new relic

Por que o distributed tracing é essencial para o APM?

Redução do MTTR em ambientes complexos de aplicativos distribuídos

e-book/novembro de 2022

Conteúdo

Reduzir a complexidade

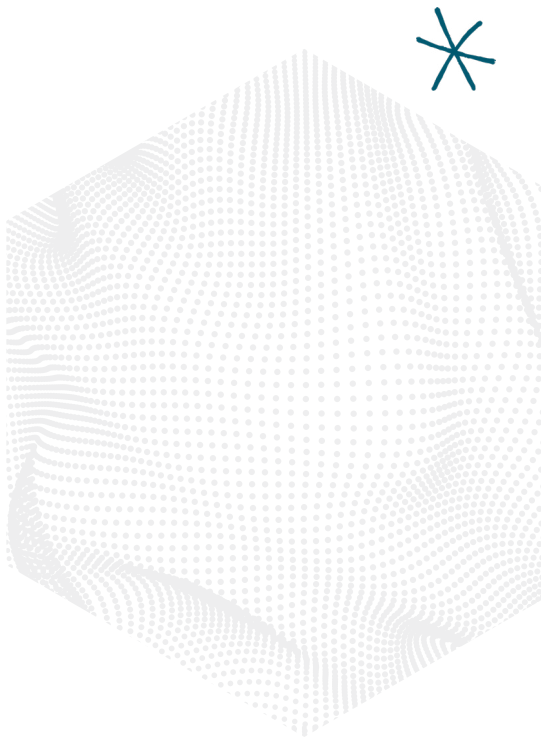
As arquiteturas e os ambientes de software modernos, como os microsserviços, têm o potencial de acelerar o desenvolvimento de aplicativos. Entretanto, em muitas organizações, as equipes de engenharia de software encaram um ambiente complexo, que torna difícil o diagnóstico e a resolução de problemas de desempenho e erros antes que afetem a confiabilidade e a experiência do cliente.

Os ambientes de microsserviços podem incluir de dezenas a centenas de serviços, dificultando a determinação dos caminhos de solicitação e dos problemas de diagnóstico. E o peso do monitoramento do desempenho de aplicativos (APM) apenas aumenta com a orquestração, a automação e o CI/CD para implantações frequentes de software. Sem a instrumentação de monitoramento adequada, as organizações correm o risco de fazer com que suas equipes procurem respostas repetidamente em sistemas distribuídos, o que aumenta o tempo médio de resolução (MTTR) e rouba o tempo de desenvolvimento de softwares inovadores.

A observabilidade acaba com a complexidade do software e fornece visibilidade de ponta a ponta, o que possibilita que as equipes resolvam problemas mais rapidamente, trabalhem de modo mais inteligente e criem experiências digitais melhores para seus clientes. A observabilidade cria contexto e insights acionáveis ao, dentre outras coisas, combinar quatro tipos essenciais de dados de observabilidade: métricas, eventos, logs e traces (MELT).

Os traces, mais precisamente o distributed tracing, são essenciais para as equipes de software que fizeram a transição (ou estão considerando fazer) para a nuvem e adotaram a arquitetura de microsserviços. Isso porque o distributed tracing é o melhor modo de entender rapidamente o que acontece com as solicitações à medida que passam pelos microsserviços que compõem os aplicativos distribuídos.

Líderes de negócios, engenheiros de DevOps, proprietários de produtos, engenheiros de SRE, líderes de equipes de software e outras partes interessadas podem usar o distributed tracing para encontrar gargalos ou erros e obter vantagem com a resolução mais rápida de problemas.



Estabelecer o caminho pelos sistemas distribuídos

O distributed tracing agora é uma grande aposta para ambientes de aplicativos modernos de monitoramento e operação. Quando as equipes monitoram o desempenho do sistema e do software quanto à observabilidade, o trace é um modo de monitorar e analisar solicitações à medida que elas se propagam por um ambiente distribuído e pulam de um serviço para outro.

O distributed tracing é a capacidade de acompanhar uma solução para rastrear e observar as solicitações de serviço à medida que elas fluem pelos sistemas distribuídos, coletando dados enquanto as solicitações vão de um serviço para outro. Os dados de trace ajudam as equipes a entender o fluxo de solicitações pelo ambiente de microsserviços e encontrar onde ocorrem falhas ou problemas de desempenho no sistema, além do motivo.

Quando as equipes instrumentam sistemas para distributed tracing, todas as transações geram telemetria de trace, do usuário de frontend até as chamadas de banco de dados de backend. Por exemplo, quando os clientes clicam em um carrinho para fazer uma compra em um aplicativo de e-commerce, essa solicitação passa por vários serviços distintos de backend e frontend, por muitos contêiners, ambientes serverless, máquinas virtuais, diferentes provedores de nuvem, fluxos locais (on-prem) ou qualquer combinação deles. A solicitação pode incluir o serviço de inventário para garantir que há inventário disponível, serviço de pagamento e serviço de entrega. E, por fim, a solicitação é concluída e volta para o usuário. Sempre que uma solicitação vai de um serviço para outro, ela emite um span com telemetria de trace. Assim que a solicitação é concluída, os spans são reunidos para criar um trace completo da jornada da solicitação pelo sistema.

Com o distributed tracing, as equipes podem:

- Rastrear o caminho de uma solicitação enquanto ela passa por um sistema complexo.
- Entender as dependências de serviços upstream e downstream.
- Descobrir a latência de componentes ao longo do caminho.
- Entender onde ocorrem os gargalos no caminho da solicitação.
- Ver e analisar onde os erros acontecem na transação no nível individual do serviço.

