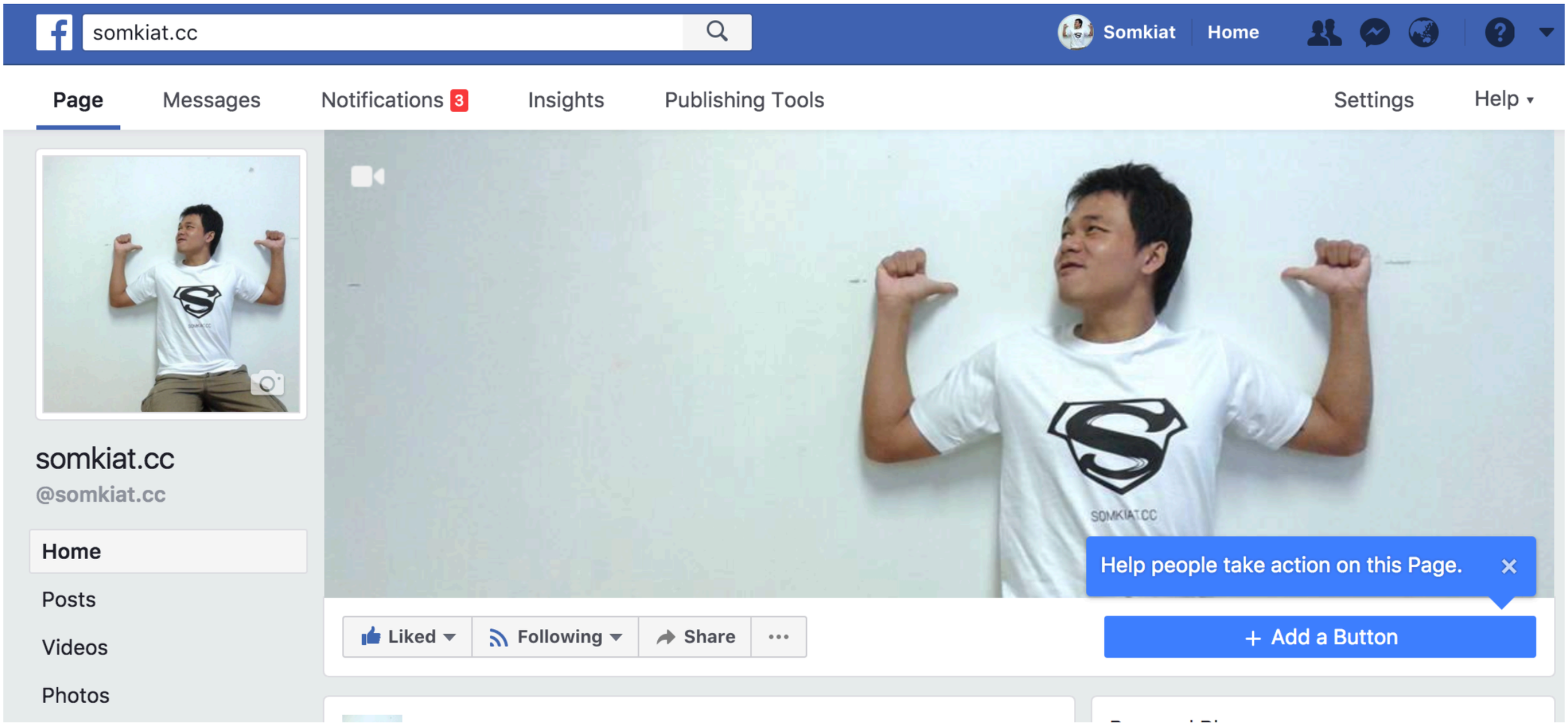
สยามชำนาญกิจ
SIAM CHAMNAN KIT

Monitoring and Observability
ELK Stack

สยามชำนาญกิจ
SIAM CHAMNAN KIT

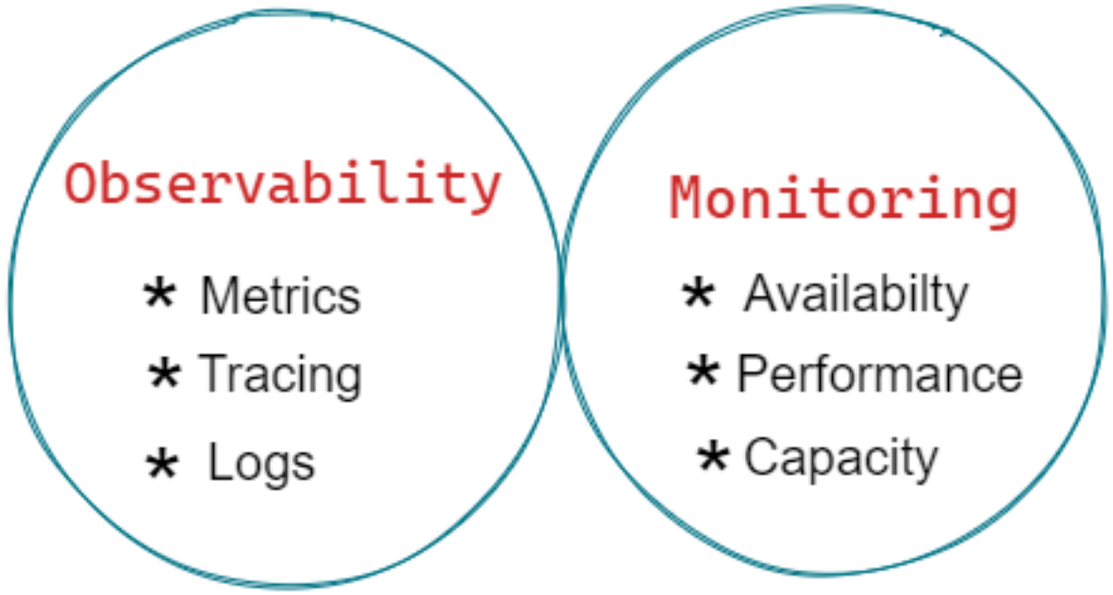
SHU HA RI
SOFTWARE DEVELOPMENT SCHOOL



How to find and solve Issues/incidents ?

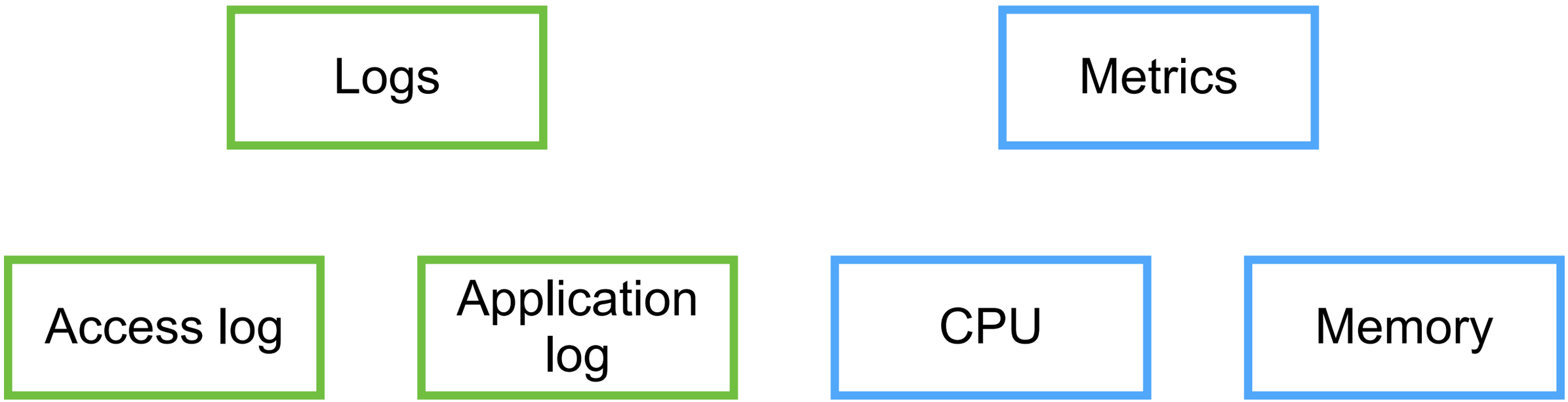
Reactive Vs. Proactive

Monitoring and Observability



Monitoring ?

Gathering data from your organization’s whole **infrastructure**



Monitoring Provide ..

Insight into your system working
Notify when have issues/errors
Measure there performance of application
Improve user experience
Help business

Observability ?

Assess how well you can understand a system's **internal states**

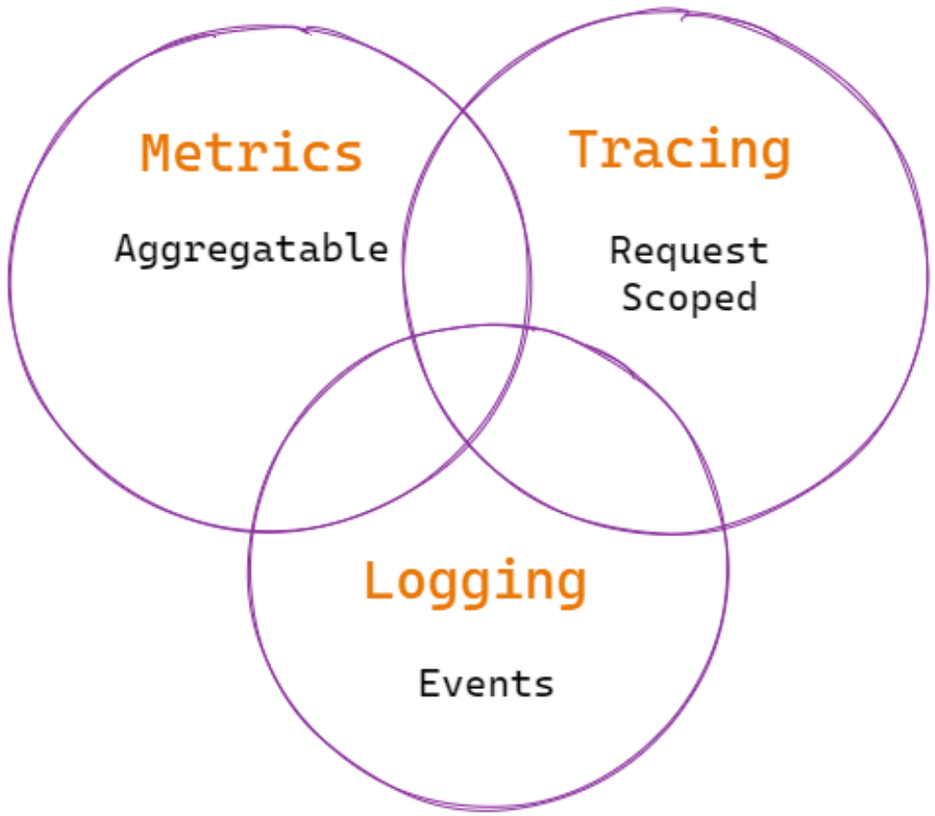
Help to understand system/application behaviors

