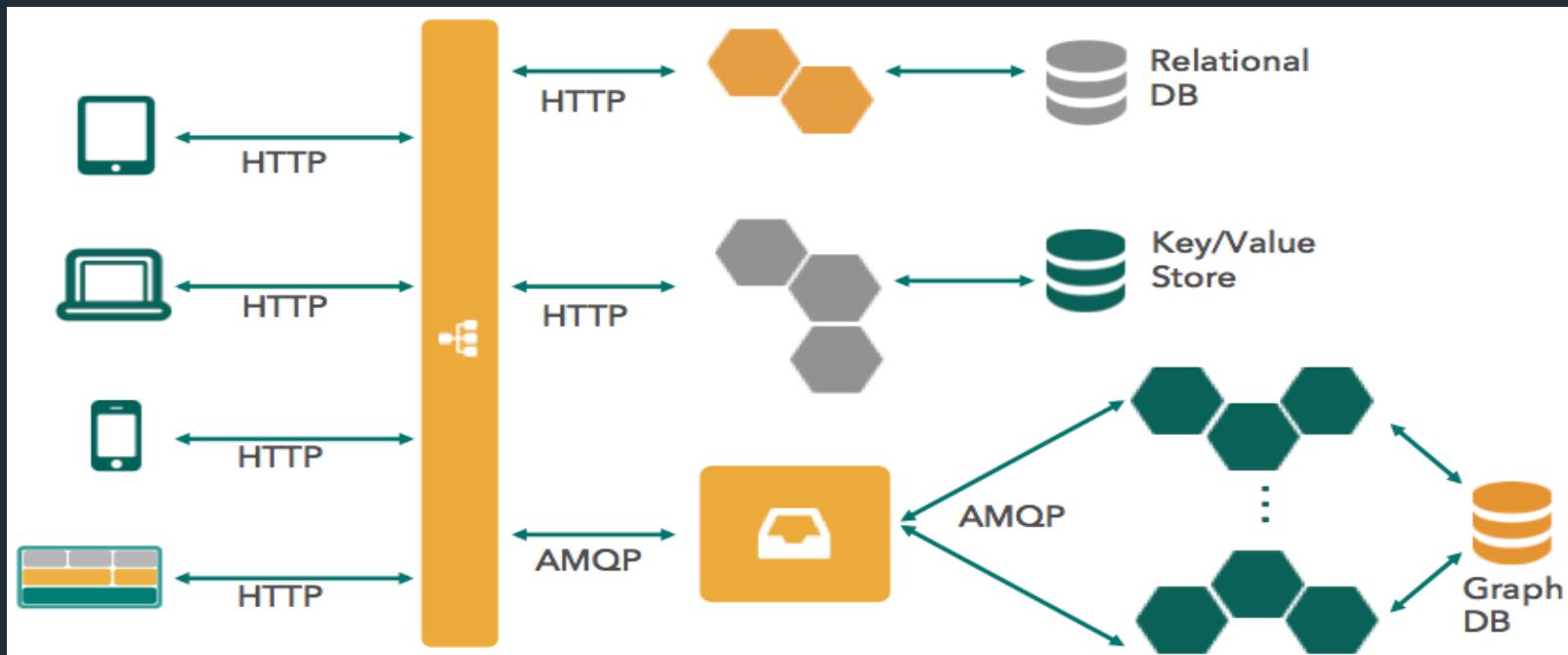
Challenges Of Microservices

- Configuration Management
- Service Registration & Discovery
- Routing & Load Balancing
- Fault Tolerance
- Monitoring
- Latency Analysis and tracing

Pivotal™

Distributed Tracing

Microservice architectures are often a graph of components distributed across a network



Pivotal™

Dapper Paper By Google

This paper described Dapper, which is Google's **production** distributed systems tracing infrastructure

Design Goals :

