

Мониторинг 2.0

Трейсинг в распределённых системах

Олег Федосеев

👉🐱🐦 olegfedoseev

oleg.fedoseev@me.com

Кто я?

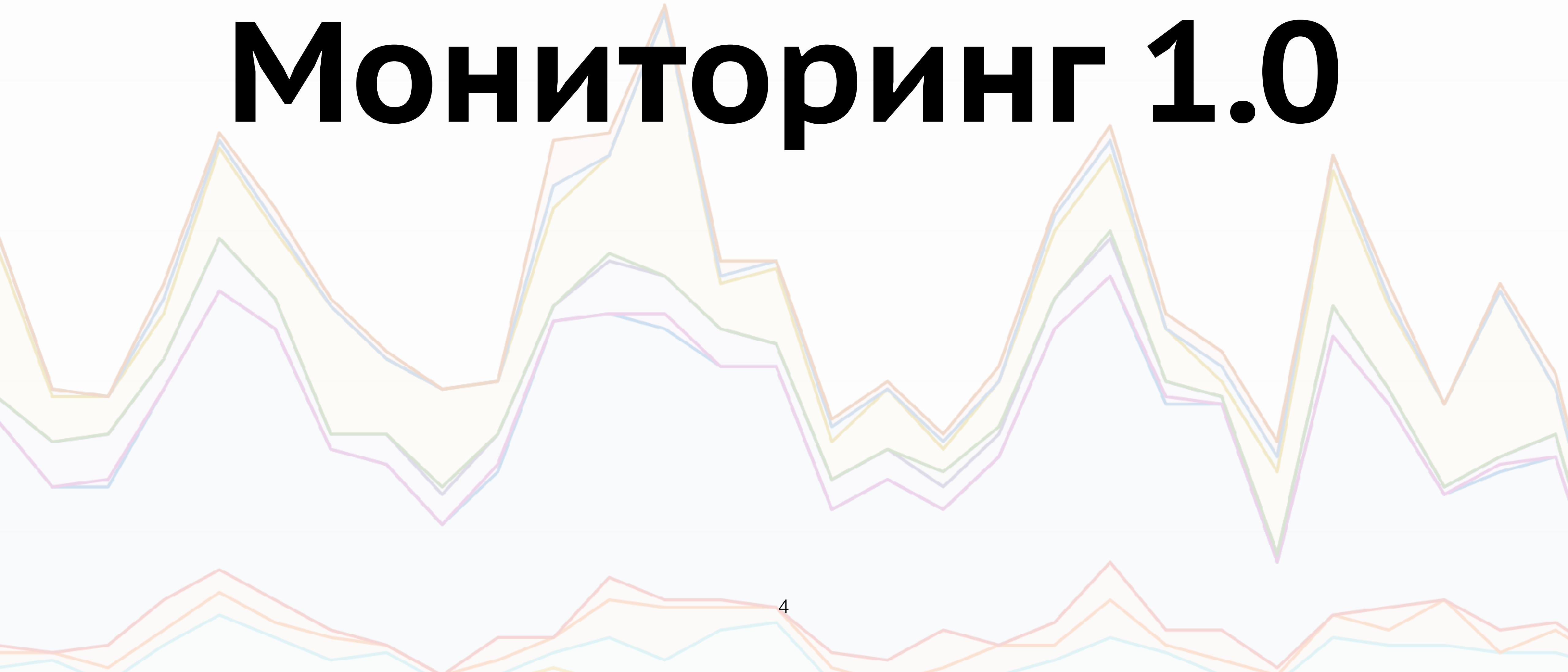
- с 2009 делал мониторинг в НГС
- с 2017 делаю мониторинг в N1.RU
- на PHP и на Go делаю



План

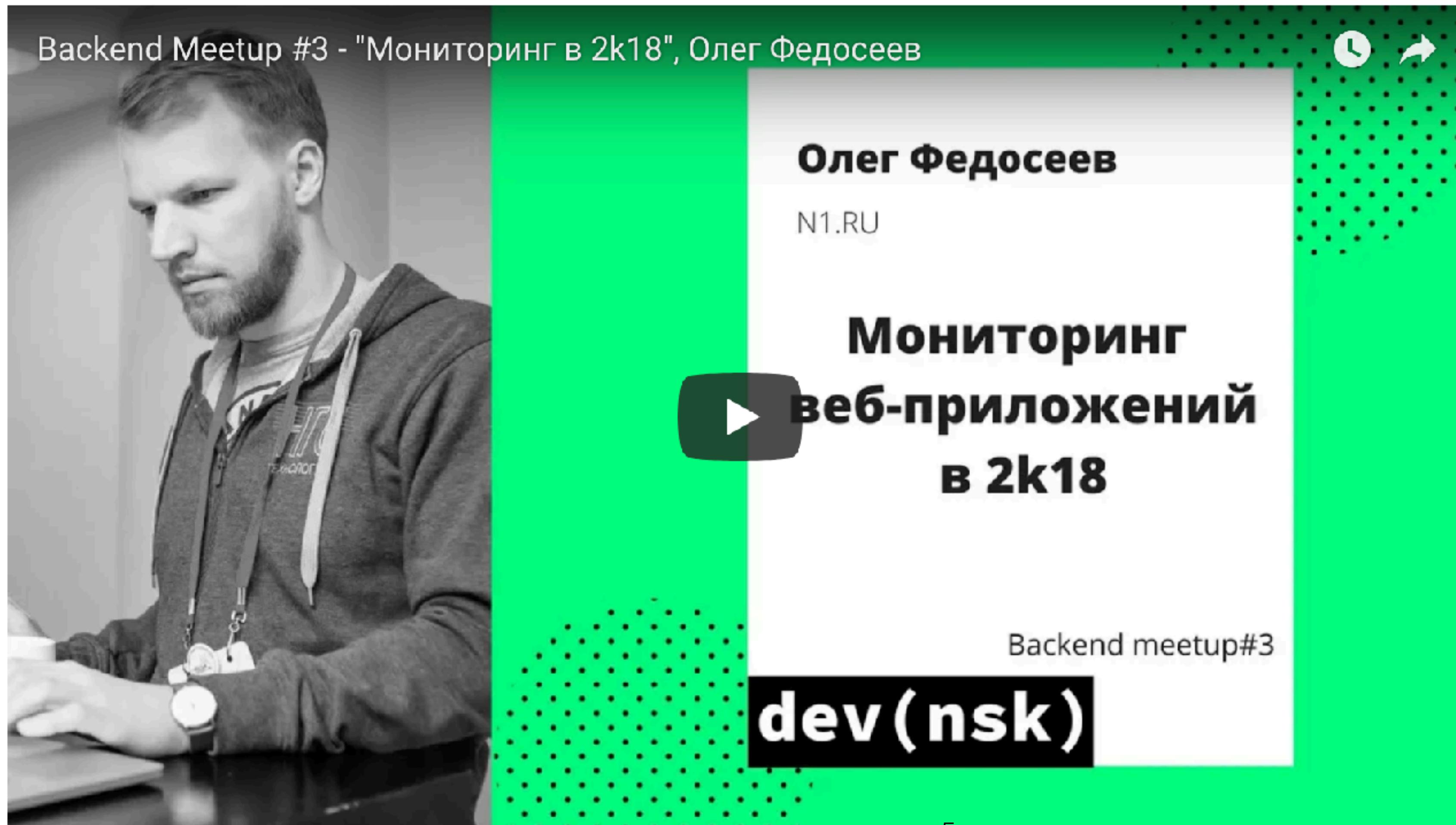
- Мониторинг 1.0 (краткое содержание прошлой серии)
- Зачем нам трейсинг?
- Инструменты и стандарты
- Пример внедрения

Мониторинг 1.0



Краткое содержание прошлой серии

https://youtu.be/pPaG09n0_E8



Краткое содержание прошлой серии

- Можно мониторить всё от железа до контейнеров
- Для PHP есть Pinba с таймерами
- Много инструментов для сбора метрик
- Акцент на мониторинг хоста/контейнера

