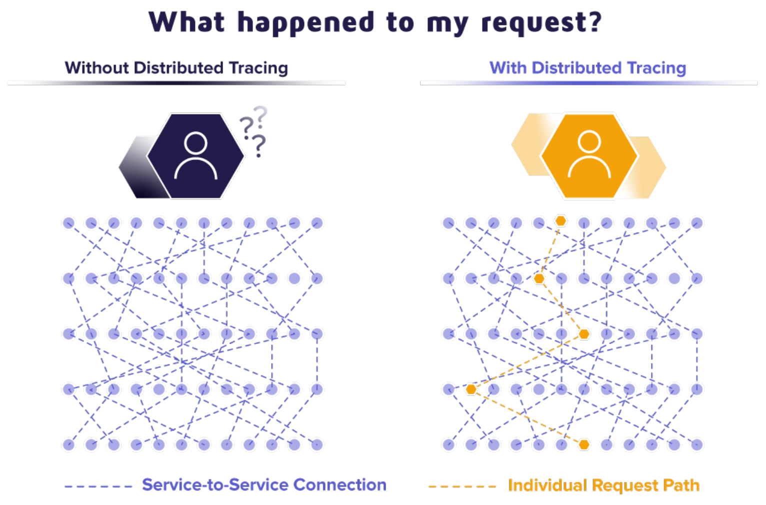
Distributed Tracing in Microservices Application

Contents -

What is Distributed Tracing?

Distributed Tracing is the process of tracing and analyzing every single request from the point of origin up to all the services it touches by analyzing the data. Distributed tracing allows us to track the lifecycle of network calls, one can see how long the request spans across a particular microservice, and also identify performance issues by getting visibility on bottlenecks. Every traces will have a Trace ID, timestamp, and other useful metadata.



What happened to my request?
Without Distributed Tracing
With Distributed Tracing
Service-to-Service Connection
Individual Request Path

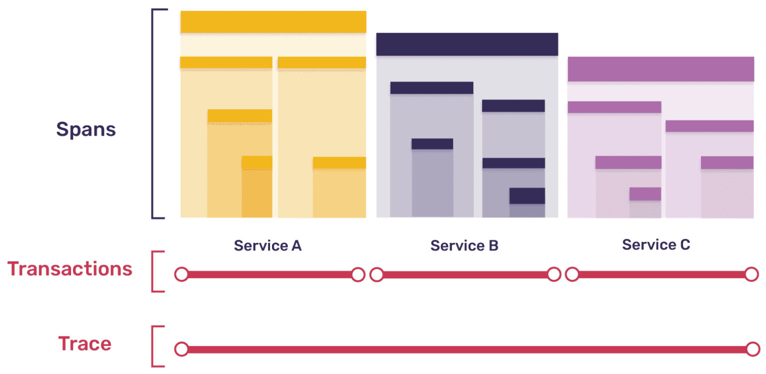
Why do we need Distributed Tracing ?

The requirement for data tracing within a highly distributed system became increasingly obvious with the significant change from monolithic to microservices design. In a microservices architecture, a request made by user may pass through numerous microservices in order to create and provide a response to the user. It is very challenging to identify the specific microservice where a problem arises.

Key Components of Distributed Tracing

The basic terminology or key components of Distributed Tracing is as follows -

Components	Description
1. Request	This denotes how various cloud applications, microservices, and other functions communicate with each other.
2. Trace	A trace represents the record of the entire operation you want to measure or track - like page load, an instance of a user completing some action in your application. When a trace includes work in multiple services, such as those listed above, it's called a distributed trace. A trace can have multiple spans.

3. Span	Each trace consists of one or more tree-like structures called transactions, the nodes of which are called spans. In most cases, each transaction represents a single instance of a service being called, and each span within that transaction represents that service performing a single unit of work, whether calling a function within that service or making a call to a different service. In short, it informs about the work done by a single service with respect to time intervals and corresponding meta-data. Spans are the basic building blocks of trace. Below diagram shows the trace and spans.  Spans Service A Service B Service C Transactions Trace
4. Child span	Spans can be further divided into child spans. Each function or a request can make a call to a child request/function. Below diagram shows the pictorial representation of parent span and child span.
5. Tags	These are the pieces of information i.e metadata associated with each span(recorded along the path) that provide a detailed overview of the actions performed during a span.

Benefits of Distributed Tracing

1. End-to-End Visibility of the user request across the entire system of microservices
2. Provides information about service dependencies
3. Metrics and observability - Like, How long did each request take?
4. Resiliency when the system encounters a failure - Like, Where are the bottlenecks? How much time is lost due to network lag during communication between services?

References -

1. Squadcast - using-distributed-tracing-in-microservices-architecture
2. Sentry - Distributed Tracing
3. https://denuwanhimangahettiarachchi.medium.com/distributed-tracing-in-microservices-4f0b741f07c6#:~:text=Distributed%20tracing%20using%20Spring%20Cloud,with%20few%20lines%20of%20code.
4. https://medium.com/swlh/microservices-observability-with-distributed-tracing-32ae467bb72a
5. https://www.splunk.com/en_us/data-insider/what-is-distributed-tracing.html