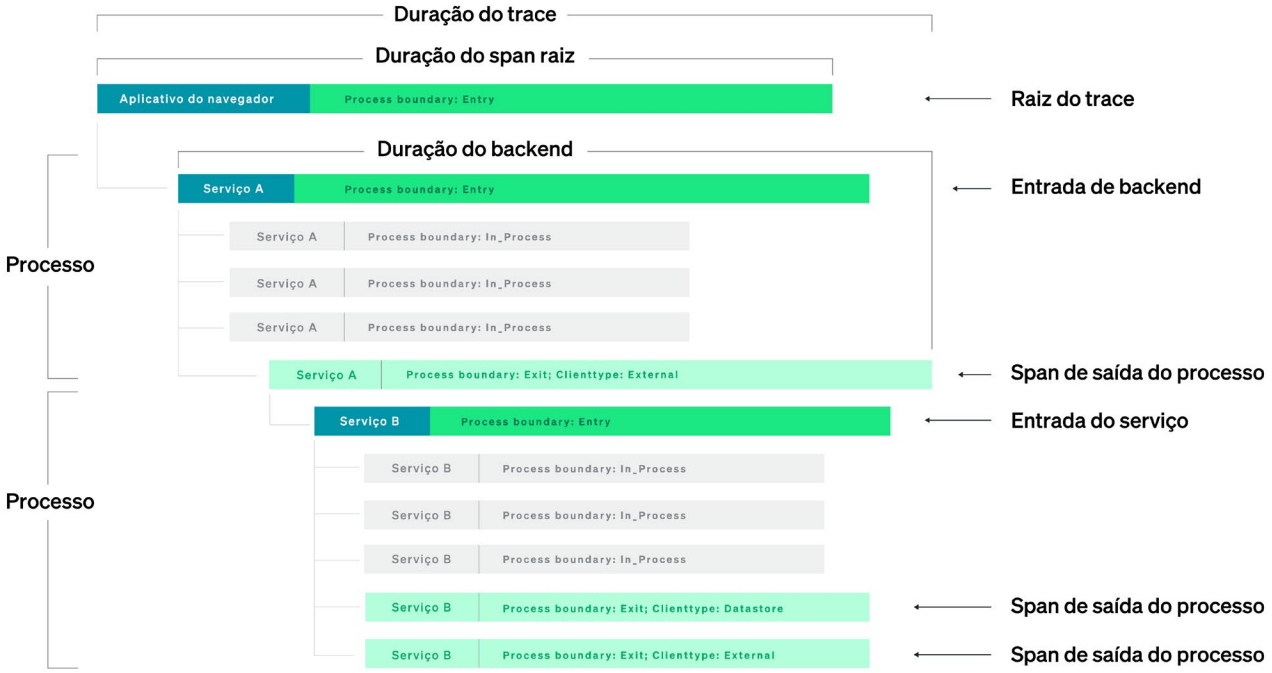
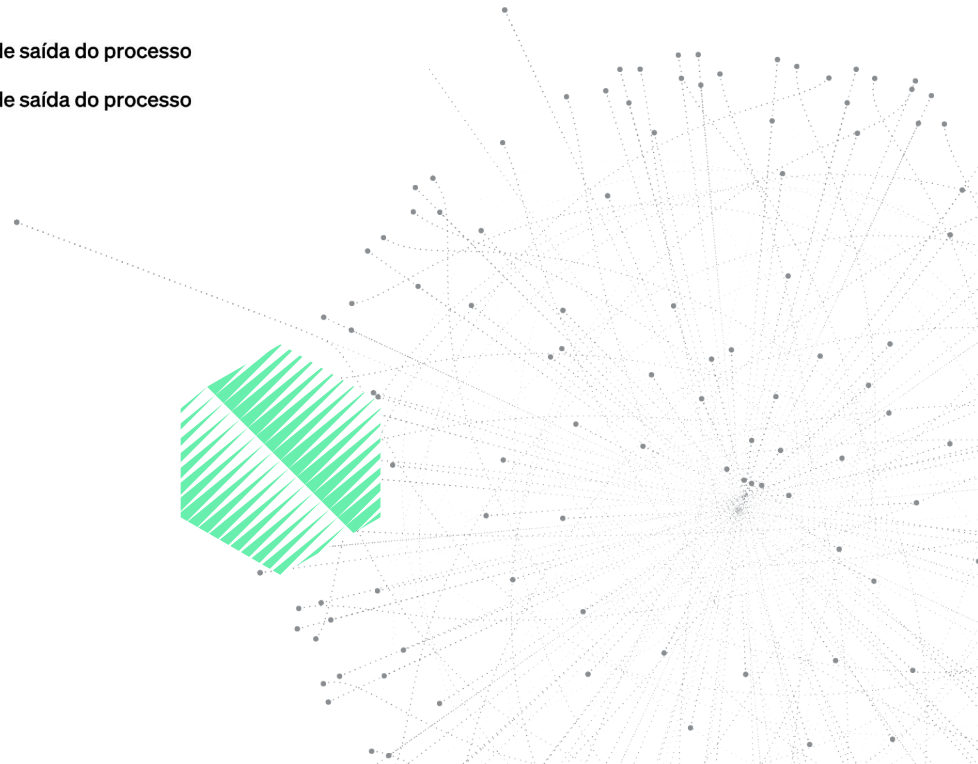


Gráfico de dispersão e visualização em cascata mostrando quanto tempo cada solicitação levou em cada etapa pelos serviços do aplicativo



Quando usar traces

Em geral, o distributed tracing é o melhor modo para as equipes de DevOps, operações, software e SRE obterem respostas para perguntas específicas rapidamente, em ambientes nos quais o software é distribuído ou depende de arquiteturas serverless. Quando uma solicitação envolve vários microsserviços, ter uma maneira de ver como todos os diferentes serviços estão funcionando juntos é essencial.

Os dados de trace fornecem contexto sobre o que está acontecendo no aplicativo como um todo e entre os serviços e as entidades. Se houvesse apenas eventos brutos para cada serviço isolado, não seria possível reconstruir uma única cadeia entre serviços para uma transação específica.

