Entwicklerhandbuch

# AWS X-Ray

# AWS X-Ray: Entwicklerhandbuch

# Table of Contents

# Was ist AWS X-Ray?

AWS X-Ray bietet Ablaufverfolgungsinformationen zu allen empfangenen Antworten und Aufrufen, die eine instrumentierte Anwendung tätigt, einschließlich der folgenden Informationen:

- Nachgeschaltete Ressourcen AWS
- Microservices
- Datenbanken
- Web-APIs

Verwenden Sie Trace-Daten und Visualisierungen, um Einblicke in die Leistung Ihrer Anwendung zu gewinnen, Probleme zu identifizieren und Optimierungsmöglichkeiten zu finden. Verwenden Sie die Analysetools in X-Ray, um Details zu jeder verfolgten Anfrage an Ihre Anwendung anzuzeigen, zu filtern und zu untersuchen.

## So funktioniert X-Ray

Um X-Ray verwenden zu können, müssen Sie zuerst Ihre Anwendung instrumentieren, damit X-Ray verfolgen kann, wie Ihre Anwendung eine Anfrage verarbeitet. Durch Hinzufügen von Instrumentierung zu Ihrer Anwendung kann X-Ray Trace-Daten und Metadaten für eingehende und ausgehende Anfragen und andere Ereignisse innerhalb Ihrer Anwendung senden. Sie können beispielsweise alle eingehenden HTTP-Anfragen und Downstream-Aufrufe Ihrer Java-Anwendung instrumentieren. AWS-Services Sie können Ihre Anwendung auch automatisch instrumentieren. Weitere Informationen finden Sie unter Instrumentierung Ihrer Anwendung.

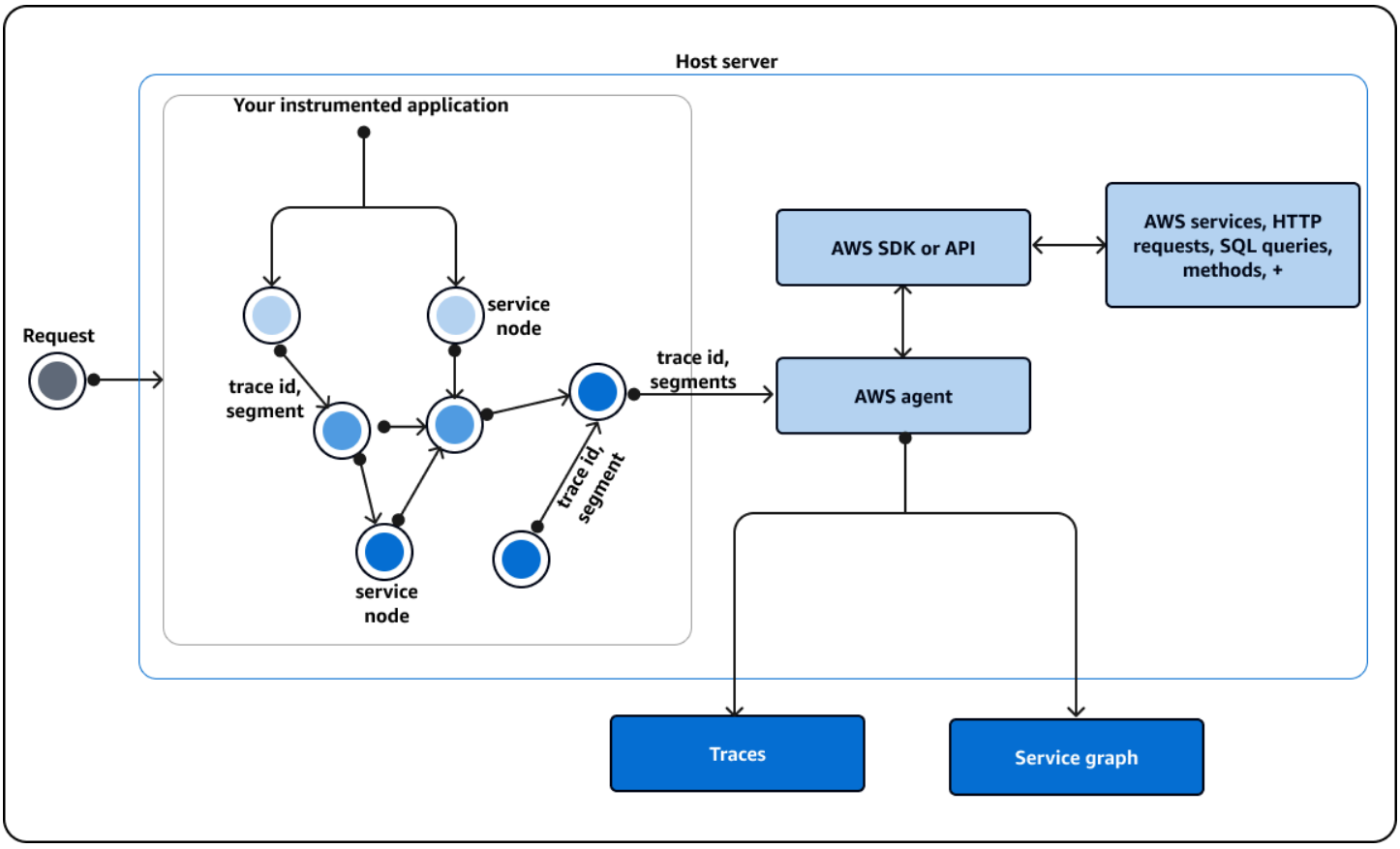
X-Ray weist jeder Anfrage, die Ihre instrumentierte Anwendung empfängt, eine Trace-ID zu. Wenn Ihre Anwendung mit einer anderen Komponente interagiert, erstellt X-Ray ein Segment. Dieses Segment ist mit der ursprünglichen Trace-ID verknüpft und verfolgt die Qualität der Interaktion mit dieser Komponente.

X-Ray verfolgt die Trace-ID und die Segmente während Ihres gesamten Anwendungsworkflows. Sie können den gesamten Arbeitsablauf analysieren oder einzelne Teile für eine detaillierte Analyse isolieren. Weitere Informationen zu Segmenten finden Sie im folgenden Konzepte Abschnitt.

X-Ray verfolgt Ihre Anwendung bei der Interaktion mit Serviceknoten oder Komponenten, um eingehende Anfragen wie folgt zu bearbeiten:

1. X-Ray verwendet eine Trace-ID und Segmente, um einzelne Interaktionen zu verfolgen.
2. Ein AWS Agent sammelt die Trace-ID und die zugehörigen Segmente und leitet sie dann an ein SDK- oder API-Trace-Framework weiter.
3. X-Ray verfolgt auch Interaktionen mit allen AWS Diensten, die in X-Ray integriert sind.
4. Der Agent sendet Daten an eine grafische Benutzeroberfläche der Konsole, auf der Sie Informationen über Ihre Traces, Segmente und Untersegmente sowie darüber, wie diese Komponenten interagieren, einsehen können.

Die vorherigen Schritte sind in der folgenden Abbildung dargestellt:




## Wie X-Ray mit Ihrer instrumentierten Anwendung interagiert

Wenn Ihre instrumentierte Anwendung eine Anfrage erhält, geht X-Ray wie folgt vor:

1. Nachdem Ihre Anwendung die Anfrage bearbeitet hat, sendet das X-Ray SDK Trace-Daten an einen AWS Collector oder Agenten. Anschließend erfasst der Agent die Trace-ID und die Segmente. Sie können aus den folgenden drei Agenten wählen:

- AWS Distribution for OpenTelemetry (ADOT) Collector — Ein Open-Source-Collector, der optimiert und gesichert ist und auf einem standardisierten Open-Source-Agenten basiert. AWS OpenTelemetry Verwenden Sie den, ADOT Collector wenn Sie Sprache und herstellerunabhängigen standardisierten Code verwenden möchten, um mit einem Agenten zu interagieren, aber dennoch das Vertrauen in die AWS Sicherheit und Optimierung haben möchten, die in das Endprodukt integriert sind. Sie können es auch verwenden, ADOT um einen Endpunkt für verschiedene Agenten und Backends zu konfigurieren.
- CloudWatchAmazon-Agent — Ein Open-Source-Collector, der Protokolle, Metriken und Traces integriert, alle Telemetriedaten unterstützt und diese ADOT Collector integriert hat.
- X-Ray-Daemon — Ein Collector, der mit dem X-Ray SDK und den X-Ray-APIs arbeitet. Verwenden Sie den X-Ray-Daemon, wenn Sie älteren Code haben oder eine Anwendung haben, die eine benutzerdefinierte Ablaufverfolgung erfordert und daher die X-Ray-APIs verwenden muss. Der Daemon ist fürLinux, und Microsoft WindowsmacOS, verfügbar und auf AWS Elastic Beanstalk den Plattformen und enthalten. AWS Lambda

2. Anschließend sendet der Agent diese Daten an ein Tracing-Framework, das entweder aus einer AWS API oder einem AWS SDK besteht, das auf einer API aufbaut. Dieses Framework interagiert mit anderen AWS Diensten. Die X-Ray-API bietet Zugriff auf alle X-Ray-Funktionen über das AWS SDK oder direkt überHTTPS. AWS Command Line Interface Verwenden Sie die X-Ray-API, wenn Sie eine Sprache verwenden oder eine Operation benötigen, die nicht von einem SDK unterstützt wird.

   Sie können die folgenden SDKs verwenden:

   - Das ADOT SDK — Verwenden Sie das ADOT SDK, um mit verschiedenen Agenten von Anbietern zu interagieren, die nicht mit AWS diesem Unternehmen verbunden sind. Das ADOT SDK unterstützt auch mehrere Backend-Dienste.
   - Das X-Ray SDK — Ein klassisches Produkt, das keine weiteren Funktionen oder Sprachen mehr hinzufügt. Verwenden Sie das X-Ray SDK, wenn Sie Ihren Anwendungscode nicht aktualisieren möchten.

   Wenn Sie ein X-Ray oder ein ADOT SDK verwenden, haben Sie in Kombination mit einem Agenten die folgenden Optionen:

   - Verwenden Sie X-Ray oder ADOT SDK mit einem CloudWatch Agenten — empfohlen.

- Verwenden Sie das ADOT SDK mit einem ADOT Collector — empfohlen, wenn Sie herstellerunabhängige Software mit Sicherheits- und AWS Optimierungsebenen verwenden möchten.
- Verwenden Sie das X-Ray SDK mit einem CloudWatch Agenten — Der CloudWatch Agent ist mit dem X-Ray SDK kompatibel.
- Verwenden Sie das X-Ray-SDK mit dem X-Ray-Daemon — Verwenden Sie dies, wenn Sie das X-Ray-SDK weiterhin verwenden möchten.

3. (Optional) Das Tracing-Framework kann mit anderen AWS Diensten, HTTP Servern, anderen Methoden und Abfragen interagieren. Einige AWS Dienste, die in X-Ray integriert werden können, umfassen Amazon EC2, Amazon SNS und API Gateway. Das SDK oder die API verfolgt die Trace-Daten während dieser Interaktionen.

   AWS Dienste, die in X-Ray integriert sind, können eingehenden Anfragen Tracing-Header hinzufügen, Trace-Daten an X-Ray senden oder einen Agenten zum Sammeln von Trace-Daten ausführen. AWS Lambda Kann beispielsweise Trace-Daten über Anfragen an Ihre Lambda-Funktionen senden.

   Weitere Informationen zu anderen Diensten, die mit X-Ray funktionieren, finden Sie unterIntegrieren Sie AWS X-Ray mit anderen AWS-Services.

4. Sie können in der Konsole Daten zu Ihren Traces, Segmenten und Untersegmenten in einer grafischen Benutzeroberfläche (GUI) anzeigen. Sie können die folgenden Optionen verwenden:

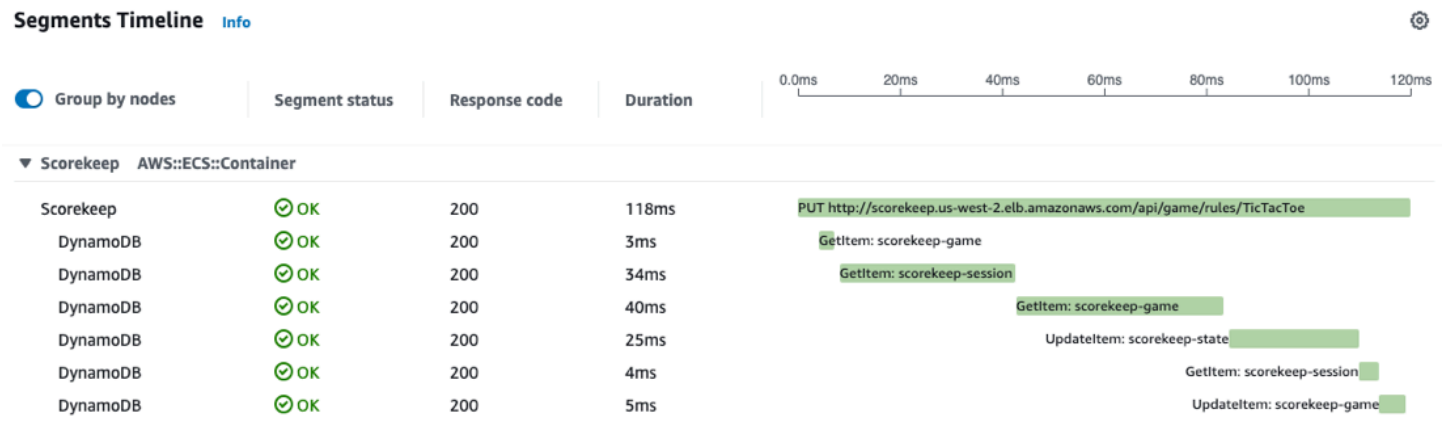
   - https://console.aws.amazon.com/cloudwatch/ — Eine grafische Benutzeroberfläche, mit der Sie Traces, Logs und Metriken an einem zentralen Ort einsehen können. Die X-Ray-Servicemaps und die CloudWatch ServiceLens Legacy-Map werden in der X-Ray-Trace-Map in der CloudWatch Konsole kombiniert.
   - https://console.aws.amazon.com/xray/home — Eine grafische Benutzeroberfläche, in der Sie Informationen zu Ihren Traces einsehen können. Sie können sich Informationen anzeigen lassen, die Einblicke in Ihre Traces, eine Trace-Map, eine Service Map und Analysen beinhalten. AWS entwickelt dieses Konsolenerlebnis nicht mehr.

X-Ray verwendet Trace-Daten von AWS Ressourcen, mit denen Ihre Anwendung interagiert, um eine detaillierte Trace-Map zu erstellen. Die Trace-Map zeigt den Client, Ihren Frontend-Dienst und die Back-End-Dienste, die Ihr Frontend-Dienst aufruft, in einer einzigen Anfrage. Verwenden Sie die Trace-Map, um Engpässe, Latenzspitzen und andere Probleme zu identifizieren und so die Leistung Ihrer Anwendungen zu lösen oder zu verbessern.

X-Ray generiert außerdem eine Service-Map, die einen Gesamtüberblick darüber bietet, wie Ihre Anwendung mit Ihren Serviceknoten interagiert. Kanten in der Service Map verbinden die Serviceknoten. Sie zeigen, wie oft die Knoten miteinander kommunizieren und wie lange die Kommunikation dauert.

Die folgende Abbildung zeigt ein Beispiel für eine Service Map, die zeigt, wie Ihre Anwendung mit verschiedenen Komponenten interagiert. Sie können eine Service Map in der Konsole anzeigen. Das Bild zeigt eine Anwendung, die eine Anfrage von einem Client empfängt. Anschließend zeigt das Bild, wie die Anwendung mit zwei DynamoDB-Tabellen und Amazon SNS interagiert.


Die folgende Abbildung zeigt ein Beispiel für die Daten, die in der Konsole für ein einzelnes Segment in einer Ablaufverfolgung verfügbar sind. Das Bild zeigt eine Zeitleiste, in der mehrere Segmente sowie die Startzeit und Dauer der einzelnen Segmente im Verhältnis zu den anderen aufgeführt sind. Das Bild zeigt auch den Segmentstatus und den HTTP-Antwortcode.




# Konzepte

AWS X-Ray empfängt Daten von Diensten als Segmente. X-Ray gruppiert dann Segmente, die eine gemeinsame Anforderung haben, in Traces. X-Ray verarbeitet die Traces, um ein Service-Diagramm zu erstellen, das eine visuelle Darstellung Ihrer Anwendung bietet.

Konzepte

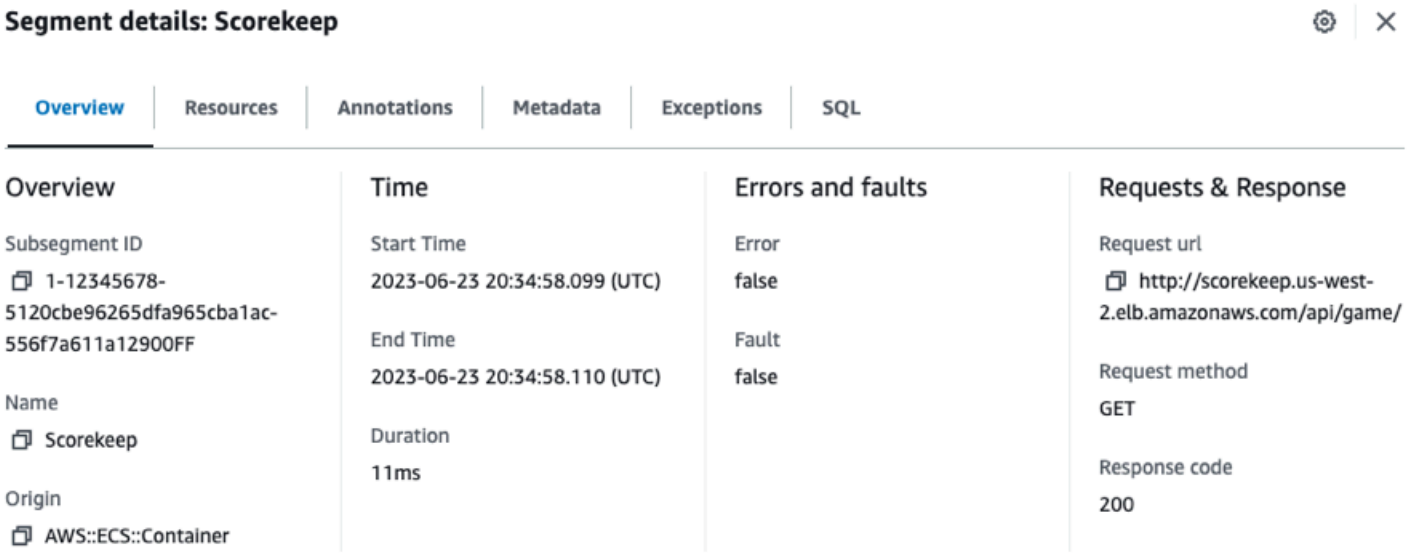
- Segmente
- Untersegmente
- Service-Diagramm
- Ablaufverfolgungen
- Sampling
- Ablaufverfolgungs-Header
- Filterausdrücke
- Gruppen
- Anmerkungen und Metadaten
- Fehler und Ausnahmen

## Segmente

Die Datenverarbeitungsressourcen, die Ihre Anwendungslogik ausführen, senden Daten zu ihrer Arbeit als Segment. Ein Segment enthält den Namen der Ressource, Details zu der Anforderung und Details zu den erledigten Aufgaben. Wenn zum Beispiel eine HTTP-Anforderung Ihre Anwendung erreicht, können Daten über Folgendes erfasst werden:

- Der Host — Hostname, Alias oder IP-Adresse.
- Die Anfrage — Methode, Client-Adresse, Pfad, Benutzeragent.
- Die Antwort — Status, Inhalt.
- Die geleistete Arbeit — Start- und Endzeiten, Untersegmente.
- Auftretende Probleme — Fehler, Störungen und Ausnahmen, einschließlich der automatischen Erfassung von Ausnahmestapeln.

Die folgende Abbildung zeigt ein Beispiel für Übersichtsinformationen, die zu einem Segment zurückgegeben wurden. Das Bild zeigt Informationen über eine ID, Start- und Endzeit, etwaige Fehler oder Störungen sowie den Anfrage- und Antwortcode einer HTTP-Anfrage:




Das Tracing-Framework, das aus SDKs oder APIs besteht, sammelt Informationen aus Anfrage- und Antwort-Headern, dem Code in Ihrer Anwendung und Metadaten über die AWS Ressourcen, auf denen Ihre Anwendung ausgeführt wird. Sie entscheiden, welche Daten X-Ray sammelt, indem Sie Ihre Anwendungskonfiguration oder Ihren Code ändern, um eingehende Anfragen, nachgelagerte Anfragen und AWS Dienste zu instrumentieren.

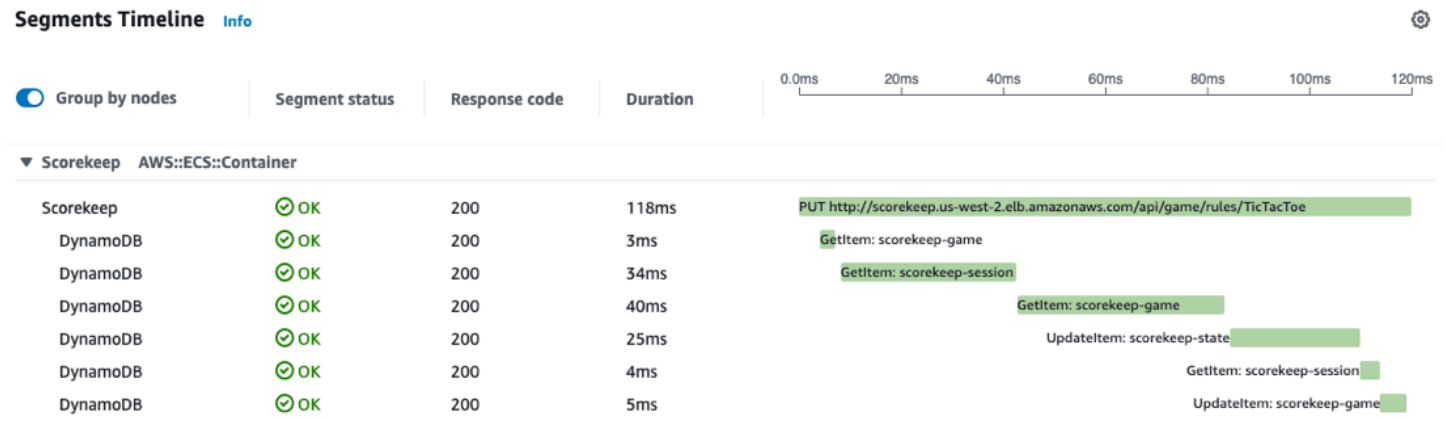
> **Weitergeleitete Anfragen**
>
> Wenn ein Load Balancer oder ein anderer Vermittler eine Anfrage an Ihre Anwendung weiterleitet, nimmt X-Ray die Client-IP aus dem `X-Forwarded-For` Header in der Anfrage und nicht aus der Quell-IP im IP-Paket. Die Client-IP, die für eine weitergeleitete Anfrage

aufgezeichnet wird, kann gefälscht sein und sollte daher nicht als vertrauenswürdig eingestuft werden.

Sie können ein Tracing-Framework wie ein SDK oder eine API verwenden, um weitere Informationen, einschließlich Anmerkungen und Metadaten, aufzuzeichnen. Einzelheiten zur Struktur von Segmenten und Untersegmenten sowie zu aufgezeichneten Informationen finden Sie unter. Dokumente für Röntgensegmente Segmentdokumente können bis zu 64 kB groß sein.

## Untersegmente

Sie können ein Segment in Untersegmente unterteilen. Untersegmente bieten detailliertere Zeitinformationen und Details zu Downstream-Aufrufen, die Ihre Anwendung zur Bearbeitung der ursprünglichen Anfrage durchführt. Ein Untersegment enthält zusätzliche Details zu einem Aufruf einer AWS-Service, einer externen HTTP-API oder einer SQL-Datenbank. Sie können auch Untersegmente definieren, um bestimmte Funktionen oder Codezeilen in Ihrer Anwendung zu instrumentieren.



