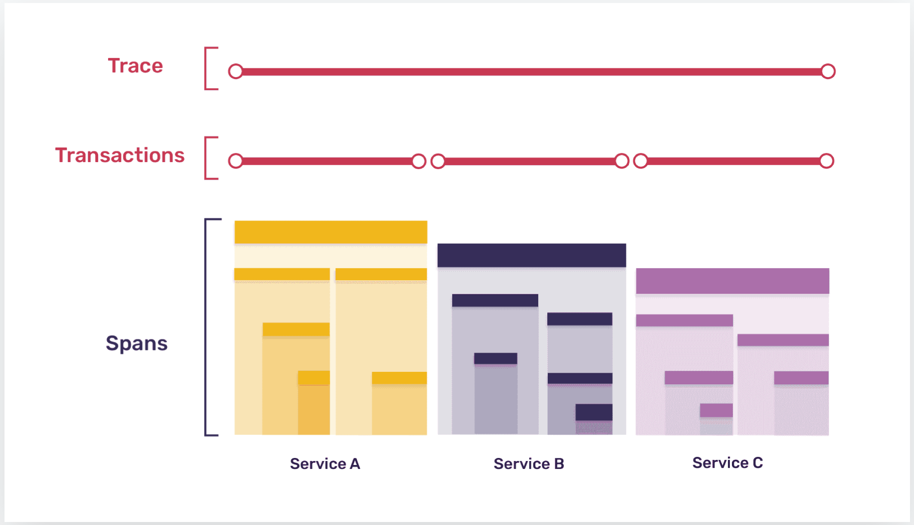
SENTRY

Part 1: Distributed Tracing 101

A single event can cause a domino effect impacting every operation in your application. Relying on logs alone, it can be next to impossible to know if the root cause of an issue in one part of your stack is due to a bug in another part.

Distributed tracing not only tells your team where an error or slowdown is happening, but if it's affecting other parts of your stack.



What Makes Up A Trace

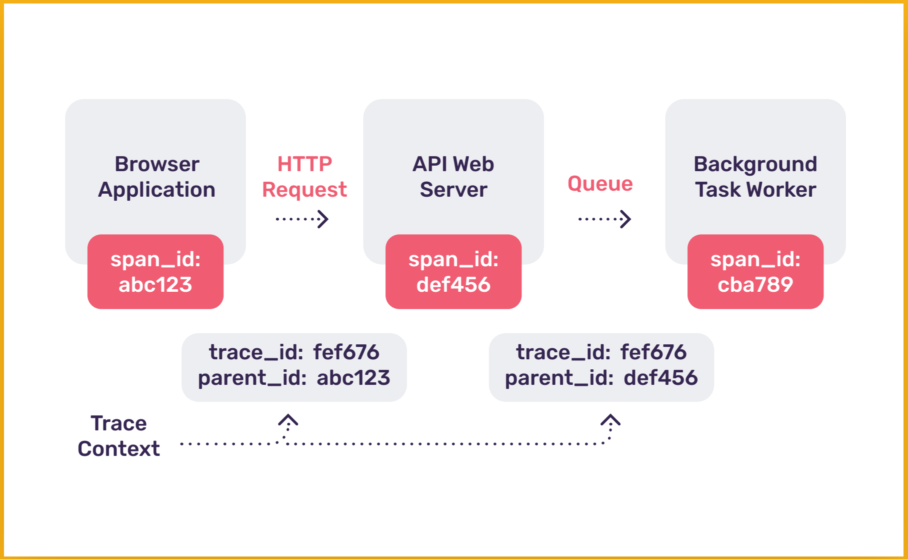
A trace represents a log of events occurring during the execution of a program. Think of it as a chain of operations performed during a specific task. Every trace is uniquely identified by a UUID (Universally Unique Identifier). This trace identifier allows you to differentiate between different traces.