Data from observability delivered to monitoring

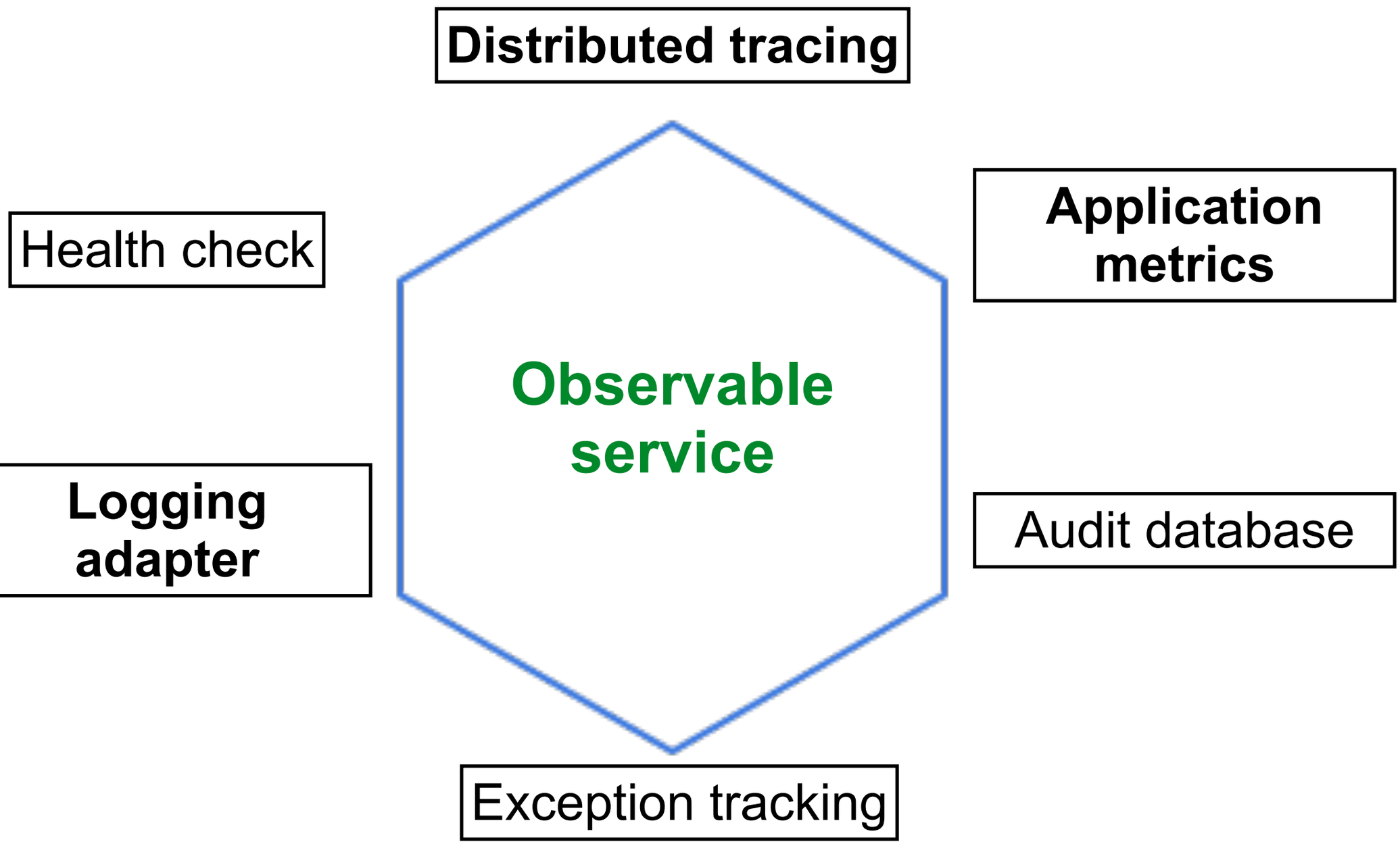
Observability ?

Assess how well you can understand a system’s internal states

Data from observability delivered to monitoring



Observable Services



Observability provide

Better visibility

Better alerting

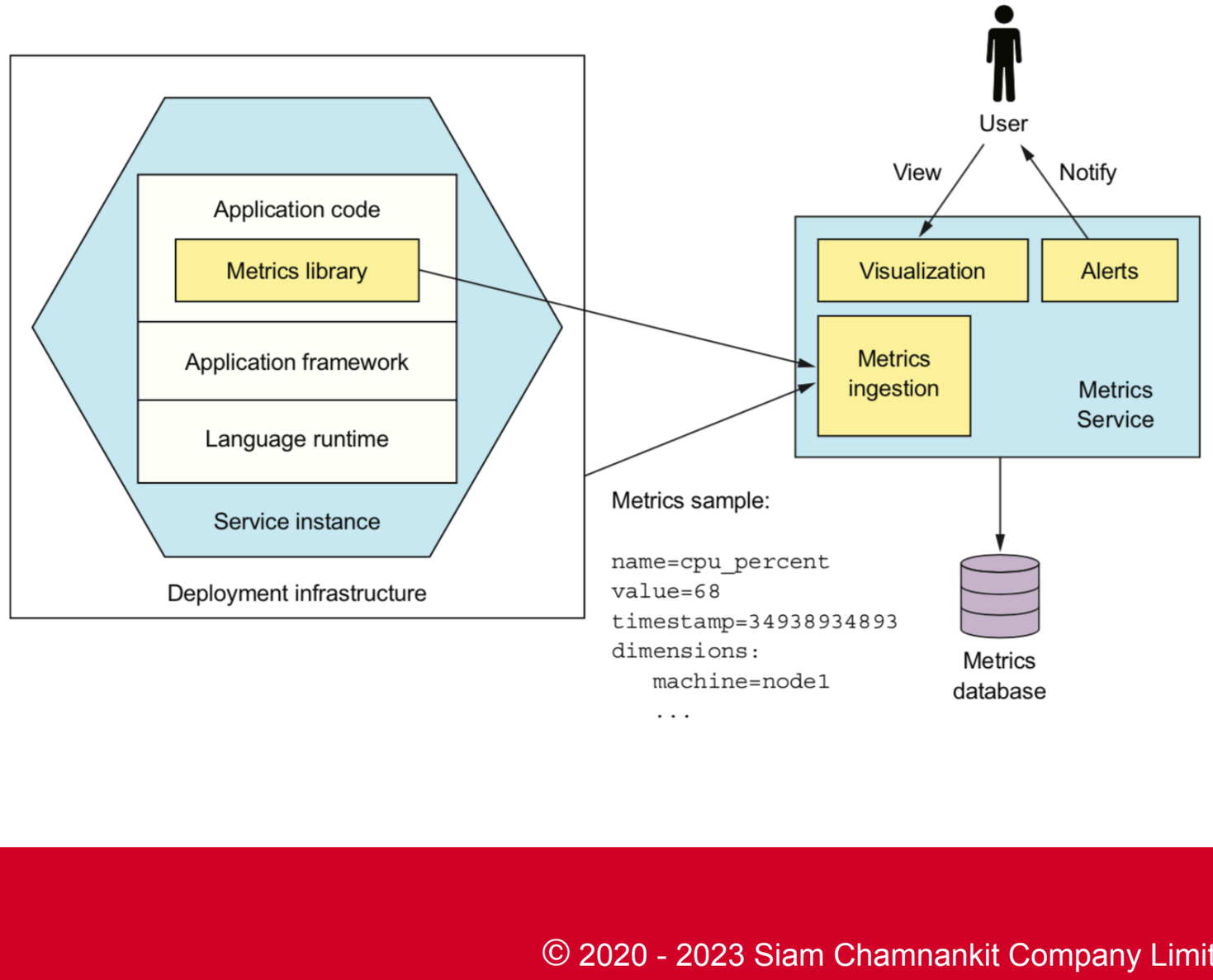
Less time to recovery

Less time in meeting

Better workflow

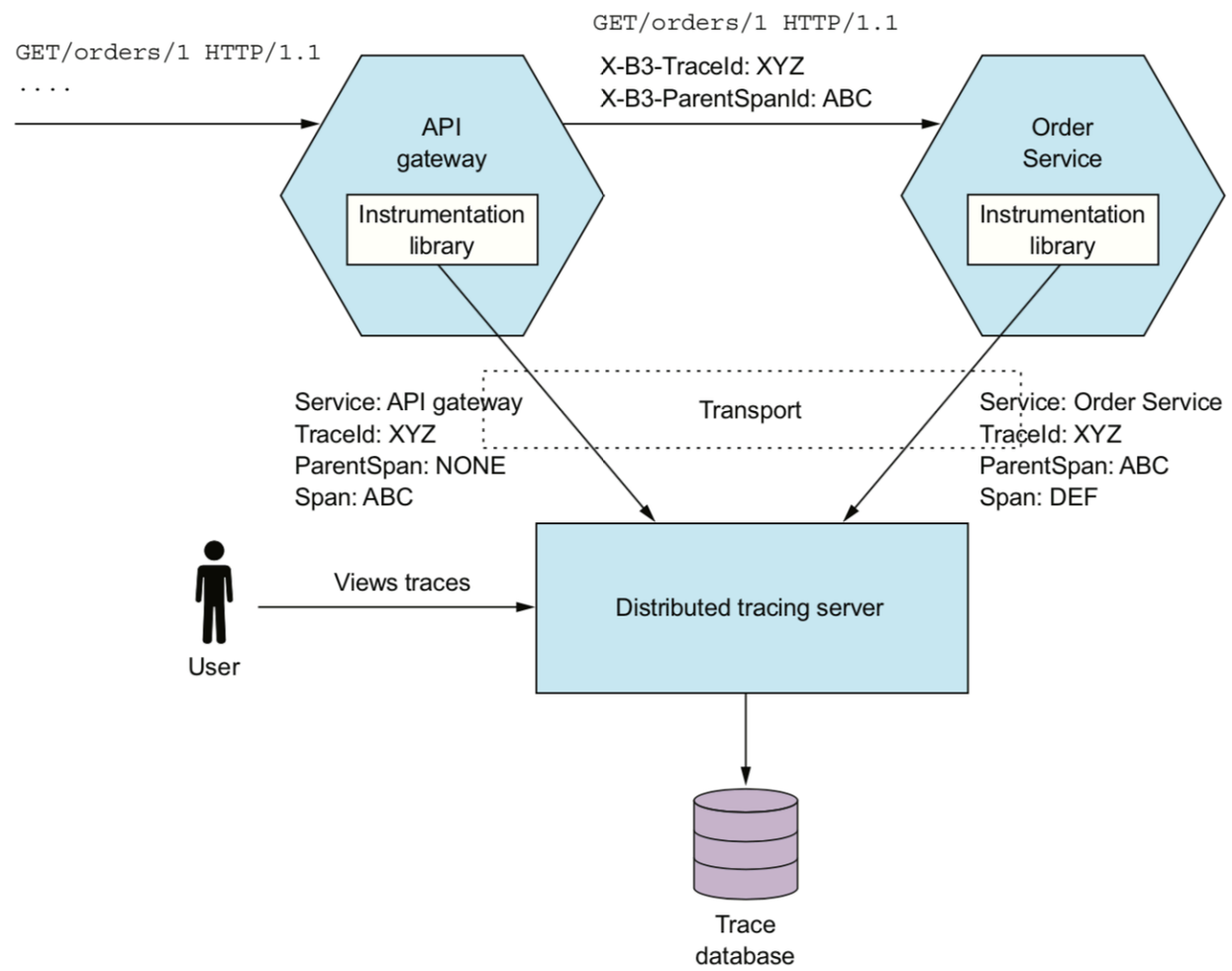
Application metrics

Services maintain metrics and expose to metric server (counters, gauges)