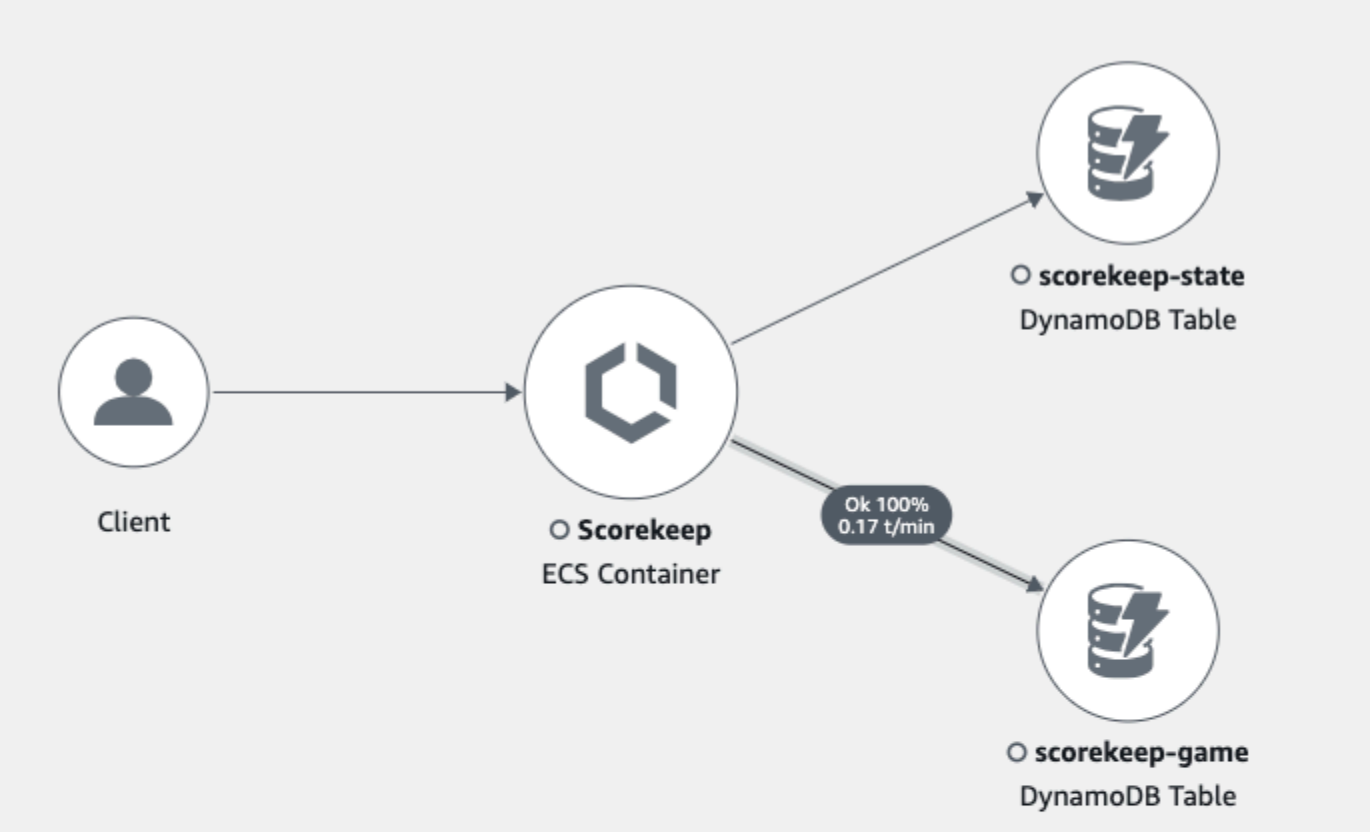

X-Ray verwendet Untersegmente, um abgeleitete Segmente und Downstream-Knoten auf der Trace-Map für Services zu generieren, die keine eigenen Segmente senden, wie Amazon DynamoDB. Mithilfe von Untersegmenten können Sie alle Ihre Downstream-Abhängigkeiten sehen, auch wenn die Abhängigkeiten das Tracing nicht unterstützen oder extern sind. AWS

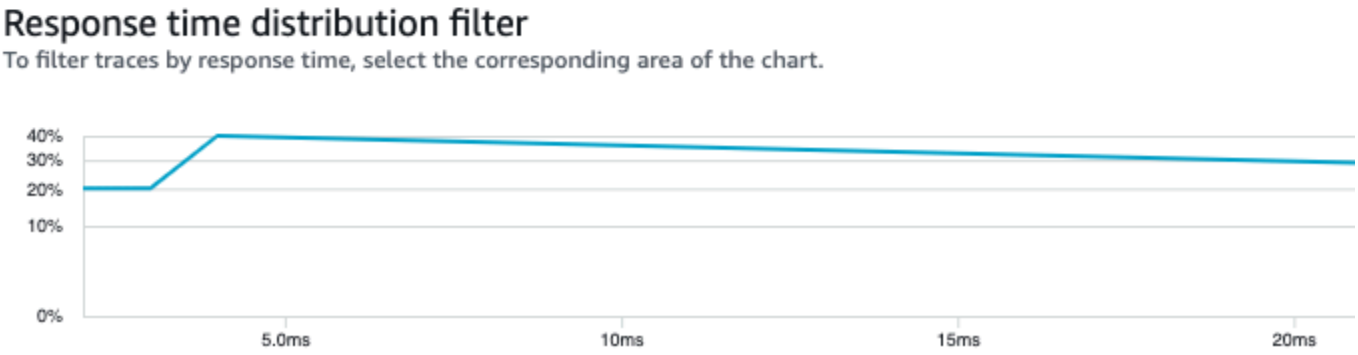
Untersegmente geben die Ansicht eines nachgelagerten Aufrufs Ihrer Anwendung als Client wieder. Wenn der Downstream-Service ebenfalls instrumentiert ist, ersetzt sein Segment das abgeleitete Segment aus dem Untersegment des Upstream-Clients. Der Knoten im Service-Graph verwendet Informationen aus dem Segment des Services, sofern verfügbar. Die Kante zwischen den beiden Knoten verwendet das Untersegment des Upstream-Dienstes.

Wenn Sie DynamoDB beispielsweise mit einem instrumentierten AWS SDK-Client aufrufen, zeichnet das X-Ray-SDK ein Untersegment für diesen Aufruf auf. DynamoDB sendet kein Segment, daher enthält das Untersegment Informationen zu folgenden Themen:

- Das abgeleitete Segment in der Ablaufverfolgung.
- Der DynamoDB; -Knoten im Service-Graph.
- Der Vorteil zwischen Ihrem Service und DynamoDB.

Das folgende Diagramm zeigt die Service-Map für eine Beispielanwendung. In der Abbildung stellt der Client eine Anfrage an eine Scorekeep-Beispielanwendung. Die Scorekeep-Anwendung ruft DynamoDB zweimal auf. Eine Kante in der Service Map steht für jeden dieser Aufrufe. Wählen Sie einen Edge aus, um den Integritätsstatus, die Anzahl und die Häufigkeit von Aufrufen an eine DynamoDB-Tabelle zu sehen. Die folgende Abbildung zeigt Spuren, die einer nach Reaktionszeit gefilterten Kante entsprechen.

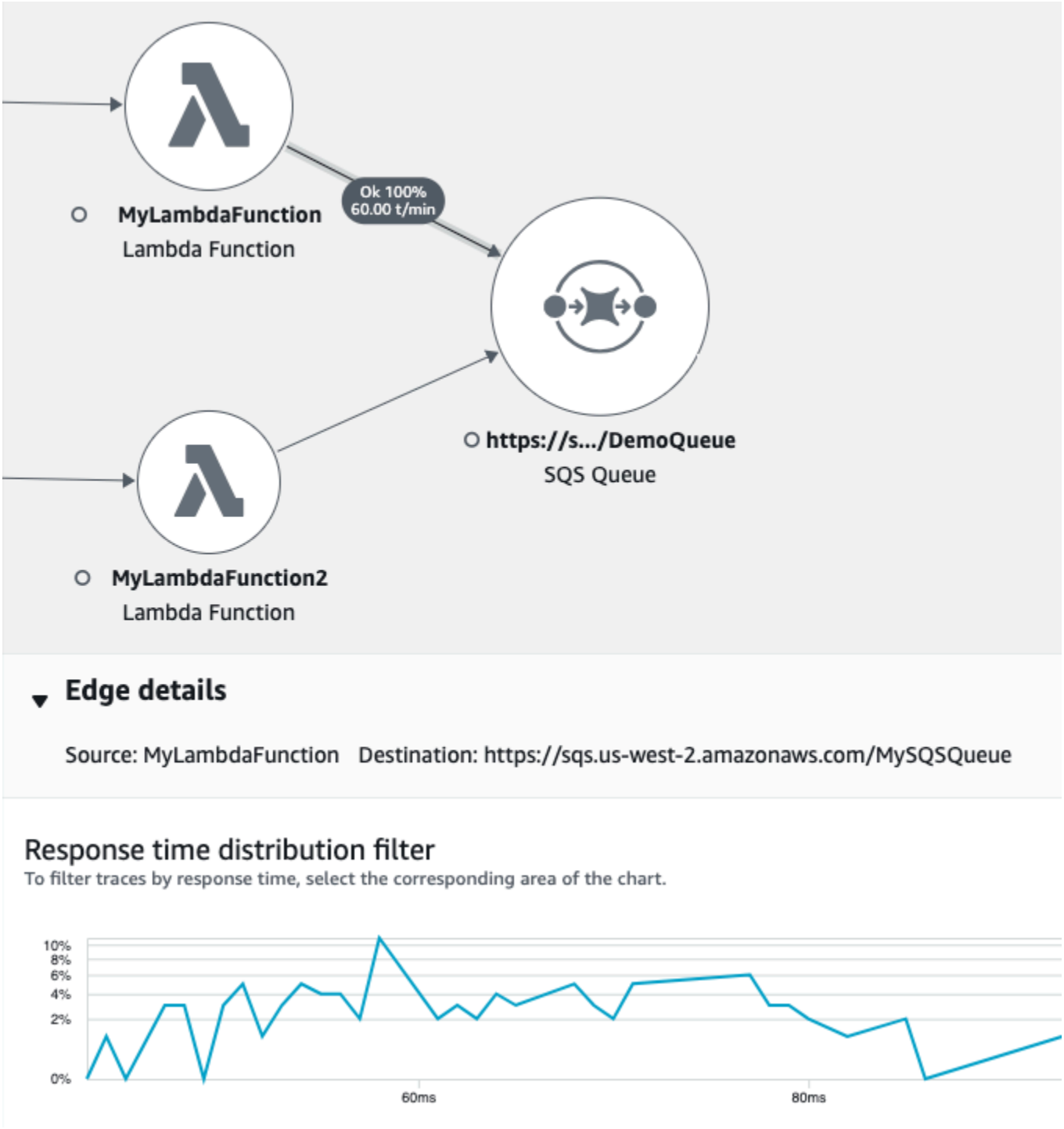



Wenn Sie einen anderen instrumentierten Dienst mit einer instrumentierten Anwendung aufrufen, sendet der nachgelagerte Dienst sein eigenes Segment. Dieses Segment zeichnet seine Ansicht desselben Anrufs auf, den der vorgelagerte Dienst in einem Untersegment aufgezeichnet hat. Im Dienstdiagramm enthalten die Knoten beider Dienste Timing- und Fehlerinformationen aus ihren Segmenten. Die Kante zwischen ihnen enthält Informationen aus dem Untersegment des Upstream-Service. Der nachgelagerte Dienst zeichnet auf, wann er die Bearbeitung der Anfrage gestartet und

beendet hat. Der Upstream-Dienst zeichnet die Roundtrip-Latenz auf, einschließlich der Zeit, die die Anfrage für die Übertragung zwischen den beiden Diensten aufgewendet hat.

Die folgende Abbildung zeigt nach der Antwortzeit gefilterte Trace-Informationen von einer Kante, die einer Upstream-Lambda-Funktion entspricht.



# Service-Diagramm

X-Ray verwendet die Daten, die Ihre Anwendung sendet, um ein Service-Diagramm zu generieren. Jede AWS Ressource, die Daten an X-Ray sendet, wird im Diagramm als Dienstknoten angezeigt. Edges verbinden die Dienste, die zusammenarbeiten, um Anfragen zu bearbeiten, verbinden Clients mit Ihrer Anwendung und verbinden Ihre Anwendung mit den nachgelagerten Diensten und Ressourcen, die sie verwendet.

> **Namen der Dienste**
>
> Der eines Segments `name` sollte mit dem Domainnamen oder dem logischen Namen des Dienstes übereinstimmen, der das Segment generiert. Dies wird jedoch nicht erzwungen. Jede Anwendung, die über die entsprechende Berechtigung verfügt, `PutTraceSegments`kann Segmente mit einem beliebigen Namen senden.

Eine Service-Graphik ist ein JSON-Dokument, das Informationen zu den Services und Ressourcen enthält, aus denen Ihre Anwendung besteht. Die X-Ray-Konsole verwendet den Service Graph, um eine Visualisierung oder Service Map zu generieren.

Die folgende Abbildung zeigt eine Service-Map. In der Service-Map wird die Beziehung zwischen der Anfrage des Clients an Ihre Anwendung und den Diensten angezeigt, mit denen Ihre Anwendung interagiert, um die Anfrage zu bearbeiten. In der folgenden Abbildung interagiert eine Scorekeep-Beispielanwendung mit zwei DynamoDB-Tabellen und Amazon SNS.


In einer verteilten Anwendung kombiniert X-Ray Knoten aller Dienste, die Anfragen mit derselben Trace-ID verarbeiten, zu einem einzigen Service-Graphen. Der erste Dienst, mit dem die Anfrage interagiert, fügt einen Tracing-Header hinzu, der zwischen dem Frontend und den von ihm aufgerufenen Diensten weitergegeben wird.

Beispielsweise führt Scorekeep eine Web-API aus, die eine AWS Lambda Funktion aufruft, um einen zufälligen Namen zu generieren. Anschließend generiert das X-Ray-SDK die Trace-ID und verfolgt Aufrufe der Lambda-Funktion. AWS Lambda übergibt die Ablaufverfolgungsdaten und die Trace-ID an die Lambda-Funktion. Das X-Ray SDK verwendet auch dieselbe Trace-ID, um Daten an einen Agenten oder Collector zu senden. Daher werden Knoten für die API, den AWS Lambda Service und die Lambda-Funktion alle als separate, aber verbundene Knoten auf der Trace-Map angezeigt.

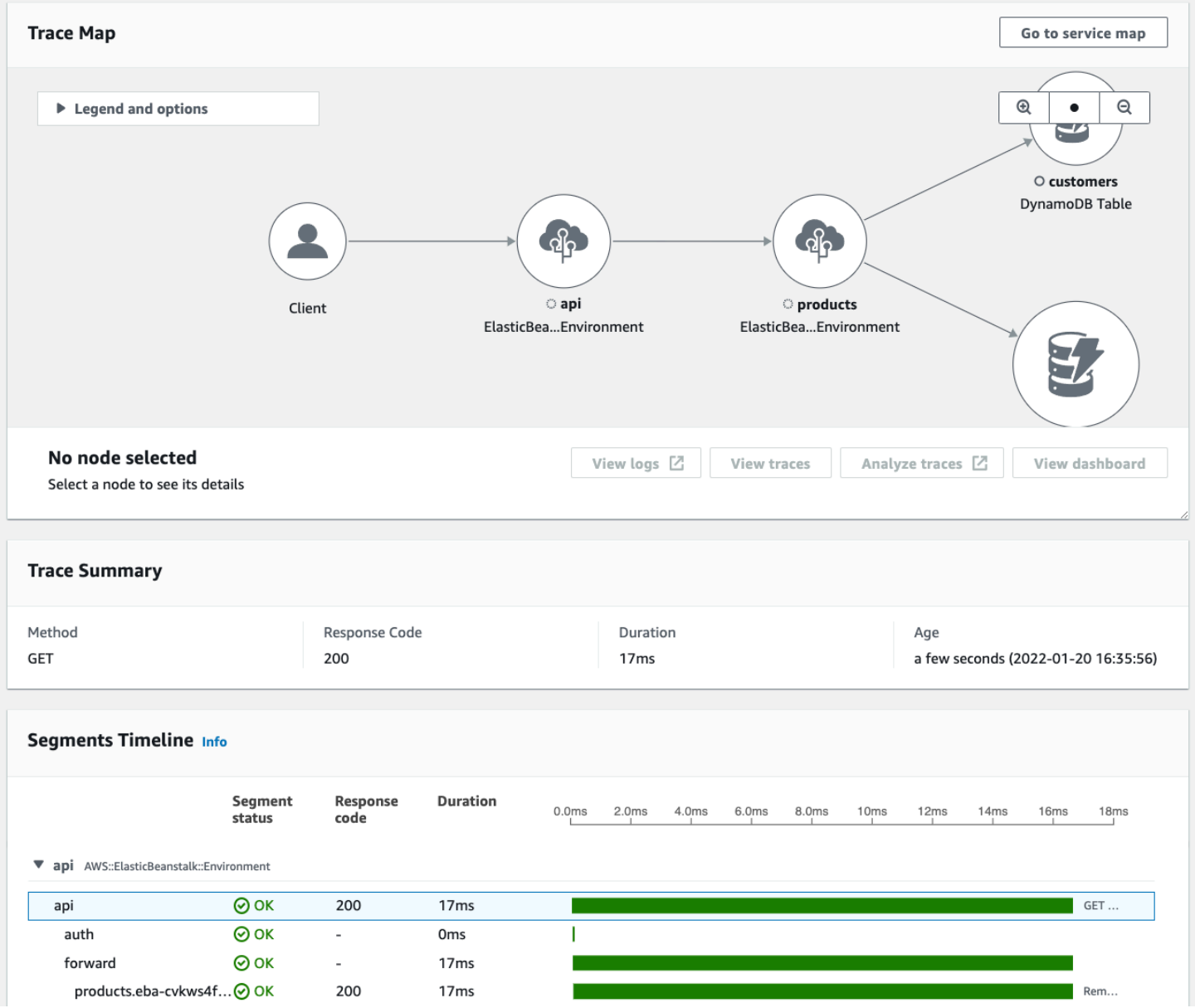
Die Daten des Service-Diagramms werden 30 Tage lang aufbewahrt.

## Ablaufverfolgungen

Eine Ablaufverfolgung sammelt alle von einer einzelnen Anforderung generierten Segmente. Der Trace verwendet eine Trace-ID, um den Pfad einer Anfrage durch Ihre Anwendung zu verfolgen.

Bei dieser Anfrage handelt es sich in der Regel um eine HTTP-GET- oder POST-Anforderung, die einen Load Balancer durchläuft, mit Ihrem Anwendungscode interagiert und Downstream-Aufrufe an andere AWS Dienste oder externe Web-APIs generiert. Der erste unterstützte Dienst, mit dem die HTTP-Anfrage interagiert, fügt der Anfrage einen Trace-ID-Header hinzu. Der Dienst leitet die Trace-ID anschließend weiter, um Latenz, Disposition und andere Anforderungsdaten nachzuverfolgen.

Die folgende Abbildung zeigt ein Beispiel für eine Anwendung, die eine HTTP Anfrage bearbeitet. Die Trace-Zusammenfassung enthält den HTTP Antwortcode, den Zeitpunkt, zu dem die Anfrage bearbeitet wurde, und wie lange es her ist, dass die Anwendung die Anfrage bearbeitet hat. Die folgende Abbildung zeigt auch eine Zeitleiste für jedes Trace-Segment. Die Zeitleiste zeigt den Status, den HTTP Antwortcode und die Zeit, die das Segment bis zur Fertigstellung benötigt hat. In einem Diagramm werden Dauer, Start- und Endzeit jedes Segments im Trace im Vergleich zu den anderen Segmenten angezeigt.




Weitere Informationen darüber, wie X-Ray-Rechnungen die Abholung verfolgen, finden Sie unter AWS X-Ray Preise für Informationen darüber, wie X-Ray-Nachverfolgungen abgerechnet werden. Trace-Daten werden 30 Tage lang aufbewahrt.

## Sampling

Das X-Ray SDK verwendet einen Sampling-Algorithmus, um eine effiziente Nachverfolgung zu gewährleisten und eine repräsentative Stichprobe der Anfragen bereitzustellen, die Ihre Anwendung bedient. Dieser Algorithmus bestimmt, welche Anfragen verfolgt werden. Standardmäßig zeichnet das X-Ray-SDK die erste Anfrage auf, die zu Beginn jeder Sekunde empfangen wird, und fünf Prozent aller weiteren Anfragen.

Um zu vermeiden, dass bei den ersten Schritten Servicegebühren anfallen, ist die Standard-Samplingrate konservativ. Sie können X-Ray so konfigurieren, dass die standardmäßige Abtastrate geändert wird, und zusätzliche Regeln konfigurieren, die das Sampling auf der Grundlage der Eigenschaften des Dienstes oder der Anfrage anwenden.

Möglicherweise möchten Sie das Sampling deaktivieren und alle Anfragen für Anrufe verfolgen, die den Status ändern oder Benutzer oder Transaktionen verarbeiten. Für Anrufe mit hohem Volumen, die nur lesbar sind, wie z. B. Hintergrundabfragen, Integritätsprüfungen oder Verbindungswartung.

Weitere Informationen finden Sie unter Konfigurieren Sie Stichprobenregeln und in der API. CreateSamplingRule

## Ablaufverfolgungs-Header

Alle Anfragen werden bis zu einer Mindestanzahl, die Sie konfigurieren können, nachverfolgt. Sobald dieses Minimum erreicht ist, verfolgt X-Ray nur einen bestimmten Prozentsatz der Anfragen, um zusätzliche Kosten zu vermeiden. X-Ray fügt HTTP-Anfragen in Tracing-Headern, die mit beginnen, die Sampling-Entscheidung und die Trace-ID hinzu. `X-Amzn-Trace-Id` X-Ray fügt diese Header hinzu, wenn eine Anfrage mit dem ersten AWS Dienst interagiert, der in X-Ray integriert ist. Das X-Ray SDK liest diese Header und nimmt sie in die Antwort auf.

Example Ablaufverfolgungs-Header mit Stammablaufverfolgungs-ID und Samplingentscheidung

```
X-Amzn-Trace-Id: Root=1-5759e988-bd862e3fe1be46a994272793;Sampled=1
```

> **Sicherheit des Ablaufverfolgungs-Headers**
>
> Ein Tracing-Header kann vom X-Ray SDK AWS-Service, einer oder der Client-Anfrage stammen. Ihre Anwendung kann `X-Amzn-Trace-Id` aus eingehenden Anforderungen entfernen, um Probleme zu vermeiden, die von Benutzern verursacht werden, die ihren Anforderungen Ablaufverfolgungs-IDs oder Samplingentscheidungen hinzufügen.

Der Ablaufverfolgungs-Header kann auch eine übergeordnete Segment-ID enthalten, wenn die Anforderung von einer instrumentierten Anwendung stammte. Wenn Ihre Anwendung beispielsweise eine Downstream-HTTP-Web-API mit einem instrumentierten HTTP-Client aufruft, fügt das X-Ray-SDK die Segment-ID für die ursprüngliche Anfrage zum Tracing-Header der Downstream-Anfrage hinzu. Eine instrumentierte Anwendung, die die Downstream-Anfrage bedient, verwendet die ID des übergeordneten Segments, um die beiden Anfragen zu verbinden.

Example Ablaufverfolgungs-Header mit Stammablaufverfolgungs-ID, übergeordnete Segment-ID und Samplingentscheidung

```
X-Amzn-Trace-Id: Root=1-5759e988-
bd862e3fe1be46a994272793;Parent=53995c3f42cd8ad8;Sampled=1
```

Lambda oder andere AWS-Services könnten einen Teil eines Headers, der mit 1 beginnt, `Lineage` als Teil ihrer Verarbeitungsmechanismen anhängen. Sie sollten den angehängten Teil des Trace-Headers nicht direkt verwenden.