Краткое содержание прошлой серии

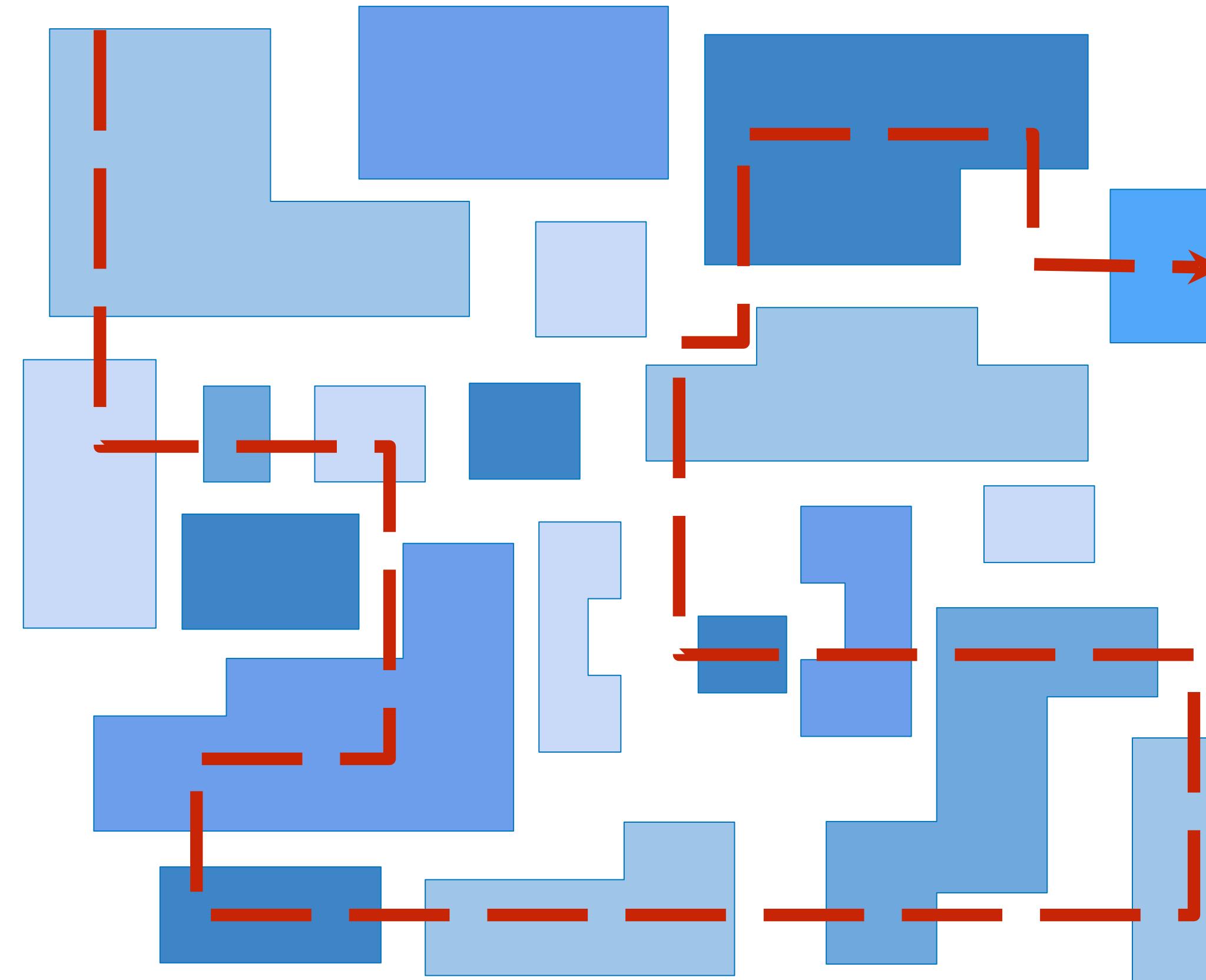
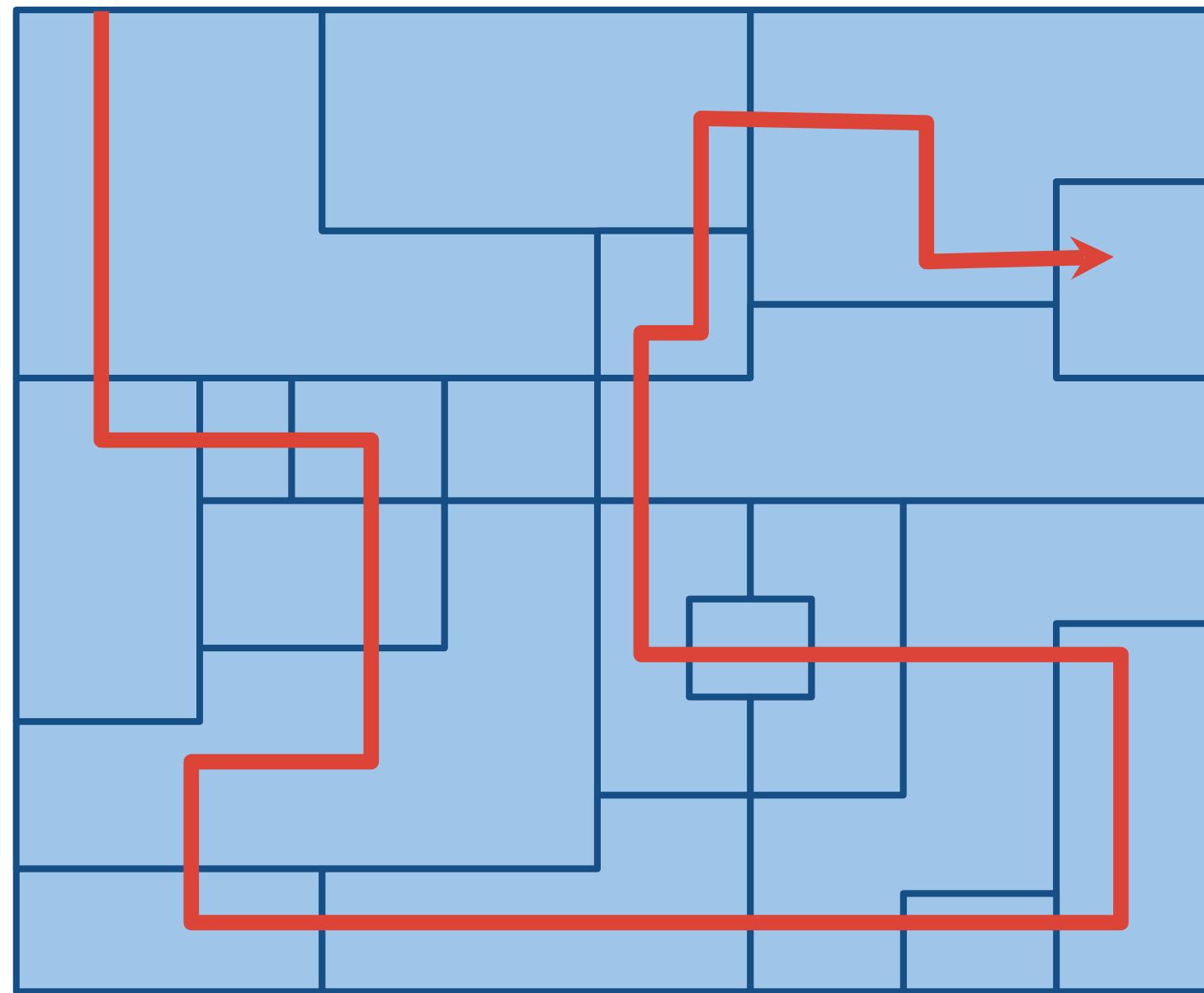
- Основная единица мониторинга это метрика
- Метрика это {время, имя, значение}
- Она атомарна и неделима
- Хорошо подходит для монолитов :-)

Зачем нужен трейсинг*?