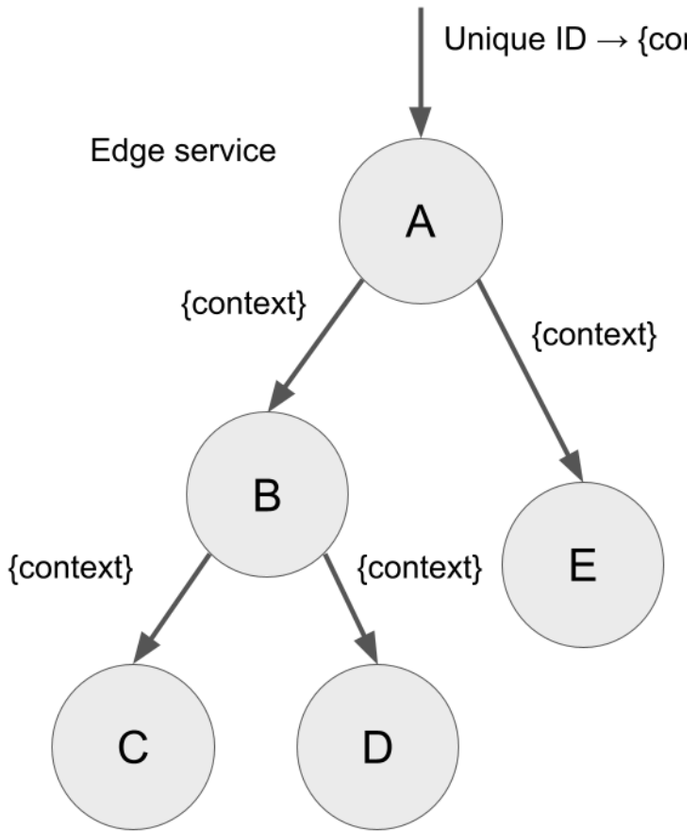


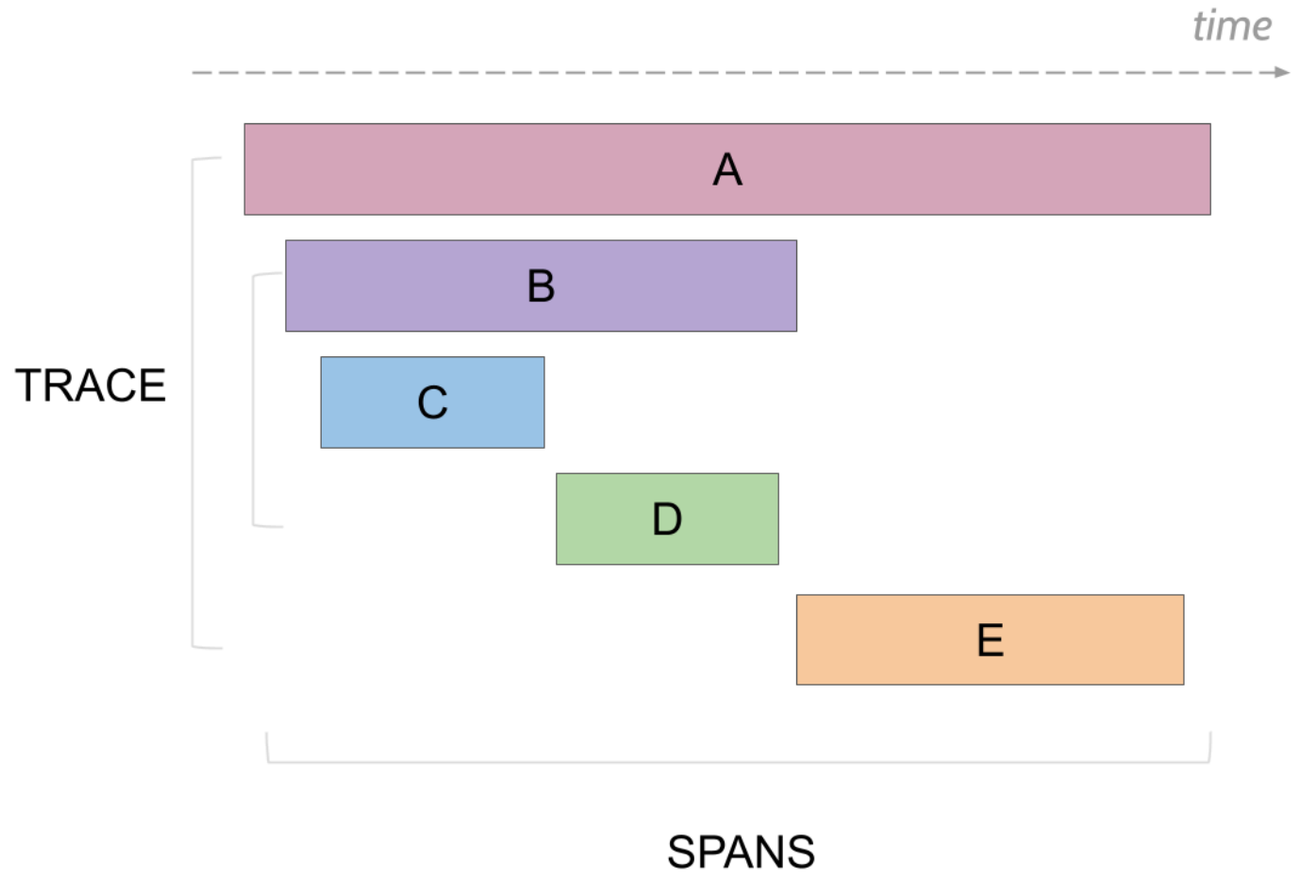
Distributed tracing

Assign each external request a **unique ID** and trace requests as flow between services



Distributed tracing





Distributed tracing tools

Format standard with OpenTelemetry
Zipkin, Jaeger, Tempo, AWS X-Ray, Elastic APM

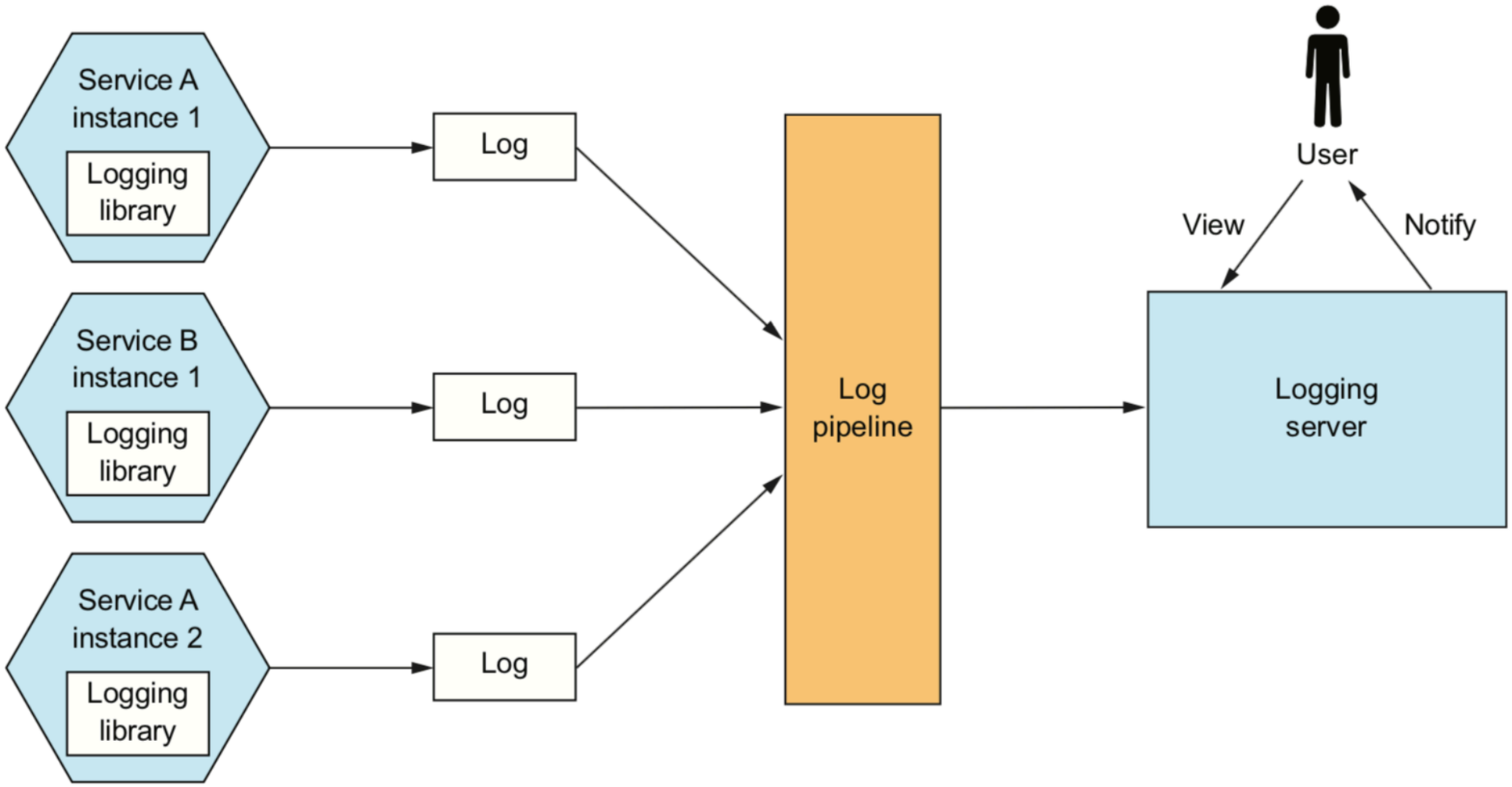
JAEGER

Grafana Tempo

OpenTelemetry

Log aggregation

Log service activity and write logs into a centralized logging server. (searching, alerting)

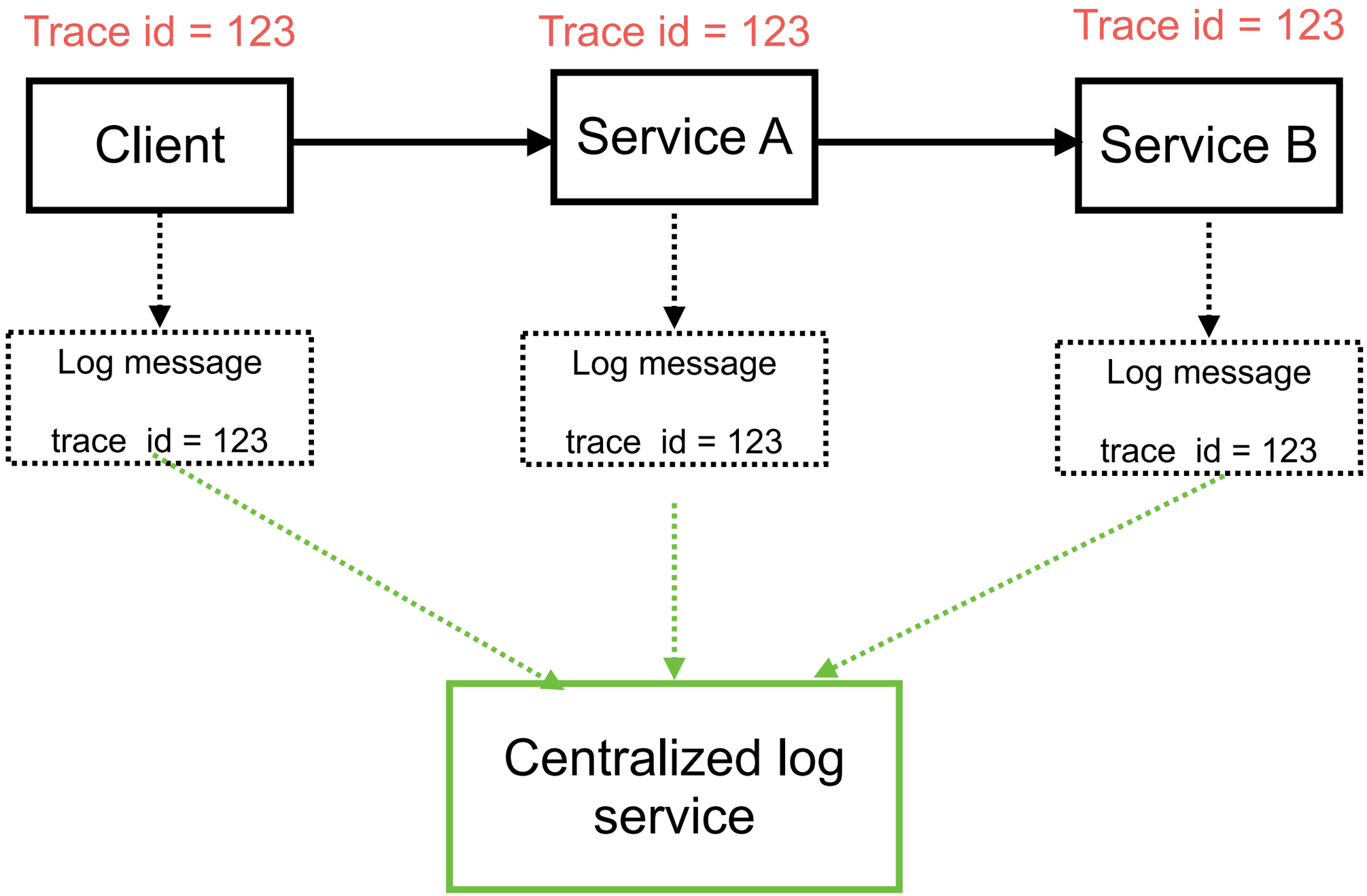


Effective Log Aggregation

Define event to log
Use structured logging
Exclude sensitive information
Log at the correct level
Be specific in your message
Don’t log large message
Make sure you keep trace Id in the log

