How Distributed Tracing Works

The trouble with modern systems has technologies increased growth complexities for monitoring software and systems.

CONTINUOUS DEPLOYMENT  DISTRIBUTED SYSTEMS  THOUSANDS OF MICROSERVICES

ON-PREM  HYBRID  CLOUD / SaaS

New Relic Edge with Infinite Tracing

is a fully managed, cloud-native, tail-based tracing solution that observes every trace and visualizes the most actionable data so you can find and resolve issues faster.

Learn more: newrelic.com/products/edge-infinite-tracing

What is distributed tracing?
Distributed tracing refers to the method of tracking, observing, and collecting data about requests as they flow through distributed systems.

How to troubleshoot faster with New Relic
Troubleshooting is much easier and faster when you can see software requests end-to-end. Follow these steps to understand the behavior and performance of your distributed systems.

The trouble with modern systems
New technologies introduce greater complexity for monitoring software and systems.

How Distributed Tracing Works

1. INSTRUMENT
Auto-instrument your services with New Relic agents, or use open instrumentation, to begin collecting trace data.

2. CONNECT
Utilize the W3C trace context standard to make sure each span can be connected to create complete trace paths.

3. COLLECT
Collect trace telemetry from every source across your system including microservices, containers, serverless functions, messaging queues, service meshes, etc.

4. VISUALIZE
Use New Relic Edge with Infinite Tracing to analyze and visualize your trace data.

What is distributed tracing?
Distributed tracing refers to the method of tracking, observing, and collecting data about requests as they flow through distributed systems.

1. WHAT IS A TRACE?
Services in a distributed system talk to one another by sending requests. A trace is data that tracks the complete path of a request as it travels from service to service. It’s composed of spans that represent time spent in each operation, or segment, along the path.

Types of sampling
Because distributed tracing processes massive amounts of data, it captures and gives you a representative “sample” of activity. Here are the two most common types of sampling:

TRADITIONAL HEAD-BASED SAMPLING
- Traces are sampled randomly, before they are fully executed
- Random samples give sufficient visibility to some systems but can miss traces with errors or high latency
- Works well with a blend of monoliths and microservices

TAIL-BASED SAMPLING
- Data is only collected after the trace has fully completed, and only the traces containing errors or high latency are collected
- Usually requires users to deploy and operate complex tracing infrastructure
- Works well in highly distributed microservices-based systems

How to troubleshoot faster with New Relic
Troubleshooting is much easier and faster when you can see software requests end-to-end. Follow these steps to understand the behavior and performance of your distributed systems.

1. INSTRUMENT
Auto-instrument your services with New Relic agents, or use open instrumentation, to begin collecting trace data.

2. CONNECT
Utilize the W3C trace context standard to make sure each span can be connected to create complete trace paths.

3. COLLECT
Collect trace telemetry from every source across your system including microservices, containers, serverless functions, messaging queues, service meshes, etc.

4. VISUALIZE
Use New Relic Edge with Infinite Tracing to analyze and visualize your trace data.

New Relic Edge with Infinite Tracing is a fully managed, cloud-native, tail-based tracing solution that observes every trace and visualizes the most actionable data so you can find and resolve issues faster.

Learn more: newrelic.com/products/edge-infinite-tracing