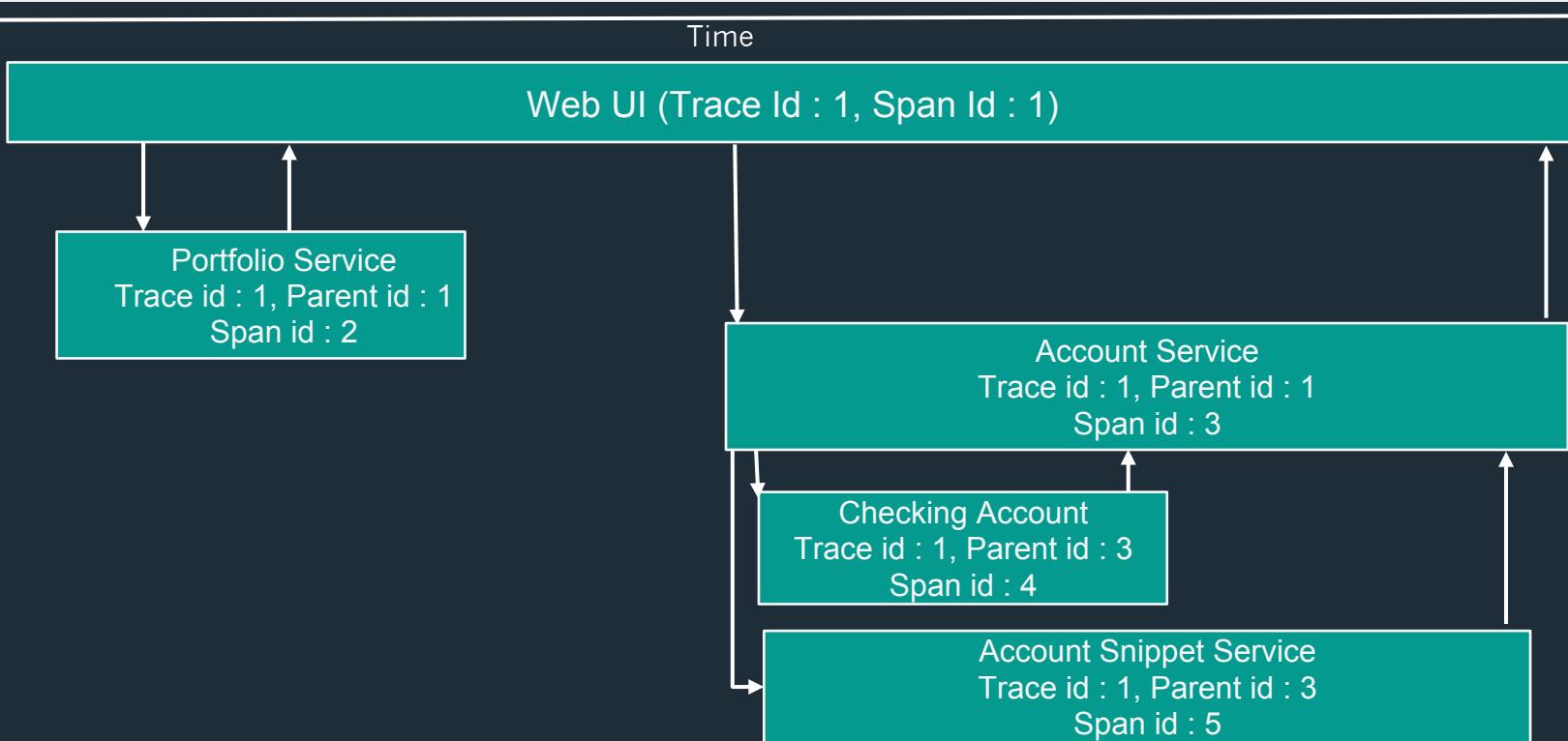
- Low overhead
- Application-level transparency
- Scalability

Based on :