Зачем нужен трейсинг?

- Каждый запрос затрагивает кучу систем
- Нужна полная картина происходящего
- С приходом микросервисов появляются распределённые транзакции и запросы

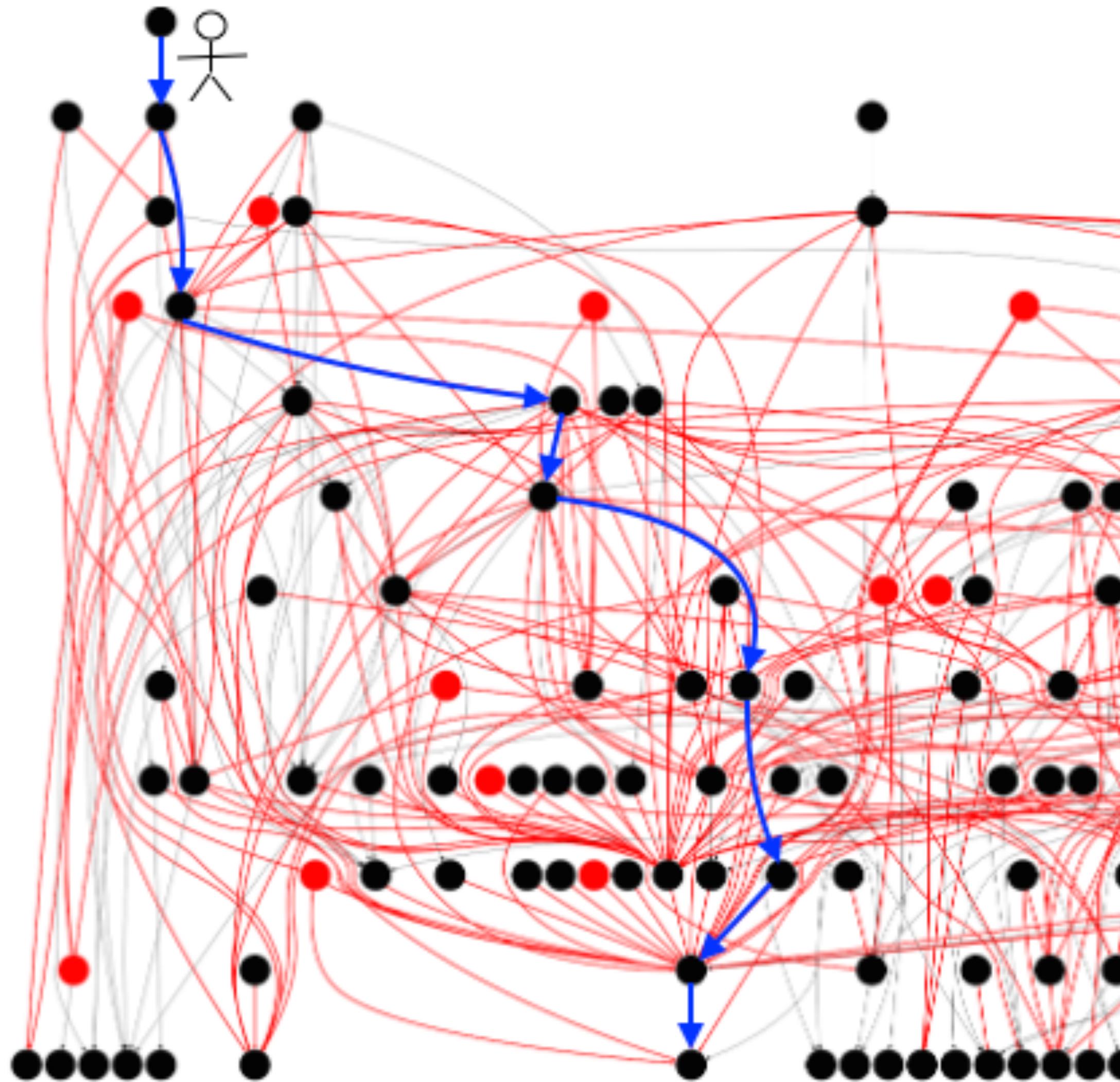
Монолит или микросервисы?



Зачем нужен трейсинг?

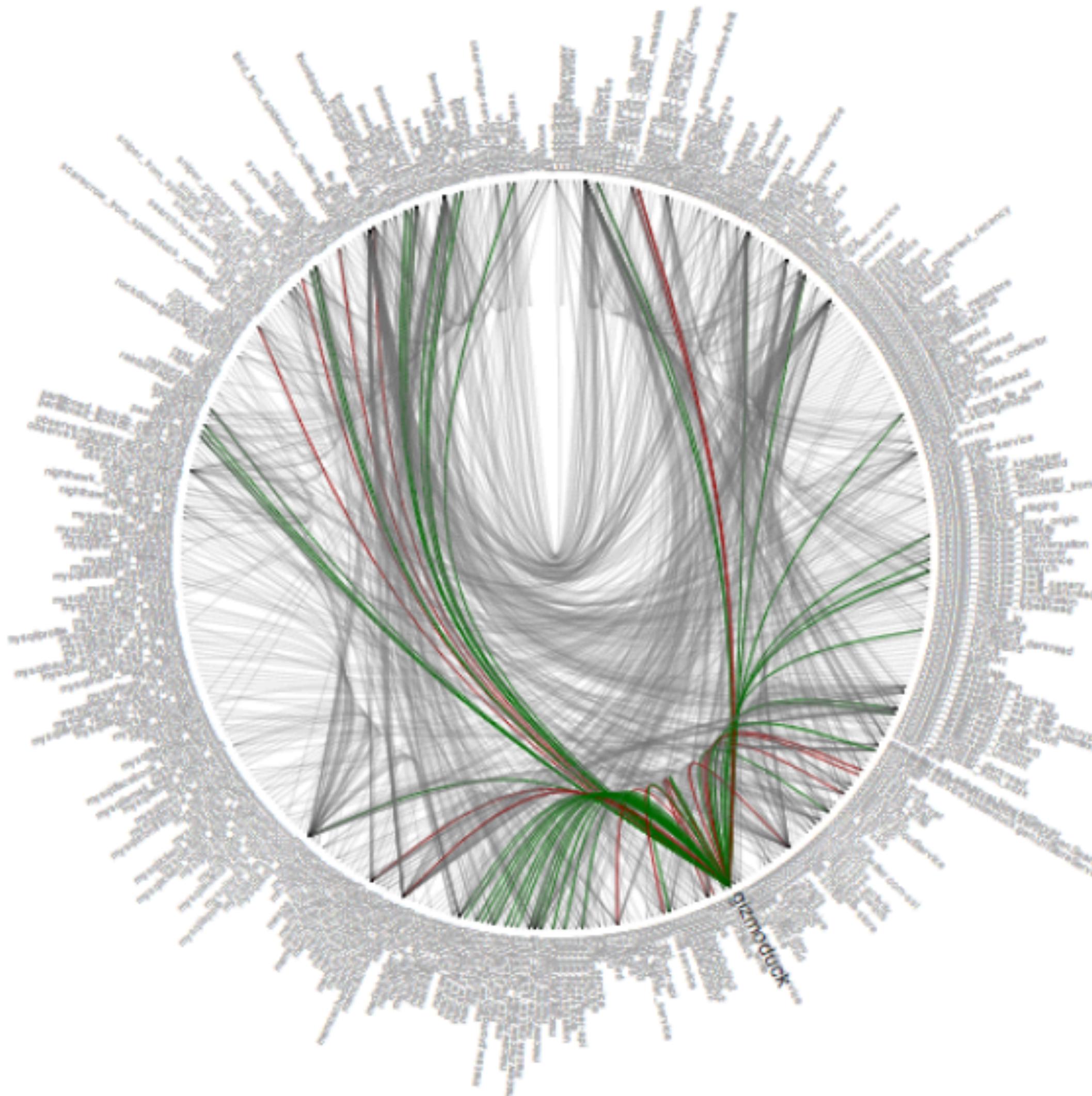


Зачем нужен трейсинг?