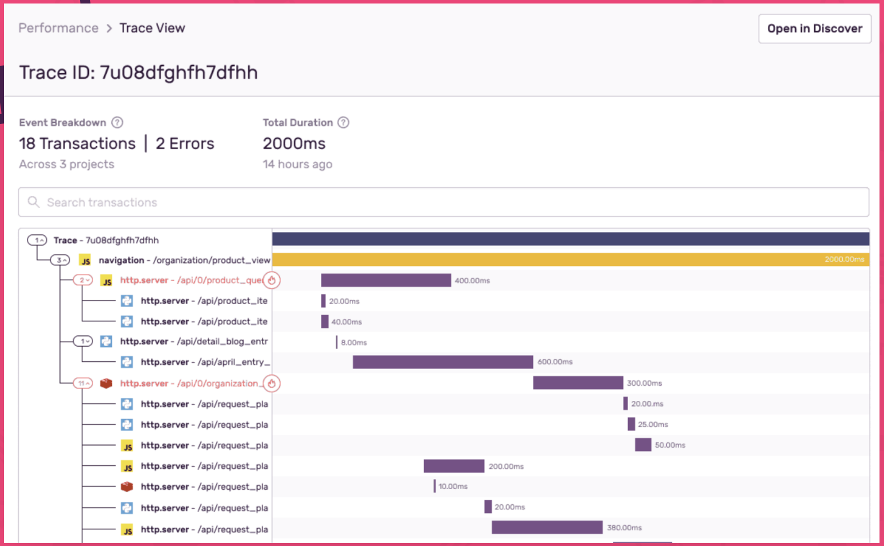
A trace consists of multiple transactions, and each transaction is composed of multiple spans. Spans are the atomic units of work and represent individual resources being loaded, UI component lifecycle events, file I/O operations, and more.

How traces are distributed

For distributed tracing to work, the application must propagate trace context between transactions.

The trace context includes three values: the trace identifier, the parent identifier (which corresponds to the span identifier of the parent span), and the sampled value. The sampled value determines whether a particular transaction should be included in the trace. By passing the trace context, our services know to which trace they should append their transactions and spans.





Trace ID: 7u08dfghfh7dfhh

Event Breakdown
18 Transactions | 2 Errors
Across 3 projects

Total Duration
2000ms
14 hours ago

Search transactions

Trace - 7u08dfghfh7dfhh
navigation - /organization/product_view 2000.00ms
http.server - /api/0/product_que 400.00ms
http.server - /api/product_ite 20.00ms
http.server - /api/product_ite 40.00ms
http.server - /api/detail_blog_entr 8.00ms
http.server - /api/april_entry_ 600.00ms
http.server - /api/0/organization_ 300.00ms
http.server - /api/request_pla 20.00.ms
http.server - /api/request_pla 25.00ms
http.server - /api/request_pla 50.00ms
http.server - /api/request_pla 200.00ms
http.server - /api/request_pla 10.00ms
http.server - /api/request_pla 20.00ms
http.server - /api/request_pla 380.00ms

Visualizing distributed tracing

In addition to errors, distributed tracing also helps you identify performance bottlenecks, because spans also hold values for how long they took. The trace and its spans are visualized using a Gantt chart, so if a span takes longer than you'd expect it, it'll be obvious.

Distributed tracing is a powerful technique for understanding the flow of requests through complex systems. With the right instrumentation and monitoring tools, distributed tracing enables faster debugging and reliable application performance

Want to learn more about distributed tracing at Sentry?

READ MORE

GET STARTED

SENTRY

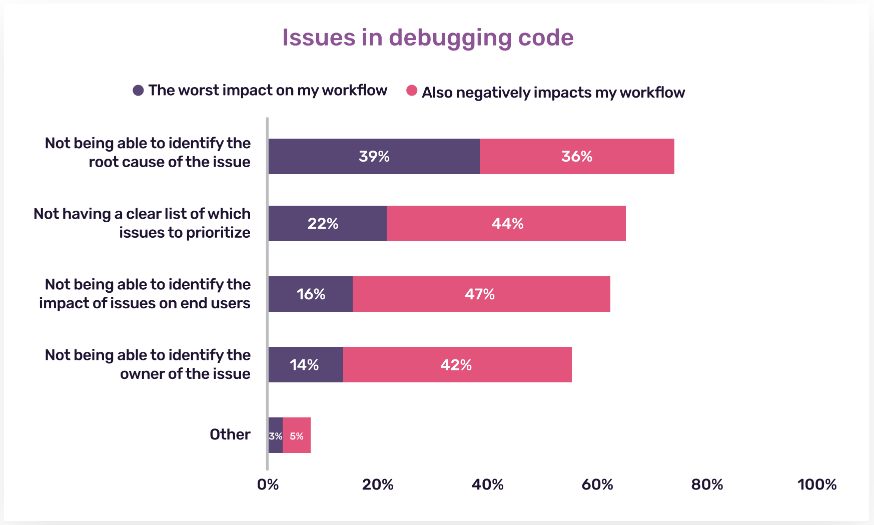
SENTRY

Part 2: Why you need it

Less time debugging

In Sentry's **State of Developer Happiness Survey**, not being able to identify the root cause of an issue has the worst impact on a developer's workflow.