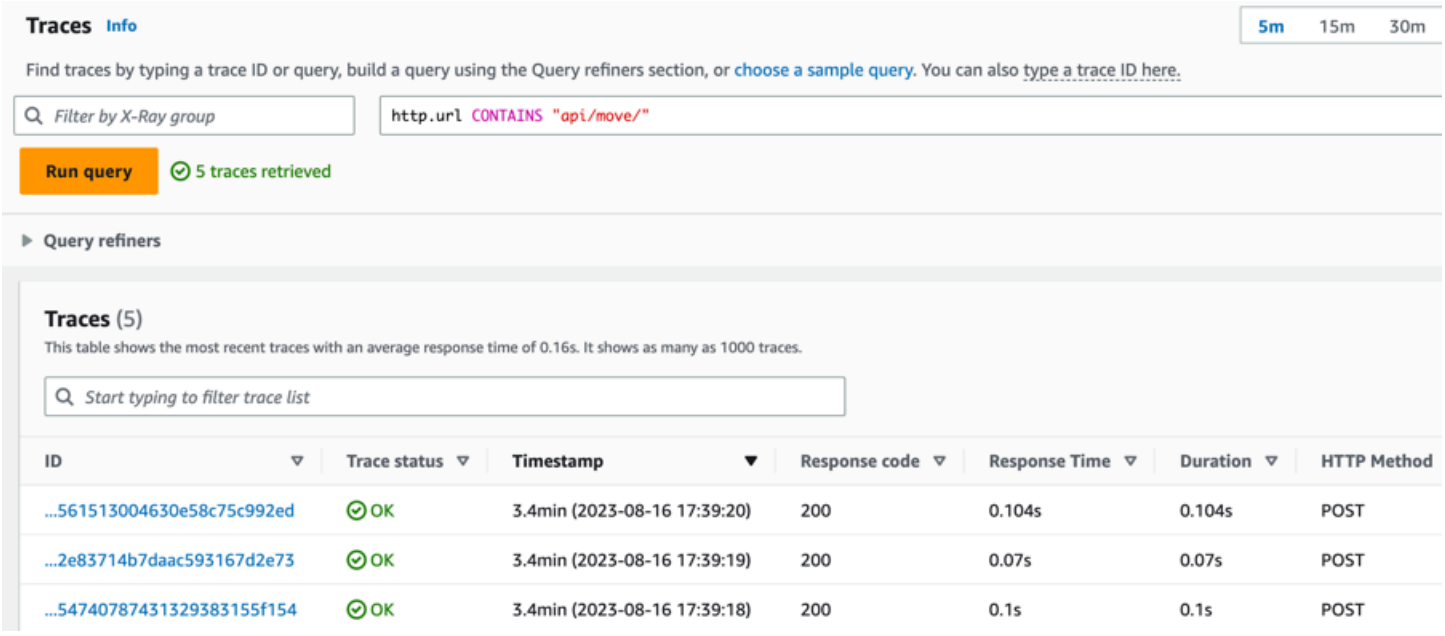
Example Tracing-Header mit Lineage

```
X-Amzn-Trace-Id: Root=1-5759e988-bd862e3fe1be46a994272793;Sampled=1;Lineage=a87bd80c:1|
68fd508a:5|c512fbe3:2
```

## Filterausdrücke

Selbst wenn Sie nur eine kleine Teilmenge von Daten abfragen, kann eine komplexe Anwendung viele Trace-Daten generieren. Verwenden Sie Filterausdrücke, um bestimmte Traces zu finden, einschließlich solcher für einzelne Anfragen, bestimmte Pfade oder Benutzer.

Die folgende Abbildung zeigt ein Textfeld in der X-Ray-Konsole, mit dem Sie nach einer von Ihnen definierten Gruppe filtern können. Weitere Informationen zu Gruppen finden Sie im folgenden Abschnitt Gruppen.



## Gruppen

Sie können eine Gruppe innerhalb eines Filterausdrucks verwenden, um die Menge der Trace-Daten zu reduzieren und sich auf Daten zu konzentrieren, die den Gruppenkriterien entsprechen.

Verwenden Sie eine Gruppe, um Servicegraphen, Trace-Zusammenfassungen und CloudWatch Kennzahlen zu generieren, die für diese Gruppe spezifisch sind. Sie können mit dem Namen oder mit dem Amazon Resource Name (ARN) anrufen. X-Ray überprüft eingehende Traces anhand des Gruppenfilterausdrucks, während sie im X-Ray-Dienst gespeichert werden. CloudWatchveröffentlicht jede Minute Metriken für Traces, die den Gruppenkriterien entsprechen.

Durch das Aktualisieren des Filterausdrucks einer Gruppe werden die bereits aufgezeichneten Daten nicht geändert. Die Aktualisierung gilt nur für nachfolgende Ablaufverfolgungen. Dies kann dazu führen, dass im Diagramm neue und alte Ausdrücke zusammengeführt werden. Um zu vermeiden, dass nicht verbundene Gruppen in einem einzigen Diagramm zusammengeführt werden, löschen Sie die aktuelle Gruppe und https://docs.aws.amazon.com/xray/latest/api/API_CreateGroup.html erstellen Sie eine neue.

> **Note**
>
> Die Abrechnung für Gruppen basiert auf der Anzahl der abgerufenen Traces, die dem Filterausdruck entsprechen. Weitere Informationen finden Sie unter AWS X-Ray Preise.

Weitere Informationen zu Gruppen finden Sie unterGruppen konfigurieren.

## Anmerkungen und Metadaten

Wenn Sie Ihre Anwendung instrumentieren, zeichnet das X-Ray SDK Informationen über eingehende und ausgehende Anfragen auf. Das SDK zeichnet auch Informationen über die verwendeten AWS Ressourcen und die Anwendung selbst auf. Sie können dem Segmentdokument weitere Informationen hinzufügen, wie etwa Anmerkungen und Metadaten. Anmerkungen und Metadaten werden auf der Trace-Ebene kombiniert. Sie können zu jedem Segment oder Untersegment hinzugefügt werden.

Anmerkungen sind Schlüssel-Wert-Paare, die für die Verwendung mit Filterausdrücken indexiert sind. Berücksichtigen Sie Anmerkungen, um Daten zur Gruppierung von Ablaufverfolgungen in der Konsole zu verwenden, oder wenn Sie die `GetTraceSummaries`-API aufrufen.

X-Ray indexiert bis zu 50 Anmerkungen pro Spur.

Metadaten sind Schlüssel-Wert-Paare mit Werten beliebigen Typs, einschließlich Objekten und Listen, die nicht indexiert sind. Verwenden Sie Metadaten zum Aufzeichnen von Daten in der Ablaufverfolgung, die nicht zur Ablaufsuche erforderlich sind.

Sie können Anmerkungen und Metadaten im Fenster mit den Segment- oder Untersegmentdetails auf der Seite mit den Trace-Details in der Konsole anzeigen. CloudWatch Weitere Informationen finden Sie unter Traces und Trace-Details anzeigen unter. Erkunden Sie die X-Ray-Konsole

## Fehler und Ausnahmen

X-Ray verfolgt Fehler in Ihrem Anwendungscode und solche, die von nachgelagerten Diensten zurückgegeben werden. X-Ray verfolgt die folgenden HTTP Antwortcodes von Anfragen:

- **`Error`**— Client-Fehler (Fehler der Serie 400) deuten darauf hin, dass der Server die Anfrage des Clients nicht verstehen oder verarbeiten konnte, da die Anfrage selbst einen Fehler enthielt. Diese Fehler können durch Syntaxfehler, fehlende Informationen oder einen fehlerhaften Anfragetext verursacht werden.
- **`Fault`**— Serverfehler (Fehler der Serie 500) deuten darauf hin, dass der Server eine gültige Anfrage aufgrund eines Problems mit dem Server selbst nicht verarbeiten konnte. Diese Fehler können durch Probleme verursacht werden, zu denen Software- oder Hardwarefehler oder Ressourcenbeschränkungen des Servers gehören.
- **`Throttle`**— Drosselungsfehler (429 Too Many Requests) sind eine bestimmte Art von Client-Fehlern, die auftreten, wenn ein Client über einen bestimmten Zeitraum zu viele Anfragen an einen Server oder eine API sendet.

Wenn eine Ausnahme auftritt, während Ihre Anwendung eine instrumentierte Anfrage bearbeitet, zeichnet das X-Ray-SDK Details zu der Ausnahme auf, einschließlich der Stack-Trace-ID, falls verfügbar. Sie können Ausnahmen unter Segmentdetails in der X-Ray-Konsole anzeigen.

# Erste Schritte mit X-Ray

Um X-Ray zu verwenden, müssen Sie wie folgt vorgehen:

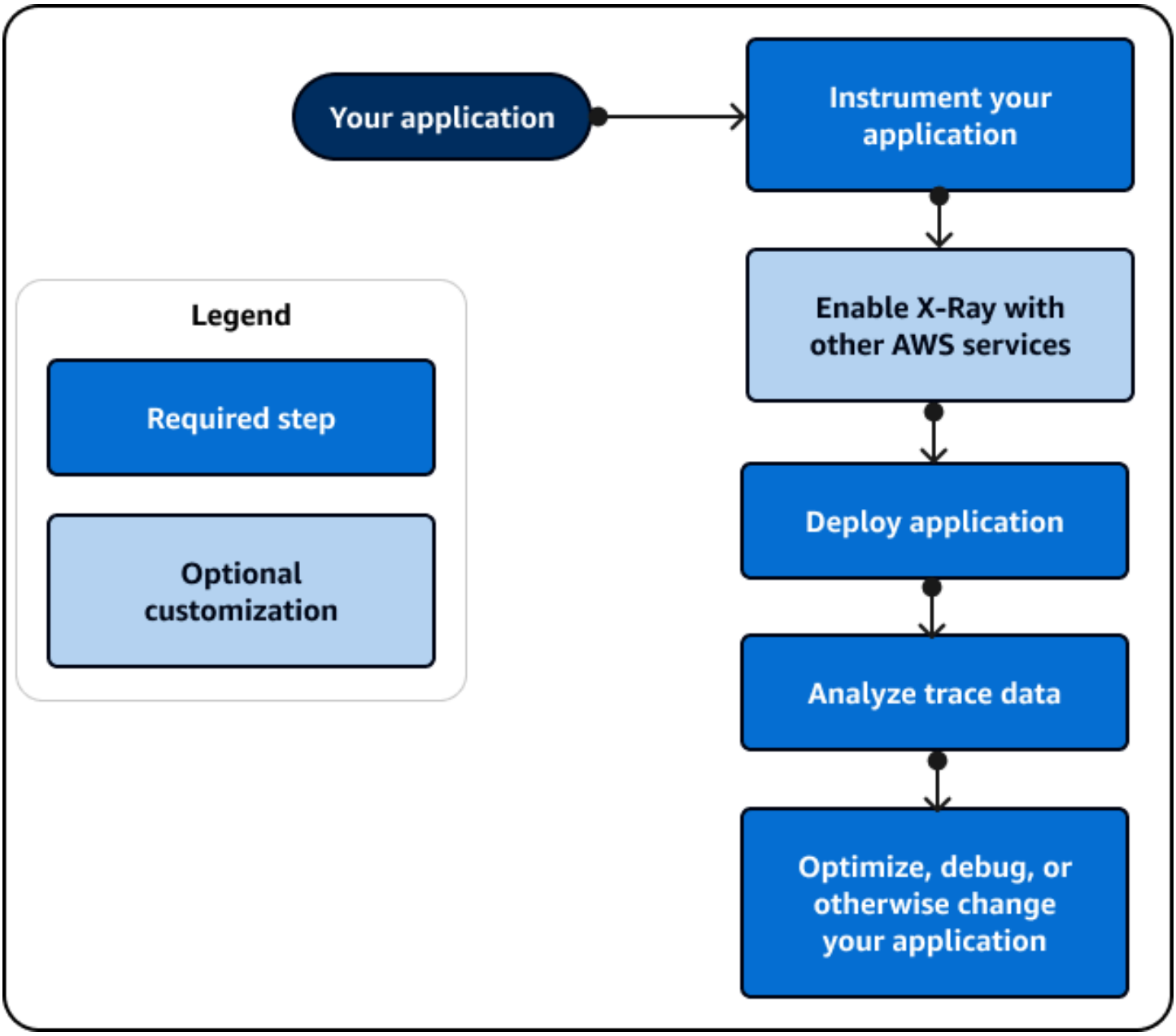
1. Instrumentieren Sie Ihre Anwendung, sodass X-Ray verfolgen kann, wie Ihre Anwendung eine Anfrage bearbeitet.

   - Verwenden Sie die X-Ray SDKs, X-Ray APIs ADOT oder CloudWatch Application Signals, um Trace-Daten an X-Ray zu senden. Weitere Informationen darüber, welche Schnittstelle verwendet werden soll, finden Sie unterWählen Sie eine Schnittstelle.

   Weitere Hinweise zur Instrumentierung finden Sie unterInstrumentieren Sie Ihre Bewerbung für AWS X-Ray.
2. (Optional) Konfigurieren Sie X-Ray so, AWS-Services dass es mit anderen Geräten funktioniert, die in X-Ray integriert sind. Sie können Traces testen und Header zu eingehenden Anfragen hinzufügen, einen Agenten oder Collector ausführen und Trace-Daten automatisch an X-Ray senden. Weitere Informationen finden Sie unter Integrieren Sie AWS X-Ray mit anderen AWS-Services.
3. Stellen Sie Ihre instrumentierte Anwendung bereit. Wenn Ihre Anwendung Anfragen empfängt, zeichnet das X-Ray SDK Trace-, Segment- und Subsegmentdaten auf. In diesem Schritt müssen Sie möglicherweise auch eine IAM-Richtlinie einrichten und einen Agenten oder Collector bereitstellen.

   - Beispiele für Skripts zur Bereitstellung einer Anwendung mithilfe des AWS Distro for OpenTelemetry (ADOT) -SDK und des CloudWatch Agenten auf verschiedenen Plattformen finden Sie unter Application Signals Demo Scripts.
   - Ein Beispielskript für die Bereitstellung einer Anwendung mit dem X-Ray SDK und dem X-Ray-Daemon finden Sie unterAWS X-Ray Beispielanwendung.
4. (Optional) Öffnen Sie eine Konsole, um die Daten anzuzeigen und zu analysieren. Sie können sich eine grafische Darstellung einer Trace-Map, einer Service Map und mehr ansehen, um zu überprüfen, wie Ihre Anwendung funktioniert. Verwenden Sie die grafischen Informationen in der Konsole, um Ihre Anwendung zu optimieren, zu debuggen und zu verstehen. Weitere Informationen zur Auswahl einer Konsole finden Sie unterVerwenden Sie ein AWS Management Console.

Das folgende Diagramm zeigt die ersten Schritte mit X-Ray:



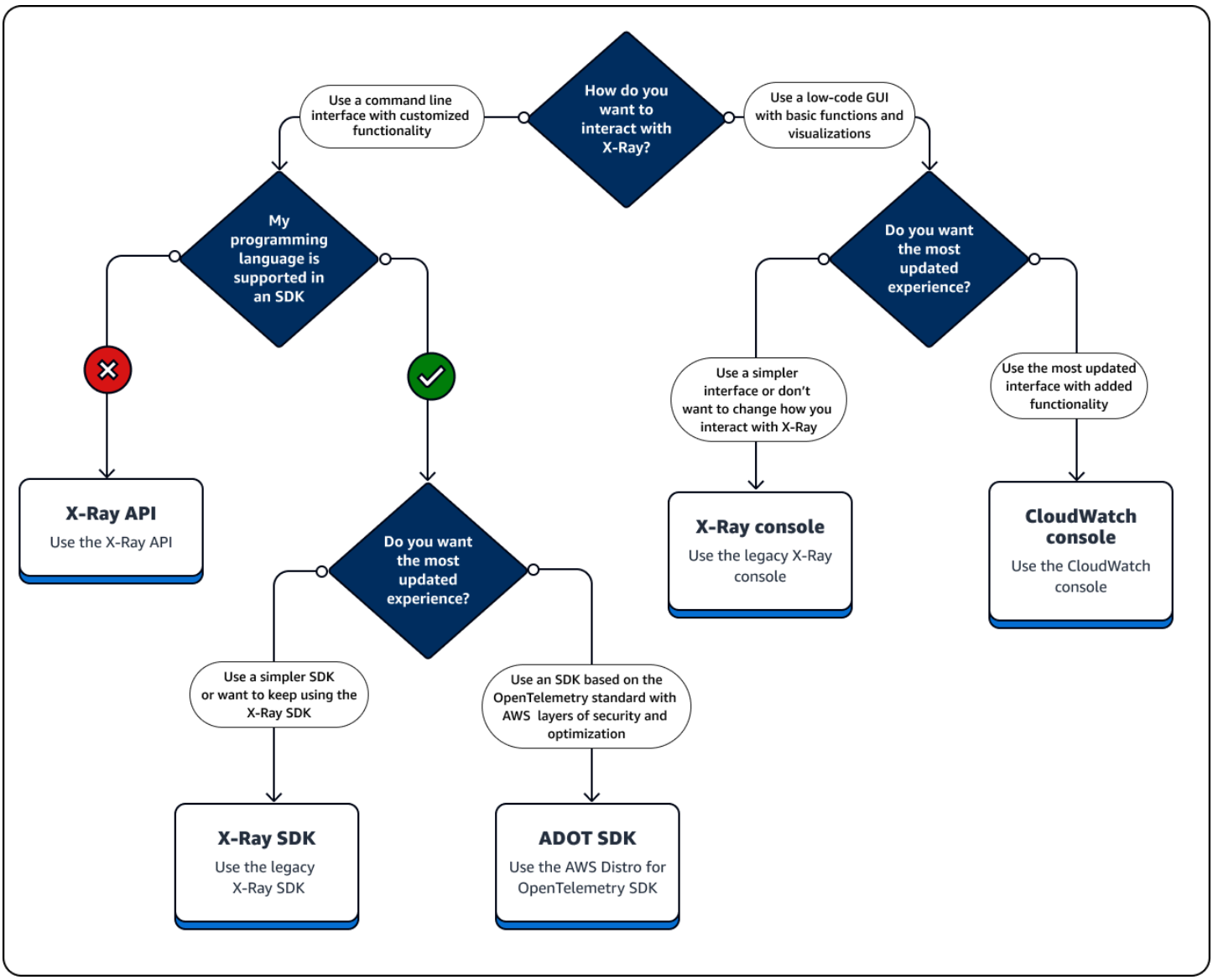

Ein Beispiel für die Daten und Karten, die in der Konsole verfügbar sind, erhalten Sie, wenn Sie eine Beispielanwendung starten, die bereits für die Generierung von Trace-Daten ausgestattet ist. In wenigen Minuten können Sie Traffic generieren, Segmente an X-Ray senden und eine Trace- und Servicekarte anzeigen.

# Wählen Sie eine Schnittstelle

AWS X-Ray kann Ihnen Einblicke geben, wie Ihre Anwendung funktioniert und wie gut sie mit anderen Diensten und Ressourcen interagiert. Nachdem Sie Ihre Anwendung instrumentiert oder konfiguriert haben, sammelt X-Ray Trace-Daten, während Ihre Anwendung Anfragen bearbeitet. Sie können diese Trace-Daten analysieren, um Leistungsprobleme zu identifizieren, Fehler zu beheben und Ihre Ressourcen zu optimieren. Diese Anleitung zeigt Ihnen, wie Sie mit X-Ray anhand der folgenden Richtlinien interagieren können:

- Verwenden Sie eine, AWS Management Console wenn Sie schnell loslegen möchten oder vorgefertigte Visualisierungen für grundlegende Aufgaben verwenden können.
  - Wählen Sie die CloudWatch Amazon-Konsole für die aktuellste Benutzererfahrung, die alle Funktionen der X-Ray-Konsole enthält.
  - Verwenden Sie die X-Ray-Konsole, wenn Sie eine einfachere Oberfläche wünschen oder die Art und Weise, wie Sie mit X-Ray interagieren, nicht ändern möchten.
- Verwenden Sie ein SDK, wenn Sie mehr benutzerdefinierte Funktionen für die Ablaufverfolgung, Überwachung oder Protokollierung benötigen, als Ihnen ein anderes bieten AWS Management Console kann.
  - Wählen Sie das ADOT SDK, wenn Sie ein herstellerunabhängiges SDK benötigen, das auf dem OpenTelemetry Open-Source-SDK basiert und zusätzliche Sicherheits- und Optimierungsebenen bietet. AWS
  - Wählen Sie das X-Ray-SDK, wenn Sie ein einfacheres SDK wünschen oder Ihren Anwendungscode nicht aktualisieren möchten.
- Verwenden Sie X-Ray-API-Operationen, wenn ein SDK die Programmiersprache Ihrer Anwendung nicht unterstützt.

Das folgende Diagramm hilft Ihnen bei der Auswahl, wie Sie mit X-Ray interagieren möchten:




Erkunden Sie die Schnittstellentypen

- Verwenden Sie ein AWS Management Console
- Verwenden Sie ein SDK
- Verwenden Sie die X-Ray-API

# Verwenden Sie ein AWS Management Console

Verwenden Sie eine AWS Management Console , wenn Sie eine grafische Benutzeroberfläche (GUI) wünschen, die nur minimale Codierung erfordert. Benutzer, die mit X-Ray noch nicht vertraut

sind, können mithilfe vorgefertigter Visualisierungen schnell loslegen und grundlegende Aufgaben ausführen. Sie können die folgenden Aktionen direkt von der Konsole aus ausführen:

- X-Ray aktivieren.
- Sehen Sie sich allgemeine Zusammenfassungen der Leistung Ihrer Anwendung an.
- Überprüfen Sie den Status Ihrer Anwendungen.
- Identifizieren Sie Fehler auf hoher Ebene.
- Sehen Sie sich grundlegende Trace-Zusammenfassungen an.

Sie können entweder die CloudWatch Amazon-Konsole unter https://console.aws.amazon.com/cloudwatch/ oder die X-Ray-Konsole unter https://console.aws.amazon.com/xray/home verwenden, um mit X-Ray zu interagieren.

## Verwenden Sie die CloudWatch Amazon-Konsole

Die CloudWatch Konsole enthält neue X-Ray-Funktionen, die gegenüber der X-Ray-Konsole neu gestaltet wurden, um die Bedienung zu vereinfachen. Wenn Sie die CloudWatch Konsole verwenden, können Sie CloudWatch Protokolle und Metriken zusammen mit X-Ray-Trace-Daten anzeigen. Verwenden Sie die CloudWatch Konsole, um Daten wie die folgenden anzuzeigen und zu analysieren:

- X-Ray-Traces — Sie können die mit Ihrer Anwendung verknüpften Traces anzeigen, analysieren und filtern, während sie eine Anfrage bearbeitet. Verwenden Sie diese Traces, um hohe Latenzen zu finden, Fehler zu debuggen und Ihren Anwendungsworkflow zu optimieren. Sehen Sie sich eine Trace Map und eine Service Map an, um visuelle Darstellungen Ihres Anwendungsworkflows zu sehen.
- Protokolle — Sie können die von Ihrer Anwendung erstellten Protokolle anzeigen, analysieren und filtern. Verwenden Sie Protokolle, um Fehler zu beheben und die Überwachung auf der Grundlage bestimmter Protokollwerte einzurichten.
- Metriken — Messen und überwachen Sie die Leistung Ihrer Anwendung anhand von Metriken, die Ihre Ressourcen ausgeben, oder erstellen Sie Ihre eigenen Messwerte. Sehen Sie sich diese Metriken in Grafiken und Diagrammen an.
- Überwachung von Netzwerken und Infrastruktur — Überwachen Sie wichtige Netzwerke auf Ausfälle und den Zustand und die Leistung Ihrer Infrastruktur, einschließlich containerisierter Anwendungen, anderer AWS Dienste und Clients.

- Alle Funktionen der X-Ray-Konsole, die im folgenden Abschnitt „Verwenden Sie die X-Ray-Konsole“ aufgeführt sind.

Weitere Informationen zur CloudWatch Konsole finden Sie unter Erste Schritte mit Amazon CloudWatch.

Melden Sie sich bei der CloudWatch Amazon-Konsole unter https://console.aws.amazon.com/cloudwatch/ an.

## Verwenden Sie die X-Ray-Konsole

Die X-Ray-Konsole bietet verteiltes Tracing für Anwendungsanfragen. Verwenden Sie die X-Ray-Konsole, wenn Sie eine einfachere Konsolenerfahrung wünschen oder Ihren Anwendungscode nicht aktualisieren möchten. AWS entwickelt die X-Ray-Konsole nicht mehr. Die X-Ray-Konsole enthält die folgenden Funktionen für instrumentierte Anwendungen:

- Einblicke — Erkennen Sie automatisch Anomalien in der Leistung Ihrer Anwendung und finden Sie die zugrunde liegenden Ursachen. Insights sind in der CloudWatch Konsole unter Insights enthalten. Weitere Informationen finden Sie unter Verwenden von X-Ray Insights inErkunden Sie die X-Ray-Konsole.
- Dienstübersicht — Zeigt eine grafische Struktur Ihrer Anwendung und ihrer Verbindungen zu Clients, Ressourcen, Diensten und Abhängigkeiten an.
- Ablaufverfolgungen — Sehen Sie sich eine Übersicht der Traces an, die von Ihrer Anwendung bei der Bearbeitung einer Anfrage generiert werden. Verwenden Sie Trace-Daten, um zu verstehen, wie Ihre Anwendung im Vergleich zu grundlegenden Kennzahlen wie HTTP Antwort- und Antwortzeit abschneidet.
- Analytik — Interpretieren, untersuchen und analysieren Sie Trace-Daten mithilfe von Diagrammen zur Verteilung der Antwortzeiten.
- Konfiguration — Erstellen Sie benutzerdefinierte Traces, um die Standardkonfigurationen für Folgendes zu ändern:
  - Probenahme — Erstellen Sie eine Regel, die festlegt, wie oft Ihre Anwendung nach Trace-Informationen durchsucht werden soll. Weitere Informationen finden Sie unter Probenahmeregeln konfigurieren unterErkunden Sie die X-Ray-Konsole.
  - Verschlüsselung — Verschlüsseln Sie Daten im Ruhezustand mit einem Schlüssel, den Sie überprüfen oder deaktivieren können. AWS Key Management Service

- Gruppen — Verwenden Sie einen Filterausdruck, um eine Gruppe von Traces mit einem gemeinsamen Merkmal wie dem Namen einer URL oder einer Antwortzeit zu definieren. Weitere Informationen finden Sie unter Gruppen konfigurieren.