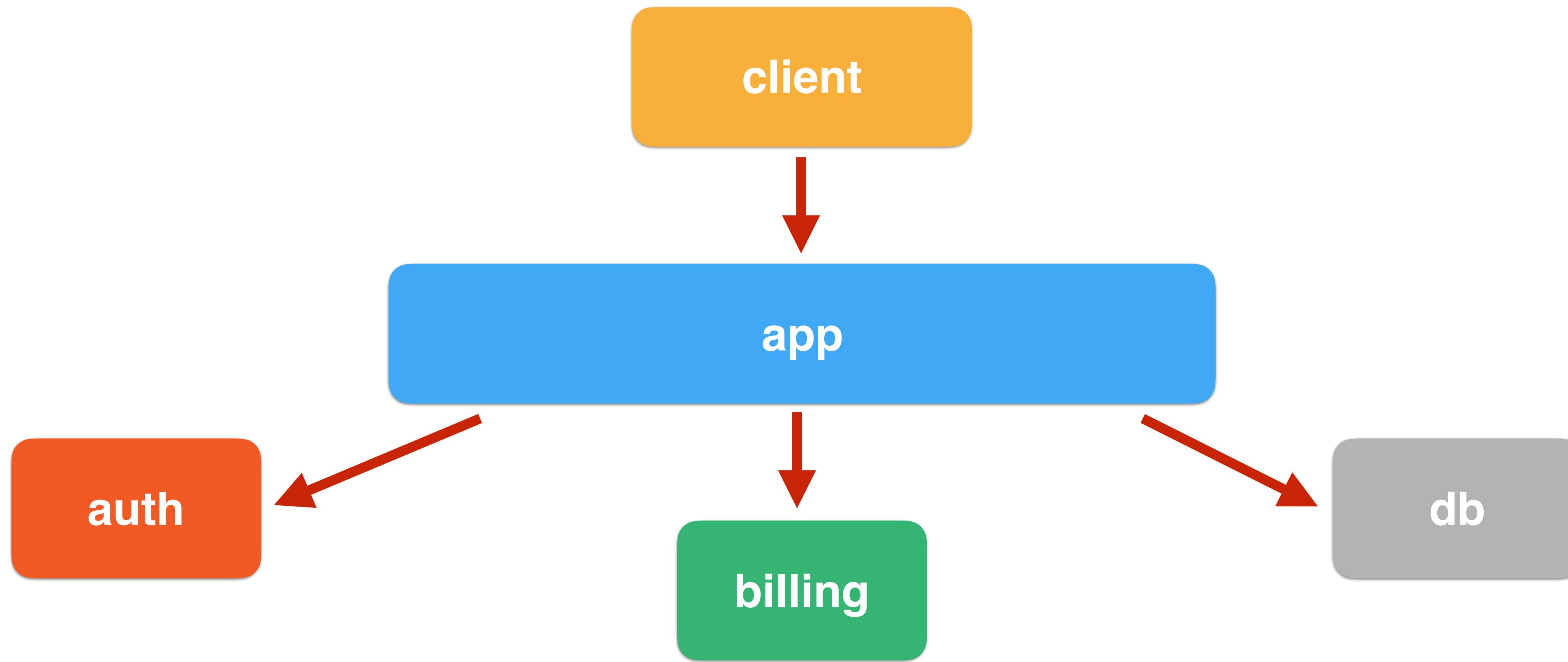
Трейсинг в
распределённой системе
позволяет отследить путь
выполнения запроса

Зачем нужен трейсинг?