Distributed tracing shows how different projects connect, so your team can identify the source, not the symptom of an issue.



"Sometimes errors on the front-end have roots on the backend. We use Sentry's tags and metadata about a request that comes in to pass along a version of distributed tracing to link these back."

TONY STUCK
ENGINEERING MANAGER AT CLOUDFLARE

CLOUDFLARE

Scaling with microservices

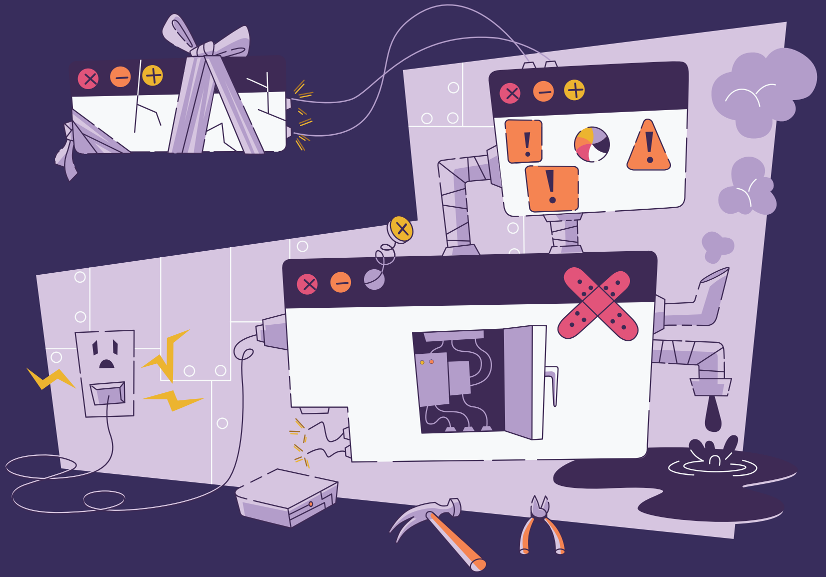
Microservices architecture can give your team the flexibility to scale. In a perfect world, APIs are well-defined, giving you insight into where errors occur in any microservice. Unfortunately, this isn't a perfect world.

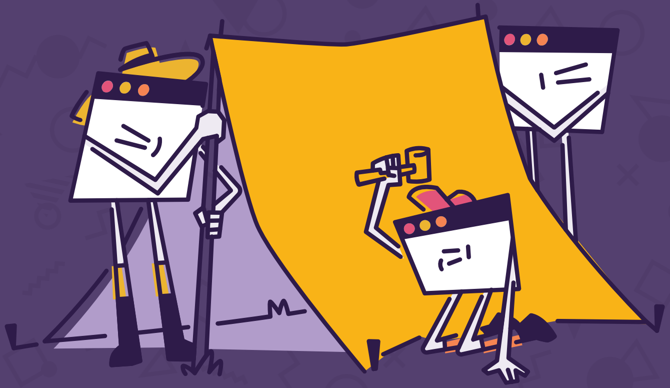
Distributed tracing allows you to pass along unique identifiers and metadata across different services and languages, so you and your team can understand all the service-to-service chatter.

Resource allocation

Know which parts of your app you need to invest in.

Whether you are deciding what needs more or less infrastructure capacity or where your team should spend their time, distributing tracing can help.





Context you and your team need

Distributed tracing gives your team the cross-project insight needed to solve for what's causing errors and slowdowns.

As an engineering manager, distributed tracing helps you identify cross-project and team dependencies. With a single view of how your services interact, you can improve alignment with other teams and dev ops.

Want to learn more about distributed tracing at Sentry?

READ MORE

GET STARTED

SENTRY