Loggen Sie sich unter https://console.aws.amazon.com/xray/home in die X-Ray-Konsole ein.

## Erkunden Sie die X-Ray-Konsole

Verwenden Sie die X-Ray-Konsole, um eine Übersicht der Dienste und der zugehörigen Traces für Anfragen anzuzeigen, die Ihre Anwendungen bearbeiten, und um Gruppen und Sampling-Regeln zu konfigurieren, die beeinflussen, wie Traces an X-Ray gesendet werden.

> **Note**
>
> Die X-Ray-Service-Karte und die CloudWatch ServiceLens Karte wurden in der CloudWatch Amazon-Konsole zur X-Ray-Trace-Map zusammengefasst. Öffnen Sie die CloudWatchKonsole und wählen Sie im linken Navigationsbereich unter X-Ray-Traces die Option Trace Map aus.
> CloudWatch enthält jetzt Application Signals, mit denen Sie Ihre Anwendungsdienste, Clients, Synthetics-Kanarien und Serviceabhängigkeiten erkennen und überwachen können. Verwenden Sie Application Signals, um eine Liste oder eine visuelle Übersicht Ihrer Services zu erhalten, Zustandsmetriken auf der Grundlage Ihrer Servicelevel-Ziele (SLOs) einzusehen und eine detaillierte Darstellung korrelierter X-Ray-Traces für eine detailliertere Fehlerbehebung durchzuführen.

Die primäre X-Ray-Konsolenseite ist die Trace-Map, eine visuelle Darstellung des JSON-Dienstdiagramms, das X-Ray aus den von Ihren Anwendungen generierten Trace-Daten generiert. Die Übersicht besteht aus Service-Knoten für jede Anwendung in Ihrem Konto, das Anforderungen verarbeitet, aus vorgelagerten Knoten, die die Ursprünge der Anforderungen repräsentieren, und aus nachgelagerten Service-Knoten, die Web-Services und Ressourcen darstellen, die von einer Anwendung während der Verarbeitung einer Anforderung verwendet werden. Es gibt zusätzliche Seiten zum Anzeigen von Traces und Trace-Details sowie zum Konfigurieren von Gruppen und Stichprobenregeln.

Sehen Sie sich das Konsolenerlebnis für X-Ray an und vergleichen Sie es mit der CloudWatch Konsole in den folgenden Abschnitten.

## Verwenden Sie die X-Ray-Trace-Map

Sehen Sie sich die X-Ray-Trace-Map an, um Dienste zu identifizieren, bei denen Fehler auftreten, Verbindungen mit hoher Latenz oder Traces für Anfragen, die nicht erfolgreich waren.

> **Note**
>
> CloudWatch enthält jetzt Application Signals, mit denen Sie Ihre Anwendungsservices, Clients, synthetischen Datenbanken und Dienstabhängigkeiten erkennen und überwachen können. Verwenden Sie Application Signals, um eine Liste oder eine visuelle Übersicht Ihrer Services zu erhalten, Zustandsmetriken auf der Grundlage Ihrer Servicelevel-Ziele (SLOs) einzusehen und eine detaillierte Darstellung korrelierter X-Ray-Traces für eine detailliertere Fehlerbehebung durchzuführen.
> Die X-Ray-Servicekarte und die CloudWatch ServiceLens Karte werden in der CloudWatch Amazon-Konsole zur X-Ray-Trace-Map kombiniert. Öffnen Sie die CloudWatchKonsole und wählen Sie im linken Navigationsbereich unter X-Ray-Traces die Option Trace Map aus.

### Anzeigen der Ablaufverfolgungskarte

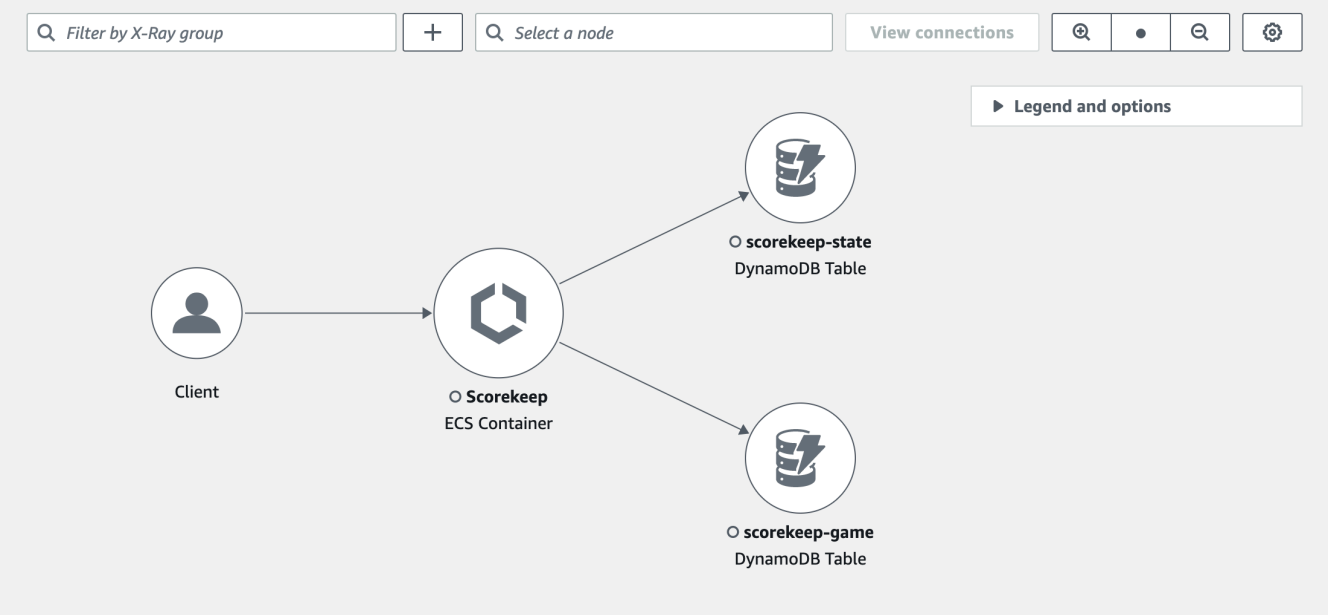
Die Trace-Map ist eine visuelle Darstellung der Trace-Daten, die von Ihren Anwendungen generiert werden. Die Karte zeigt Dienstknoten, die Anfragen bearbeiten, vorgelagerte Clientknoten, die den Ursprung der Anfragen darstellen, und Downstream-Serviceknoten, die Webdienste und Ressourcen darstellen, die von einer Anwendung bei der Bearbeitung einer Anfrage verwendet werden.

Die Trace-Map zeigt eine verknüpfte Ansicht von Traces für ereignisgesteuerte Anwendungen, die Amazon SQS und Lambda verwenden. Weitere Informationen finden Sie im folgenden Abschnitt zu ereignisgesteuerten Trace-Anwendungen. Die Trace-Map unterstützt auch die kontenübergreifende Ablaufverfolgung, bei der Knoten mehrerer Konten in einer einzigen Map angezeigt werden.

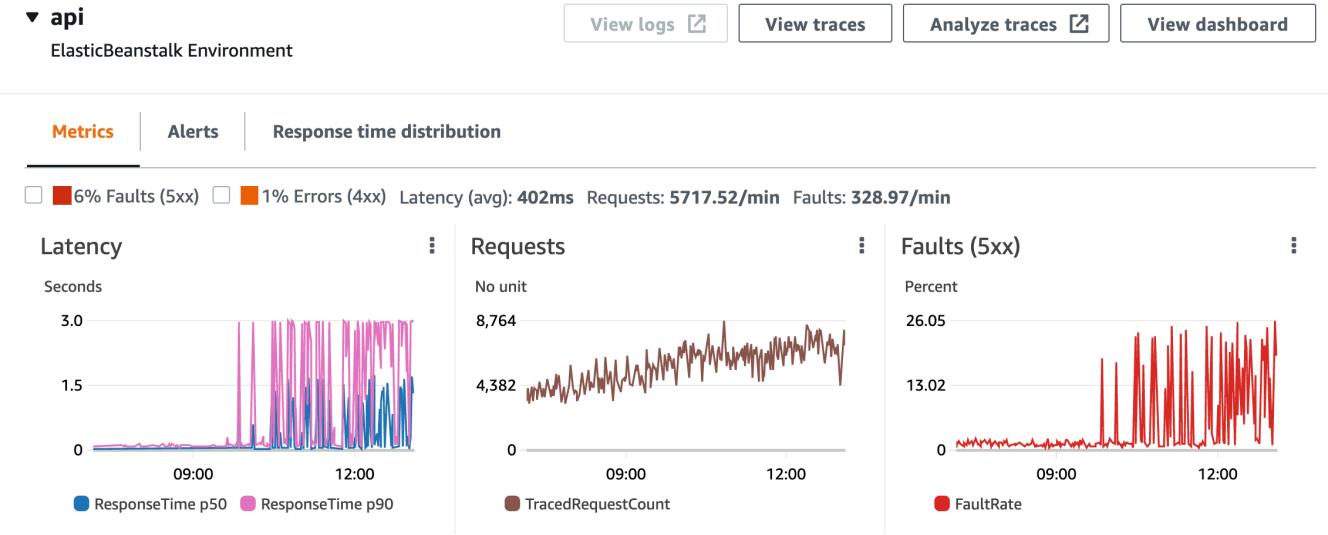
#### CloudWatch console

Um die Trace-Map in der Konsole anzuzeigen CloudWatch

1. Öffnen Sie die CloudWatch -Konsole. Wählen Sie im linken Navigationsbereich im Bereich X-Ray Traces die Option Trace Map aus.




2. Wählen Sie einen Service-Knoten, um Anforderungen für diesen Knoten anzuzeigen, oder eine Edge zwischen zwei Knoten, um Anforderungen anzuzeigen, die diese Verbindung passiert haben.
3. Unter der Trace-Map werden zusätzliche Informationen angezeigt, darunter Registerkarten für Messwerte, Warnmeldungen und die Verteilung der Antwortzeiten. Wählen Sie auf der Registerkarte Metriken einen Bereich innerhalb jedes Diagramms aus, um weitere Details anzuzeigen, oder wählen Sie die Optionen Fehler oder Fehler, um die Ablaufverfolgung zu filtern. Wählen Sie auf der Registerkarte Verteilung der Antwortzeiten einen Bereich innerhalb des Diagramms aus, um die Ablaufverfolgung nach Antwortzeit zu filtern.




4. Zeigen Sie Traces an, indem Sie Traces anzeigen wählen, oder wenn ein Filter angewendet wurde, wählen Sie Gefilterte Traces anzeigen.

5. Wählen Sie Protokolle anzeigen, um die mit dem ausgewählten Knoten verknüpften CloudWatch Protokolle anzuzeigen. Nicht alle Trace-Map-Knoten unterstützen das Anzeigen von Protokollen. Weitere Informationen finden Sie in den CloudWatch Protokollen zur Fehlerbehebung.

In der Trace-Map werden Probleme innerhalb der einzelnen Knoten farblich dargestellt:

- Rot für Server-Fehler (500er-Fehler)
- Gelb für Client-Fehler (400er-Fehler)
- Violett für Ablehnungsfehler („Fehler 429 – Zu viele Anfragen”)

Wenn Ihre Trace-Map groß ist, verwenden Sie die Steuerelemente auf dem Bildschirm oder die Maus, um sie zu vergrößern und zu verkleinern und die Karte zu verschieben.

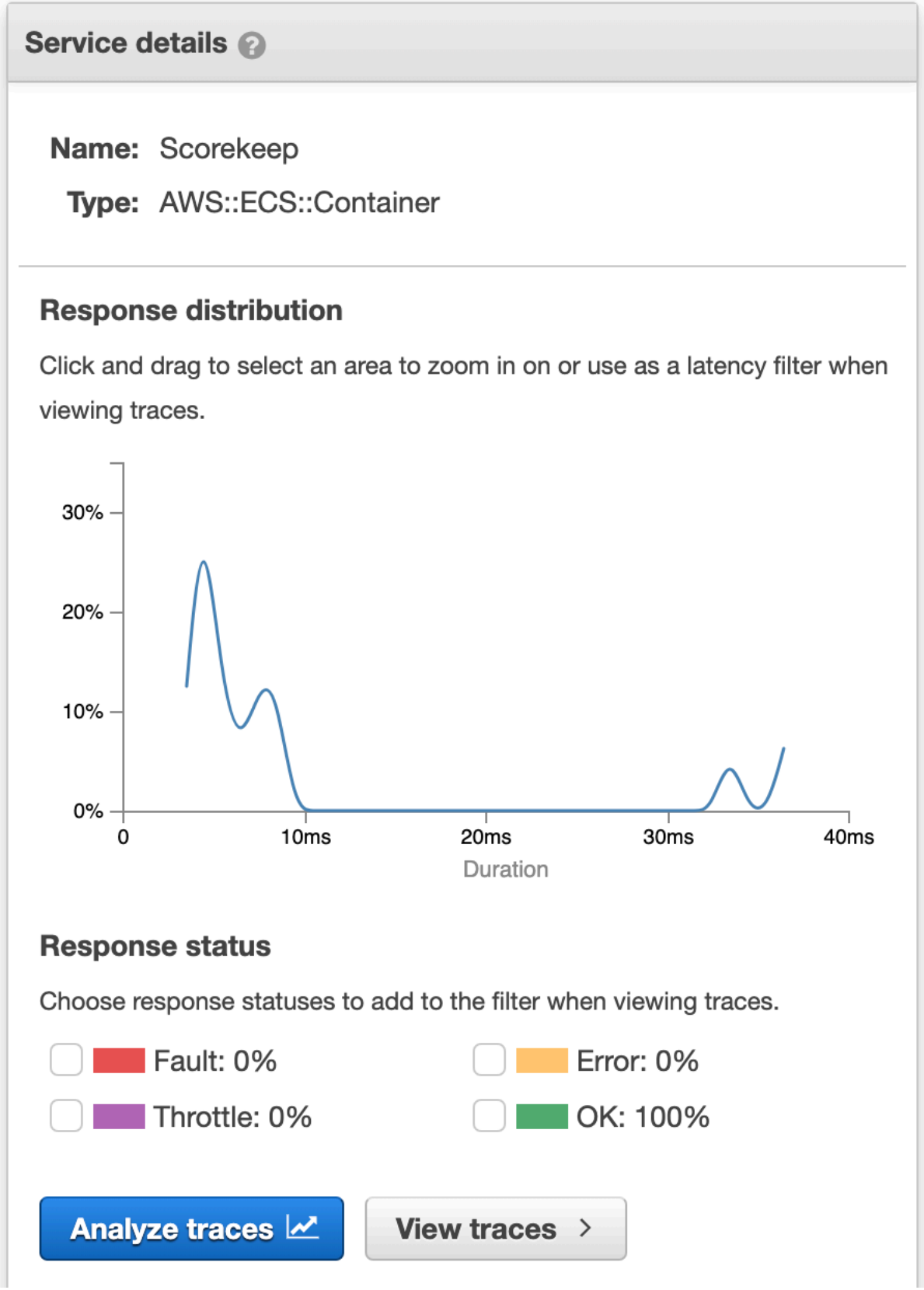
X-Ray console

Um die Serviceübersicht anzuzeigen

1. Öffnen Sie die X-Ray-Konsole. Die Serviceübersicht wird standardmäßig angezeigt. Sie können Service Map auch im linken Navigationsbereich auswählen.


2. Wählen Sie einen Service-Knoten, um Anforderungen für diesen Knoten anzuzeigen, oder eine Edge zwischen zwei Knoten, um Anforderungen anzuzeigen, die diese Verbindung passiert haben.

3. Verwenden Sie ein Histogramm zur Verteilung von Antworten, um Traces nach Dauer zu filtern, und wählen Sie die Statuscodes aus, für die Sie Traces anzeigen möchten. Klicken Sie anschließend auf View traces, um die Ablaufverfolgungsliste mit angewendetem Filterausdruck zu öffnen. Weitere Informationen zu Verteilungshistogrammen finden Sie unter. ???


Service details
Name: Scorekeep
Type: AWS::ECS::Container
Response distribution
Click and drag to select an area to zoom in on or use as a latency filter when viewing traces.
30%
20%
10%
0%
0
10ms
20ms
30ms
40ms
Duration
Response status
Choose response statuses to add to the filter when viewing traces.
Fault: 0%
Error: 0%
Throttle: 0%
OK: 100%
Analyze traces
View traces

Die Service-Karte zeigt den Zustand jedes Knotens an und markiert ihn farblich auf Grundlage des Verhältnisses zwischen erfolgreichen Anrufen und Fehlern:

- Grün für erfolgreiche Anrufe
- Rot für Server-Fehler (500er-Fehler)
- Gelb für Client-Fehler (400er-Fehler)
- Violett für Ablehnungsfehler („Fehler 429 – Zu viele Anfragen”)

Wenn Ihre Service-Map groß ist, können Sie die Karte mithilfe der Steuerelemente auf dem Bildschirm oder mit der Maus vergrößern und verkleinern und verschieben.

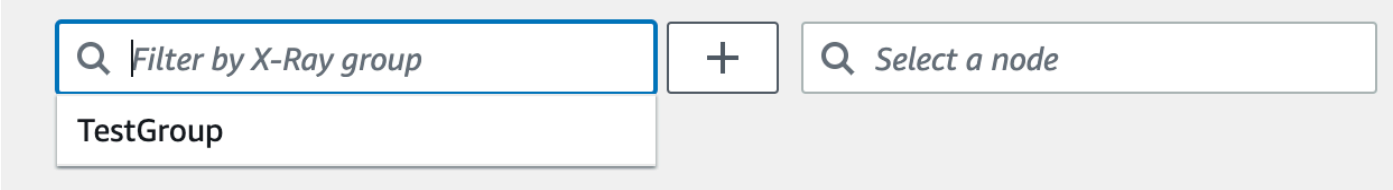
> **Note**
>
> Die X-Ray-Trace-Map kann bis zu 10.000 Knoten anzeigen. In seltenen Fällen, in denen die Gesamtzahl der Serviceknoten diesen Grenzwert überschreitet, kann es vorkommen, dass Sie eine Fehlermeldung erhalten und keine vollständige Trace-Map in der Konsole anzeigen können.

### Die Trace-Map nach Gruppen filtern

Mithilfe eines Filterausdrucks können Sie Kriterien definieren, anhand derer Traces in eine Gruppe aufgenommen werden sollen. Weitere Informationen zu Filterausdrücken finden Sie unter Verwenden von Filterausdrücken. Gehen Sie als Nächstes wie folgt vor, um diese bestimmte Gruppe dann in der Trace-Map anzuzeigen.

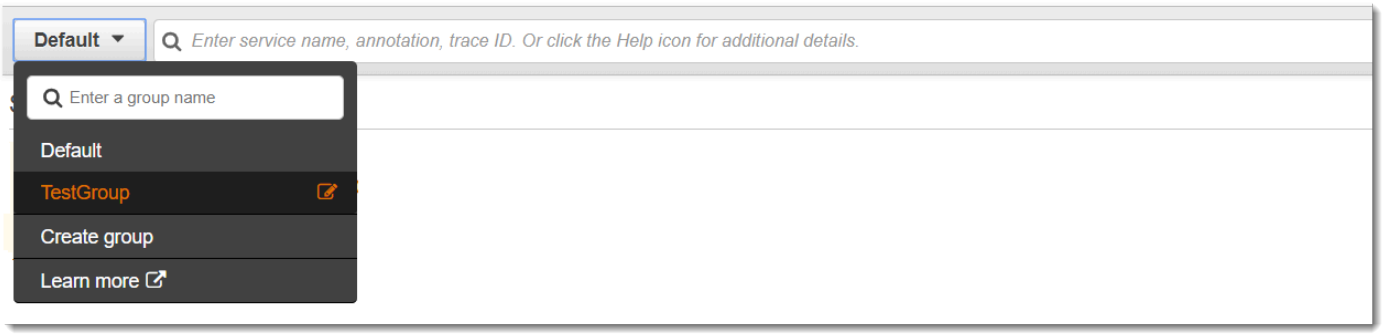
CloudWatch console

Wählen Sie einen Gruppennamen aus dem Gruppenfilter oben links auf der Trace-Map aus.




X-Ray console

Wählen Sie einen Gruppennamen aus dem Dropdown-Menü links neben der Suchleiste aus.




Die Service-Map wird nun so gefiltert, dass Traces angezeigt werden, die dem Filterausdruck der ausgewählten Gruppe entsprechen.

Legende und Optionen der Trace-Map

Die Trace-Map enthält eine Legende und mehrere Optionen zum Anpassen der Kartenanzeige.

CloudWatch console

Wählen Sie oben rechts auf der Karte das Drop-down-Menü Legende und Optionen aus. Wählen Sie aus, was in den Knoten angezeigt wird, darunter:

- Metrics zeigt die durchschnittliche Antwortzeit und die Anzahl der pro Minute während des ausgewählten Zeitraums gesendeten Traces an.
- Nodes zeigt das Dienstsymbol in jedem Knoten an.

Wählen Sie zusätzliche Karteneinstellungen aus dem Bereich Einstellungen aus, auf den Sie über das Zahnradsymbol oben rechts in der Karte zugreifen können. Zu diesen Einstellungen gehört die Auswahl, anhand welcher Metrik die Größe der einzelnen Knoten bestimmt wird und welche Kanarienvögel auf der Karte angezeigt werden sollen.

X-Ray console

Zeigen Sie die Legende der Servicekarte an, indem Sie oben rechts auf der Karte auf den Link Kartenlegende klicken. Unten rechts auf der Trace-Map können Optionen für die Servicekarte ausgewählt werden, darunter:

- Mit den Dienstsymbolen wird umgeschaltet, was innerhalb der einzelnen Knoten angezeigt wird. Dabei wird entweder das Servicesymbol oder die durchschnittliche Antwortzeit und die Anzahl der pro Minute während des ausgewählten Zeitraums gesendeten Traces angezeigt.

- Knotengröße: Bei „Keine“ werden alle Knoten auf dieselbe Größe gesetzt.
- Knotengröße: Health bewertet Knoten anhand der Anzahl der betroffenen Anfragen, einschließlich Fehlern, Störungen oder gedrosselten Anfragen.
- Knotengröße: Der Traffic berechnet die Größe der Knoten anhand der Gesamtzahl der Anfragen.

### Traces und Trace-Details anzeigen

Verwenden Sie die Seite „Traces“ in der X-Ray-Konsole, um Traces anhand von URL, Antwortcode oder anderen Daten aus der Trace-Zusammenfassung zu suchen. Nachdem Sie einen Trace aus der Trace-Liste ausgewählt haben, zeigt die Seite mit den Trace-Details eine Karte der Service-Knoten, die dem ausgewählten Trace zugeordnet sind, sowie eine Zeitleiste mit Trace-Segmenten an.

#### Anzeigen von Ablaufverfolgungen

CloudWatch console

Um Traces in der CloudWatch Konsole anzuzeigen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im linken Navigationsbereich X-Ray Traces und dann Traces aus. Sie können nach Gruppen filtern oder einen Filterausdruck eingeben, der die Spuren filtert, die im Abschnitt „Traces“ unten auf der Seite angezeigt werden. Weitere Informationen finden Sie unter Verwenden von Filterausdrücken.

   Alternativ können Sie die Service Map verwenden, um zu einem bestimmten Serviceknoten zu navigieren und sich dann die Traces anzusehen. Dadurch wird die Seite „Traces“ geöffnet, auf der bereits eine Abfrage angewendet wurde.
3. Verfeinern Sie Ihre Abfrage im Bereich Abfrageverfeinerungen. Um Traces nach einem gemeinsamen Attribut zu filtern, wählen Sie eine Option aus dem Abwärtspfeil neben Abfrage verfeinern nach. Es gibt die folgenden Optionen:

   - Knoten — Filtert Traces nach Dienstknoten.
   - Ressourcen-ARN — Filtert Traces nach einer Ressource, die einer Ablaufverfolgung zugeordnet ist. Beispiele für diese Ressourcen sind eine Amazon Elastic Compute Cloud (Amazon EC2) -Instance, eine AWS Lambda Funktion oder eine Amazon DynamoDB Tabelle.