Concept of Traces and Spans

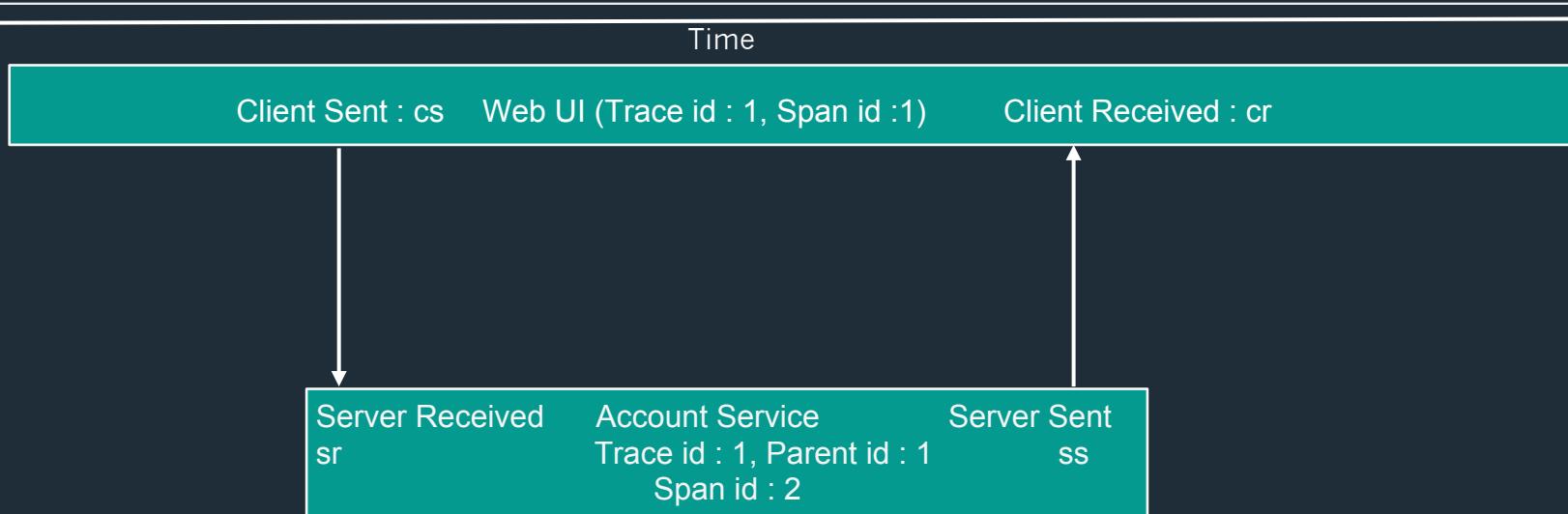
Pivotal™

Traces And Spans



Pivotal™

Traces And Spans Annotations



Pivotal™

Example Span – An individual operation

Operation

Events

Tags

POST /things

wombats:10.2.3.47:8080

Server Received

Server Sent

peer.ipv4 1.2.3.4

http.request-id abcd-fde

http.request.size 15 MiB

http.url ...&features=HD-uploads

Pivotal™

Twitter

 Clip slide

Front end



Back end



Pivotal™

Screenshot of a web browser showing a news article from The Atlantic and a Network Performance Monitor tool.

The main content area displays a news article titled "Americans Are as Likely to Be Killed by Their Own Furniture as by Terrorism". The article discusses terrorist attacks killing 17 U.S. civilians last year and 15 the year before. It includes a small image of a destroyed building and a link to theatlantic.com/international/... .

The sidebar on the left lists navigation links: Home, Connect, Discover, Stories, Activity, Who to follow, Find friends, Browse categories, and Trends - Change.