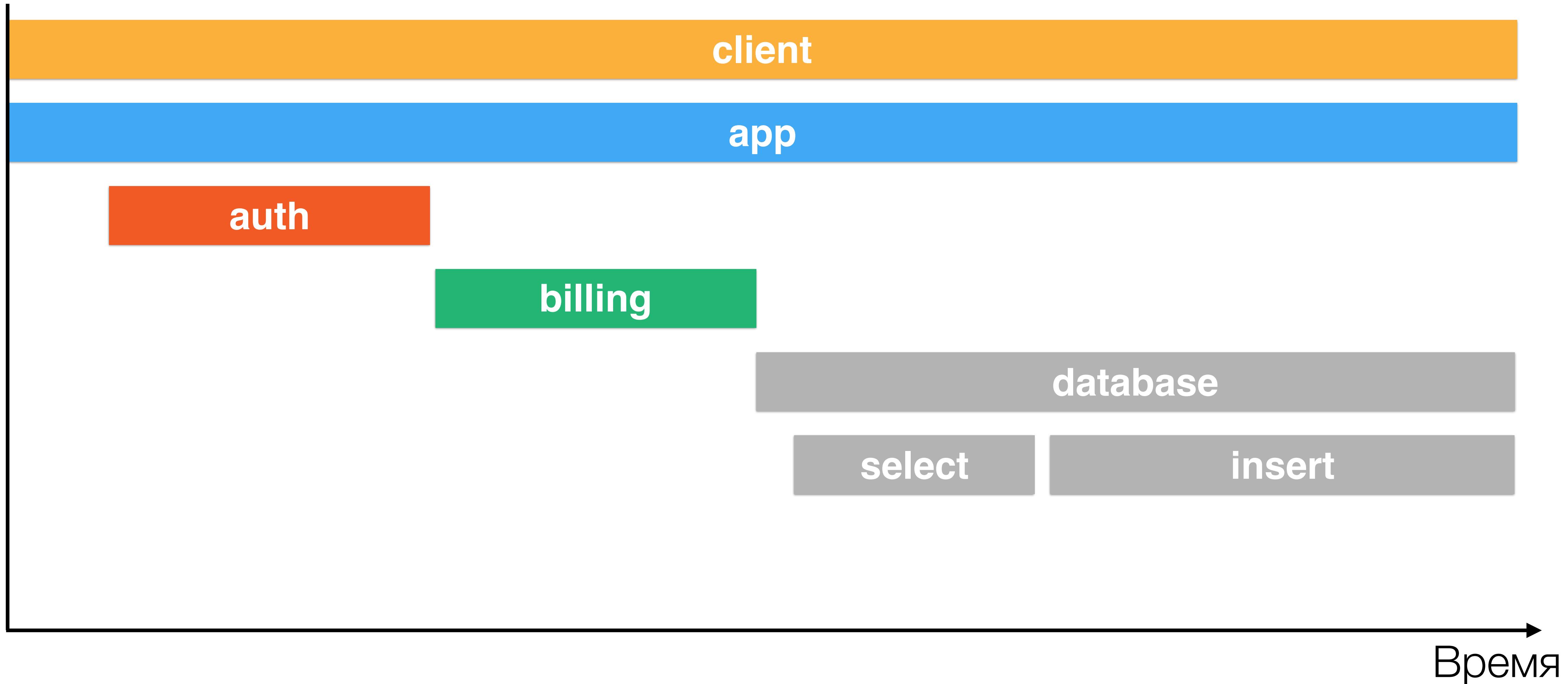


Distributed tracing в
условиях микросервисов
поможет быстро выявить
проблему

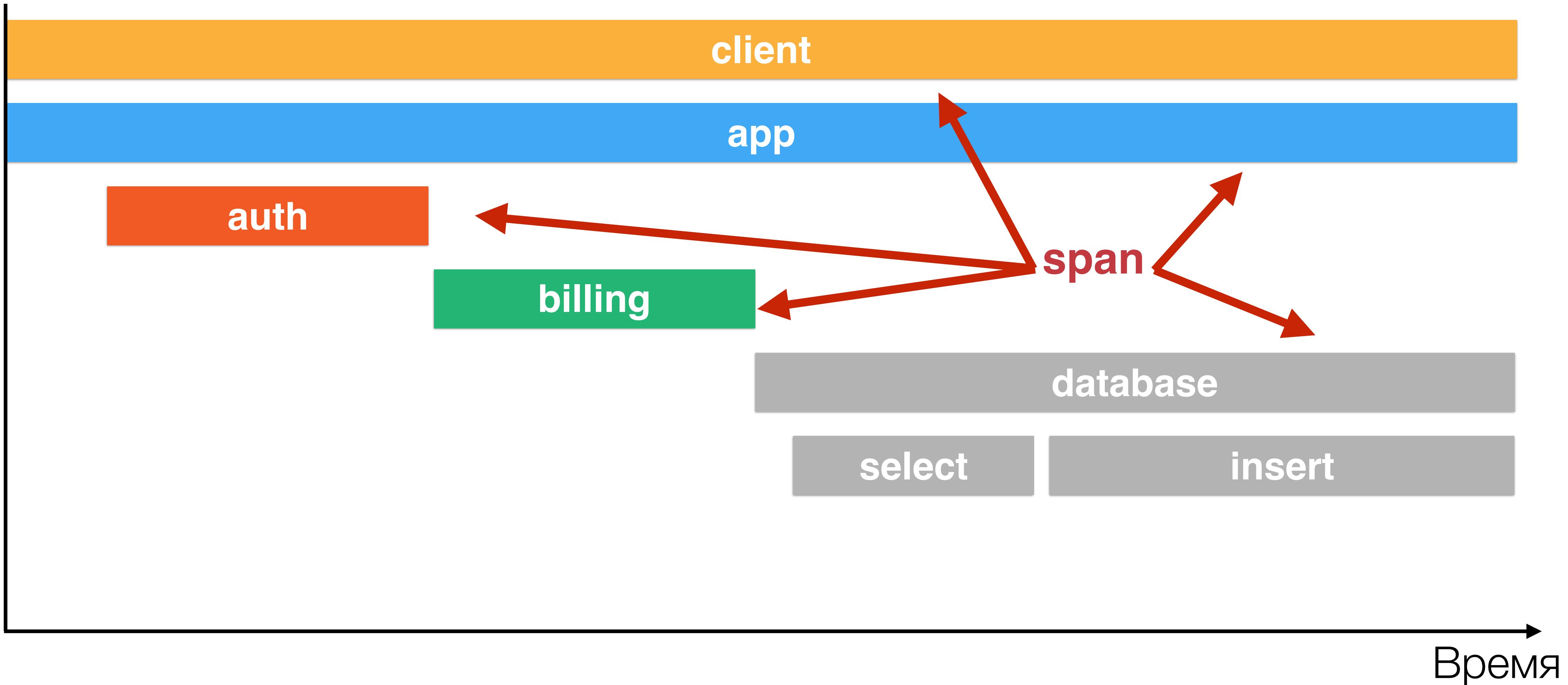
Что такое трейс?



Что такое трейс?



Что такое трейс?

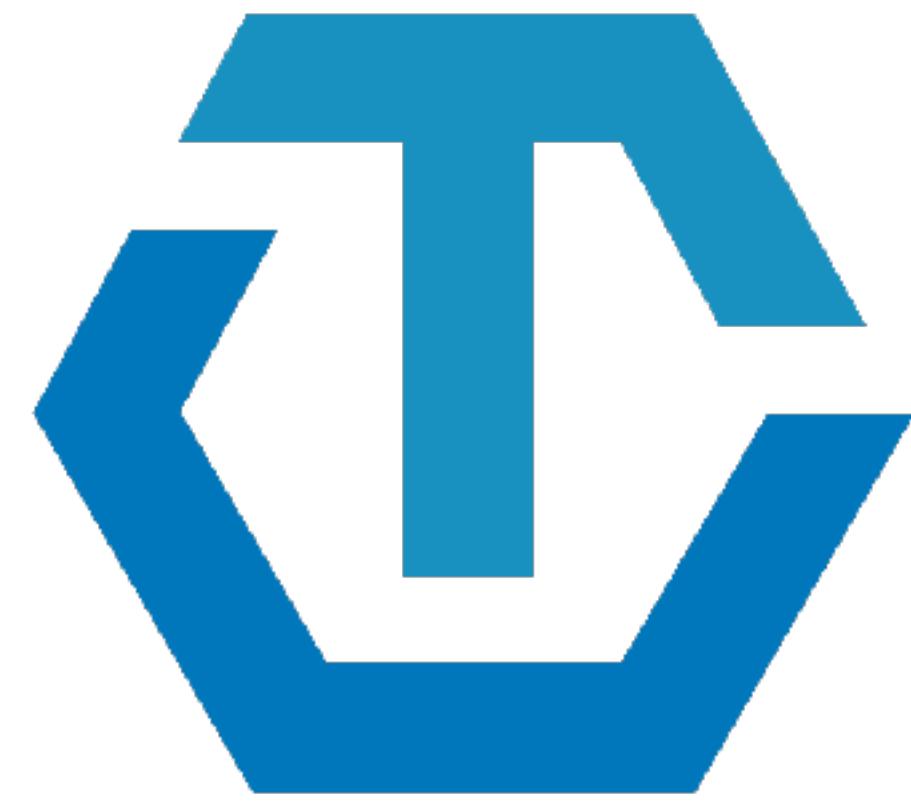


Понятия трейсинга

- **Trace** - информация о транзакции в распределённой системе
- **Span** - именованный кусок работы, с временем начала и продолжительностью. Может иметь теги.
- **Tags/Attributes** - пара ключ-значения
- **Context** - информация передающаяся от сервиса к сервису

Инструменты и стандарты

Стандарты



OPENTRACING

OpenTracing.io



OpenCensus

OpenCensus.io

OpenTracing

- Проект от Uber
- Спецификация для трейсинга
- Библиотеки для инструментации
- Поддержка Jaeger, Appdash, LightStep и др.
- Только трейсинг
- ★1301



 Joyent  workiva

OpenCensus

- Проект от Google и Microsoft
- Набор библиотек
- Стандартный протокол
- Умеет в трейсинг и метрики
- Поддержка Jaeger, Zipkin, Prometheus и др.
- ★1015