- Benutzer — Filtert Traces mit einer Benutzer-ID.
- Meldung zur Fehlerursache — Filtert die Ablaufverfolgung nach der Fehlerursache.
- URL — Filtert Traces nach einem URL-Pfad, der von Ihrer Anwendung verwendet wird.
- HTTP-Statuscode — Filtert Traces nach dem von Ihrer Anwendung zurückgegebenen HTTP-Statuscode. Sie können einen benutzerdefinierten Antwortcode angeben oder aus den folgenden Optionen wählen:
  - `200`— Die Anfrage war erfolgreich.
  - `401`— Der Anfrage fehlten gültige Authentifizierungsdaten.
  - `403`— Der Anfrage fehlten gültige Berechtigungen.
  - `404`— Der Server konnte die angeforderte Ressource nicht finden.
  - `500`— Der Server ist auf einen unerwarteten Fehler gestoßen und hat einen internen Fehler generiert.

Wählen Sie einen oder mehrere Einträge aus und klicken Sie dann auf Zur Abfrage hinzufügen, um den Filterausdruck oben auf der Seite zu ergänzen.

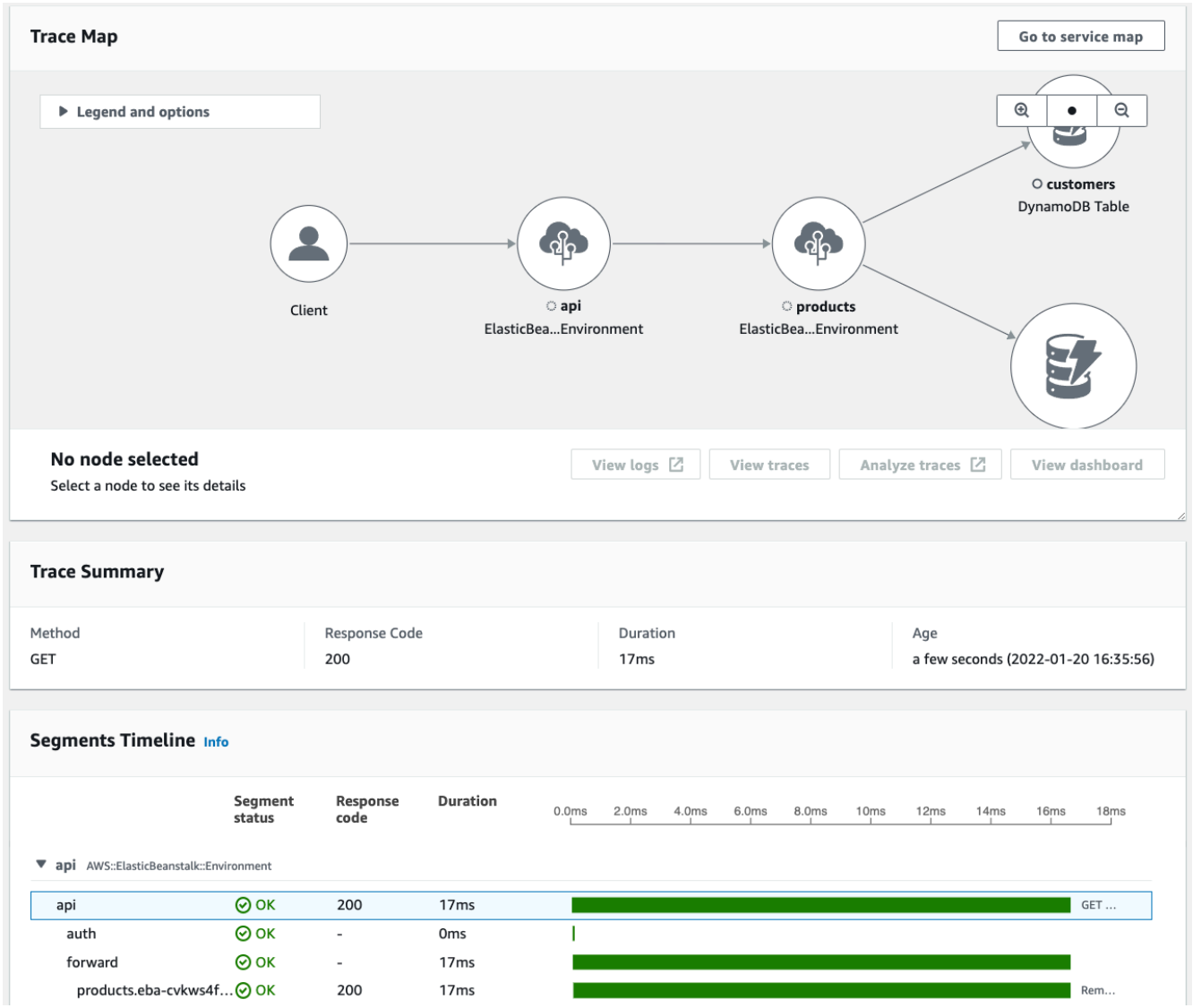
4. Um einen einzelnen Trace zu finden, geben Sie eine Trace-ID direkt in das Abfragefeld ein. Sie können das X-Ray-Format oder das World Wide Web Consortium (W3C) -Format verwenden. Ein Trace, der mit dem AWS Distro for erstellt wurde, hat beispielsweise das OpenTelemetry W3C-Format.

   > **Note**
   >
   > Wenn Sie Traces abfragen, die mit einer Trace-ID im W3C-Format erstellt wurden, zeigt die Konsole die entsprechende Ablaufverfolgung im X-Ray-Format an. Wenn Sie beispielsweise `4efaaf4d1e8720b39541901950019ee5` im W3C-Format abfragen, zeigt die Konsole das X-Ray-Äquivalent an:`1-4efaaf4d-1e8720b39541901950019ee5`.

5. Wählen Sie „Abfrage jederzeit ausführen“, um im Bereich „Traces“ unten auf der Seite eine Liste der passenden Traces anzuzeigen.
6. Um die Seite mit den Trace-Details für einen einzelnen Trace anzuzeigen, wählen Sie eine Trace-ID aus der Liste aus.

   Die folgende Abbildung zeigt eine Trace-Map mit Serviceknoten, die mit der Trace verknüpft sind, und Kanten zwischen den Knoten, die den Pfad der Segmente, aus denen sich die

Trace zusammensetzt, darstellen. Auf die Trace-Map folgt eine Trace-Zusammenfassung. Die Zusammenfassung enthält Informationen über einen GET Beispielvorgang, seinen Antwortcode, die Dauer der Ausführung der Ablaufverfolgung und das Alter der Anfrage. Die Segment-Timeline folgt der Trace-Zusammenfassung, die die Dauer der Trace-Segmente und -Untersegmente anzeigt.



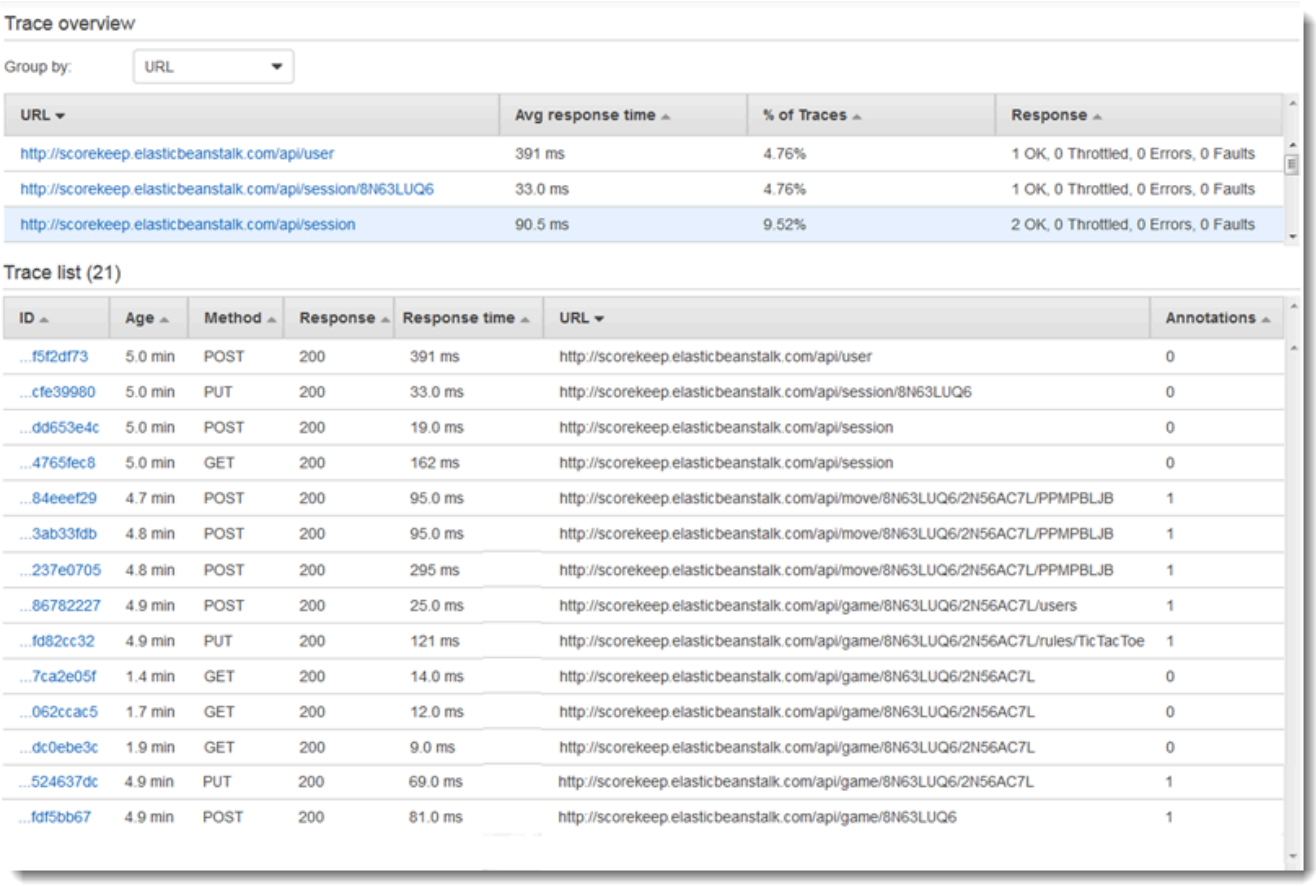

Wenn Sie über eine ereignisgesteuerte Anwendung verfügen, die Amazon SQS und Lambda verwendet, können Sie in der Trace-Map eine verknüpfte Ansicht der Traces für jede Anfrage sehen. In der Karte werden die Spuren von Nachrichtenerstellern mit den Spuren von AWS Lambda Verbrauchern verknüpft und als gestrichelte Linie dargestellt. Weitere Informationen zu ereignisgesteuerten Anwendungen finden Sie unter. Verfolgen Sie ereignisgesteuerte Anwendungen

Die Seiten Traces und Trace-Details unterstützen auch die kontenübergreifende Ablaufverfolgung, bei der Traces von mehreren Konten in der Trace-Liste und in einer einzigen Trace-Map aufgeführt werden können. Weitere Informationen finden Sie unter Kontenübergreifende Rückverfolgung.

## X-Ray console

So zeigen Sie Spuren in der X-Ray-Konsole an

1. Öffnen Sie die Traces-Seite in der X-Ray-Konsole. Im Bereich Trace-Übersicht wird eine Liste von Traces angezeigt, die nach gemeinsamen Merkmalen wie Fehlerursachen, ResourceARN und InstanceIdgruppiert sind.
2. Um ein allgemeines Merkmal auszuwählen, um einen gruppierten Satz von Traces anzuzeigen, erweitern Sie den Abwärtspfeil neben Gruppieren nach. Die folgende Abbildung zeigt eine Trace-Übersicht der Traces, die nach der URL für gruppiert sindAWS X-Ray Beispielanwendung, sowie eine Liste der zugehörigen Traces.

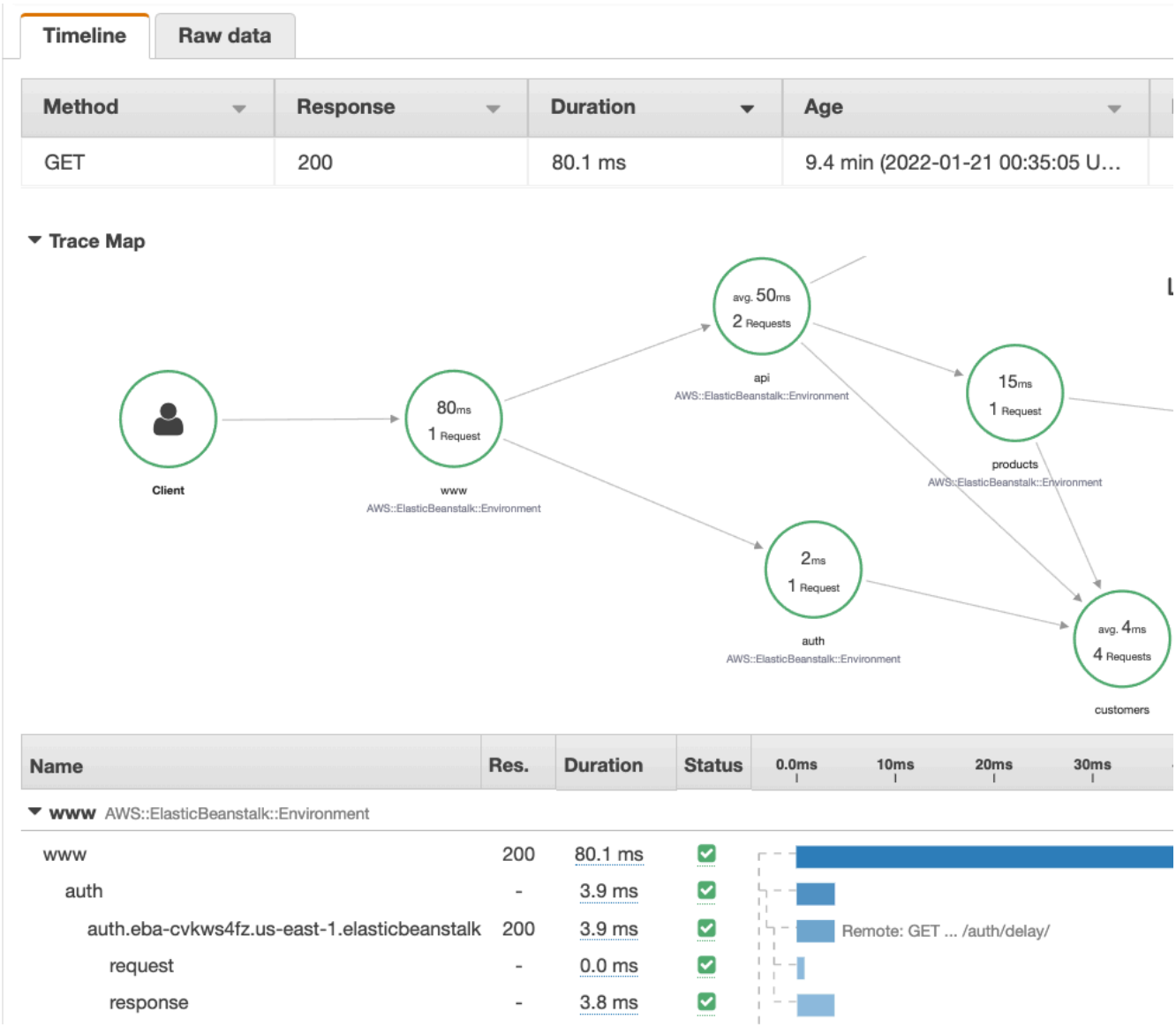


3. Wählen Sie die ID einer Ablaufverfolgung aus, um sie unter der Trace-Liste anzuzeigen. Sie können auch im Navigationsbereich die Option Service Map auswählen, um die Traces für einen bestimmten Serviceknoten anzuzeigen. Anschließend können Sie die mit diesem Knoten verknüpften Traces anzeigen.

   Auf der Registerkarte Zeitleiste wird der Anforderungsablauf für den Trace angezeigt. Er umfasst Folgendes:

   - Eine Karte des Pfads für jedes Segment im Trace.
   - Wie lange es gedauert hat, bis das Segment einen Knoten in der Trace-Map erreicht hat.
   - Wie viele Anfragen wurden an den Knoten in der Trace-Map gestellt.

   Die folgende Abbildung zeigt ein Beispiel für eine Trace-Map, die mit einer `GET` Anforderung verknüpft ist, die an eine Beispielanwendung gestellt wurde. Die Pfeile zeigen den Pfad, den jedes Segment zurückgelegt hat, um die Anfrage abzuschließen. Die Dienstknoten zeigen die Anzahl der Anfragen an, die während der `GET` Anfrage gestellt wurden.




Weitere Informationen zur Registerkarte „Zeitleiste“ finden Sie im folgenden Abschnitt „Überblick über die Ablaufverfolgung“.

Auf der Registerkarte Rohdaten werden Informationen über die Spur und die Segmente und Untersegmente, aus denen sich die Spur zusammensetzt, im JSON Format angezeigt. Diese Informationen können Folgendes beinhalten:

- Zeitstempel
- Eindeutige IDs
- Ressourcen, die dem Segment oder Untersegment zugeordnet sind
- Die Quelle oder der Ursprung des Segments oder Untersegments

- Zusätzliche Informationen über die Anfrage an Ihre Anwendung, z. B. die Antwort auf eine HTTP-Anfrage

### Erkunden Sie den Trace-Zeitplan

Der Abschnitt Zeitleiste zeigt eine Hierarchie von Segmenten und Untersegmenten neben einem horizontalen Balken, der der Zeit entspricht, die sie für die Erledigung ihrer Aufgaben aufgewendet haben. Der erste Eintrag in der Liste ist das Segment, das alle vom Service für eine einzelne Anforderung aufgezeichneten Daten darstellt. Untersegmente sind eingerückt und werden im Anschluss an das Segment aufgelistet. Spalten enthalten Informationen zu den einzelnen Segmenten.

CloudWatch console

In der CloudWatch Konsole bietet die Segment-Timeline die folgenden Informationen:

- Die erste Spalte: Listet die Segmente und Untersegmente in der ausgewählten Spur auf.
- Die Spalte Segmentstatus: Listet das Statusergebnis jedes Segments und Untersegments auf.
- Die Spalte Antwortcode: Listet einen HTTP-Antwortstatuscode auf eine Browseranfrage auf, die von dem Segment oder Untersegment gestellt wurde, sofern verfügbar.
- Die Spalte Dauer: Listet auf, wie lange das Segment oder Untersegment lief.
- Die Spalte Gehostet in: Listet den Namespace oder die Umgebung auf, in der das Segment oder Untersegment ausgeführt wird, falls zutreffend. Weitere Informationen finden Sie unter https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/AppSignals-StandardMetrics.html#AppSignals-StandardMetrics-Dimensions.
- Die letzte Spalte: Zeigt horizontale Balken an, die der Dauer entsprechen, in der das Segment oder Untersegment ausgeführt wurde, im Verhältnis zu den anderen Segmenten oder Untersegmenten in der Zeitleiste.

Um die Liste der Segmente und Untersegmente nach Serviceknoten zu gruppieren, aktivieren Sie die Option Nach Knoten gruppieren.

X-Ray console

Wählen Sie auf der Seite mit den Trace-Details die Registerkarte Timeline aus, um die Timeline für jedes Segment und Untersegment anzuzeigen, aus denen ein Trace besteht.

In der X-Ray-Konsole bietet die Timeline die folgenden Informationen:

- Die Spalte Name: Listet die Namen der Segmente und Untersegmente im Trace auf.
- Die Spalte Res.: Listet einen HTTP-Antwortstatuscode auf eine Browseranfrage auf, die von dem Segment oder Untersegment gestellt wurde, sofern verfügbar.
- Die Spalte Dauer: Listet auf, wie lange das Segment oder Untersegment lief.
- Die Spalte Status: Listet das Ergebnis des Status eines Segments oder Untersegments auf.
- Die letzte Spalte: Zeigt horizontale Balken an, die der Dauer entsprechen, in der das Segment oder Untersegment im Verhältnis zu den anderen Segmenten oder Untersegmenten in der Zeitleiste ausgeführt wurde.

Um die rohen Trace-Daten anzuzeigen, die die Konsole zur Generierung der Zeitleiste verwendet, wählen Sie die Registerkarte Rohdaten. Die Rohdaten zeigen Ihnen Informationen über den Trace und die Segmente und Untersegmente, aus denen sich der Trace zusammensetzt, im `JSON` Format. Diese Informationen können Folgendes beinhalten:

- Zeitstempel
- Eindeutige IDs
- Ressourcen, die dem Segment oder Untersegment zugeordnet sind
- Die Quelle oder der Ursprung des Segments oder Untersegments
- Zusätzliche Informationen über die Anfrage an Ihre Anwendung, z. B. die Antwort auf eine HTTP-Anfrage.

Wenn Sie ein instrumentiertes AWS SDK oder einen SQL Client verwendenHTTP, um externe Ressourcen aufzurufen, zeichnet das X-Ray-SDK Untersegmente automatisch auf. Sie können das X-Ray SDK auch verwenden, um benutzerdefinierte Untersegmente für jede Funktion oder jeden Codeblock aufzuzeichnen. Zusätzliche Untersegmente, die aufgezeichnet werden, während ein benutzerdefiniertes Untersegment geöffnet ist, werden zu untergeordneten Segmenten des benutzerdefinierten Untersegments.

Anzeigen von Segmentdetails

Wählen Sie in der Trace-Zeitleiste den Namen eines Segments aus, um dessen Details anzuzeigen.