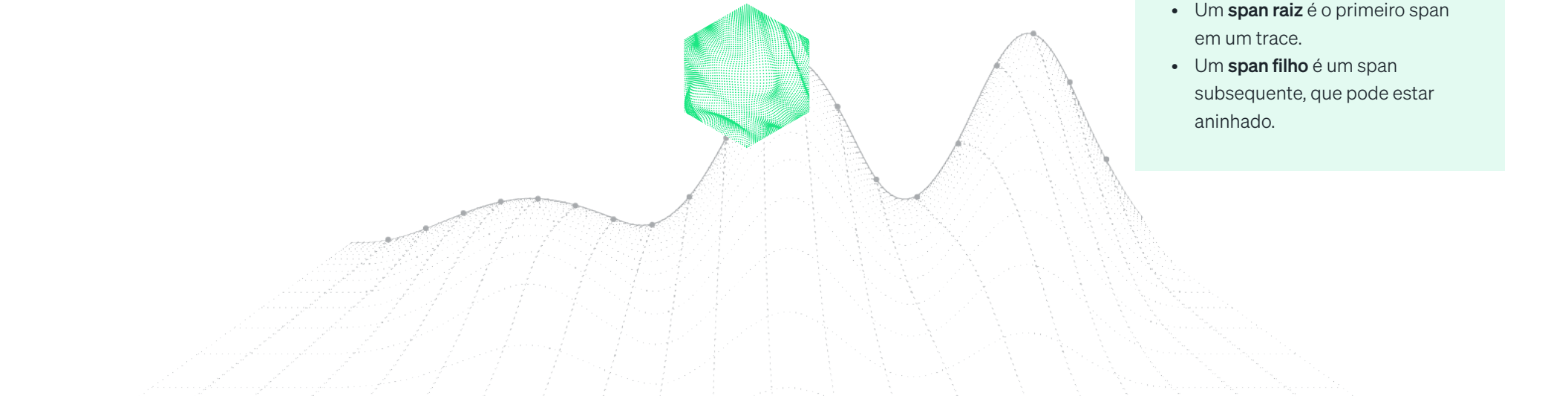
Muitas vezes, os aplicativos chamam diversos outros dependendo da tarefa que estão tentando realizar. Eles também costumam processar dados paralelamente. Então a cadeia de chamados pode ser inconsistente e os tempos podem ser duvidosos para realizar uma correlação. A única maneira de garantir uma cadeia de chamados consistente é passar o contexto do trace entre cada serviço para identificar exclusivamente uma única transação em toda a cadeia.

Isso significa que as equipes devem usar o distributed tracing para obter respostas para perguntas como:

- Como está a saúde dos serviços que compõem um sistema distribuído?
- Qual é a causa raiz dos erros e dos defeitos em um sistema distribuído?
- Quais são os gargalos de desempenho que poderiam afetar a experiência do cliente?
- Quais serviços apresentam código ineficiente ou problemático e que devem ser priorizados para otimização pelas equipes?

Guia rápido sobre a terminologia relacionada ao distributed tracing:

- Uma **transação** é formada pelas chamadas de método e função que compõem a unidade de trabalho em um aplicativo de software. Ela começa quando o método é chamado e termina quando o método retorna ou dá erro.
- Uma **solicitação** é como os aplicativos, os microsserviços e as funções se comunicam umas com as outras.
- Um **trace** é composto pelos dados de desempenho sobre as solicitações à medida que elas passam pelos microsserviços.
- Um **span** representa operações ou segmentos que fazem parte de um trace.
- Um **span raiz** é o primeiro span em um trace.
- Um **span filho** é um span subsequente, que pode estar aninhado.



Como os traces funcionam

A união de traces forma eventos especiais chamados "spans". Os spans ajudam a rastrear uma cadeia casual através de um ecossistema de microsserviços para uma única transação. Para isso, cada serviço passa identificadores de correlação, conhecidos como "contexto do trace", um para o outro. Esse contexto do trace é usado para adicionar atributos ao span.

Timestamp	EventType	TraceID	SpanID	ParentID	ServiceID	Duração
08/11/2022 15:34:23	Span	2ec68b32	aaa111	bbb111	Máquina de venda automática	23
08/11/2022 15:34:22	Span	2ec68b32	bbb111	aaa111	Backend da máquina de venda automática	18
08/11/2022 15:34:20	Span	2ec68b32	ccc111	bbb111	Operadora de cartão de crédito	15
08/11/2022 15:34:19	Span	2ec68b32	ddd111	ccc111	Banco emissor	3

Exemplo de um distributed tracing composto por spans em uma transação com cartão de crédito

Na tabela acima, o timestamp e os dados de duração mostram que a operadora de cartão de crédito tem o serviço mais lento na transação com 12 de 23 segundos, mais da metade do tempo para todo o trace.

Como chegamos a 12 segundos?
O span para entrar em contato com o banco emissor é o que chamamos de span filho. O span para entrar em contato com a empresa de cartão de crédito é o span pai. Então, se a solicitação do banco levou 3 segundos e a da empresa de cartão de crédito 15 segundos, e nós subtrairmos o filho do pai, vemos que demorou 12 segundos para processar a transação com cartão de crédito.