https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Logging_Cheat_Sheet.md

Log aggregation



Consistent Structure across all logs

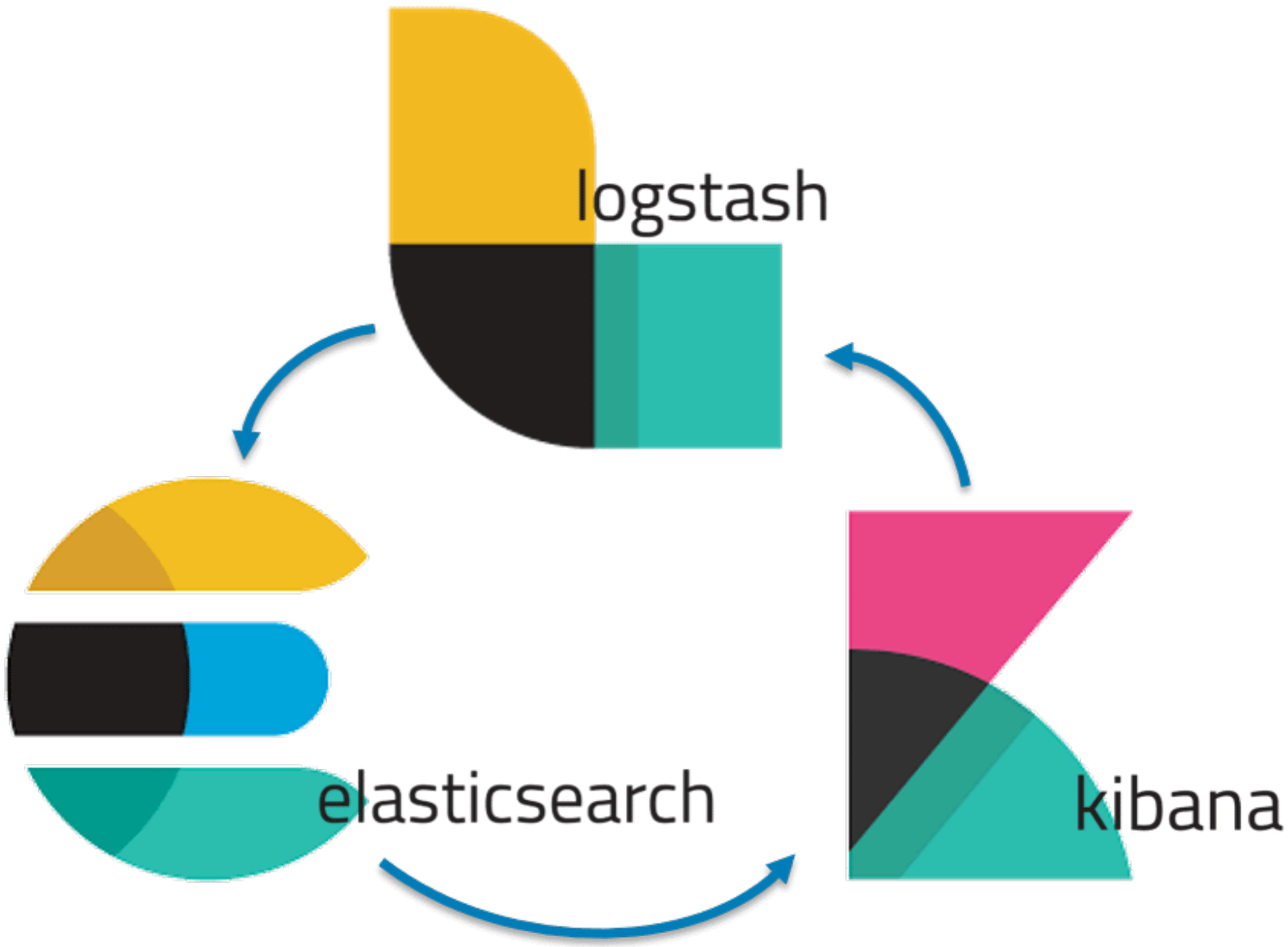
Property	Description	Example
Timestamp	Date and time of the log	2023-07-01
Log level	DEBUG, INFO, ERROR	
Trace Id or Correlation Id	Unique identifier that refer to other logs from all services	
Event/Action Name	Identify to event or action of log	Authentication fail
Service ID/ Name	Identify to service	
Request path	Path for the request	/api/products

Don’t keep !!

```
com.framework.FrameworkException: Error in web request
    at com.framework.ApplicationStarter.lambda$start$0(ApplicationStarter.java:15)
    at spark.RouteImpl$1.handle(RouteImpl.java:72)
    at spark.http.matching.Routes.execute(Routes.java:61)
    at spark.http.matching.MatcherFilter.doFilter(MatcherFilter.java:134)
    at spark.embeddedserver.jetty.JettyHandler.doHandle(JettyHandler.java:50)
    at org.eclipse.jetty.server.session.SessionHandler.doScope(SessionHandler.java:1568)
    at org.eclipse.jetty.server.handler.ScopedHandler.handle(ScopedHandler.java:144)
    at org.eclipse.jetty.server.handler.HandlerWrapper.handle(HandlerWrapper.java:132)
    at org.eclipse.jetty.server.Server.handle(Server.java:503)
    at org.eclipse.jetty.server.HttpChannel.handle(HttpChannel.java:364)
    at org.eclipse.jetty.server.HttpConnection.onFillable(HttpConnection.java:260)
    at org.eclipse.jetty.io.AbstractConnection$ReadCallback.succeeded(AbstractConnection.java:305)
    at org.eclipse.jetty.io.FillInterest.fillable(FillInterest.java:103)
    at org.eclipse.jetty.io.ChannelEndPoint$2.run(ChannelEndPoint.java:118)
    at org.eclipse.jetty.util.thread.QueuedThreadPool.runJob(QueuedThreadPool.java:765)
    at org.eclipse.jetty.util.thread.QueuedThreadPool$2.run(QueuedThreadPool.java:683)
    at java.base/java.lang.Thread.run(Thread.java:834)
Caused by: com.project.module.MyProjectFooBarException: The number of FooBars cannot be zero
    at com.project.module.MyProject.anotherMethod(MyProject.java:20)
    at com.project.module.MyProject.someMethod(MyProject.java:12)
    at com.framework.ApplicationStarter.lambda$start$0(ApplicationStarter.java:13)
    ... 16 more
Caused by: java.lang.ArithmeticException: The denominator must not be zero
    at org.apache.commons.lang3.math.Fraction.getFraction(Fraction.java:143)
    at com.project.module.MyProject.anotherMethod(MyProject.java:18)
    ... 18 more
```