Im Bereich „Segmentdetails“ werden die Registerkarten „Übersicht“, „Ressourcen“, „Anmerkungen“, „Metadaten“, „Ausnahmen“ und „SQL“ angezeigt. Es gilt Folgendes:

- Die Registerkarte Overview (Übersicht) zeigt Informationen zu Anforderung und Antwort an. Zu den Informationen gehören der Name, die Startzeit, die Endzeit, die Anforderungs-URL, der Anforderungsvorgang, der Anforderungsantwortcode sowie etwaige Fehler und Störungen.
- Auf der Registerkarte Ressourcen für ein Segment werden Informationen aus dem X-Ray SDK und zu den AWS Ressourcen angezeigt, auf denen Ihre Anwendung ausgeführt wird. Verwenden Sie die Amazon EC2- oder Amazon ECS-Plug-ins für das X-Ray SDK AWS Elastic Beanstalk, um servicespezifische Ressourceninformationen aufzuzeichnen. Weitere Informationen zu Plug-ins finden Sie im Abschnitt Service-Plugins unter. Konfiguration des X-Ray SDK for Java
- Auf den verbleibenden Registerkarten werden Anmerkungen, Metadaten und Ausnahmen angezeigt, die für das Segment aufgezeichnet wurden. Ausnahmen werden automatisch erfasst, wenn sie aus einer instrumentierten Anfrage generiert werden. Anmerkungen und Metadaten enthalten zusätzliche Informationen, die Sie mithilfe der vom X-Ray SDK bereitgestellten Operationen aufzeichnen. Verwenden Sie das X-Ray SDK, um Anmerkungen oder Metadaten zu Ihren Segmenten hinzuzufügen. Weitere Informationen finden Sie unter dem sprachspezifischen Link unter Instrumentierung Ihrer Anwendung mit SDKs in. AWS X-Ray Instrumentieren Sie Ihre Bewerbung für AWS X-Ray

### Anzeigen von Untersegmentdetails

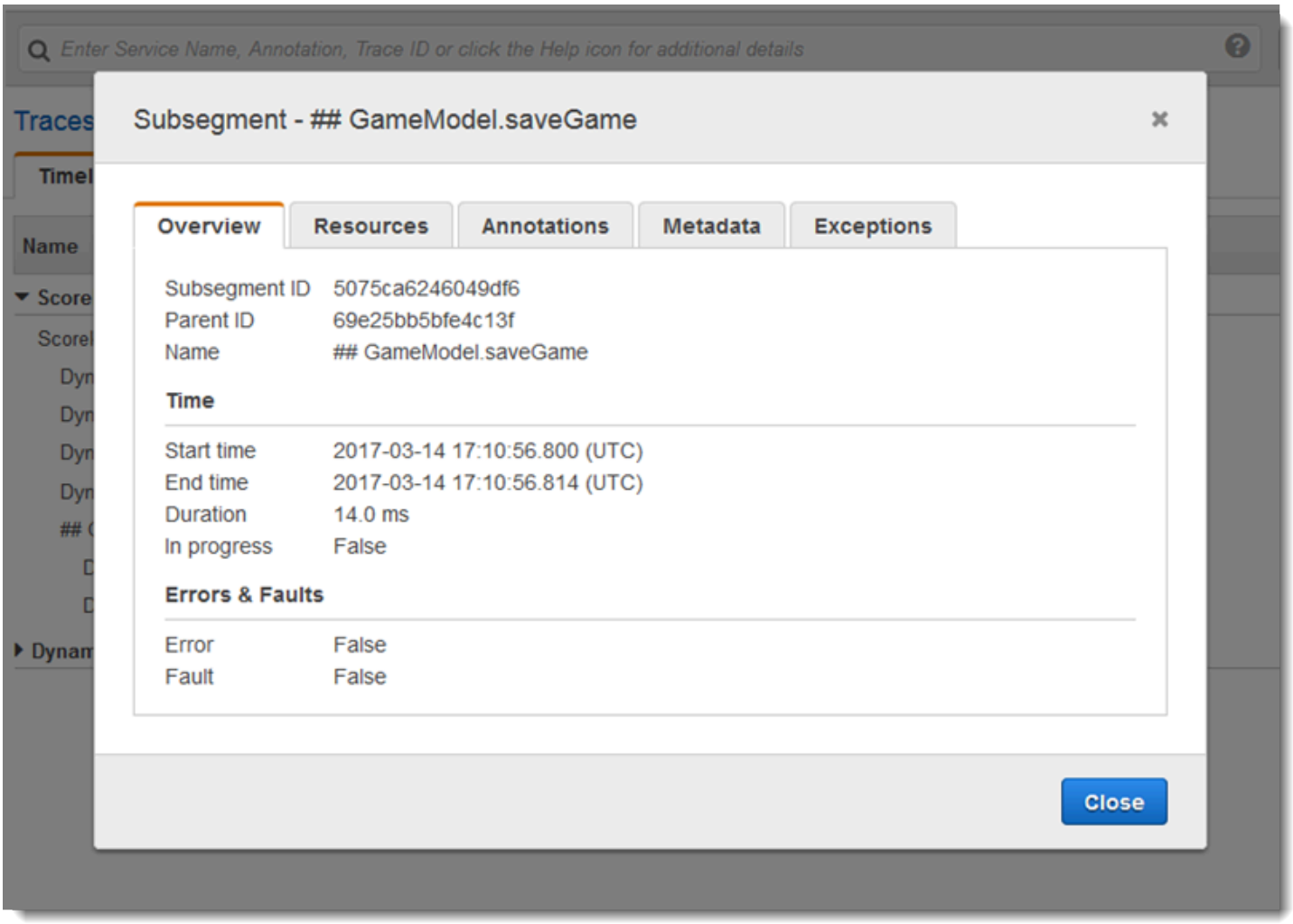
Wählen Sie in der Ablaufverfolgungszeitleiste den Namen eines Untersegments aus, um dessen Details anzuzeigen:

- Die Registerkarte „Übersicht“ enthält Informationen über die Anfrage und die Antwort. Dazu gehören der Name, die Startzeit, die Endzeit, die Dauer, die AnfrageURL, der Anforderungsvorgang, der Anforderungsantwortcode und alle Fehler und Störungen. Für die mit instrumentierten Clients generierten Untersegmente enthält die Registerkarte Overview (Übersicht) Informationen zu Anforderung und Antwort aus der Sicht Ihrer Anwendung.
- Auf der Registerkarte Ressourcen für ein Untersegment werden Details zu den AWS Ressourcen angezeigt, die zur Ausführung des Untersegments verwendet wurden. Die Registerkarte Ressourcen kann beispielsweise einen AWS Lambda Funktions-ARN, Informationen über eine DynamoDB-Tabelle, jede aufgerufene Operation und eine Anforderungs-ID enthalten.
- Auf den verbleibenden Registerkarten werden Anmerkungen, Metadaten und Ausnahmen angezeigt, die für das Untersegment aufgezeichnet wurden. Ausnahmen werden automatisch erfasst, wenn sie aus einer instrumentierten Anfrage generiert werden. Anmerkungen und Metadaten enthalten zusätzliche Informationen, die Sie mithilfe der vom X-Ray SDK bereitgestellten Operationen aufzeichnen. Verwenden Sie das X-Ray SDK, um Anmerkungen

oder Metadaten zu Ihren Segmenten hinzuzufügen. Weitere Informationen finden Sie unter dem sprachspezifischen Link unter Instrumentierung Ihrer Anwendung mit SDKs in. AWS X-Ray Instrumentieren Sie Ihre Bewerbung für AWS X-Ray

Bei benutzerdefinierten Untersegmenten zeigt die Registerkarte Overview (Übersicht) den Namen des Untersegments an, den Sie angeben können, um den Code-Bereich oder die Funktion anzugeben, der oder die aufgezeichnet wird. Weitere Informationen finden Sie unter dem sprachspezifischen Link, der unter Instrumentierung Ihrer Anwendung mit SDKs aufgeführt ist. AWS X-Ray Generieren von benutzerdefinierten Untersegmenten mit dem X-Ray SDK for Java

Die folgende Abbildung zeigt die Registerkarte „Übersicht“ für ein benutzerdefiniertes Untersegment. Die Übersicht enthält die Untersegment-ID, die übergeordnete ID, den Namen, die Start- und Endzeiten, die Dauer, den Status und die Fehler oder Störungen.



Die Registerkarte Metadaten für ein benutzerdefiniertes Untersegment enthält Informationen im JSON Format über die von diesem Untersegment verwendeten Ressourcen.

Verwenden Sie Filterausdrücke

Verwenden Sie Filterausdrücke, um eine Trace-Map oder Traces für eine bestimmte Anfrage, einen bestimmten Dienst, eine Verbindung zwischen zwei Diensten (ein Edge) oder Anfragen, die eine Bedingung erfüllen, anzuzeigen. X-Ray bietet eine Sprache für Filterausdrücke zum Filtern von Anfragen, Diensten und Kanten auf der Grundlage von Daten in Anforderungsheadern, Antwortstatus und indizierten Feldern in den ursprünglichen Segmenten.

Wenn Sie einen Zeitraum für die Anzeige von Traces in der X-Ray-Konsole auswählen, erhalten Sie möglicherweise mehr Ergebnisse, als die Konsole anzeigen kann. In der rechten oberen Ecke der Konsole wird die Anzahl der gescannten Ablaufverfolgungen angezeigt und Sie können ablesen, ob weitere Ablaufverfolgungen verfügbar sind. Sie können einen Filterausdruck verwenden, um die Ergebnisse auf die Spuren einzugrenzen, nach denen Sie suchen möchten.

Details zu Filterausdrücken

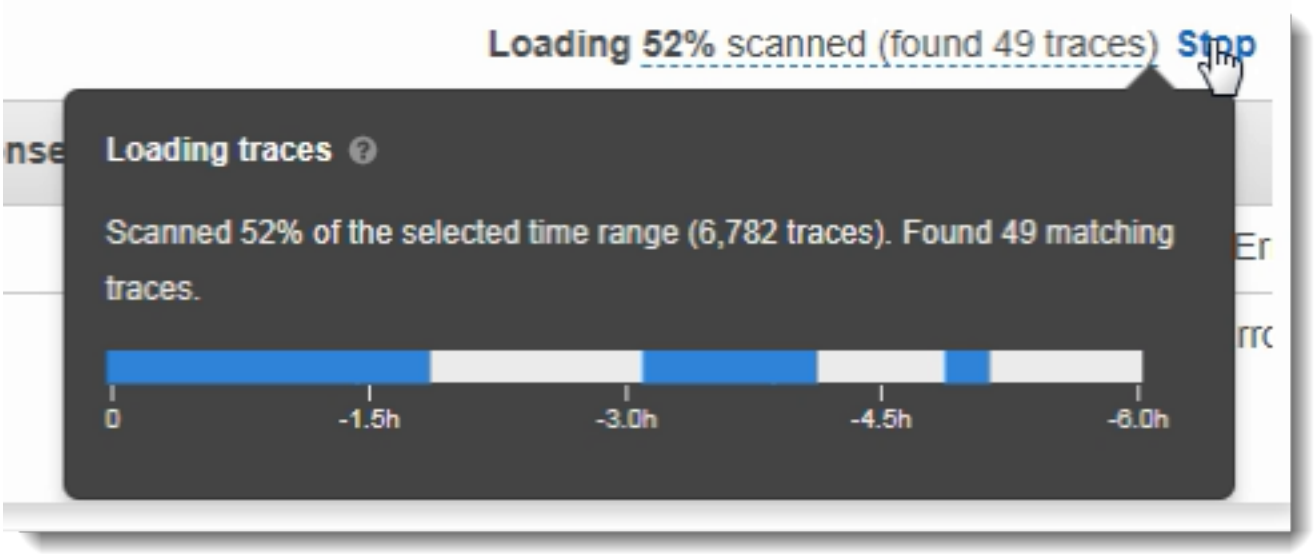
Wenn Sie einen Knoten in der Trace-Map auswählen, erstellt die Konsole einen Filterausdruck, der auf dem Dienstnamen des Knotens und den auf Ihrer Auswahl beruhenden Fehlertypen basiert. Um Ablaufverfolgungen zu finden, die Leistungsprobleme zeigen oder zu bestimmten Anfragen gehören, können Sie den von der Konsole bereitgestellten Ausdruck anpassen oder einen eigenen erstellen. Wenn Sie Anmerkungen mit dem X-Ray SDK hinzufügen, können Sie auch nach dem Vorhandensein eines Annotationsschlüssels oder dem Wert eines Schlüssels filtern.

> **Note**
>
> Wenn Sie in der Trace-Map einen relativen Zeitraum und einen Knoten auswählen, konvertiert die Konsole den Zeitraum in eine absolute Start- und Endzeit. Um sicherzustellen, dass die Ablaufverfolgungen für den Knoten in den Suchergebnissen angezeigt werden, und um das Scannen von Zeiträumen zu vermeiden, in denen der Knoten nicht aktiv war, werden nur Zeiten in den Zeitraum einbezogen, zu denen der Knoten Ablaufverfolgungen gesendet hat. Um relativ zur aktuellen Zeit zu suchen, können Sie wieder zum relativen Zeitraum auf der Seite der Ablaufverfolgungen wechseln und einen erneuten Scan durchführen.

Wenn immer noch mehr Ergebnisse vorliegen, als in der Konsole angezeigt werden können, gibt die Konsole die Anzahl der Ablaufverfolgungen an, die die Kriterien erfüllt haben, sowie die Anzahl

der gescannten Ablaufverfolgungen. Der angezeigte Prozentsatz entspricht dem Prozentsatz des ausgewählten Zeitraums, der gescannt wurde. Um sicherzustellen, dass alle passenden Ablaufverfolgungen in den Ergebnissen angezeigt werden, grenzen Sie Ihren Filterausdruck weiter ein oder wählen einen kürzeren Zeitraum aus.

Die Konsole beginnt den Scan am Ende des Zeitraums und arbeitet sich nach vorne durch, um Ihnen die aktuellsten Ergebnisse zuerst anzuzeigen. Wenn viele Ablaufverfolgungen vorliegen, aber nur wenige Ergebnisse, teilt die Konsole den Zeitraum in Abschnitte ein und scannt diese parallel. Der Fortschrittsbalken zeigt die Abschnitte des Zeitraums an, die gescannt wurden.




Verwenden Sie Filterausdrücke mit Gruppen

Gruppen sind eine Sammlung von Ablaufverfolgungen, die durch einen Filterausdruck definiert sind. Sie können Gruppen verwenden, um zusätzliche Servicegraphen zu erstellen und CloudWatch Amazon-Metriken bereitzustellen.

Gruppen werden über ihren Namen oder einen Amazon-Ressourcennamen (ARN) identifiziert und enthalten einen Filterausdruck. Der Service vergleicht eingehende Ablaufverfolgungen mit dem Ausdruck und speichert sie entsprechend.

Sie können Gruppen mit dem Dropdown-Menü links neben der Suchleiste des Filterausdrucks erstellen und ändern.

> **Note**
>
> Wenn der Service einen Fehler beim Qualifizieren einer Gruppe feststellt, ist diese Gruppe nicht mehr in der Verarbeitung eingehender Ablaufverfolgungen enthalten und es wird eine Fehlermetrik aufgezeichnet.

Weitere Informationen zu Gruppen finden Sie unterGruppen konfigurieren.

Syntax für Filterausdrücke

Filterausdrücke können ein Schlüsselwort, einen unären oder binären Operator und einen Wert für den Vergleich enthalten.

```
keyword operator value
```

Für verschiedene Arten von Schlüsselwörtern sich unterschiedliche Operatoren erhältlich. Beispielsweise ist `responsetime` ein Zahlenschlüsselwort, das mit auf Zahlen bezüglichen Operatoren verglichen werden kann.

Example — Anfragen, bei denen die Antwortzeit mehr als 5 Sekunden betrug

```
responsetime > 5
```

Sie können mehrere Ausdrücke zu einem zusammengesetzten Ausdruck verbinden, indem Sie die Operatoren `AND` oder `OR` verwenden.

Example — Anfragen mit einer Gesamtdauer von 5—8 Sekunden

```
duration >= 5 AND duration <= 8
```

Einfache Schlüsselwörter und Operatoren finden Probleme nur auf der Ebene der Ablaufverfolgungen. Wenn ein Fehler dahinter auftritt, von Ihrer Anwendung jedoch berücksichtigt und nicht an den Benutzer zurückgegeben wird, findet eine Suche nach `error` diesen Fehler nicht.

Um Spuren mit nachgelagerten Problemen zu finden, können Sie die komplexen Schlüsselwörter `service()` und verwenden. `edge()` Diese Schlüsselwörter ermöglichen die Anwendung eines Filterausdrucks auf alle nachgeordneten Knoten, einen einzelnen nachgeordneten Knoten oder

eine Edge zwischen zwei Knoten. Weitere Informationen zu diesen Schlüsselwörtern finden Sie im folgenden Abschnitt zu komplexen Schlüsselwörtern. Für mehr Granularität können Sie Services und Edges mit der id()-Funktion nach Typ filtern. Weitere Informationen finden Sie im folgenden Abschnitt zur ID-Funktion.

Boolesche Schlüsselwörter

Boolesche Schlüsselwortwerte sind entweder „true“ oder „false“. Verwenden Sie diese Schlüsselwörter, um Ablaufverfolgungen zu finden, die zu Fehlern geführt haben.

Boolesche Schlüsselwörter

- `ok`— Der Statuscode der Antwort lautete 2XX Success.
- `error`— Der Antwortstatuscode lautete 4XX Client Error.
- `throttle`— Der Antwortstatuscode lautete 429 Too Many Requests.
- `fault`— Der Antwortstatuscode lautete 5XX Server Error.
- `partial`— Die Anfrage enthält unvollständige Segmente.
- `inferred`— Die Anfrage hat abgeleitete Segmente.
- `first`— Element ist das erste Element einer aufgezählten Liste.
- `last`— Element ist das letzte Element einer Aufzählungsliste.
- `remote`— Die Ursachenentität ist entfernt.
- `root`— Service ist der Einstiegspunkt oder das Stammsegment einer Ablaufverfolgung.

Boolesche Operatoren finden Segmente, bei denen der angegebene Schlüssel `true` oder `false` ist.

Boolesche Operatoren

- none — Der Ausdruck ist wahr, wenn das Schlüsselwort wahr ist.
- `!`— Der Ausdruck ist wahr, wenn das Schlüsselwort falsch ist.
- `=`, `!=` — Vergleicht den Wert des Schlüsselworts mit der Zeichenfolge `true` oder`false`. Diese Operatoren verhalten sich genauso wie die anderen Operatoren, sind jedoch expliziter.

Example — Der Antwortstatus ist 2XX OK

```
ok
```

Example — der Antwortstatus ist nicht 2XX OK

```
!ok
```

Example — der Antwortstatus ist nicht 2XX OK

```
ok = false
```

Example — Der letzte aufgezählte Fehler-Trace hat den Fehlernamen „deserialize“

```
rootcause.fault.entity { last and name = "deserialize" }
```

Example — Anfragen mit Remote-Segmenten, bei denen die Abdeckung größer als 0,7 ist und der Dienstname „traces“ lautet

```
rootcause.responsetime.entity { remote and coverage > 0.7 and name = "traces" }
```

Example — Anfragen mit abgeleiteten Segmenten, bei denen der Diensttyp „aws:DynamoDB“ ist

```
rootcause.fault.service { inferred and name = traces and type = "AWS::DynamoDB" }
```

Example — Anfragen, die ein Segment mit dem Namen „Datenebene“ als Stamm haben

```
service("data-plane") {root = true and fault = true}
```

Zahlenschlüsselwörter

Verwenden Sie Zahlenschlüsselwörter, um nach Anforderungen mit einer bestimmten Reaktionszeit, Dauer oder einem bestimmten Reaktionsstatus zu suchen.

Zahlenschlüsselwörter

- `responsetime`— Zeit, die der Server benötigt hat, um eine Antwort zu senden.
- `duration`— Gesamtdauer der Anfrage, einschließlich aller Downstream-Aufrufe.
- `http.status`— Statuscode der Antwort.
- `index`— Position eines Elements in einer aufgezählten Liste.
- `coverage`— Dezimaler Prozentsatz der Antwortzeit der Entität im Vergleich zur Antwortzeit des Stammsegments. Gilt nur für die Reaktionszeit der Ursachenentitäten.

### Zahlenoperatoren

Zahlenschlüsselwörter verwenden Standard-Operatoren für Gleichheit und Vergleich.

- =, != — Das Schlüsselwort ist gleich oder ungleich einem Zahlenwert.
- <,<=,>, >= — Das Schlüsselwort ist kleiner oder größer als ein Zahlenwert.

Example — Der Antwortstatus ist nicht 200 OK

```
http.status != 200
```

Example — Anfrage, bei der die Gesamtdauer 5—8 Sekunden betrug

```
duration >= 5 AND duration <= 8
```

Example — Anfragen, die in weniger als 3 Sekunden erfolgreich abgeschlossen wurden, einschließlich aller Downstream-Aufrufe

```
ok !partial duration <3
```

Example — Entität mit einer aufgezählten Liste, deren Index größer als 5 ist

```
rootcause.fault.service { index > 5 }
```

Example — Anfragen, bei denen die letzte Entität, deren Abdeckung größer als 0,8 ist

```
rootcause.responsetime.entity { last and coverage > 0.8 }
```

### Zeichenfolgen-Schlüsselwörter

Verwenden Sie String-Schlüsselwörter, um Ablaufverfolgungen mit bestimmtem Text in den Anforderungsheadern oder bestimmte Benutzer-IDs zu finden.

### Zeichenfolgen-Schlüsselwörter

- `http.url`— URL anfordern.
- `http.method`— Methode anfordern.
- `http.useragent`— User-Agent-Zeichenfolge anfordern.

- `http.clientip`— Die IP-Adresse des Anforderers.
- `user`— Wert des Benutzerfeldes in einem beliebigen Segment im Trace.
- `name`— Der Name eines Dienstes oder einer Ausnahme.
- `type`— Art der Dienstleistung.
- `message`— Ausnahmemeldung.
- `availabilityzone`— Wert des AvailabilityZone-Felds für ein beliebiges Segment im Trace.
- `instance.id`— Wert des Instanz-ID-Felds für ein beliebiges Segment im Trace.
- `resource.arn`— Wert des Ressourcen-ARN-Felds für ein beliebiges Segment im Trace.

Zeichenfolgenoperatoren finden Werte, die identisch mit bestimmtem Text sind oder diesen enthalten. Werte müssen stets in Anführungszeichen angegeben werden.

Zeichenfolgenoperatoren

- `=`, `!=` — Das Schlüsselwort ist gleich oder ungleich einem Zahlenwert.
- `CONTAINS`— Das Schlüsselwort enthält eine bestimmte Zeichenfolge.
- `BEGINSWITH`, `ENDSWITH` — Das Schlüsselwort beginnt oder endet mit einer bestimmten Zeichenfolge.

Example — http.url-Filter

```
http.url CONTAINS "/api/game/"
```

Um zu testen, ob ein Feld in einer Ablaufverfolgung vorhanden ist (unabhängig von seinem Wert), prüfen Sie, ob es die leere Zeichenfolge enthält.

Example — Benutzerfilter

Findet alle Ablaufverfolgungen mit Benutzer-IDs.

```
user CONTAINS ""
```

Example — wählt Traces mit einer Fehlerursache aus, zu der auch ein Dienst mit dem Namen „Auth“ gehört

```
rootcause.fault.service { name = "Auth" }
```

Example — wählt Traces mit einer Hauptursache für die Antwortzeit aus, deren letzter Service den Typ DynamoDB hat

```
rootcause.responsetime.service { last and type = "AWS::DynamoDB" }
```

Example — wählt Traces mit einer Fehlerursache aus, deren letzte Ausnahme die Meldung „Zugriff verweigert für account_id: 1234567890" enthält

```
rootcause.fault.exception { last and message = "Access Denied for account_id:
 1234567890"
```

Komplexe Schlüsselwörter

Verwenden Sie komplexe Schlüsselwörter, um Anfragen anhand des Servicenamens, des Edge-Namens oder des Anmerkungswerts zu finden. Für Services und Edges können Sie einen zusätzlichen Filterausdruck angeben, der sich auf den Service oder die Edge bezieht. Für Anmerkungen können Sie nach dem Wert einer Anmerkung mit einem bestimmten Schlüssel filtern und dafür boolesche Werte, Zahlen oder Zeichenfolgenoperatoren verwenden.

Komplexe Schlüsselwörter

- `annotation.`*`key`*`—` *`Wert einer Anmerkung mit Feldschlüssel.`* Der Wert einer Anmerkung kann ein boolescher Wert, eine Zahl oder eine Zeichenfolge sein, sodass Sie jeden der Vergleichsoperatoren dieser Typen verwenden können. Sie können dieses Schlüsselwort in Kombination mit dem `edge` Schlüsselwort `service` or verwenden.
- `edge(`*`source`*`, `*`destination`*`) {`*`filter`*`}`— Verbindung zwischen *`Quelle`* und *`Ziel`* der Dienste. Optionale geschweifte Klammern können einen Filterausdruck enthalten, der für Segmente in dieser Verbindung gilt.
- `group.`*`name`* `/ group.`*`arn`*— Der Wert des Filterausdrucks einer Gruppe, auf den durch den Gruppennamen oder Gruppen-ARN verwiesen wird.
- `json`— JSON-Ursachenobjekt. Schritte zum programmgesteuerten Erstellen von JSON-Entitäten finden Sie unter Daten aus AWS X-Ray abrufen.
- `service(`*`name`*`) {`*`filter`*`}`*`— Dienst mit Namensname.`* Optionale geschweifte Klammern können einen Filterausdruck enthalten, der für vom Service erstellte Segmente gilt.

Verwenden Sie das Schlüsselwort service, um Traces für Anfragen zu finden, die einen bestimmten Knoten auf Ihrer Trace-Map erreichen.

Komplexe Schlüsselwort-Operatoren finden Segmente, in denen der angegebene Schlüssel gesetzt wurde oder nicht.

Komplexe Schlüsselwort-Operatoren

- none — Der Ausdruck ist wahr, wenn das Schlüsselwort gesetzt ist. Wenn das Schlüsselwort vom Typ Boolean ist, wird es als boolescher Wert ausgewertet.
- `!` — Der Ausdruck ist wahr, wenn das Schlüsselwort nicht gesetzt ist. Wenn das Schlüsselwort vom Typ Boolean ist, wird es als boolescher Wert ausgewertet.
- =, `!=` — Vergleicht den Wert des Schlüsselworts.
- `edge(`*`source`*`, `*`destination`*`) {`*`filter`*`}` — Verbindung zwischen *Quelle* und *Ziel* der Dienste. Optionale geschweifte Klammern können einen Filterausdruck enthalten, der für Segmente in dieser Verbindung gilt.
- `annotation.`*`key`* — Wert einer Anmerkung mit *Feldschlüssel*. Der Wert einer Anmerkung kann ein boolescher Wert, eine Zahl oder eine Zeichenfolge sein, sodass Sie jeden der Vergleichsoperatoren dieser Typen verwenden können. Sie können dieses Schlüsselwort in Kombination mit dem `edge` Schlüsselwort `service` or verwenden.
- `json` — JSON-Ursachenobjekt. Schritte zum programmgesteuerten Erstellen von JSON-Entitäten finden Sie unter Daten aus AWS X-Ray abrufen.

Verwenden Sie das Schlüsselwort service, um Traces für Anfragen zu finden, die einen bestimmten Knoten auf Ihrer Trace-Map erreichen.

Example — Servicefilter

Anforderungen mit einem Aufruf an `api.example.com` mit einem Fehler (Fehler der Serie 500).

```
service("api.example.com") { fault }
```

Sie können den Servicenahmen ausschließen, um einen Filterausdruck auf alle Knoten in Ihrer Service-Übersicht anzuwenden.

Example — Servicefilter

Anfragen, die irgendwo auf Ihrer Trace-Map einen Fehler verursacht haben.

```
service() { fault }
```

Das Edge-Schlüsselwort wendet einen Filterausdruck auf eine Verbindung zwischen zwei Knoten an.