The bottom section shows a Network Performance Monitor with the following data:

Name	Path	Method	Status	Type	Initiator	Size Content	Time Latency	Timeline	1.30s	1.94s	2.59s	3.24s	3.89s	4.53s	5.18s
	discover /	GET	200 OK	text/html	Other	08 204.52KB	967ms 955ms	Timeline							
	t1_core.bundle.css twimg0-a.akamaihd.net/a/134184843:	GET	304 Not Modified	text/css	discover:11 Parser	3198 121.84KB	12ms 6ms								
	t1_more.bundle.css twimg0-a.akamaihd.net/a/134184843:	GET	304 Not Modified	text/css	discover:13 Parser	3198 138.74KB	13ms 6ms								
	01457d1a0f0e533062cd0d1033fb4d twimg0-a.akamaihd.net/profile_images	GET	200 OK	image/png	discover:401 Parser	(from cache)	109ms 90ms								
	IMG_0026_2_normal.jpg twimg0-a.akamaihd.net/profile_images	GET	200 OK	image/jpeg	discover:404 Parser	(from cache)	109ms 91ms								
	4zbhy2jm5xiflnqvy8bw_normal.jpeg twimg0-a.akamaihd.net/profile_images	GET	200 OK	image/jpeg	discover:410 Parser	(from cache)	109ms 91ms								
	lisa_avatar_normal twimg0-a.akamaihd.net/profile_images	GET	200 OK	image/jpeg	discover:416 Parser	(from cache)	108ms 92ms								
	5853643591_f87872e069_b_normal.j twimg0-a.akamaihd.net/profile_images	GET	200 OK	image/jpeg	discover:419 Parser	(from cache)	109ms 92ms								6
	przeas_1a052f2a2-177f-51c2-135693:		304		discover:176	5078	281ms								

A red circle highlights the timeline bar for the first request (discover /) between 1.30s and 1.94s, indicating a significant delay in this specific network segment.

Pivotal™

Zipkin

- Zipkin is a tracing system and was created by Twitter in 2012
- Implementation is based on Dapper Paper by Google
- Adrian Cole (Spring Cloud) is the main contributor
- In 2015, OpenZipkin became the primary fork

Goals :