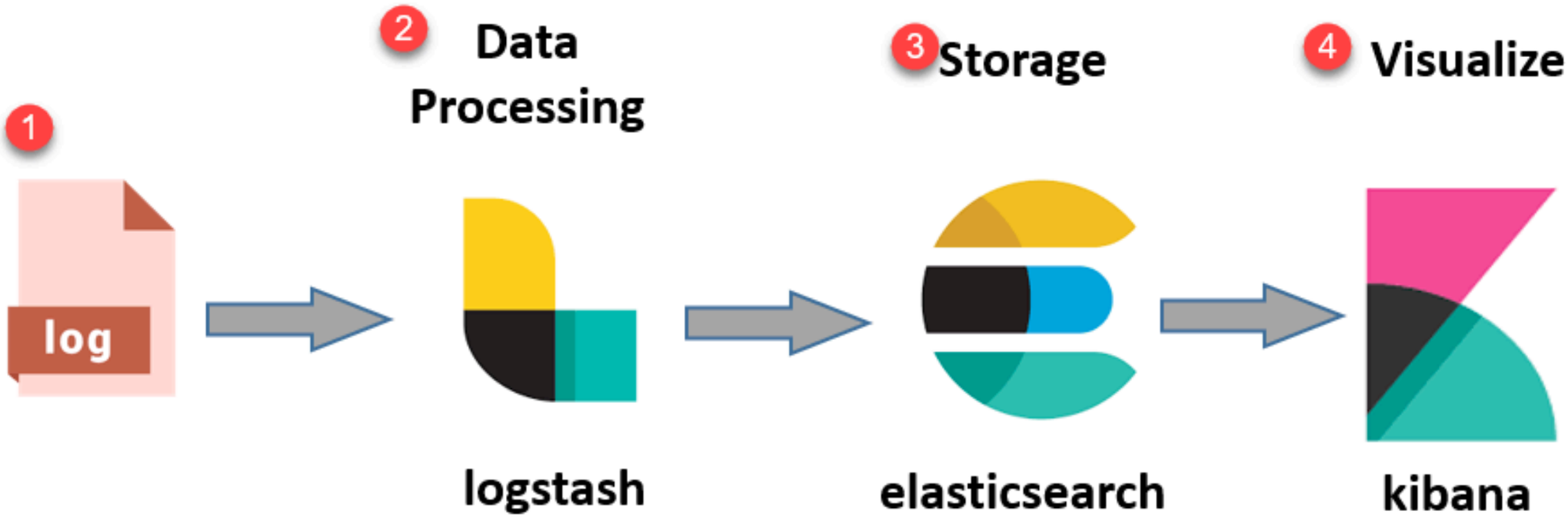
ELK Stack

ELK Stack



https://www.elastic.co/elastic-stack

ELK Stack

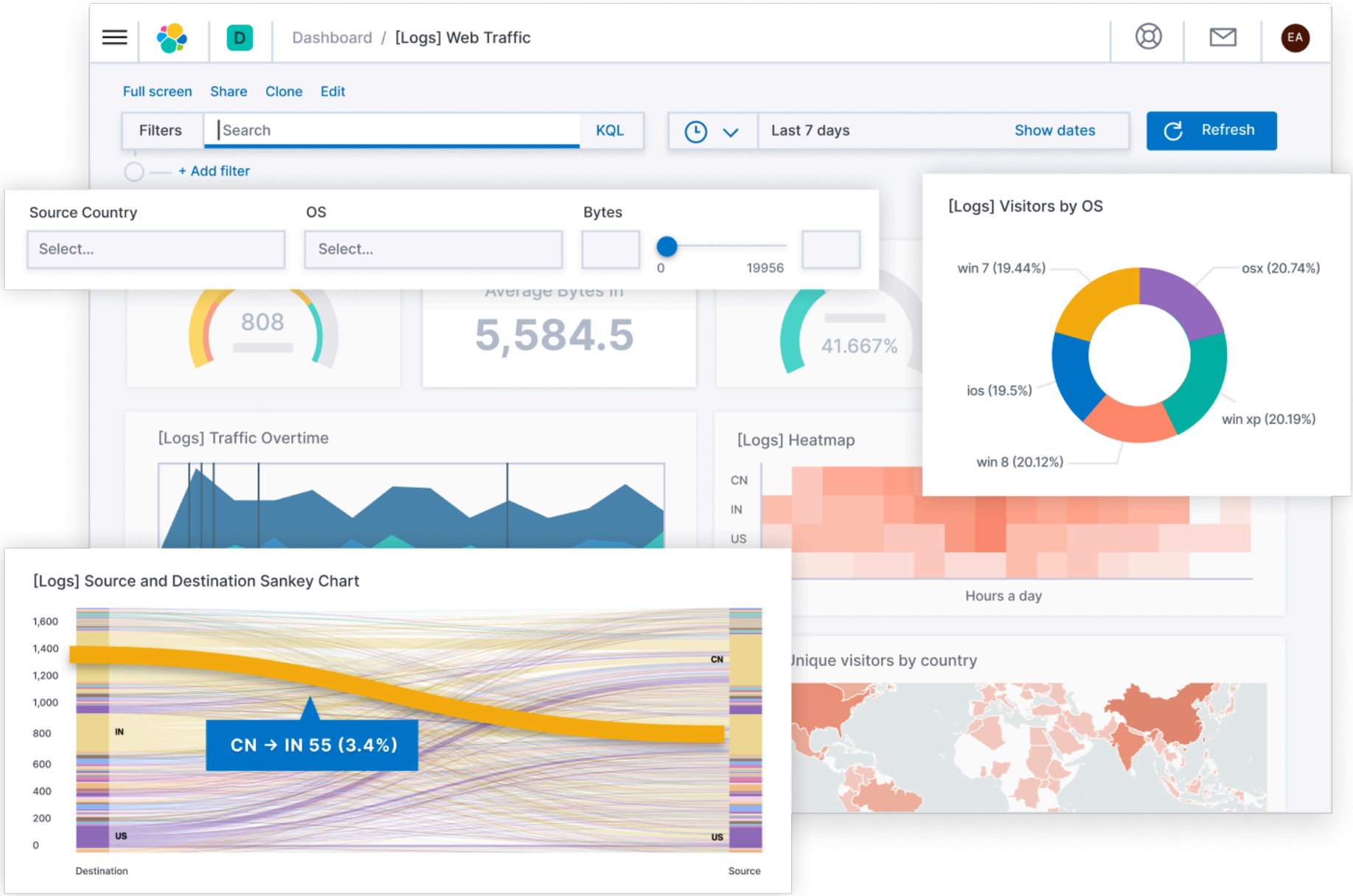


https://www.elastic.co/elastic-stack

Elasticsearch

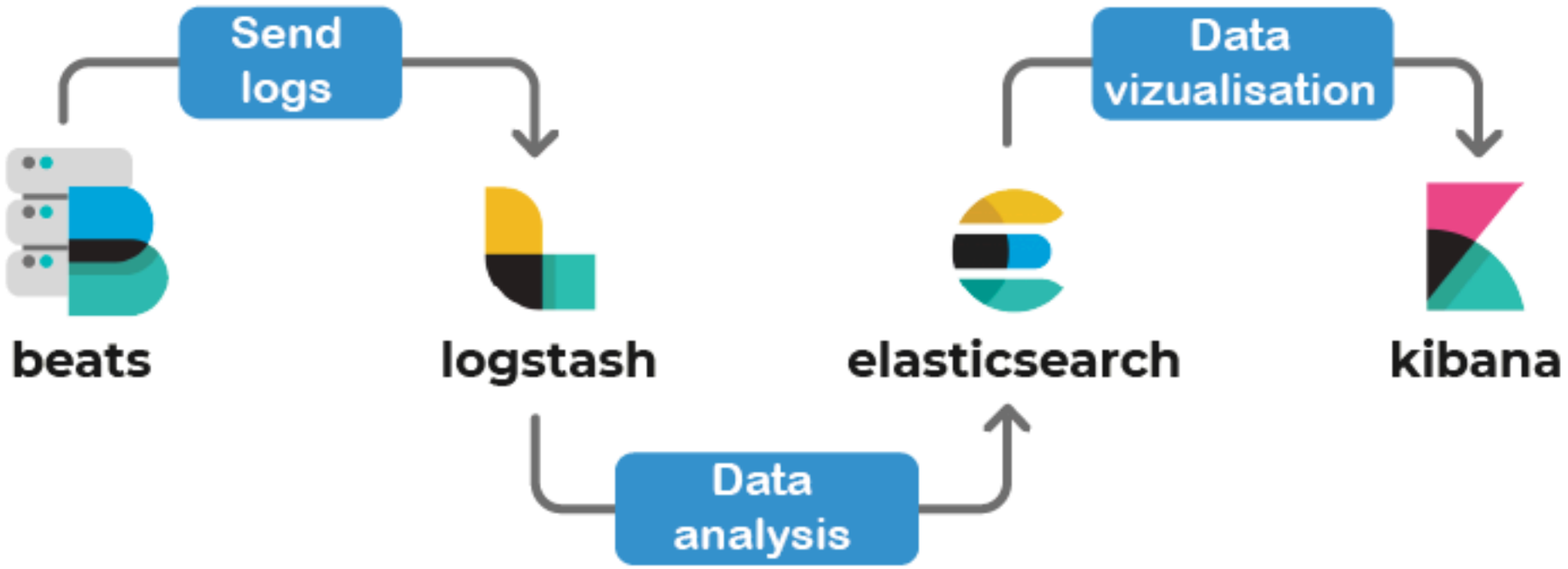
Database model for searching and aggregation
Support time series data
Scalable and Distributed database
Support JSON format