Инструменты для сбора и отображения



Zipkin.io



JAEGER

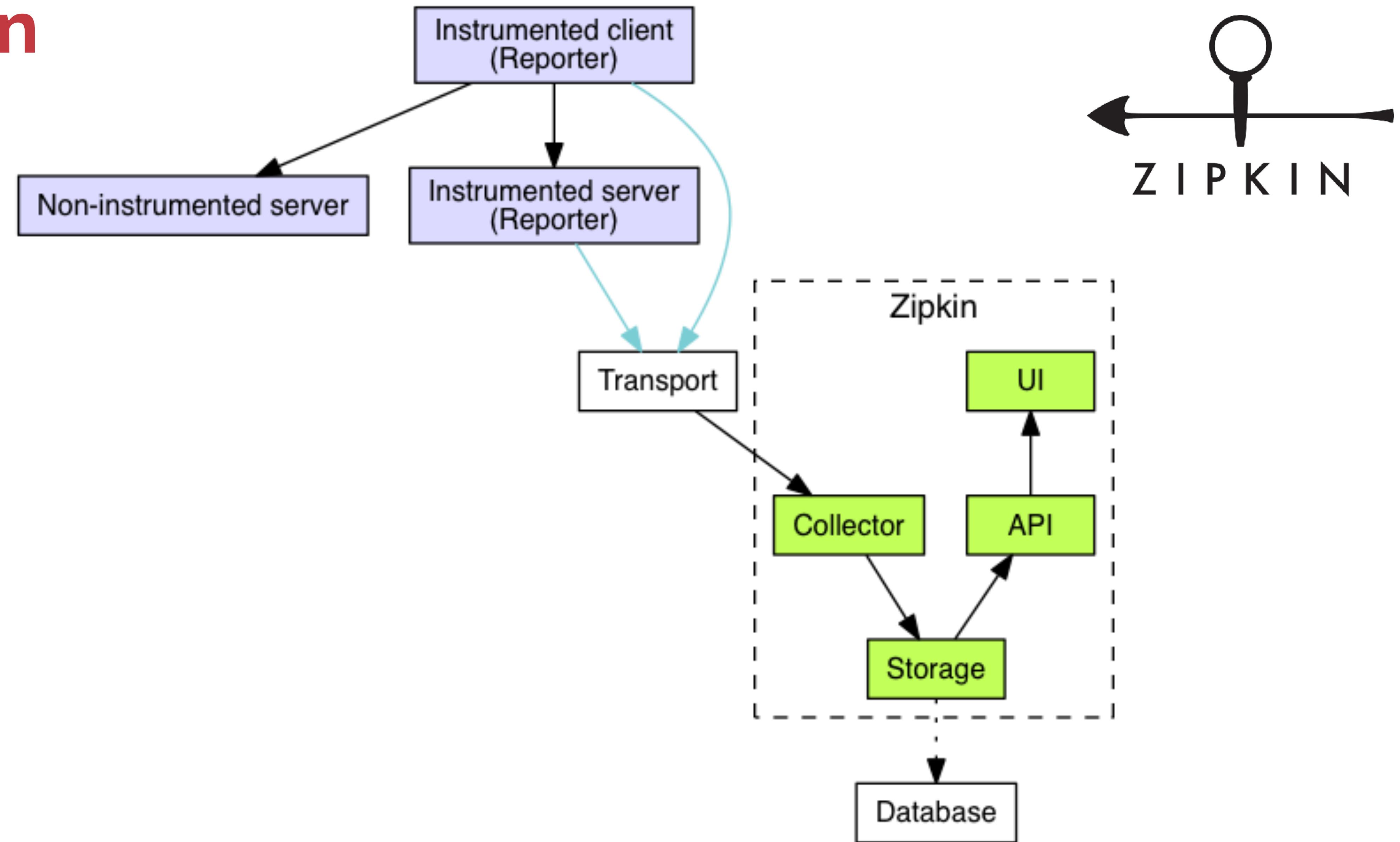
JaegerTracing.io

Zipkin

- Проект от Twitter'a
- Написан на Java
- На вход HTTP, Kafka или Scribe
- Хранение Cassandra, ElasticSearch, MySQL



Zipkin



Duration: 209.323ms

Services: 5

Depth: 7

Total Spans: 24

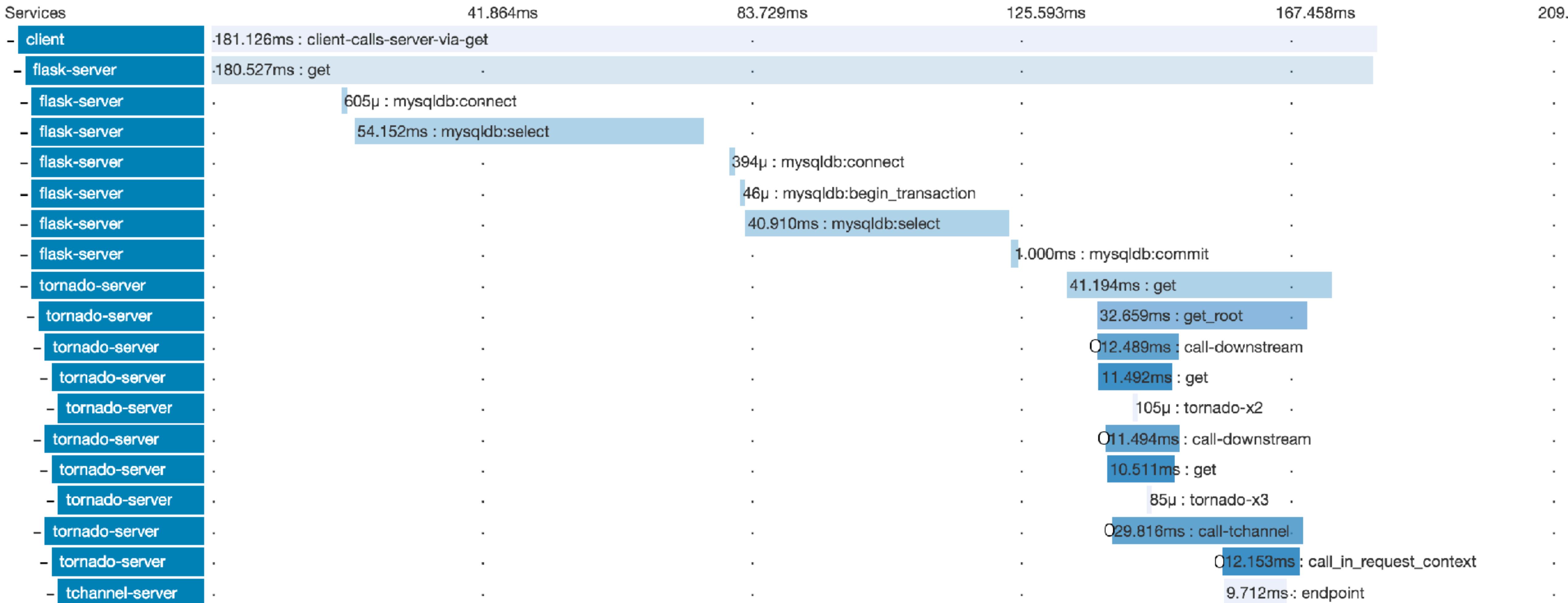
JSON

Expand All

Collapse All

Filter Service Se...

client x4 flask-server x10 missing-service-name x2 tchannel-server x2 tornado-server x11

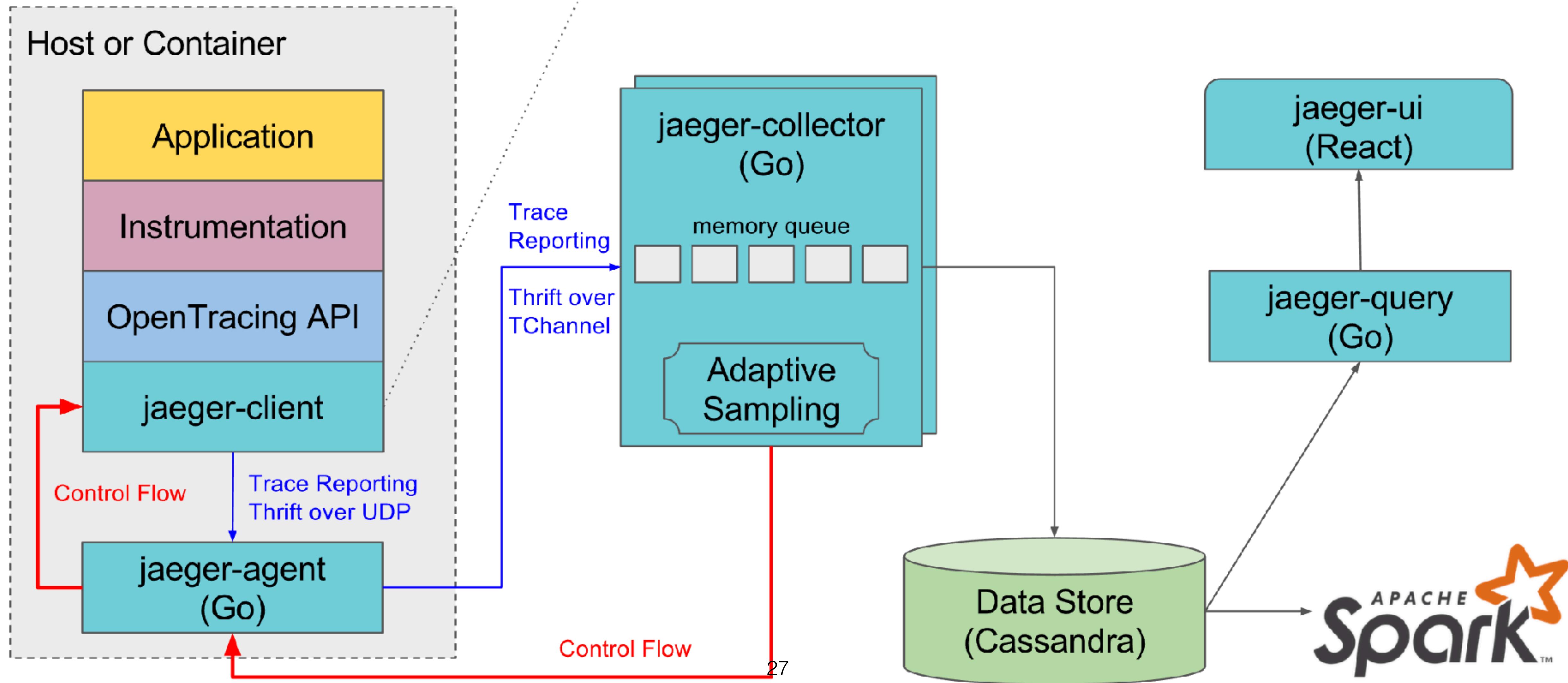


Jaeger

- Проект Uber
- Go + React
- На вход Zipkin
- Хранение Cassandra, ElasticSearch, memory