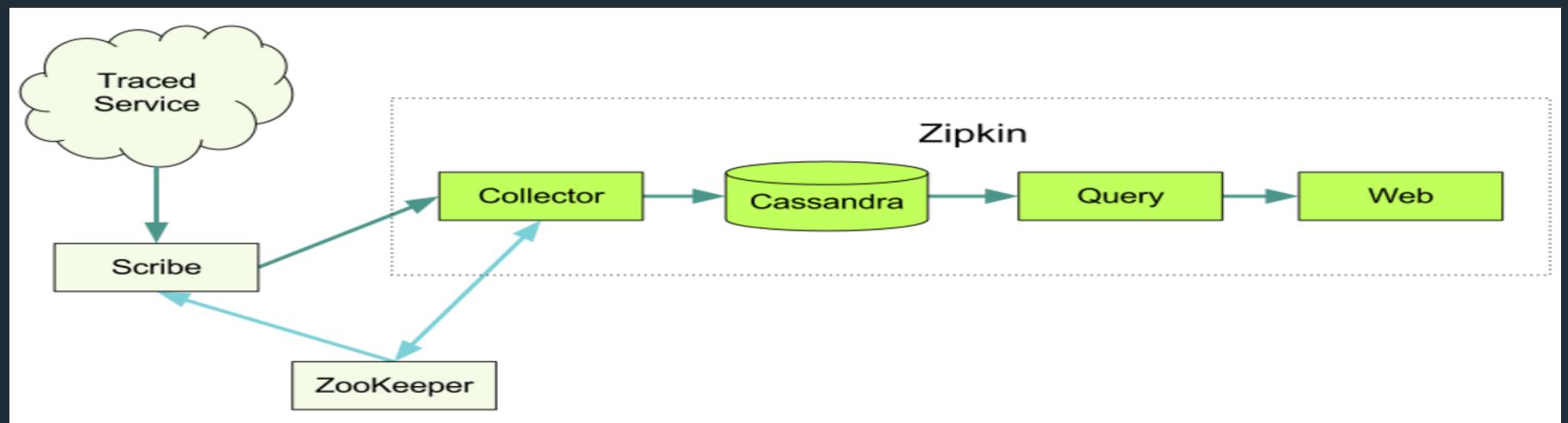
Aggregate spans into trace trees

Provide query and visualization for latency analysis

Have retention policy

Pivotal™

Zipkin Architecture



- Tracers collect timing data and transport it over HTTP, RabbitMq, Kafka
- Collectors store spans in MySQL or Cassandra
- Users query for traces via Zipkin's Web UI or API

Pivotal™

Tracers

- **Tracers** add logic to create unique trace ID
- **Trace ID** is generated when the first request is made
- **Span Id** is generated as the request arrives at each microservice
- Example tracer is Spring Cloud Sleuth
- Tracers execute in your production apps! They are written to not log too much