Example — Kantenfilter

Anforderung, bei der der Service `api.example.com` einen Aufruf an `backend.example.com` vorgenommen hat, der mit einem Fehler fehlgeschlagen ist.

```
edge("api.example.com", "backend.example.com") { error }
```

Sie können auch den `!`-Operator mit Service- und Edge-Schlüsselwörtern verwenden, um einen Service oder eine Edge aus den Ergebnissen eines anderen Filterausdrucks auszuschließen.

Example — Service- und Anfragefilter

Anforderung, bei der die URL mit `http://api.example.com/` beginnt und `/v2/` enthält, aber kein Service mit dem Namen `api.example.com` erreicht wird.

```
http.url BEGINSWITH "http://api.example.com/" AND http.url CONTAINS "/v2/" AND !
service("api.example.com")
```

Example — Service- und Antwortzeitfilter

Findet Spuren, bei denen der Wert eingestellt `http url` ist und die Reaktionszeit länger als 2 Sekunden ist.

```
http.url AND responseTime > 2
```

Für Anmerkungen können Sie alle Traces aufrufen, bei denen der Wert gesetzt `annotation.`*`key`* ist, oder die Vergleichsoperatoren verwenden, die dem Wertetyp entsprechen.

Example — Anmerkung mit Zeichenkettenwert

Anforderungen mit einer Anmerkung mit dem Namen `gameid` und dem Zeichenfolgenwert `"817DL6V0"`.

```
annotation.gameid = "817DL6V0"
```

Example — Anmerkung ist gesetzt

Anfragen mit einer Anmerkung namens `age` set.

```
annotation.age
```

Example — Anmerkung ist nicht gesetzt

Anfragen ohne eine Anmerkung namens `age` set.

```
!annotation.age
```

Example — Anmerkung mit Zahlenwert

Anforderungen mit einer Anmerkung mit einem numerischen Wert größer als 29.

```
annotation.age > 29
```

Example — Anmerkung in Kombination mit Service oder Edge

```
service { annotation.request_id = "917DL6VO" }
```

```
edge { source.annotation.request_id = "916DL6VO" }
```

```
edge { destination.annotation.request_id = "918DL6VO" }
```

Example — Gruppe mit Benutzer

Anfragen, bei denen Traces auf den `high_response_time` Gruppenfilter treffen (z. B.`responseTime > 3`) und der Benutzer den Namen Alice hat.

```
group.name = "high_response_time" AND user = "alice"
```

Example — JSON mit der Ursachenentität

Anforderungen mit übereinstimmenden Ursachenentitäten.

```
rootcause.json = #[{ "Services": [ { "Name": "GetWeatherData", "EntityPath": [{ "Name":
 "GetWeatherData" }, { "Name": "get_temperature" } ] }, { "Name": "GetTemperature",
 "EntityPath": [ { "Name": "GetTemperature" } ] } ] }]
```

id-Funktion

Wenn Sie einen Service-Namen für die Schlüsselwörter `service` oder `edge` bereitstellen, erhalten Sie Ergebnisse für alle Knoten mit diesem Namen. Für eine präzisere Filterung können Sie mit der `id`-Funktion zusätzlich zu einem Namen einen Servicetyp angeben, um zwischen Knoten mit demselben Namen zu differenzieren.

Verwenden Sie die `account.id` Funktion, um ein bestimmtes Konto für den Dienst anzugeben, wenn Sie Traces von mehreren Konten in einem Monitoring-Konto anzeigen.

```
id(name: "service-name", type:"service::type", account.id:"account-ID")
```

Sie können die `id`-Funktion anstelle eines Service-Namens in Service- und Edge-Filtern verwenden.

```
service(id(name: "service-name", type:"service::type")) { filter }
```

```
edge(id(name: "service-one", type:"service::type"), id(name: "service-two",
 type:"service::type")) { filter }
```

AWS Lambda Funktionen führen beispielsweise zu zwei Knoten in der Trace-Map; einer für den Funktionsaufruf und einer für den Lambda-Service. Die zwei Knoten haben denselben Namen, aber unterschiedliche Arten. Ein Standard-Servicefilter findet Ablaufnachverfolgungen für beide.

Example — Dienstfilter

Anforderungen, die einen Fehler bei einem Service mit dem Namen `random-name` enthalten.

```
service("function-name") { error }
```

Verwenden Sie die `id`-Funktion zur Eingrenzung der Suche auf Fehler bei der Funktion selbst, unter Ausschluss von Fehlern vom Service.

Example — Servicefilter mit ID-Funktion

Anforderungen, die einen Fehler bei einem Service mit dem Namen `random-name` und dem Typ `AWS::Lambda::Function` enthalten.

```
service(id(name: "random-name", type: "AWS::Lambda::Function")) { error }
```

Um Knoten nach Typ zu suchen, können Sie den Namen auch vollständig ausschließen.

Example — Servicefilter mit ID-Funktion und Servicetyp

Anforderungen, die einen Fehler bei einem Service des Typs `AWS::Lambda::Function` enthalten.

```
service(id(type: "AWS::Lambda::Function")) { error }
```

Um nach Knoten für einen bestimmten Knoten zu suchen AWS-Konto, geben Sie eine Konto-ID an.

Example — Servicefilter mit ID-Funktion und Konto-ID

Anfragen, die einen Dienst innerhalb einer bestimmten Konto-ID beinhalten`AWS::Lambda::Function`.

```
service(id(account.id: "account-id"))
```

Kontenübergreifende Rückverfolgung

AWS X-Ray unterstützt kontenübergreifende Beobachtbarkeit, sodass Sie Anwendungen überwachen und Fehler beheben können, die sich über mehrere Konten innerhalb eines AWS-Region Sie können Ihre Metriken, Protokolle und Traces in allen verknüpften Konten nahtlos suchen, visualisieren und analysieren, als ob Sie in einem einzigen Konto arbeiten würden. Auf diese Weise erhalten Sie einen vollständigen Überblick über Anfragen, die über mehrere Konten hinweg eingehen. Sie können kontenübergreifende Traces in der X-Ray-Trace-Map und auf den Traces-Seiten in der CloudWatchKonsole anzeigen.

Die gemeinsam genutzten Observability-Daten können jede der folgenden Telemetriearten beinhalten:

- Metriken bei Amazon CloudWatch
- Gruppen in Amazon CloudWatch Logs protokollieren
- Spuren in AWS X-Ray
- Anwendungen in Amazon CloudWatch Application Insights

Konfigurieren Sie die kontoübergreifende Beobachtbarkeit

Um die kontenübergreifende Observability zu aktivieren, richten Sie ein oder mehrere AWS Monitoring-Konten ein und verknüpfen Sie sie mit mehreren Quellkonten. Ein Monitoring-Konto ist ein zentrales System AWS-Konto , das Beobachtbarkeitsdaten, die aus Quellkonten generiert

wurden, einsehen und mit ihnen interagieren kann. Ein Quellkonto ist eine Person AWS-Konto , die Beobachtbarkeitsdaten für die darin enthaltenen Ressourcen generiert.

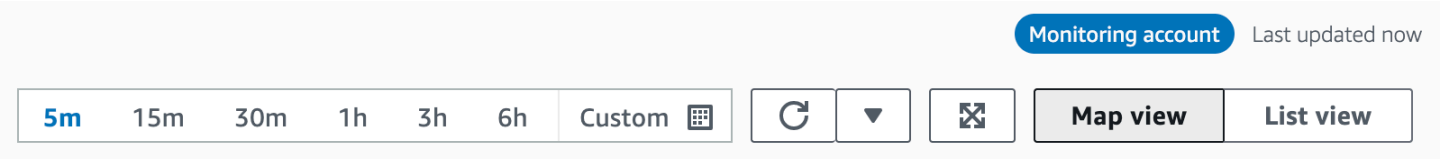
Quellkonten teilen ihre Beobachtbarkeitsdaten mit Monitoringkonten. Traces werden von jedem Quellkonto auf bis zu fünf Überwachungskonten kopiert. Kopien der Traces von den Quellkonten auf das erste Überwachungskonto sind kostenlos. Kopien von Traces, die an zusätzliche Überwachungskonten gesendet werden, werden jedem Quellkonto auf der Grundlage der Standardpreise in Rechnung gestellt. Weitere Informationen finden Sie unter AWS X-Ray Preise und CloudWatch Amazon-Preise.

Verwenden Sie die CloudWatch Konsole oder die neuen Observability Access Manager-Befehle in der AND-API, um Verknüpfungen zwischen Monitoring-Konten AWS CLI und Quellkonten zu erstellen. Weitere Informationen finden Sie unter CloudWatch kontoübergreifende Observabilität.

> **Note**
>
> Röntgenspuren werden dort abgerechnet, AWS-Konto wo sie empfangen wurden. Wenn sich eine Stichprobenanfrage über mehrere Dienste erstreckt AWS-Konto, zeichnet jedes Konto eine separate Ablaufverfolgung auf, und alle Traces haben dieselbe Trace-ID. Weitere Informationen zu kontenübergreifenden Observability-Preisen finden Sie unter Preise und AWS X-Ray CloudWatch Amazon-Preise.

### Kontoübergreifende Traces anzeigen

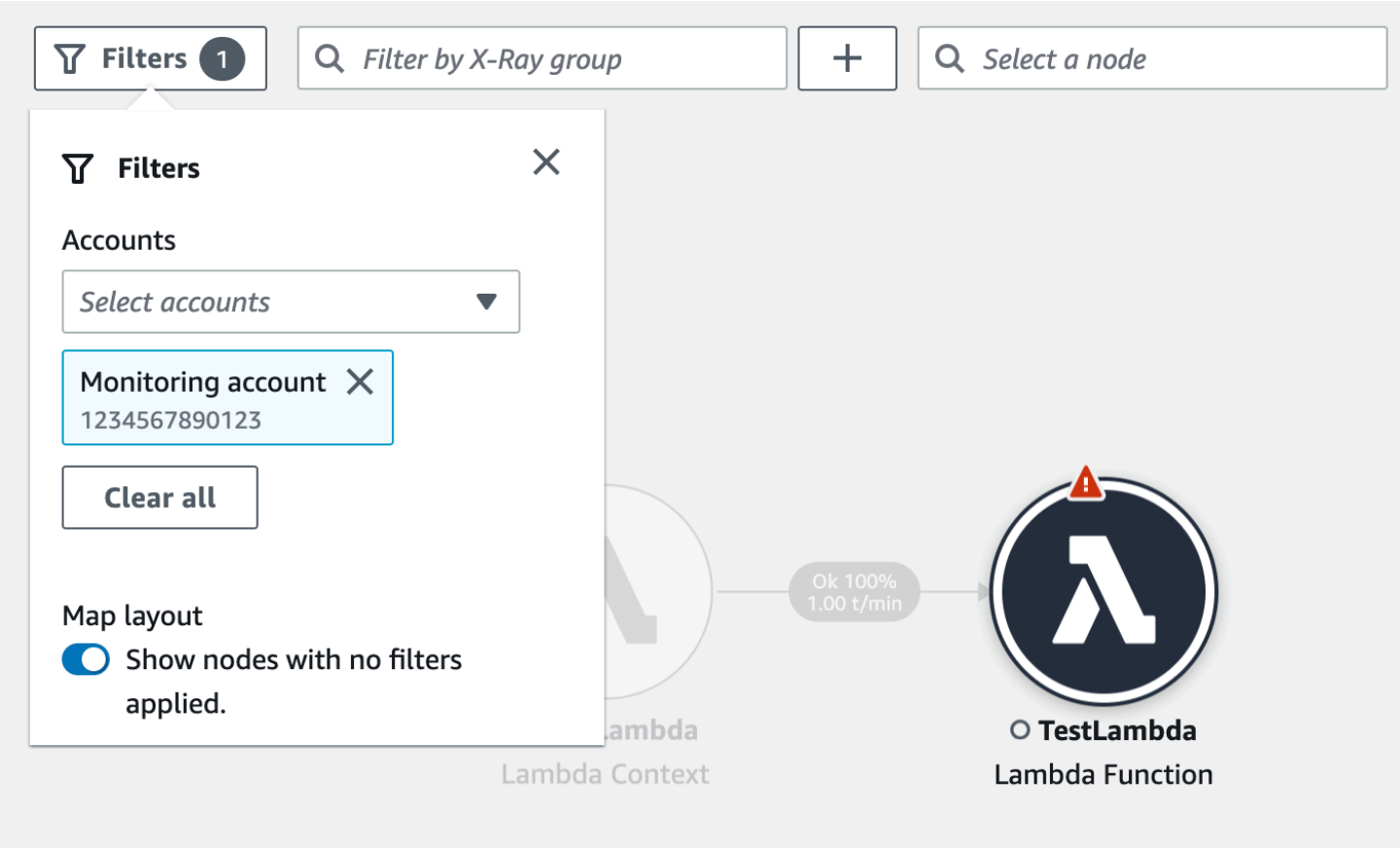
Kontoübergreifende Ablaufverfolgungen werden im Überwachungskonto angezeigt. Für jedes Quellkonto werden nur lokale Ablaufverfolgungen für dieses spezifische Konto angezeigt. In den folgenden Abschnitten wird davon ausgegangen, dass Sie beim Monitoring-Konto angemeldet sind und die CloudWatch Amazon-Konsole geöffnet haben. Sowohl auf der Trace-Map- als auch auf der Traces-Seite wird in der oberen rechten Ecke ein Abzeichen für das Überwachungskonto angezeigt.



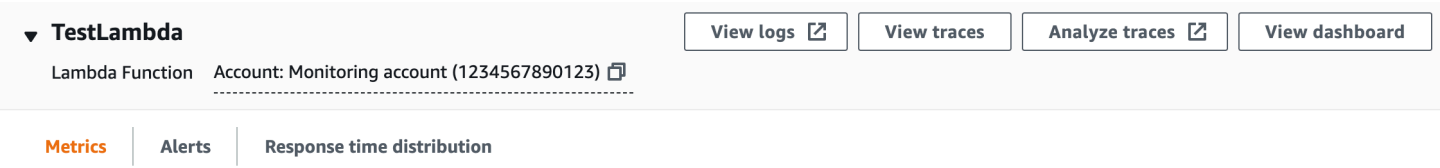

### Karte verfolgen

Wählen Sie in der CloudWatch Konsole im linken Navigationsbereich unter X-Ray-Traces die Option Trace Map aus. In der Trace-Map werden standardmäßig Knoten für alle Quellkonten angezeigt,

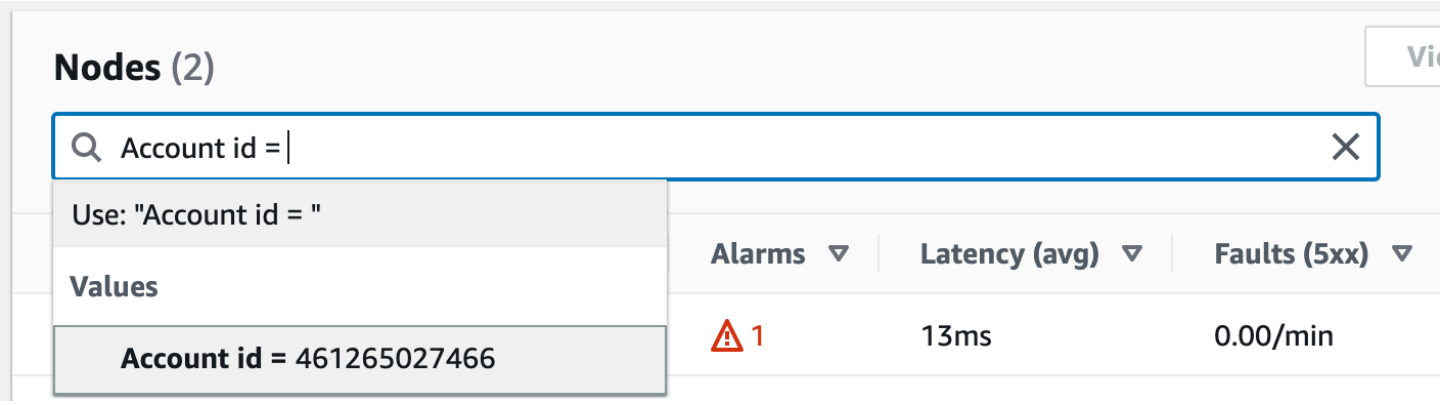
die Traces an das Überwachungskonto senden, sowie Knoten für das Überwachungskonto selbst. Wählen Sie auf der Trace-Map oben links Filter aus, um die Trace-Map mithilfe der Drop-down-Liste Konten zu filtern. Nachdem ein Kontofilter angewendet wurde, sind Dienstknoten von Konten, die nicht dem aktuellen Filter entsprechen, ausgegraut.




Wenn Sie einen Dienstknoten auswählen, enthält der Bereich mit den Knotendetails die Konto-ID und das Label des Dienstes.




Wählen Sie in der oberen rechten Ecke der Trace-Map die Option Listenansicht aus, um eine Liste der Dienstknoten anzuzeigen. Die Liste der Dienstknoten umfasst Dienste aus dem Überwachungskonto und allen konfigurierten Quellkonten. Filtern Sie die Liste der Knoten nach Kontoname oder Konto-ID, indem Sie sie aus dem Knotenfilter auswählen.

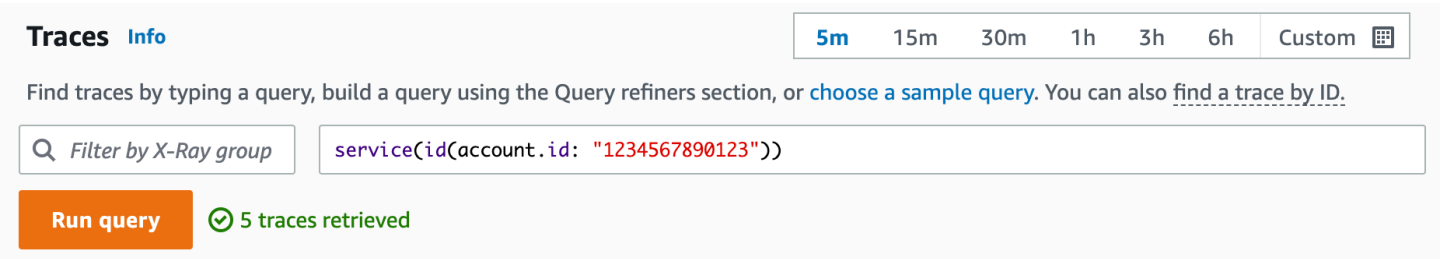



Ablaufverfolgungen

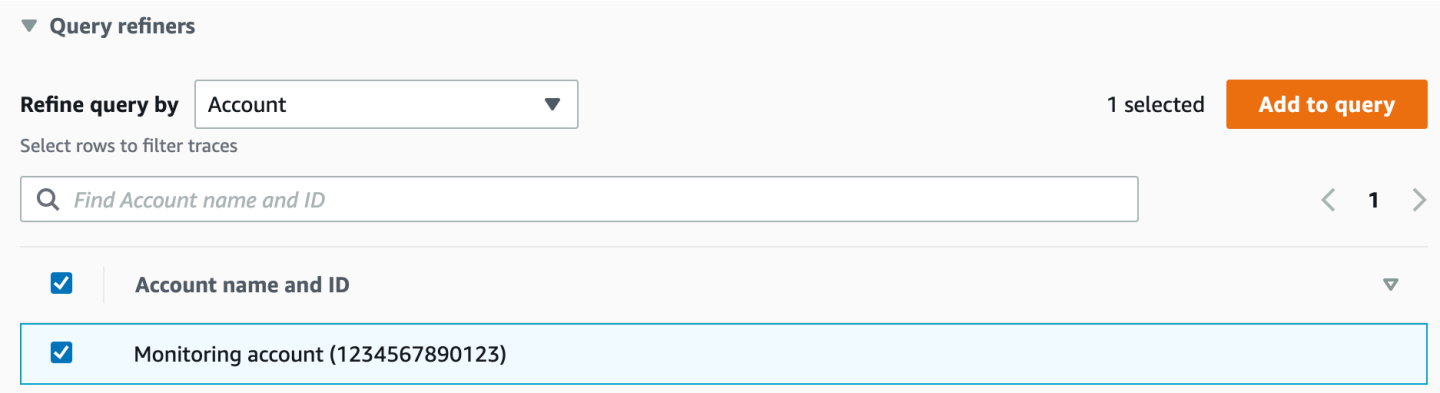
Rufen Sie die Trace-Details für Traces auf, die sich über mehrere Konten erstrecken, indem Sie die CloudWatch Konsole vom Monitoring-Konto aus öffnen und im linken Navigationsbereich unter X-Ray-Traces die Option Traces auswählen. Sie können diese Seite auch öffnen, indem Sie einen Knoten in der X-Ray-Trace-Map auswählen und dann im Bereich mit den Knotendetails die Option Traces anzeigen wählen.

Die Seite „Traces“ unterstützt Abfragen nach Konto-ID. Geben Sie zunächst eine Abfrage ein, die eine oder mehrere Konto-IDs enthält. Weitere Informationen zu Abfragen finden Sie unterVerwenden Sie Filterausdrücke. Im folgenden Beispiel werden Ablaufverfolgungen abgefragt, die die Konto-ID X oder Y durchlaufen haben:

```
service(id(account.id:"X")) OR service(id(account.id:"Y"))
```



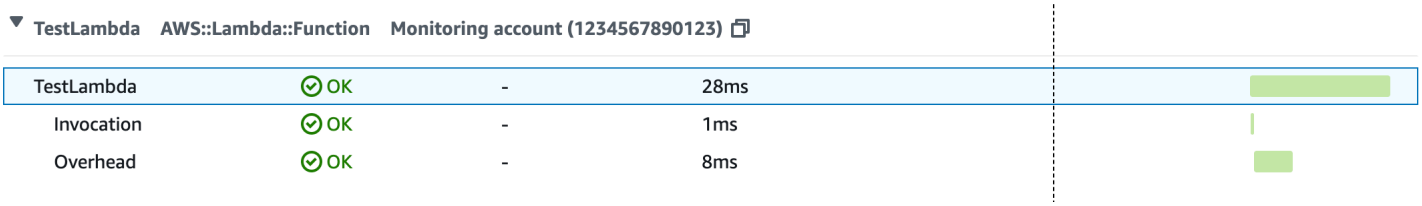

Verfeinern Sie Ihre Anfrage nach Konto. Wählen Sie ein oder mehrere Konten aus der Liste aus und klicken Sie dann auf Zur Abfrage hinzufügen.




Einzelheiten nachverfolgen

Sehen Sie sich die Details einer Ablaufverfolgung an, indem Sie sie aus der Traces-Liste unten auf der Traces-Seite auswählen. Die Trace-Details werden angezeigt, einschließlich einer Übersicht mit den Ablaufverfolgungsdetails mit Dienstknoten aller Konten, die der Trace durchlaufen hat. Wählen Sie einen bestimmten Dienstknoten aus, um das entsprechende Konto anzuzeigen.

Im Abschnitt Segment-Timeline werden die Kontodetails für jedes Segment in der Timeline angezeigt.




Verfolgen Sie ereignisgesteuerte Anwendungen

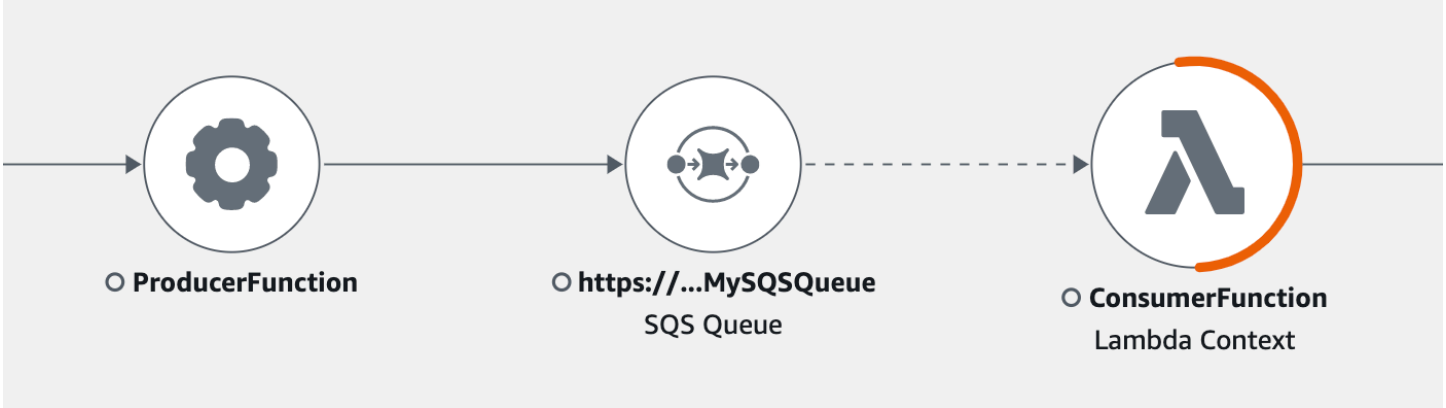
AWS X-Ray unterstützt die Ablaufverfolgung ereignisgesteuerter Anwendungen mithilfe von Amazon SQS und. AWS Lambda Verwenden Sie die CloudWatch Konsole, um eine verbundene Ansicht jeder Anfrage zu sehen, während sie bei Amazon SQS in die Warteschlange gestellt und von einer oder mehreren Lambda-Funktionen verarbeitet wird. Traces von Upstream-Nachrichtenproduzenten werden automatisch mit Traces von nachgeschalteten Lambda-Consumer-Knoten verknüpft, wodurch eine end-to-end Ansicht der Anwendung erstellt wird.