JAEGER



Find Traces

Service

frontend

Tags

http.status_code:400|http.status_code:200

Lookback

Last Hour

Min Duration

e.g. 1.2s, 100ms, 500us

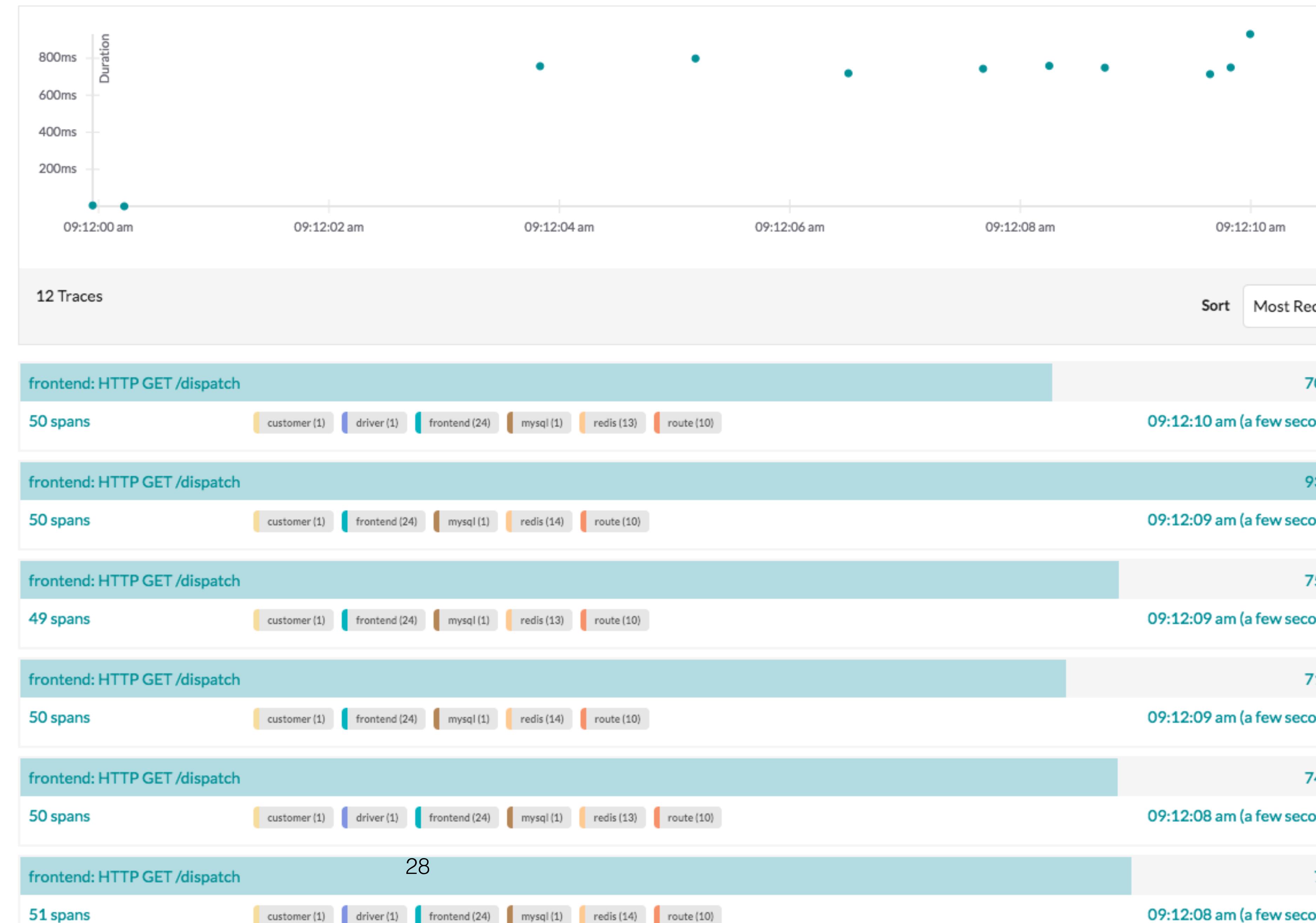
Max Duration

e.g. 1.1s

Limit Results

20

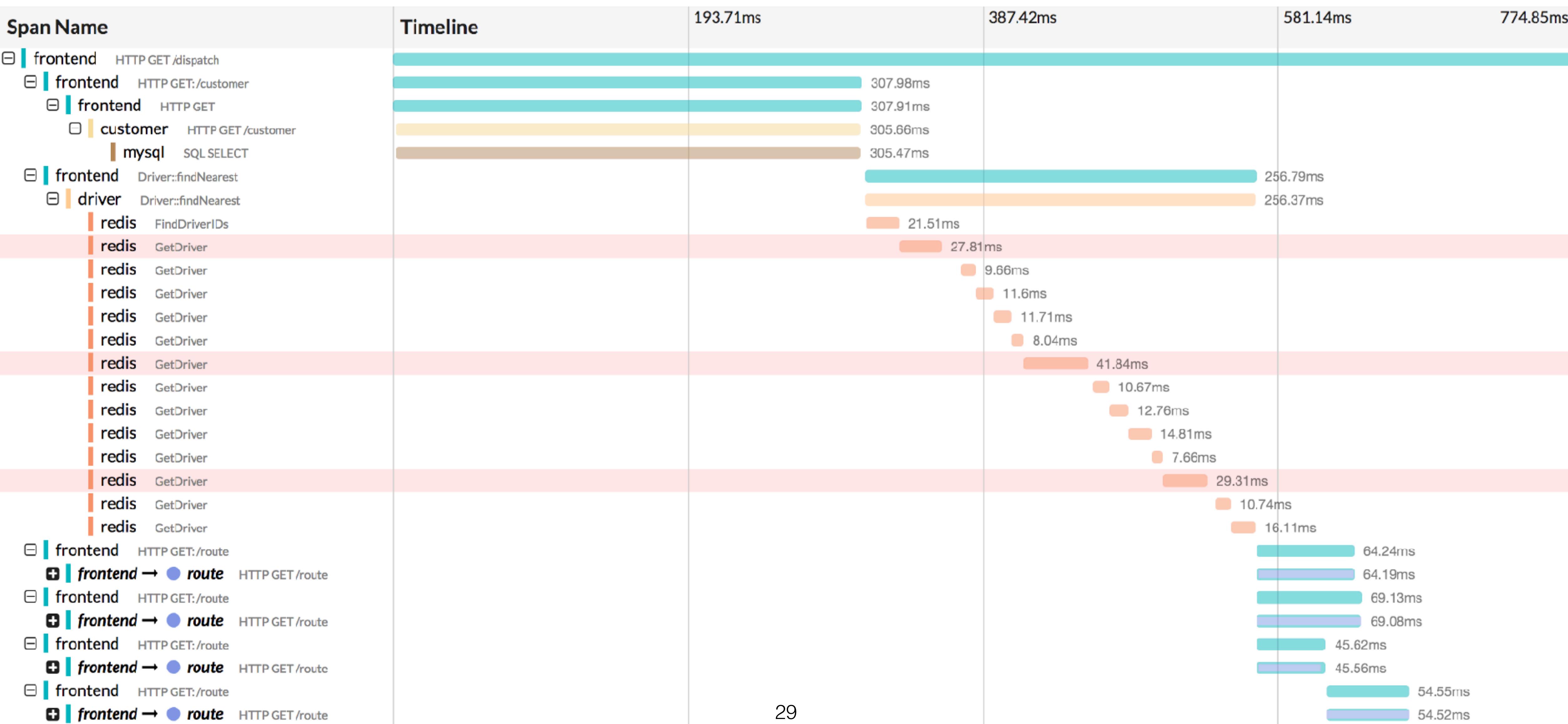
Find Traces



» frontend: HTTP GET /dispatch

View Options ▾

Search...