Conectar os pontos

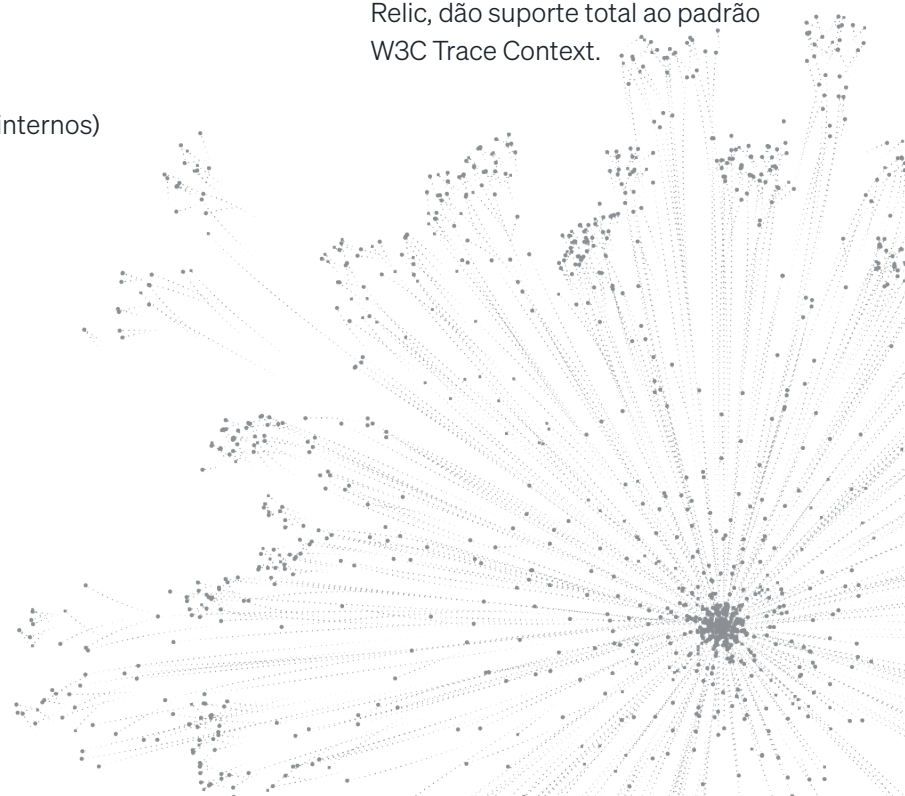
Assim que as organizações começaram a migrar para aplicativos distribuídos, elas rapidamente perceberam que precisavam de um modo para ter visibilidade dos microsserviços individuais isoladamente e em todo o fluxo de solicitação. Essa migração é o motivo pelo qual o distributed tracing se tornou uma prática recomendada para obter a visibilidade necessária sobre o que estava acontecendo. E combinar os traces com outros três tipos essenciais de dados de telemetria — métricas, eventos e logs — proporciona às equipes uma visão completa do ambiente de software e do desempenho da observabilidade total.

O distributed tracing também exige contexto de trace. Esse requisito significa atribuir uma ID única a cada solicitação, atribuir uma ID única a cada etapa em um trace, codificar essa informação contextual e passar (ou propagar) o contexto codificado de um serviço para o próximo à medida que a solicitação passa por um ambiente de aplicativo. Esse processo permite que a ferramenta de distributed tracing correlacione cada etapa de um trace na ordem correta, juntamente com outras informações necessárias para monitorar e acompanhar o desempenho.

Normalmente, um único trace captura dados sobre:

- Spans (nome do serviço, nome da operação, duração e outros metadados)
- Erros
- Duração de operações importantes em cada serviço (como funções e chamadas de métodos internos)
- Atributos personalizados

O W3C Trace Context se tornou o padrão para propagação de contexto de trace nos limites do processo. Ele permite que todos os tracers e agentes em conformidade com o padrão participem de um trace, com os dados de trace propagados de um serviço raiz para o serviço terminal. Muitos provedores de observabilidade, incluindo a New Relic, dão suporte total ao padrão W3C Trace Context.



Por que as organizações precisam de distributed tracing?

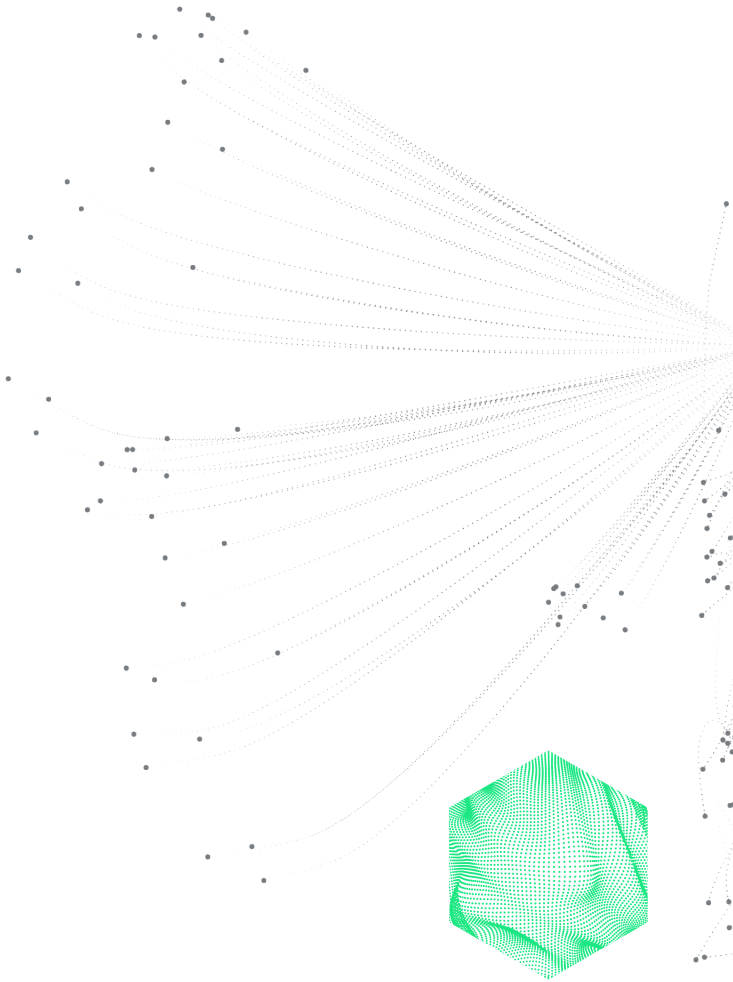
Conforme novas tecnologias e práticas — como nuvem, microsserviços, contêiners, funções serverless, DevOps, engenharia de confiabilidade de sites e mais — aceleram a velocidade e diminuem a dificuldade de criar um software do código à produção, elas também apresentam novos desafios:

- Mais pontos de falha no stack do aplicativo
- MTTR maior devido à complexidade do ambiente do aplicativo
- Menos tempo para as equipes inovarem, pois precisam de mais tempo para diagnóstico de problemas

Por exemplo, uma solicitação lenta pode afetar a experiência de um grupo de clientes. Essa solicitação é distribuída por vários microsserviços e funções serverless. Várias equipes têm e monitoram muitos serviços envolvidos na solicitação, e nenhuma relatou problemas de desempenho nos microsserviços. Sem um modo de exibir o desempenho de toda a solicitação em diferentes serviços, é quase impossível saber onde e por que a alta latência está ocorrendo e qual equipe deve lidar com o problema. Como parte de uma estratégia de observabilidade total, o distributed tracing aborda os desafios dos ambientes de aplicativos modernos.