Kibana dashboard



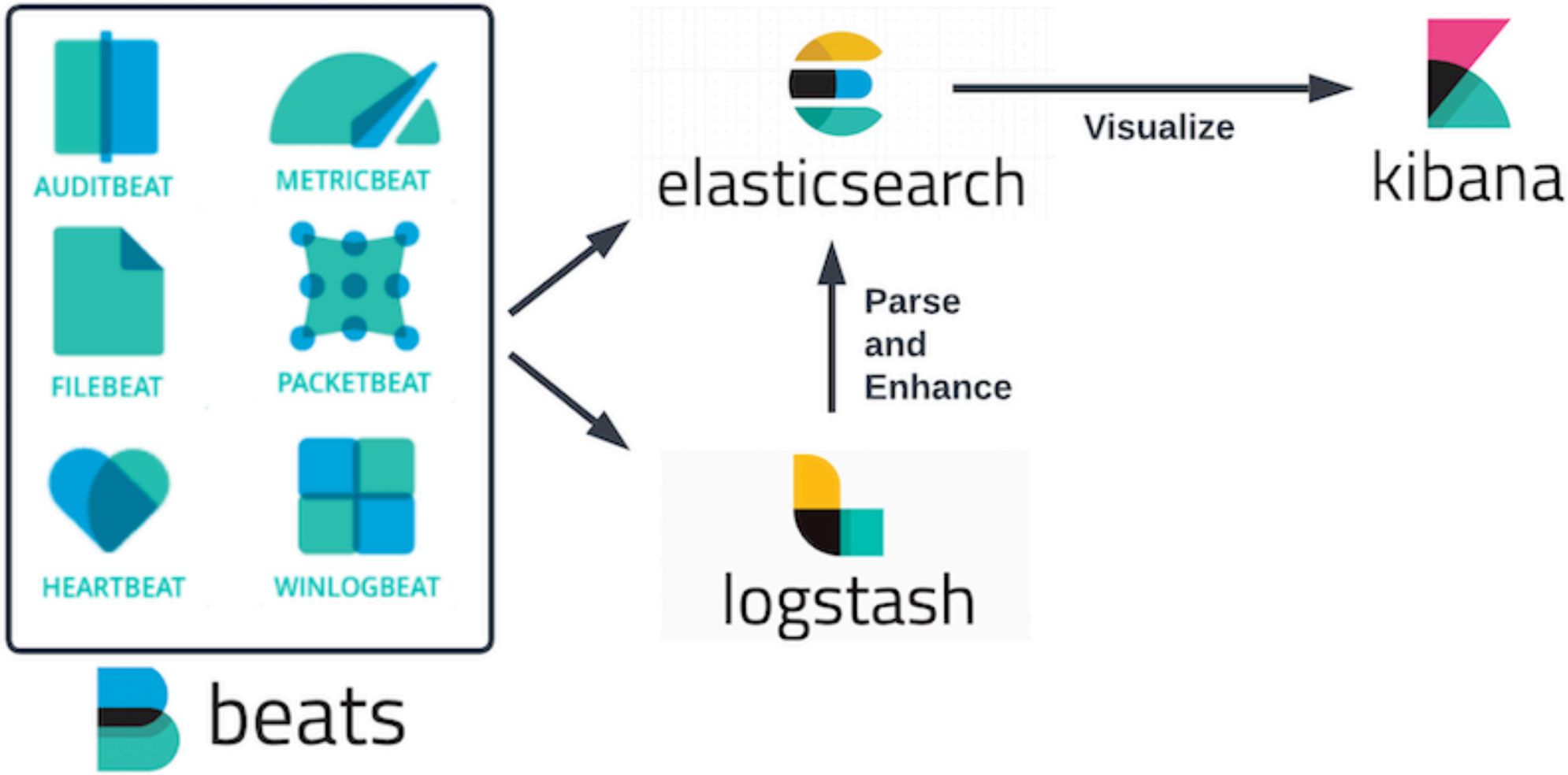
https://www.elastic.co/elastic-stack

ELK Stack + Beats



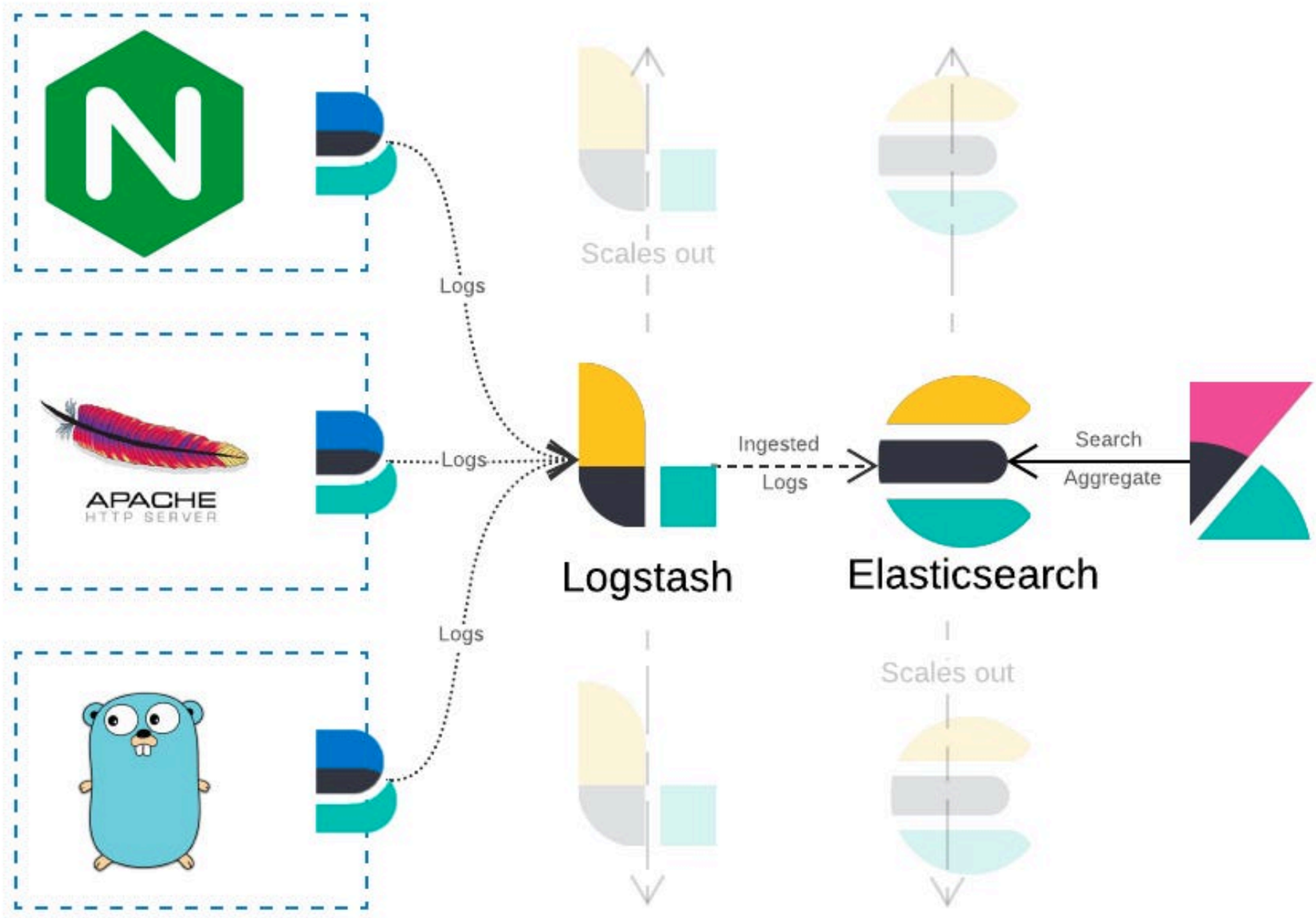
https://www.elastic.co/beats

ELK Stack + Beats



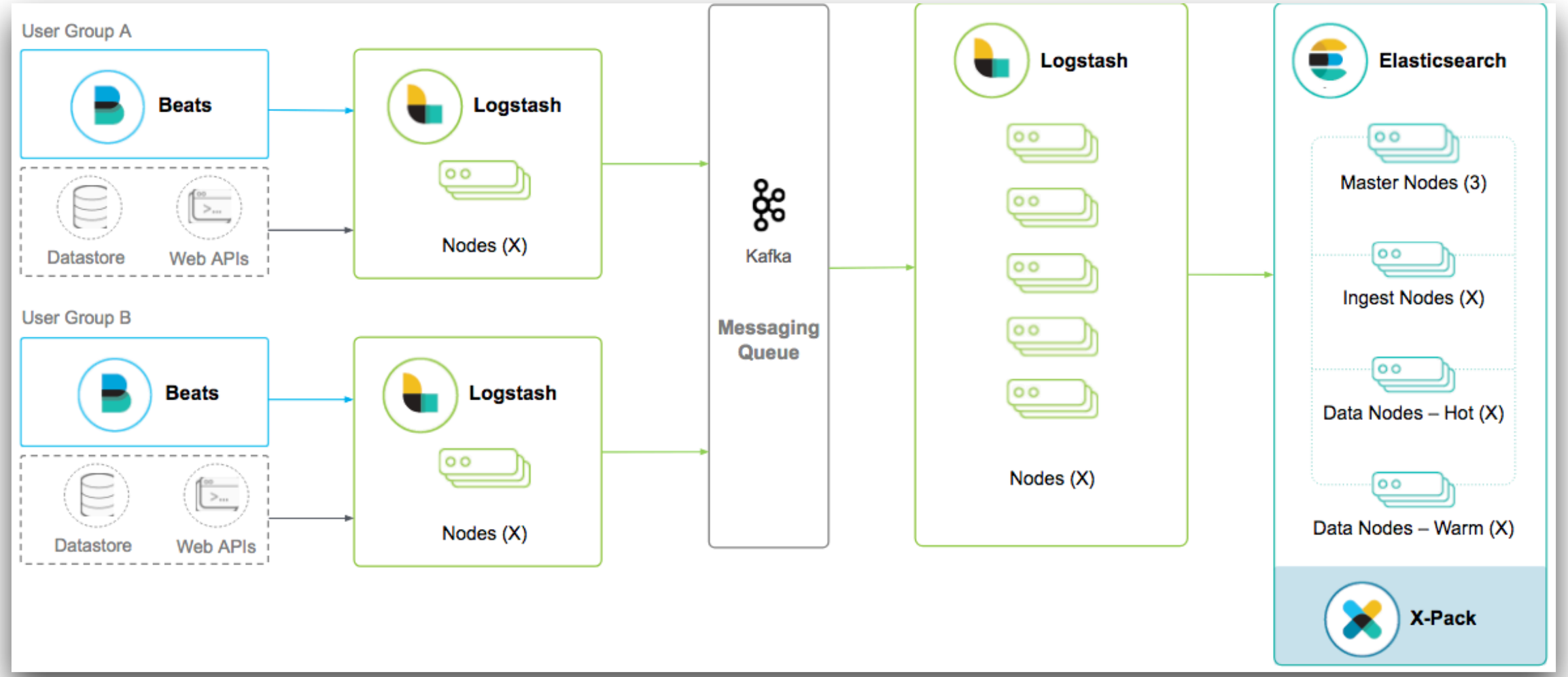
https://www.elastic.co/beats

ELK Stack + Beats



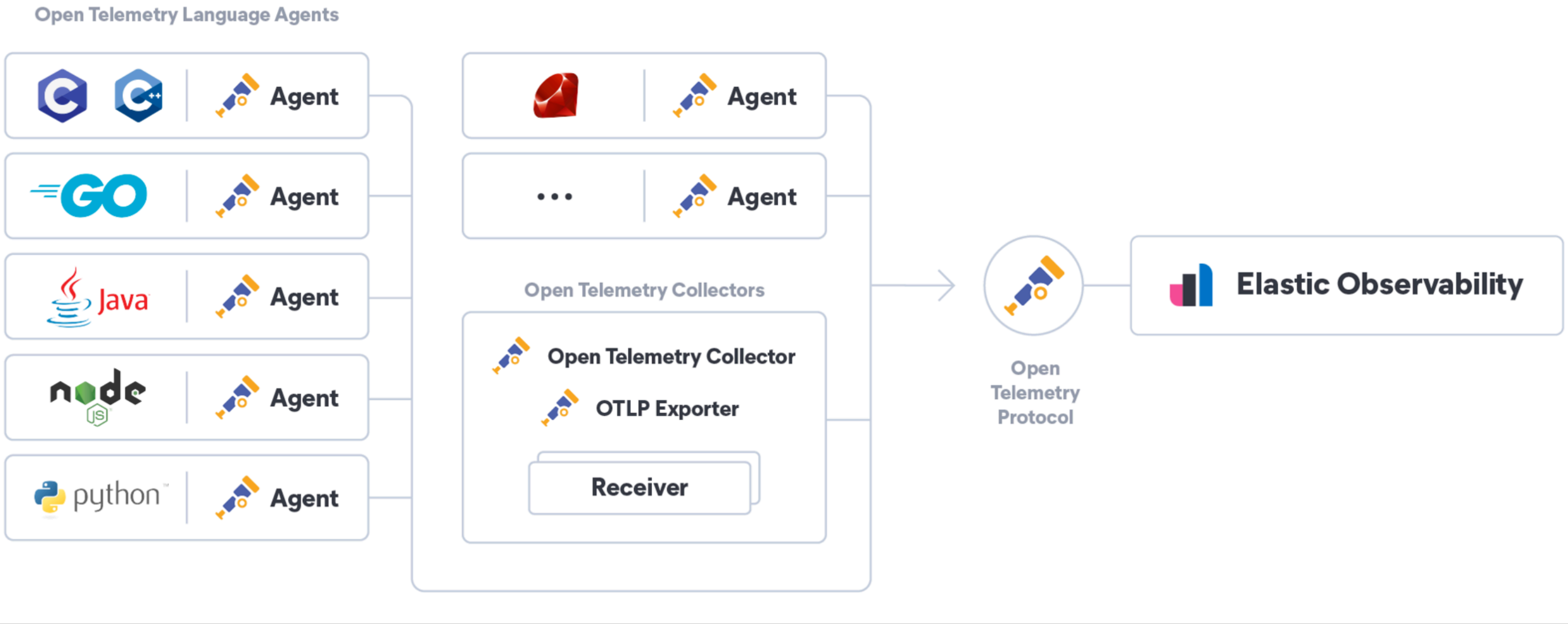
https://www.elastic.co/beats

Scaling



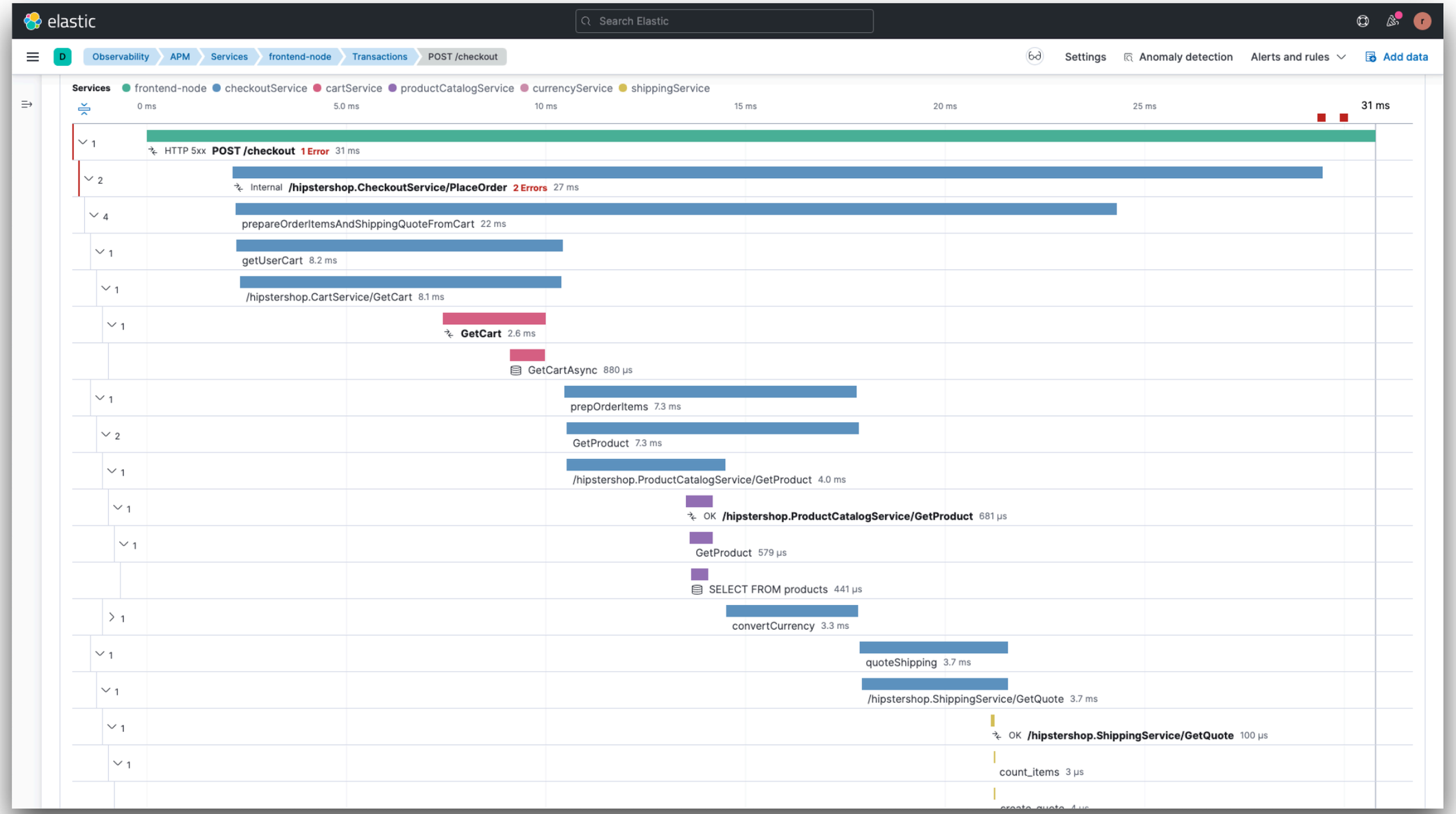
https://www.elastic.co

Elastic APM



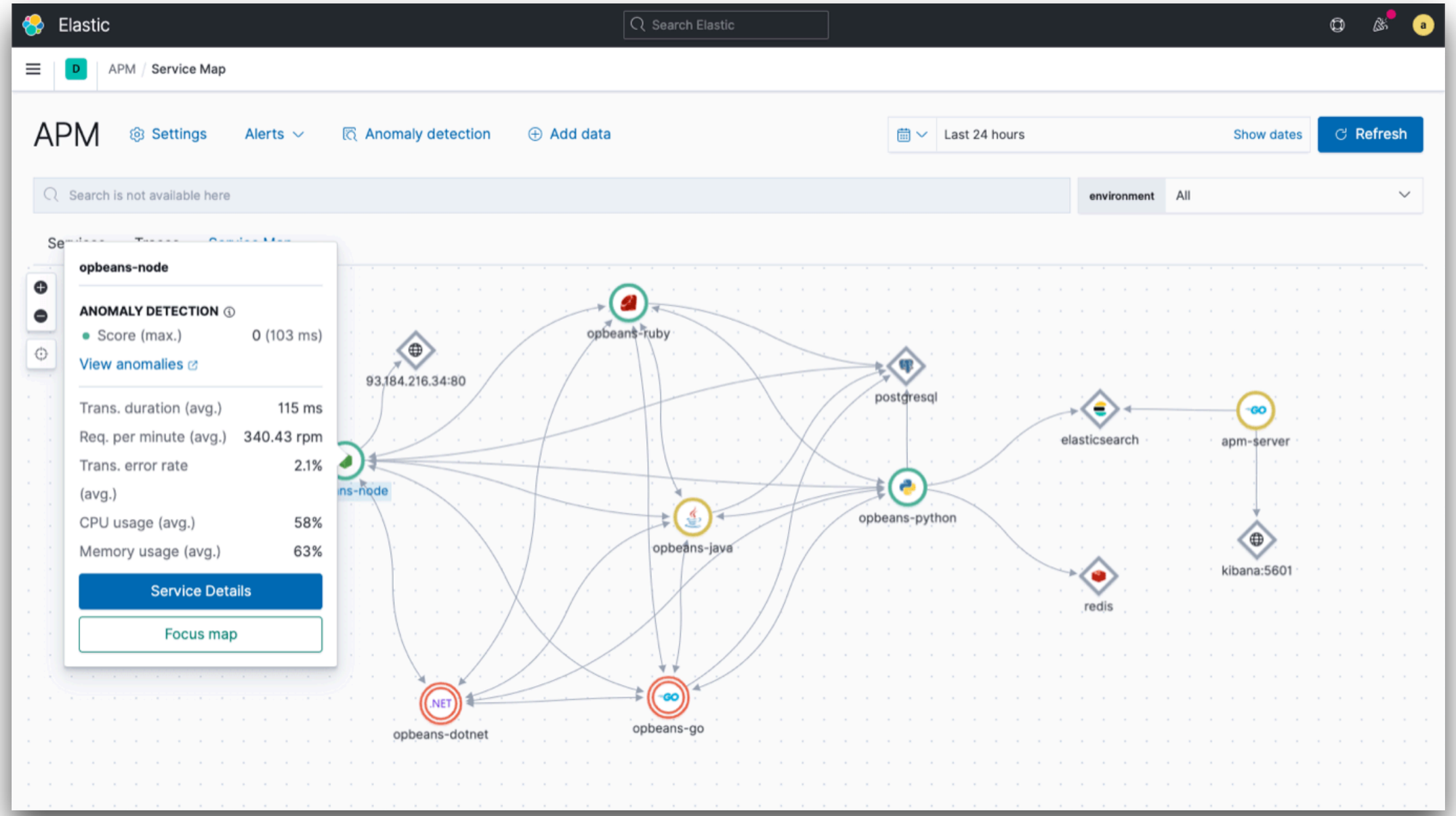