- Tracers have instrumentation or sampling policy to manage volumes of traces and spans

Pivotal™

Spring Cloud Sleuth is a Zipkin-Compatible tracer

- Spring Cloud Sleuth sets up useful log formatting for you that prints the trace ID and the span ID
- It includes instrumentation for Spring Boot, and a streaming collector
- Reports to zipkin via HTTP/stream by depending on spring-cloud-sleuth-zipkin / stream
- Transparent to application developer
 1. Include starter BOM on app build system

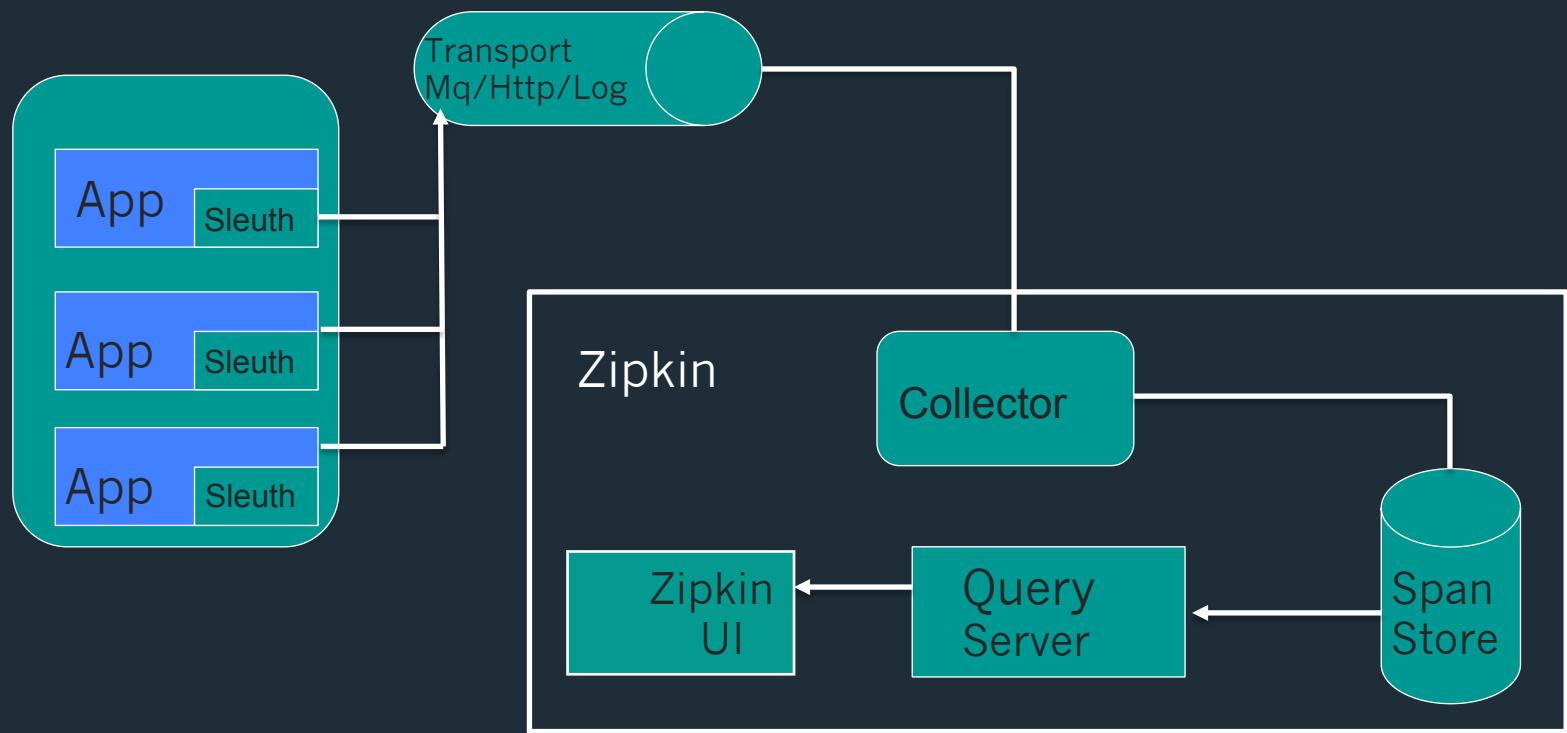
```
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-sleuth-stream</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-stream-rabbit</artifactId>
</dependency>
<dependency>
```