Ao entender profundamente o desempenho de todos os serviços, tanto upstream quanto downstream, as equipes de software podem realizar as seguintes tarefas de modo mais eficaz e rápido:

- Identificar e resolver problemas para minimizar o impacto nos resultados de negócios e na experiência do cliente.
- Mensurar a saúde geral do sistema e entender o efeito das mudanças na experiência do cliente.
- Priorizar as áreas de alto valor para melhoria, de modo a otimizar as experiências digitais dos clientes.
- Inovar continuamente com confiança para ter um desempenho melhor que o da concorrência.



Obter visibilidade no pipeline de dados

O distributed tracing exige relatório e processamento da telemetria de trace. O volume dos dados de trace pode crescer exponencialmente ao longo do tempo, visto que o volume das solicitações aumenta conforme as equipes implantam mais microsserviços no ambiente.

Por esse motivo, muitas organizações usam amostragem de dados para gerenciar a complexidade e os custos associados à transmissão da atividade de trace. De modo ideal, os dados com amostragem representam as características da população de dados maior.

As equipes de software precisam de flexibilidade para escolher amostragem tail-based ou head-based para atender aos requisitos de monitoramento para cada aplicativo.

Amostragem head-based eficiente

A amostragem head-based coleta e armazena dados de trace aleatoriamente enquanto o span raiz (o primeiro) é processado para rastrear e analisar o que acontece com a transação em todos os serviços pelos quais ele passa. Normalmente, a amostragem head-based acontece no agente responsável pela coleta da telemetria de trace ao selecionar aleatoriamente quais traces devem ter amostras coletadas para análise. As decisões de amostragem acontecem antes da conclusão dos traces. Como não há como saber qual trace pode ter um problema, as equipes podem deixar passar traces com erros ou processos lentos não usuais.

A amostragem head-based funciona bem para fornecer uma amostragem estatística geral das solicitações por um sistema distribuído. Ela pega traces com erros ou latência em aplicativos com um volume menor de transações e ambientes com uma mistura de arquitetura monolítica e arquitetura baseada em microsserviços. A amostragem head-based é um modo eficiente de coletar amostra de uma grande quantidade de trace de dados em tempo real. Além disso, há pouco a nenhum impacto no desempenho do aplicativo.

Vantagens da amostragem head-based

- Funciona bem para aplicativos com menor rendimento de transação
- Execução rápida e fácil
- Apropriada para ambientes mistos de microsserviços e monolíticos, nos quais o ambiente monolítico ainda é o principal
- Pouco a nenhum impacto no desempenho do aplicativo
- Uma solução econômica para enviar dados de trace para provedores de terceiros
- A amostragem estatística fornece transparência adequada no sistema distribuído

Limitações da amostragem head-based

- As amostras de trace são coletadas aleatoriamente
- A amostragem acontece antes de o trace concluir completamente seu caminho por muitos serviços, então não há como saber com antecedência qual trace poderá encontrar um problema
- Em sistemas com alta taxa de transferência, os traces com erros ou latência não usual podem ficar de fora da amostra

Traces acionáveis com amostragem tail-based

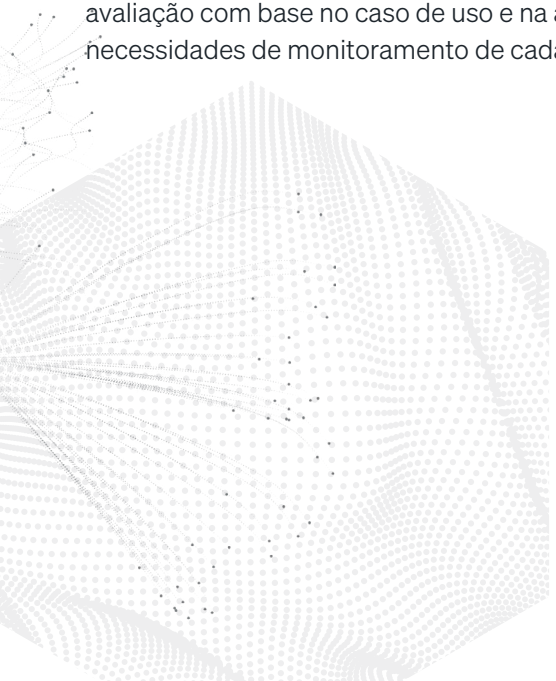
O distributed tracing com amostragem tail-based ajuda as equipes de software a resolver problemas em sistemas altamente distribuídos e com grande volume, nos quais as equipes devem observar toda a telemetria de trace e coletar amostras dos traces com erros ou latência não usual. A amostragem tail-based coleta todas as informações sobre o trace quando ele é concluído.

A amostragem tail-based não é apenas um recurso interessante, mas um requisito quando as equipes precisam do mais alto nível de granularidade para resolução de problemas.

Algumas organizações precisam da ferramenta de distributed tracing para observar e analisar todos os spans — todos os saltos entre os serviços — e trazer à tona os traces mais acionáveis para resolução de problemas, pois o período de inatividade pode custar milhões de dólares, especialmente durante eventos de pico.

Por exemplo, uma organização com uma carga média de span de três milhões de spans por minuto vê picos de 300 milhões de spans por minuto quando lança um novo produto. A amostragem head-based tradicional é inadequada para esse tipo de organização com alto volume de transação.

Nem todo trace é igual. Para escolher o melhor método de amostragem, as equipes devem fazer uma avaliação com base no caso de uso e na análise de custo em relação aos benefícios, além de considerar as necessidades de monitoramento de cada aplicativo.