https://www.elastic.co/observability/application-performance-monitoring

Elastic APM



https://www.elastic.co/observability/application-performance-monitoring

Elastic APM



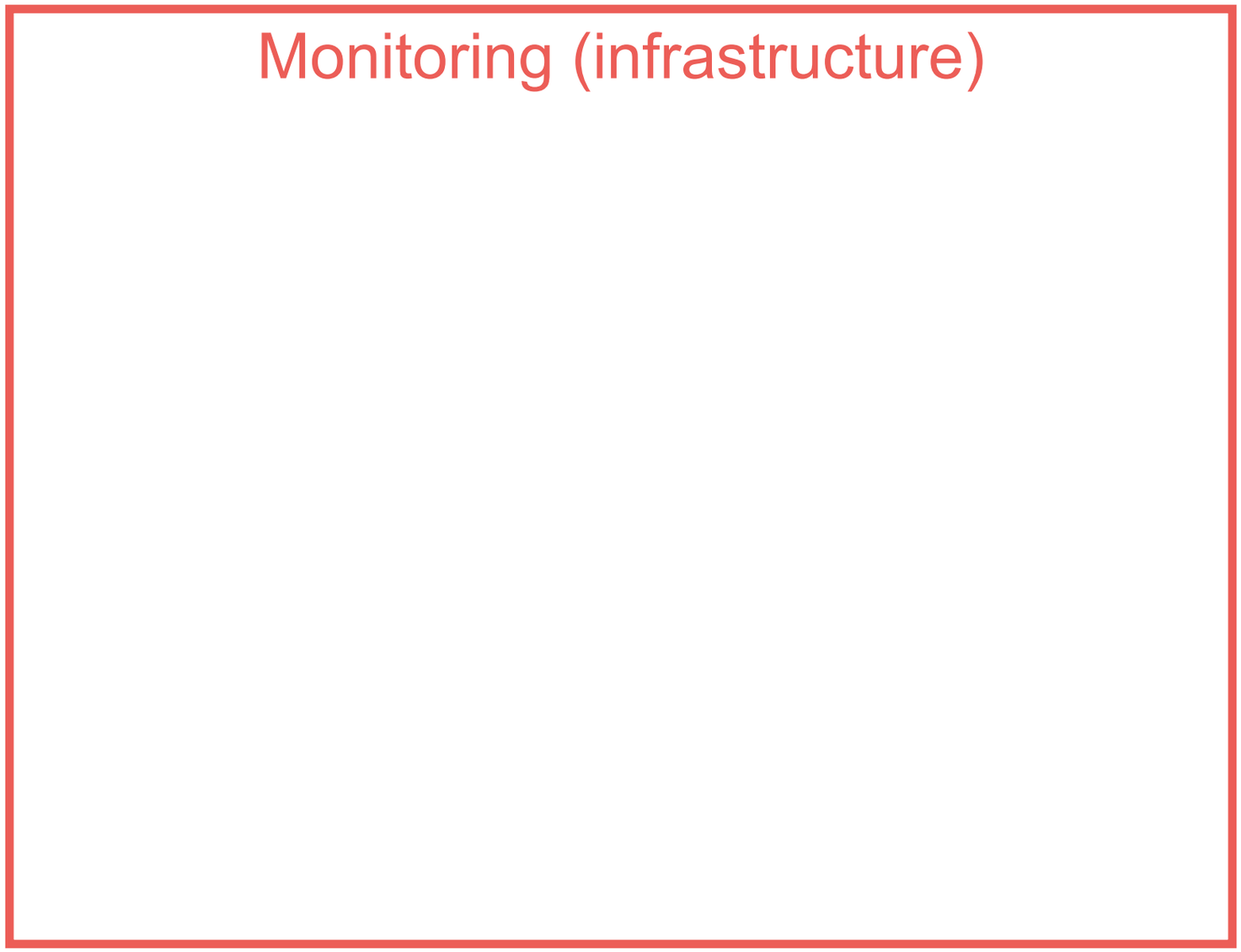
https://www.elastic.co/observability/application-performance-monitoring

More With RUM
(Real User Monitoring)

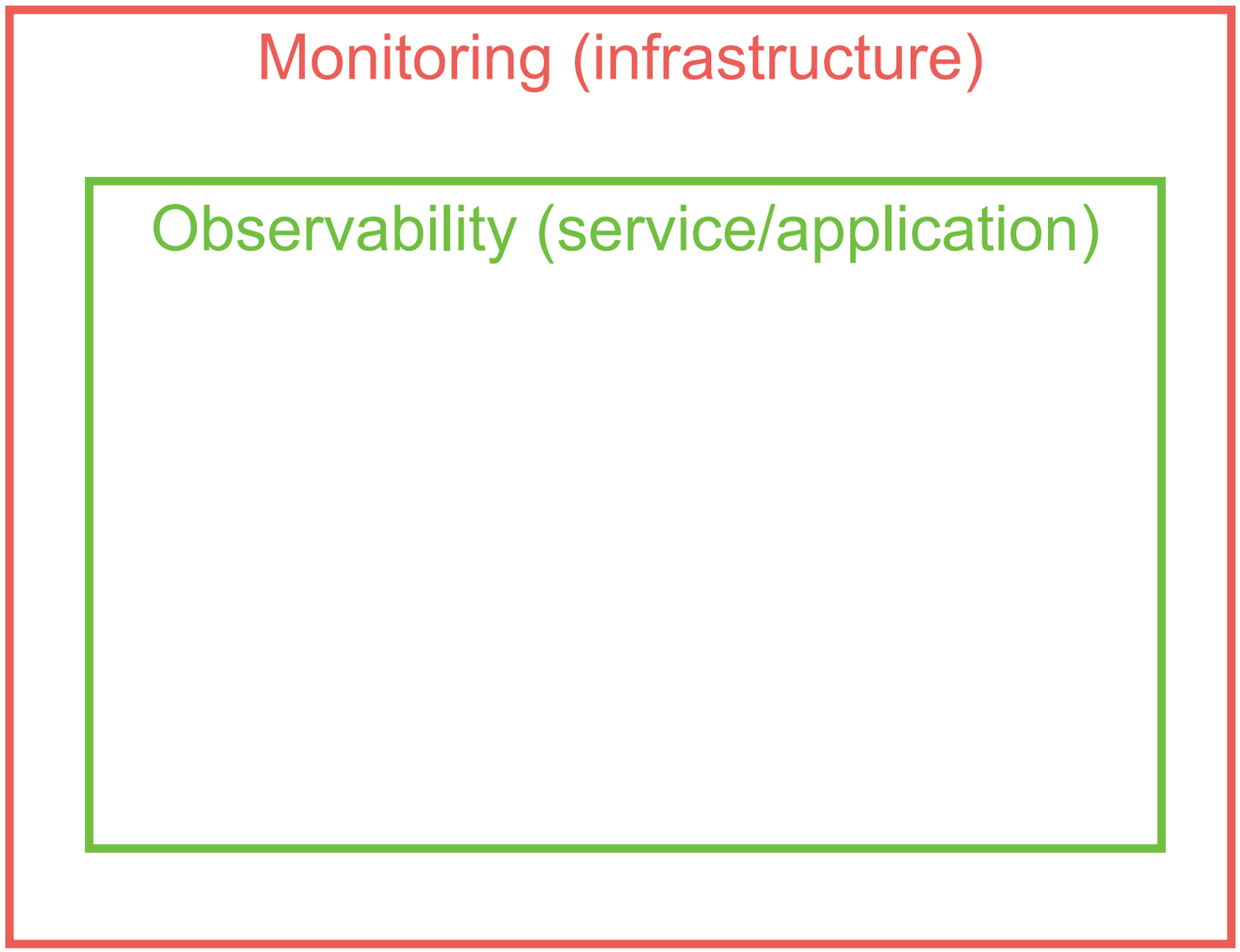
RUM

Mobile Real User Monitoring
Insight and information of end user usage patterns
Performance, analyze transaction in user level