2. Specify “sampler”

```
@Bean
public Sampler sampler(){
    return new AlwaysSampler();
}
```

Pivotal™

Spring Cloud Sleuth Stream & Zipkin Architecture



Pivotal™

Road Map (Distributed Tracing and PCF)

- Zipkin as a PCF Spring Cloud Service expected on Q4
- Apps running on PCF can integrate Spring Cloud Sleuth and bind to Zipkin
- Currently supporting only rabbit MQ, future plans to support kafka
- Future versions might merge with Metrix

Pivotal™

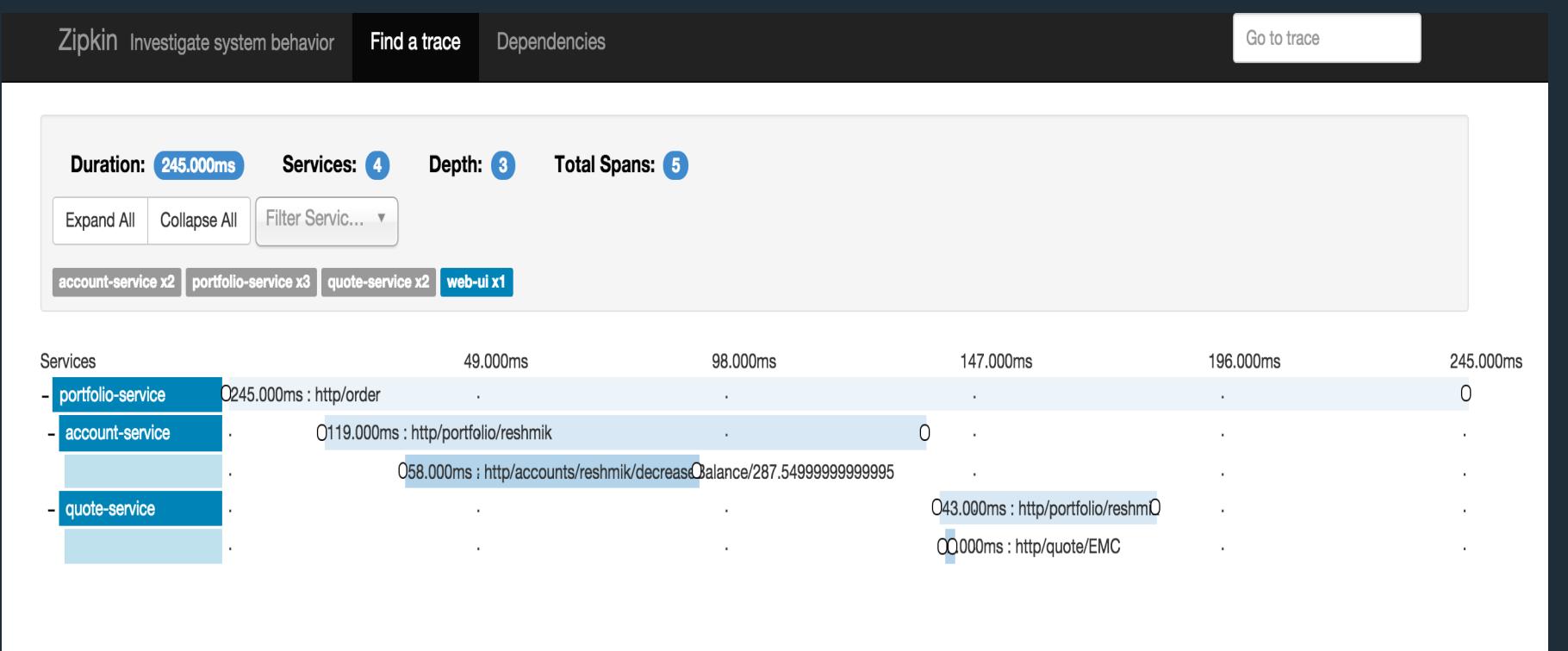
Summary

- Tracing requests in a microservice ecosystem is hard
- Distributed tracing to the rescue, by using traces and spans you can trace your requests end-end
- Zipkin is a distributed tracing system that collect, process and present data reported by tracers
- Spring Cloud Sleuth is a Zipkin Compatible tracer

Pivotal™

© Copyright 2015 Pivotal. All rights reserved.

Demo



Pivotal™

Duration: 932.000ms Services: 3 Depth: 3 Total Spans: 5

[Expand All](#)[Collapse All](#)[Filter...](#)[service1 x2](#) [service2 x2](#) [service3 x3](#)

Services		186.400ms	372.800ms	559.200ms	745.600ms	932.000ms
- service3	96.000ms : http:/compose					
- service3	794.000ms : http:/data/					
service1	9.000ms : http:/data/					
- service3	883.000ms : http:/data/					
service2	8.000ms : http:/data/					

Zipkin Investigate system behavior

Find a trace

Dependencies

Go to trace

Start time 03-31-2016

23:52

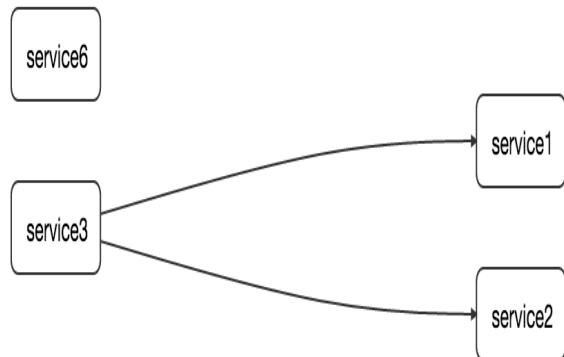
End time

04-07-2016

23:52

Analyze Dependencies

spring-cloud-sleuth-sample-stream



Pivotal™

© Copyright 2015 Pivotal. All rights reserved.

Resources

SME Content :

[Distributed Tracing SME](#)

#sme-pcf-zipkin

Github Content:

[spring-cloud-sleuth repo](#)

[twitter-zipkin](#)

[openzipkin](#)

[Dapper Paper By Google](#)

Josh Long's blog : <https://spring.io/blog/2016/02/15/distributed-tracing-with-spring-cloud-sleuth-and-spring-cloud-zipkin>

Pivotal™