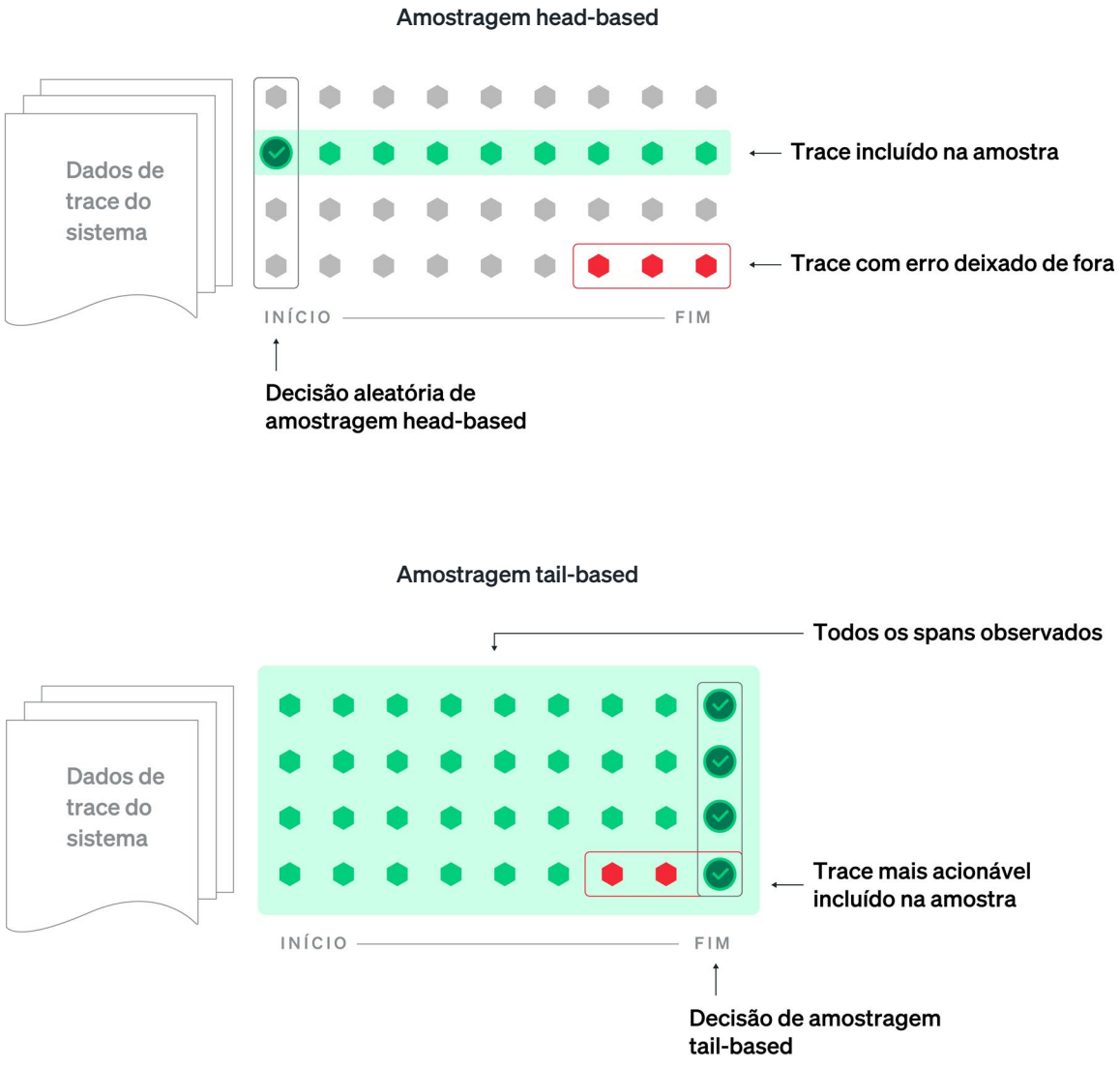


Vantagens da amostragem tail-based

- Observa e analisa todos os traces
- Coleta amostras depois que os traces foram totalmente concluídos
- Visualiza traces com erros ou lentidão não usual com mais rapidez

Limitações da amostragem tail-based

- Pode exigir que satélites, proxies e gateways adicionais executem software de amostragem
- Exige um pouco de esforço para gerenciar e dimensionar softwares de terceiros em alguns casos
- Incorre custos adicionais para transmitir e armazenar mais dados



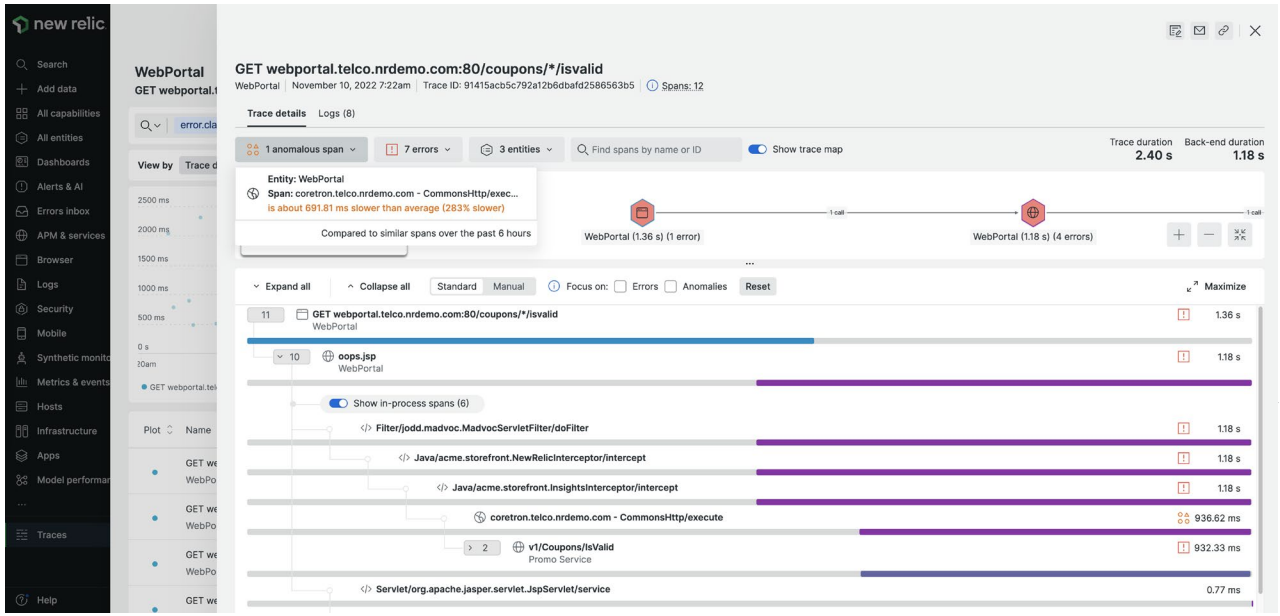
Amostragem head-based tradicional (acima) e amostragem tail-based (abaixo)