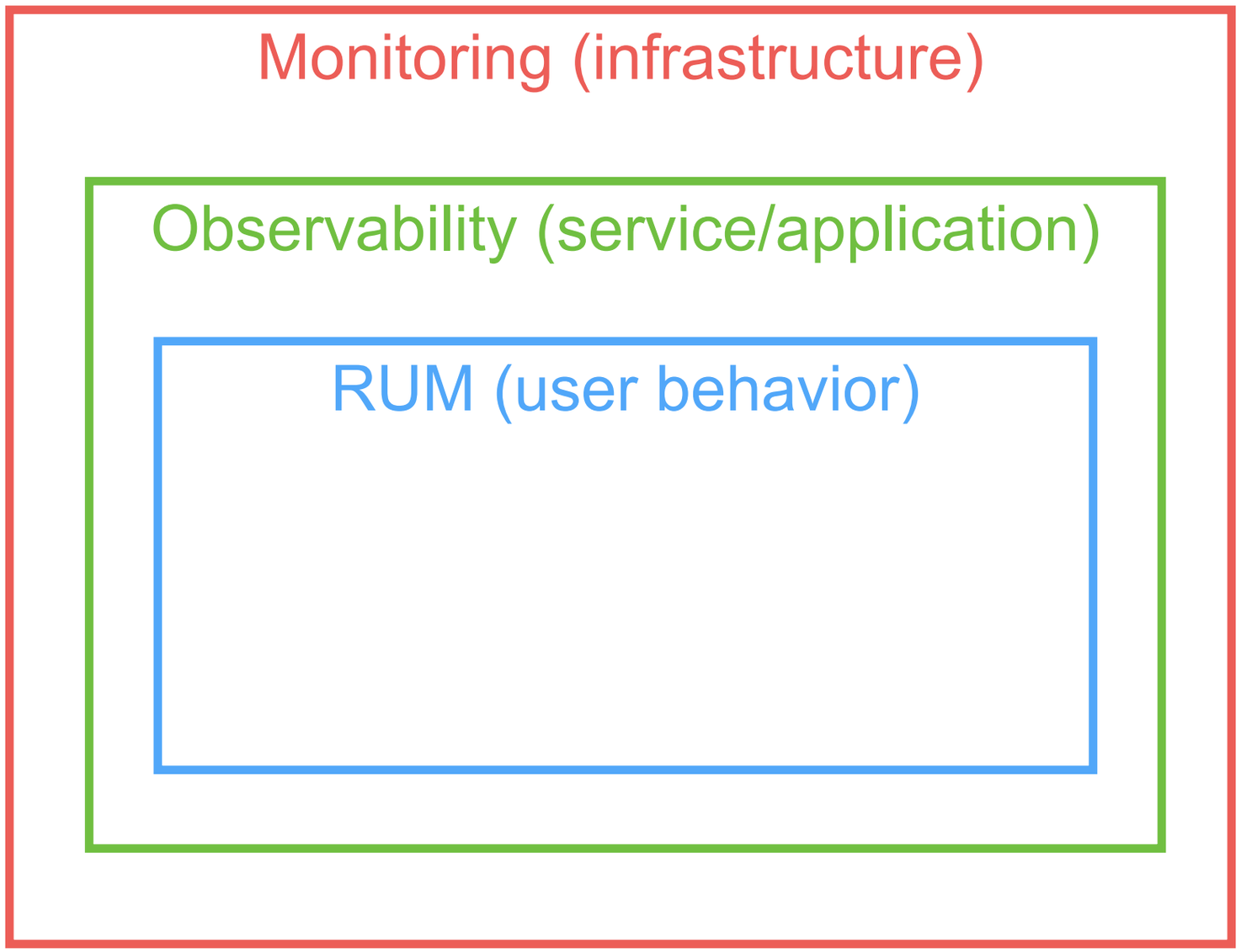
Monitoring -> Observability -> RUM



Monitoring -> Observability -> RUM



Monitoring -> Observability -> RUM



Grafana Faro

Grafana Labs | Products | Open source | Solutions | Learn | Docs | Company | Downloads | Contact us | Sign in

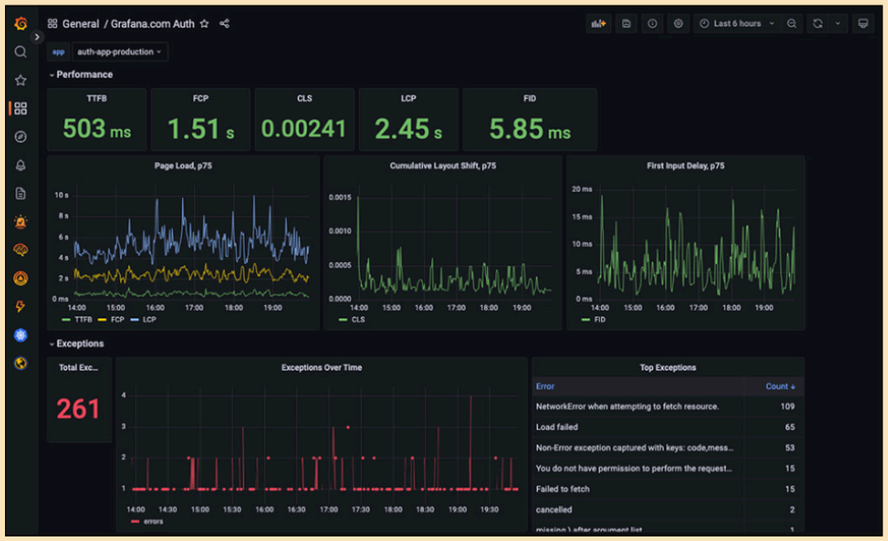
Star us on GitHub | ★ 568

Frontend monitoring

What is Grafana Faro?

A project for frontend application observability, Grafana Faro includes a highly configurable web SDK for real user monitoring (RUM) that instruments browser frontend applications to capture observability signals. The frontend telemetry can then be correlated with backend and infrastructure data for seamless, full-stack observability.

GitHub project Blog announcement Documentation



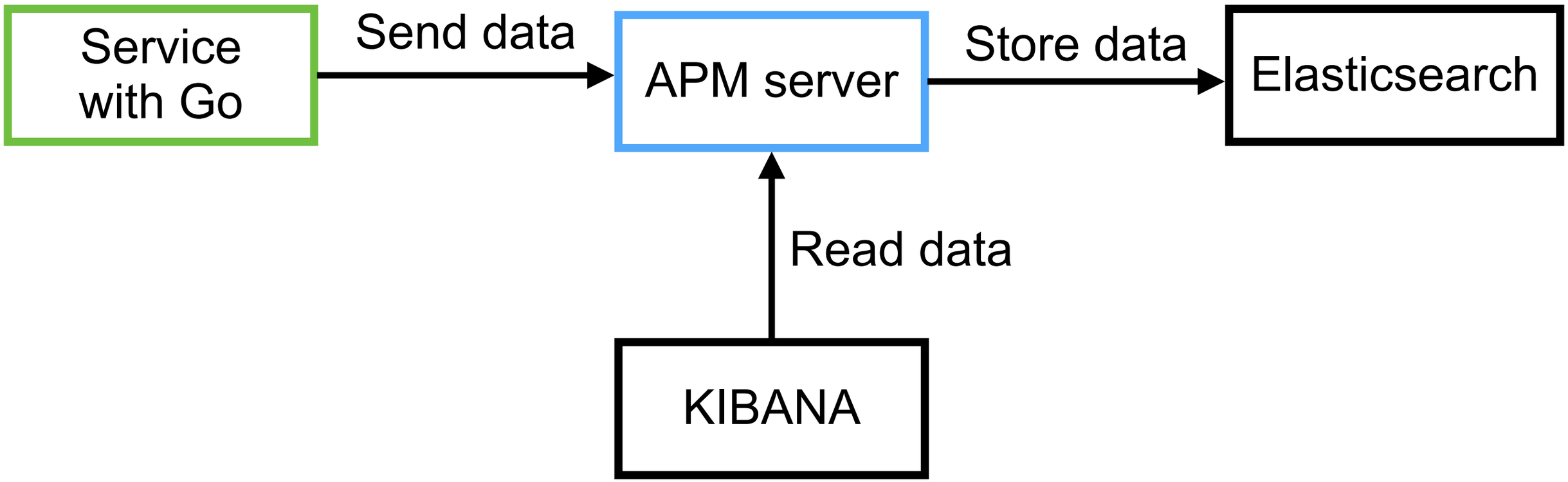
Grafana Faro overview

The Grafana Faro Web SDK is a highly configurable open source JavaScript agent that can easily be embedded in web applications to collect real user monitoring (RUM) data: performance metrics, logs, exceptions, events, and traces.

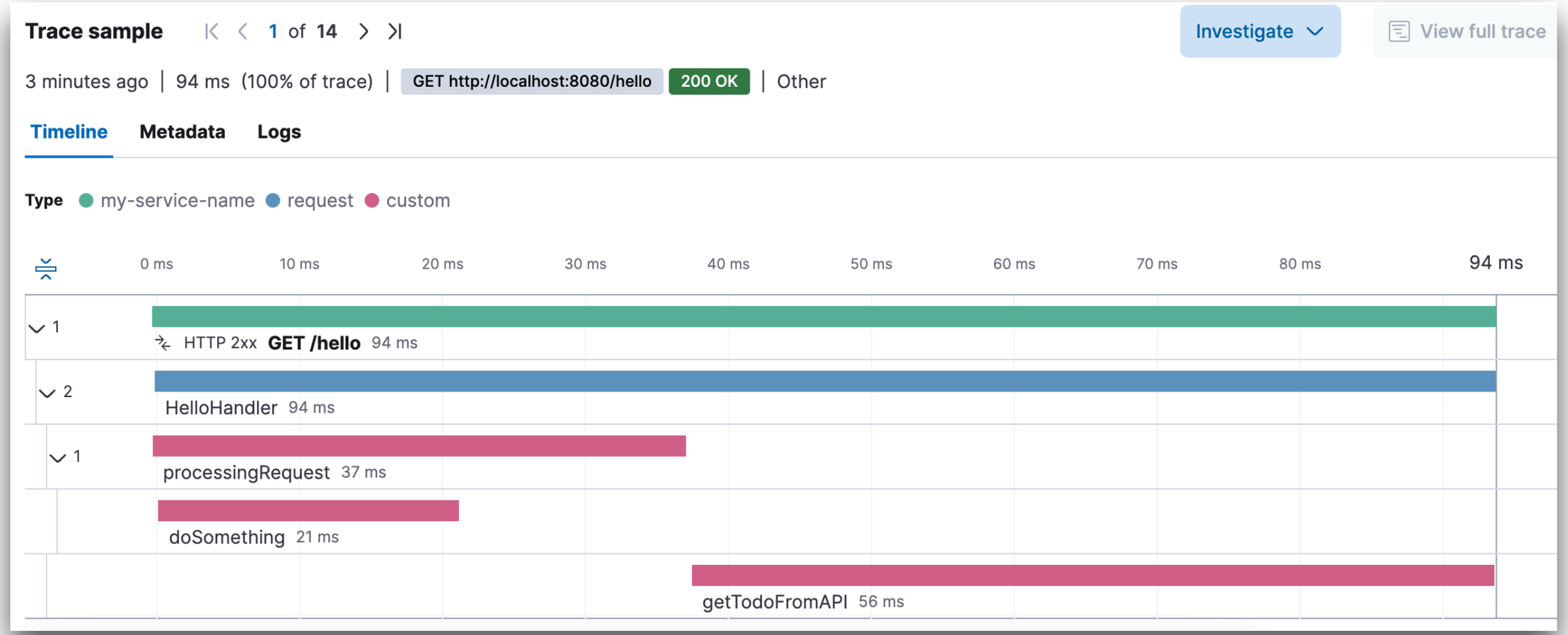
https://grafana.com/oss/faro/

Demo time with ELK + Beats

Structure of Demo



Distributed Tracing



Q/A