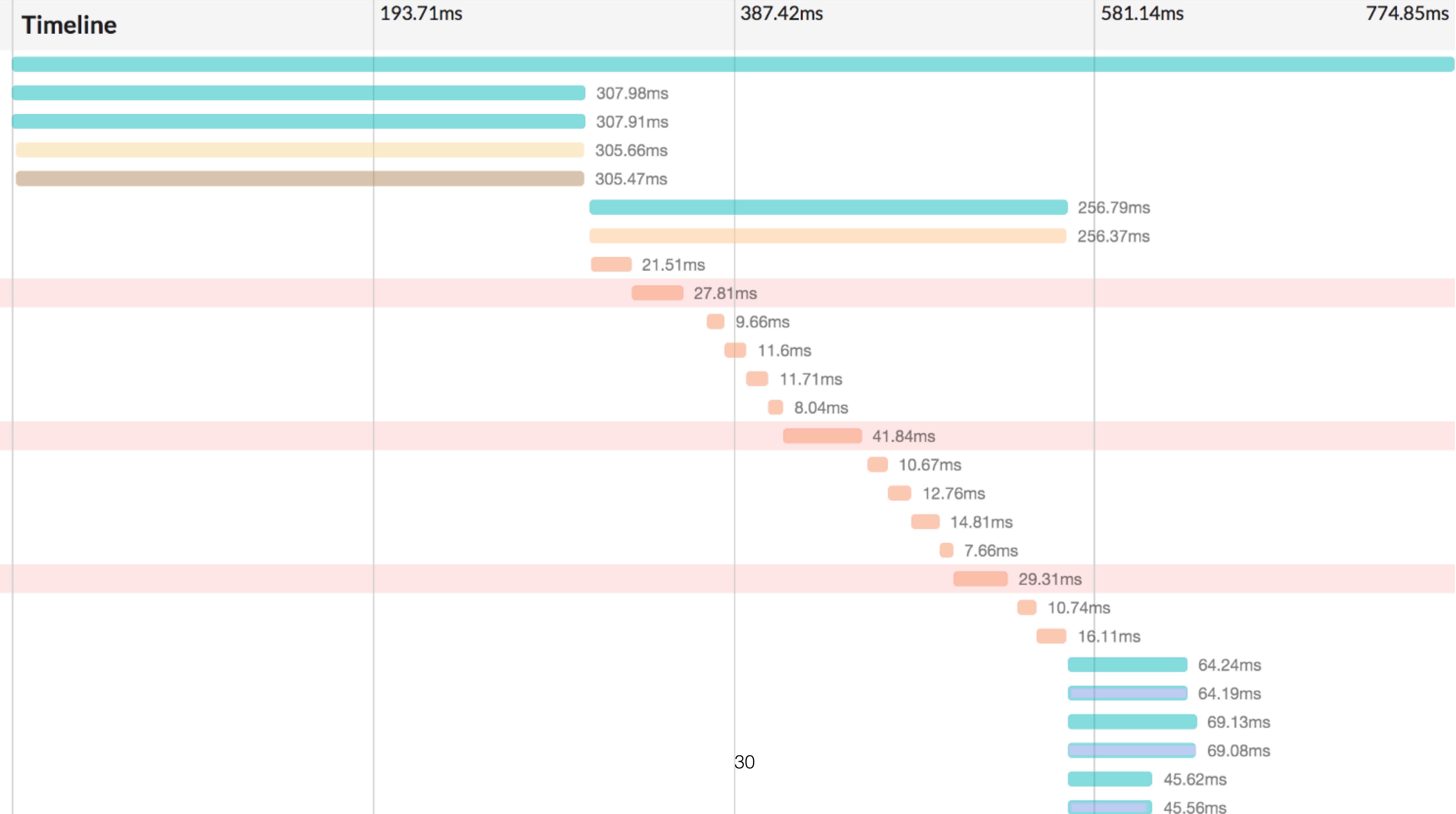
Timeline

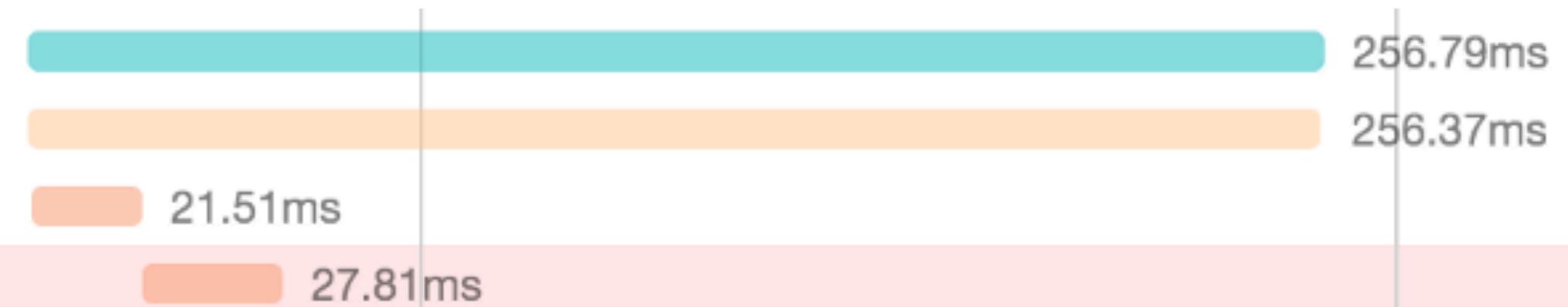
193.71ms

387.42ms

581.14ms

774.85ms





GetDriver

Service: redis Duration: 27.81ms Start Time: 332.69ms

+ Tags: param.driverID=T748713C error=true span.kind=client

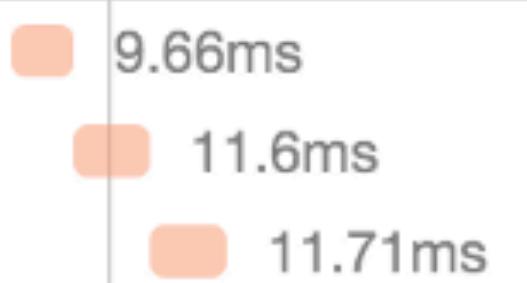
+ Process: ip=192.168.1.4

▼ Logs (1)

+ 358.54ms: event=redis timeout driver_id=T748713C error=redis timeout level=error

**Log timestamps are relative to the start time of the full trace.

+ Debug Info



Нюансы внедрения

- Данных может быть много, помним про семплинг
- Для полного эффекта нужно полное покрытие

Выводы

- Если у вас микросервисы - вам нужен трейсинг
- Если у вас большой монолит - вам тоже нужен трейсинг
- Трейсинг даст вам очень подробную картину происходящего
- Начать просто, есть куча готовых библиотек

dev(nsk)

N1.RU
НЕДВИЖИМОСТЬ

Вопросы?

Олег Федосеев
  olegfedoseev
oleg.fedoseev@me.com