Análise e visualização

A coleta de dados de trace será uma perda de tempo se as equipes de software não tiverem um modo fácil de analisar e visualizar os dados em arquiteturas complexas. Uma plataforma de observabilidade abrangente permite que as equipes vejam todos os dados de negócios e telemetria em um só local. Ela também fornece o contexto necessário para obter significado, tomar a medida certa rapidamente e trabalhar com os dados de maneiras significativas.

Idealmente, uma visualização de distributed tracing tem uma estrutura em árvore. A visualização deve incluir spans filho que fazem referência a um span pai, e permite que as equipes vejam quais spans têm alta latência e erros em um trace. Isso também ajuda as equipes a entenderem os detalhes exatos dos erros, bem como quais serviços estão lentos, com atributos detalhados para encontrar problemas e corrigi-los rapidamente.

Provedores de observabilidade como a New Relic usam essa estrutura de visualização para resolução de problemas e análise.



Distributed tracing da New Relic

Lidar com o peso do gerenciamento

Resolver problemas de sistemas distribuídos é uma situação clássica de encontrar a agulha no palheiro, e instrumentar sistemas para tracing, coleta e visualização de dados pode exigir muito trabalho e ser complexo de implementar. As soluções de software como serviço (SaaS) totalmente gerenciadas permitem que as equipes eliminem o peso de implementar, gerenciar e dimensionar satélites ou gateways de terceiros para coleta de dados.

A plataforma de observabilidade da New Relic facilita a instrumentação de aplicativos com uma única implantação de agente para praticamente qualquer framework e linguagem de programação. As equipes também podem usar ferramentas de código aberto e padrões de instrumentação aberta para instrumentar os ambientes. O OpenTelemetry é considerado o padrão para coleta de telemetria e instrumentação de código aberto.

A plataforma da New Relic também oferece um serviço de amostragem tail-based totalmente gerenciado que observa e analisa todos os spans em um sistema distribuído e, além disso, fornece exibições de traces com erros ou latência não usual para que as equipes possam identificar e resolver problemas rapidamente.

A plataforma observa todos os spans e fornece métricas, dados de erros e traces essenciais em uma única visualização. Ela fornece insights críticos ao salvar a maioria dos dados acionáveis na plataforma da New Relic. O resultado é visibilidade inigualável nos sistemas distribuídos, possibilitando que as equipes entendam o impacto da latência downstream ou dos erros com métricas detalhadas e, em seguida, detalhem os dados de trace salvos para os traces mais relevantes.

O distributed tracing está incluso no New Relic APM com transferência de dados de baixo custo e baixa latência de agentes New Relic, instrumentação com funções serverless ou qualquer outra fonte de dados, incluindo instrumentação de terceiros.

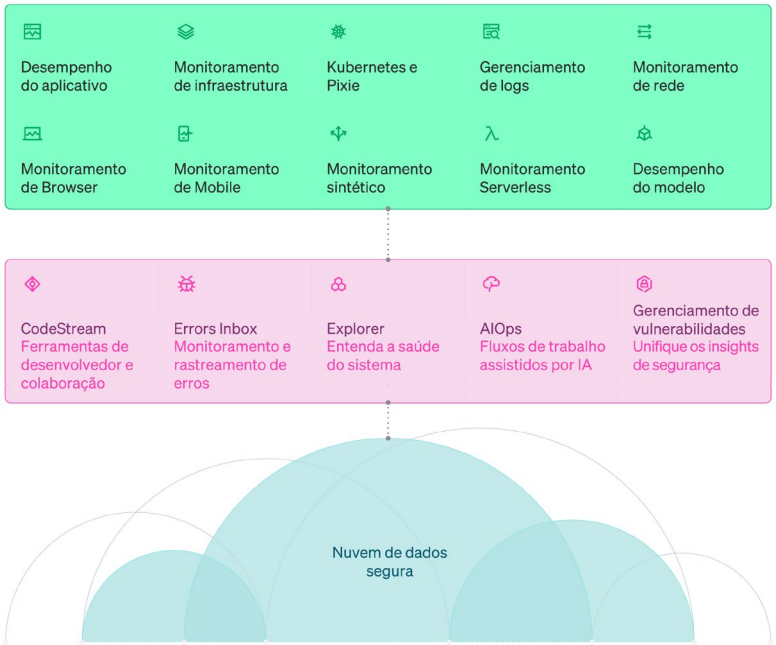
Com o New Relic, você pode:

- Aproveitar um serviço local de nuvem totalmente gerenciado que dimensiona sob demanda.
- Observar e analisar todos os traces nos sistemas distribuídos.
- Visualizar a maioria dos traces acionáveis que contêm erros ou latência não usual.
- Eliminar o esforço de implantar, gerenciar, dar suporte e dimensionar satélites ou gateways de terceiros nos ambientes.
- Aproveitar todo o suporte dos padrões e da instrumentação abertos para telemetria de trace.
- Reduzir os custos de egresso de dados de locais próximos para cargas de trabalho na nuvem.
- Resolver problemas de modo mais eficaz.
- Reduzir o tempo médio de detecção (MTTD) e o MTTR com traces acionáveis e de alta fidelidade.
- Capacitar engenheiros e desenvolvedores para que foquem trabalhos mais importantes, como o desenvolvimento de novos recursos.

Cara ou coroa? Você não precisa jogar uma moeda

A New Relic oferece opções flexíveis para distributed tracing de modo que as equipes possam tomar decisões de amostragem head ou tail-based no nível do aplicativo. Para aplicativos críticos nos quais as equipes precisam observar e analisar todos os traces, é possível selecionar a amostragem tail-based sem preocupações com o gerenciamento da infraestrutura de amostragem.

A New Relic é o único provedor de observabilidade que proporciona às equipes de software flexibilidade para selecionar distributed tracing com amostragem head-based ou amostragem tail-based totalmente gerenciada. Com menos gerenciamento, há mais oportunidades para inovar e ganhar vantagem em relação à concorrência.



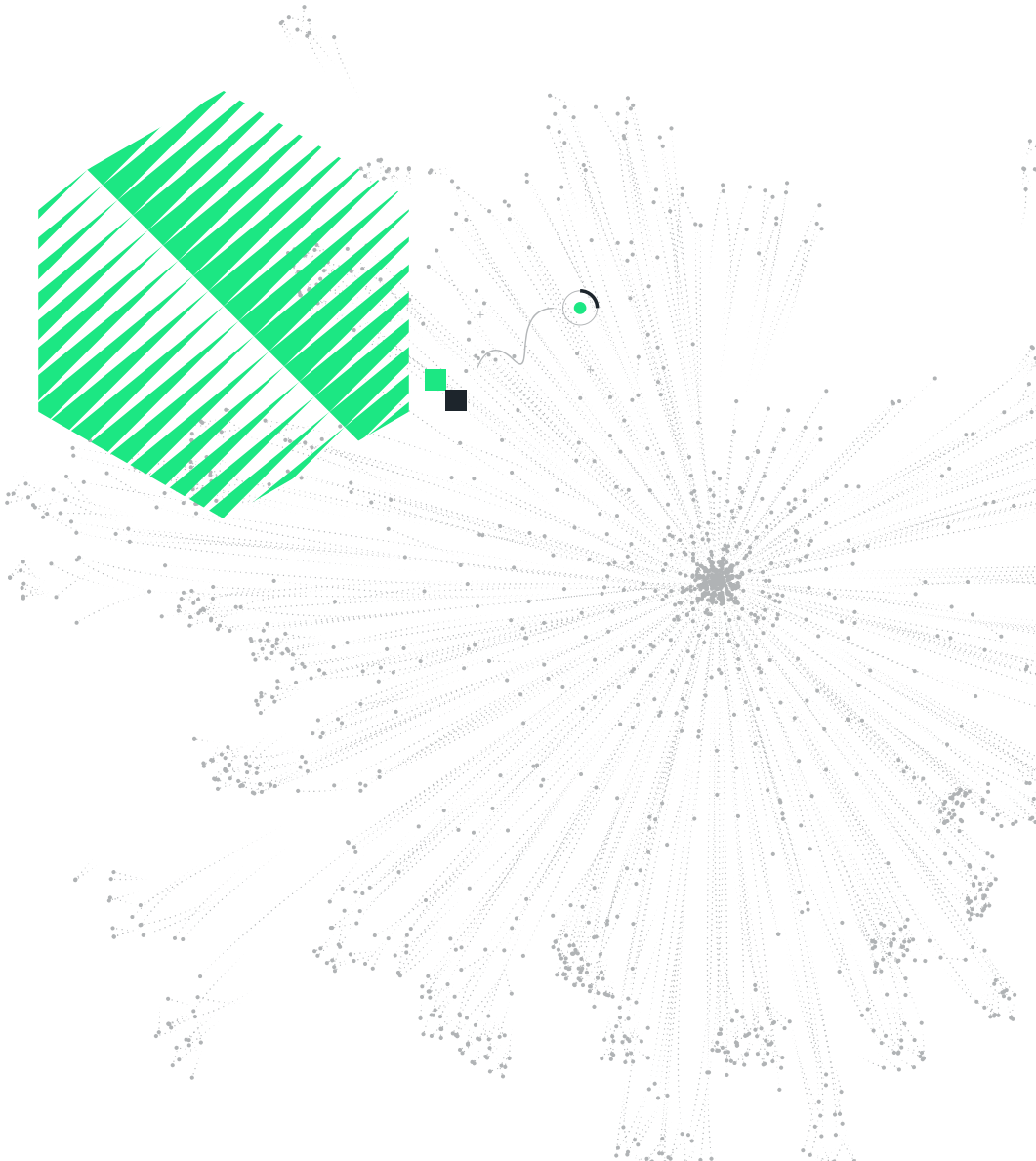
A plataforma de observabilidade da New Relic incorpora gerenciamento de logs, APM, distributed tracing, monitoramento de infraestrutura, monitoramento Serverless, monitoramento de Mobile, monitoramento de Browser, monitoramento sintético, monitoramento de Kubernetes e mais.

Próximos passos

Para começar a usar o New Relic APM com o distributed tracing, cadastre-se em uma conta gratuita hoje mesmo. As contas gratuitas incluem 100 GB/mês de ingestão de dados, um usuário Full Platform e usuários Basic ilimitados.

Já tem uma conta na New Relic? Começar a aproveitar o distributed tracing do New Relic APM é fácil, basta usar nosso agente APM mais recente. Saiba mais sobre as opções de configuração de distributed tracing.

Comece agora



Sobre a New Relic

Como líder em observabilidade, a New Relic capacita os engenheiros com uma abordagem orientada por dados para planejamento, construção, implantação e execução de grandes softwares. A New Relic oferece a única plataforma de dados unificada com toda a telemetria — métricas, eventos, logs e traces — combinada com poderosas ferramentas de análise full-stack para ajudar os engenheiros a trabalharem melhor com dados, não com opiniões.

A primeira do setor a trabalhar com preços baseados no uso de maneira intuitiva e previsível, a New Relic oferece aos engenheiros muito mais valor, ajudando a melhorar os tempos de ciclo de planejamento, as taxas de falha de alteração, a frequência de lançamento e o MTTR. Isso ajuda startups em hipercrescimento e marcas líderes do mundo todo a melhorar o tempo de operação e a confiabilidade, impulsionar a eficiência operacional e fornecer experiências excepcionais aos clientes enquanto estimulam a inovação e o crescimento.

new relic.