> **Note**
>
> Jedes Trace-Segment kann mit bis zu 20 Traces verknüpft werden, wobei ein Trace maximal 100 Links enthalten kann. In bestimmten Szenarien kann das Verknüpfen zusätzlicher Spuren dazu führen, dass die maximale Größe des Trace-Dokuments überschritten wird, was zu einer möglicherweise unvollständigen Ablaufverfolgung führen kann. Dies kann

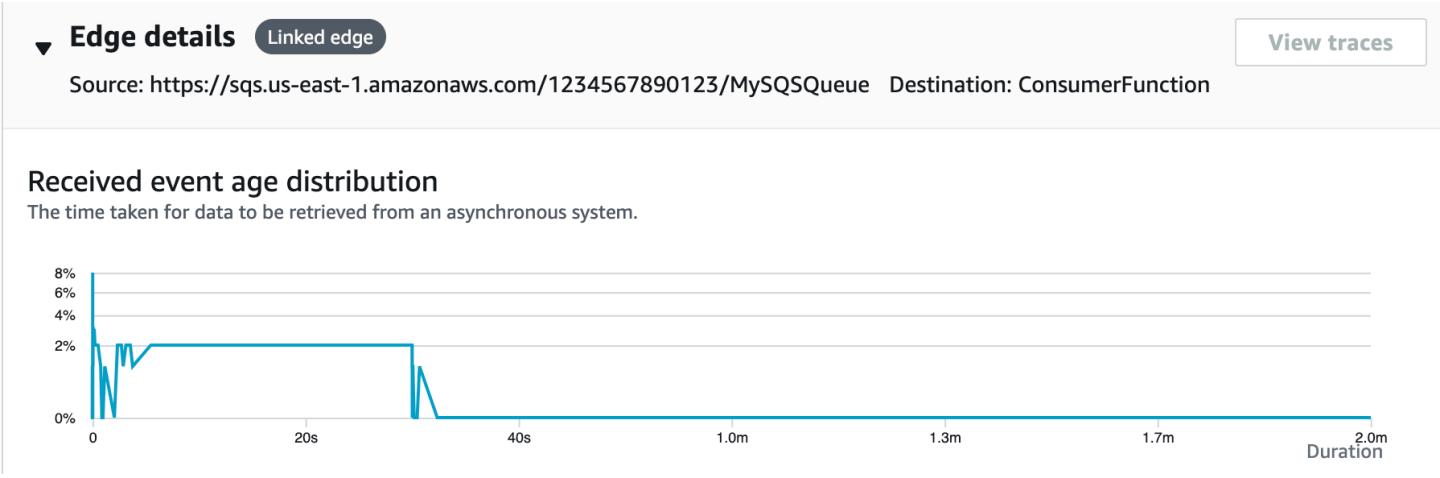
beispielsweise passieren, wenn eine Lambda-Funktion mit aktivierter Ablaufverfolgung viele SQS-Nachrichten in einem einzigen Aufruf an eine Warteschlange sendet. Wenn Sie auf dieses Problem stoßen, ist eine Abhilfemaßnahme verfügbar, die die X-Ray-SDKs verwendet. Weitere Informationen finden Sie im X-Ray-SDK für Java, Node.js, Python, Go oder .NET.

### Verknüpfte Traces in der Trace-Map anzeigen

Verwenden Sie die Trace-Map-Seite in der CloudWatchKonsole, um eine Trace-Map mit Traces von Nachrichtenproduzenten anzuzeigen, die mit Traces von Lambda-Verbrauchern verknüpft sind. Diese Links werden mit einer gestrichelten Linie angezeigt, die den Amazon SQS-Knoten mit den nachgeschalteten Lambda-Consumer-Knoten verbindet.



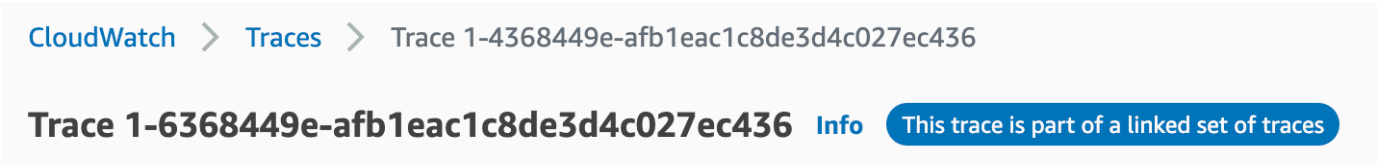

Wählen Sie eine gestrichelte Linie aus, um ein Histogramm mit dem Alter eines empfangenen Ereignisses anzuzeigen, das die Verteilung des Ereignisalters bei Empfang durch Verbraucher darstellt. Das Alter wird jedes Mal berechnet, wenn ein Ereignis empfangen wird.



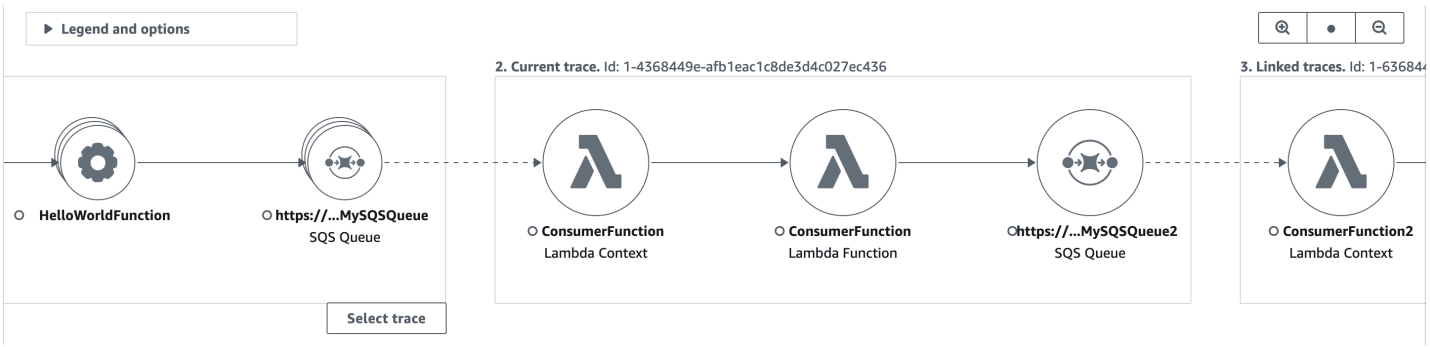
### Details der verknüpften Ablaufverfolgung anzeigen

Trace-Details anzeigen, die von einem Nachrichtenproduzenten, einer Amazon SQS SQS-Warteschlange oder einem Lambda-Consumer gesendet wurden:

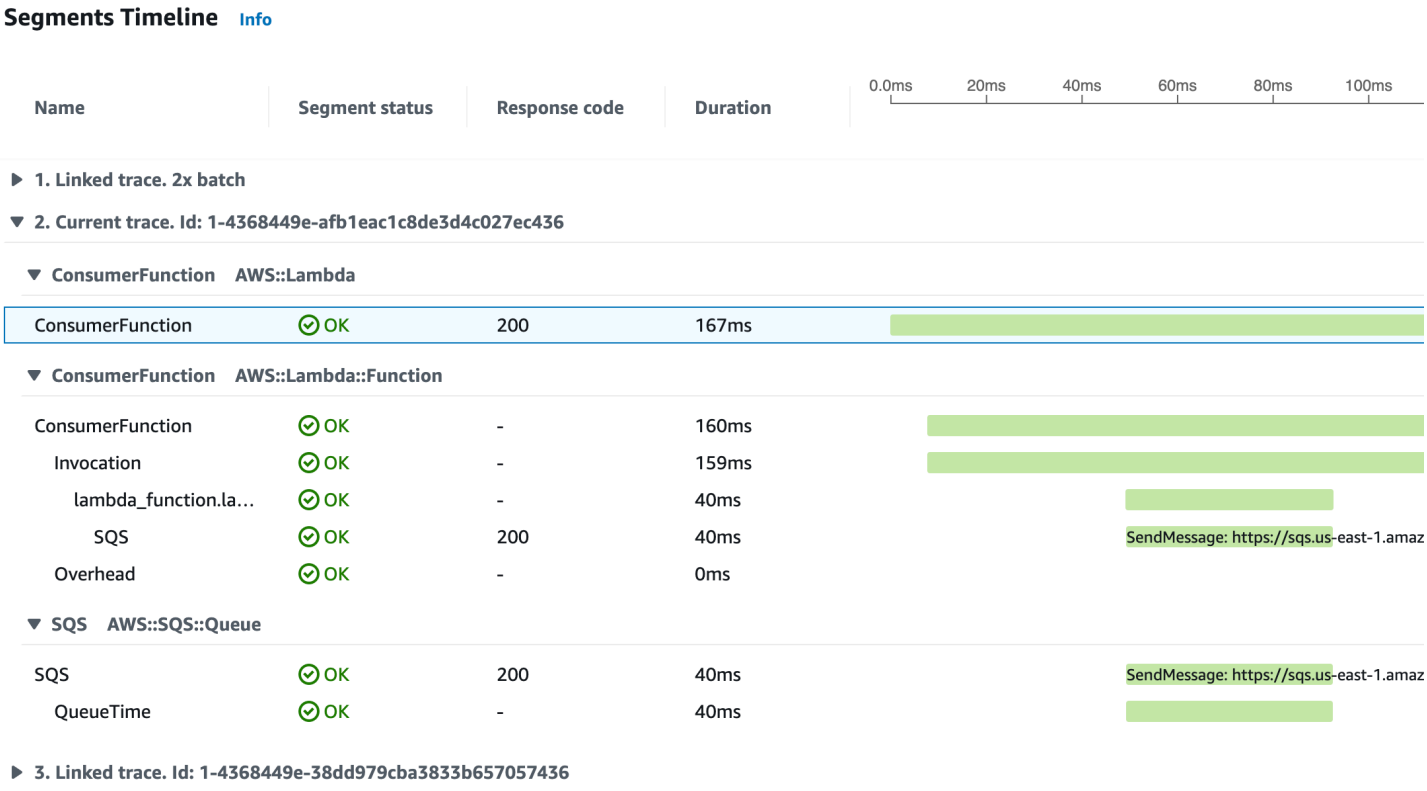
1. Verwenden Sie die Trace-Map, um einen Message Producer, Amazon SQS oder Lambda Consumer Node auszuwählen.
2. Wählen Sie im Bereich mit den Knotendetails die Option Traces anzeigen, um eine Liste von Traces anzuzeigen. Sie können in der CloudWatch Konsole auch direkt zur Seite „Traces“ navigieren.
3. Wählen Sie einen bestimmten Trace aus der Liste aus, um die Seite mit den Trace-Details zu öffnen. Auf der Seite mit den Ablaufverfolgungsdetails wird eine Meldung angezeigt, wenn der ausgewählte Trace Teil eines verknüpften Trace-Satzes ist.




In der Karte mit den Verfolgungsdetails werden die aktuelle Spur zusammen mit den verknüpften Spuren flussaufwärts und flussabwärts angezeigt, die jeweils in einem Feld enthalten sind, das die Grenzen der einzelnen Spuren angibt. Wenn die aktuell ausgewählte Spur mit mehreren Upstream- oder Downstream-Leiterbahnen verknüpft ist, werden die Knoten innerhalb der Upstream- oder Downstream-verknüpften Spuren gestapelt, und die Schaltfläche Spur auswählen wird angezeigt.



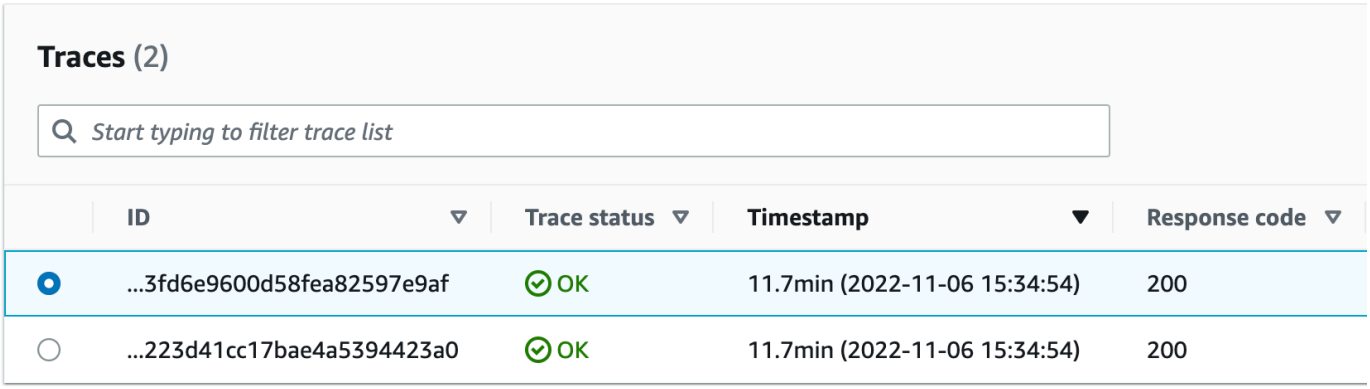

Unter der Karte mit den Spurdetails wird eine Zeitleiste mit Spurensegmenten angezeigt, einschließlich verknüpfter Spuren flussaufwärts und flussabwärts. Wenn mehrere stromaufwärts oder flussabwärts verknüpfte Spuren vorhanden sind, können deren Segmentdetails nicht angezeigt werden. Um Segmentdetails für eine einzelne Spur innerhalb einer Reihe verknüpfter Spuren anzuzeigen, wählen Sie eine einzelne Spur aus, wie im folgenden Abschnitt beschrieben.



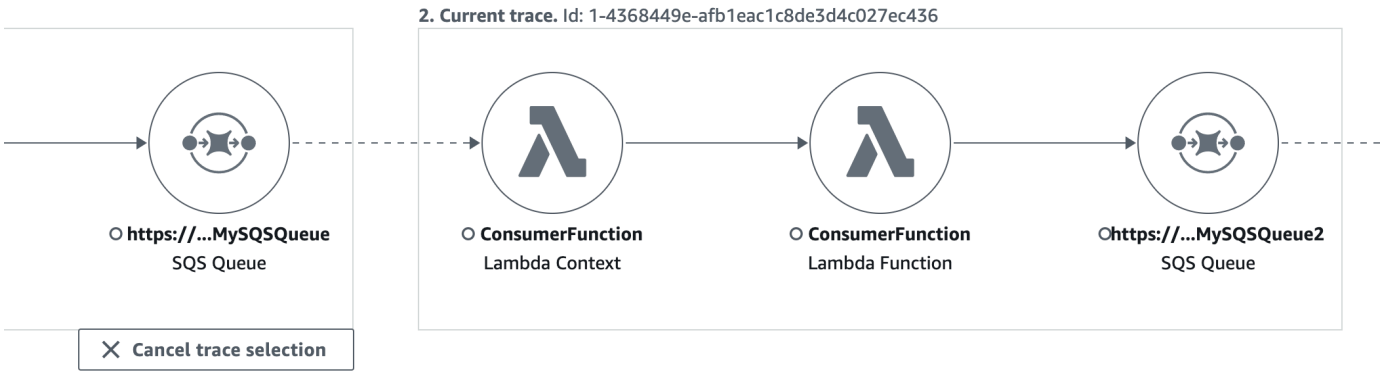

Wählen Sie eine einzelne Spur innerhalb einer Reihe verknüpfter Spuren aus

Filtert einen verknüpften Satz von Spuren zu einer einzelnen Spur, um Segmentdetails in der Timeline zu sehen.

1. Wählen Sie auf der Karte mit den Trace-Details unter den verknüpften Spuren die Option Spur auswählen aus. Eine Liste von Spuren wird angezeigt.




2. Wählen Sie das Optionsfeld neben einer Spur aus, um sie in der Karte mit den Ablaufverfolgungsdetails anzuzeigen.
3. Wählen Sie „Spurauswahl abbrechen“, um den gesamten Satz verknüpfter Spuren anzuzeigen.



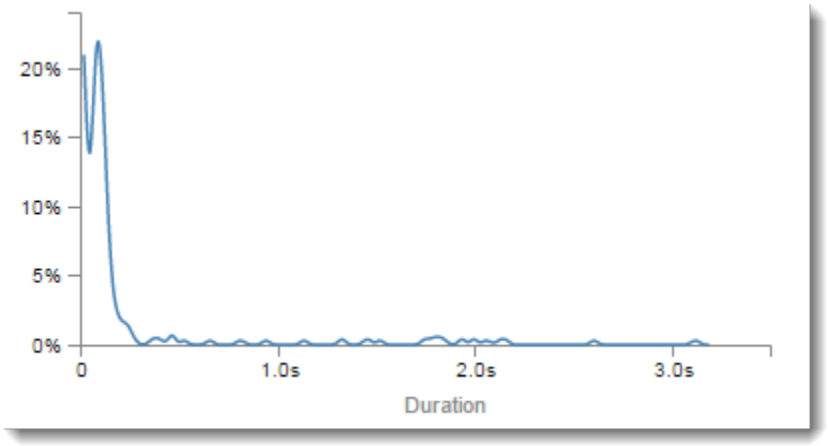

## Verwenden Sie Latenz-Histogramme

Wenn Sie einen Knoten oder eine Kante auf einer Trace-Map auswählen, zeigt die X-Ray-Konsole ein Latenzverteilungshistogramm an.

### Latency

Als Latenz wird die Zeit zwischen dem Start und dem Abschluss einer Anforderung bezeichnet. Ein Histogramm zeigt eine Verteilung von Latenzen. Er zeigt Dauer auf der x-Achse und den Prozentsatz der Anforderungen, die jeder Dauer entsprechen, auf der y-Achse.

Dieses Histogramm zeigt einen Service, der die meisten Anfragen in weniger als 300 ms abschließt. Ein kleiner Prozentsatz von Anfragen dauert 2 Sekunden, und einige Ausreißer nehmen noch mehr Zeit in Anspruch.



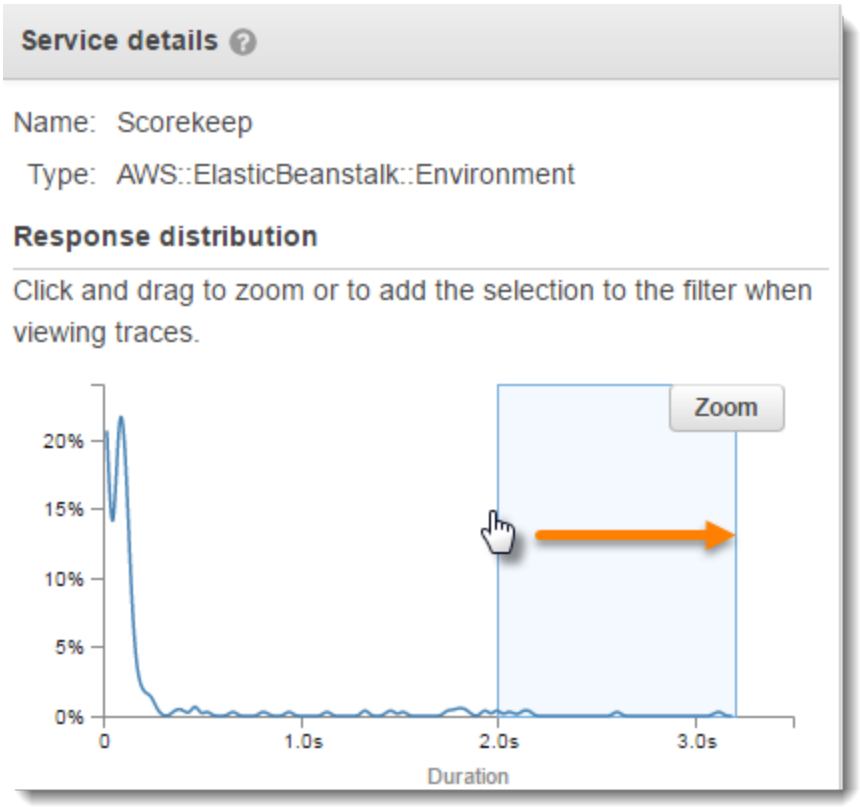

## Interpretieren von Service-Details

Service-Histogramme und Edge-Histogramme bieten eine visuelle Darstellung der Latenz aus der Sicht eines Services oder eines Anforderers.

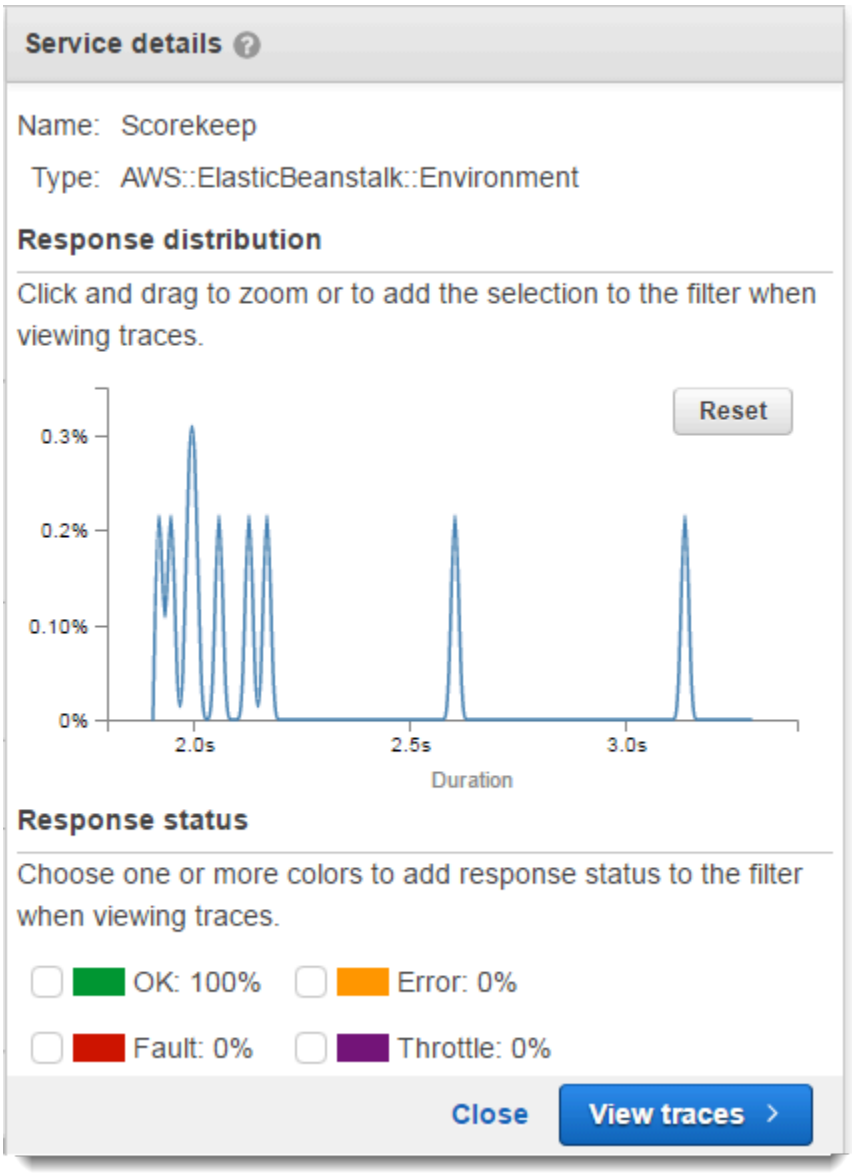
- Wählen Sie einen Dienstknoten aus, indem Sie auf den Kreis klicken. X-Ray zeigt ein Histogramm für Anfragen, die vom Dienst bedient werden. Die Latenzen sind die, die der Service aufgezeichnet hat; dazu gehört nicht die Netzwerklatenz zwischen dem Service und dem Auftraggeber.
- Wählen Sie eine Kante aus, indem Sie auf die Linie oder die Pfeilspitze der Kante zwischen zwei Diensten klicken. X-Ray zeigt ein Histogramm für Anfragen des Anforderers, die vom Downstream-Dienst bearbeitet wurden. Die Latenzen sind die vom Auftraggeber aufgezeichneten; dazu gehört die Latenz in der Netzwerkverbindung zwischen den beiden Services.

Zur Interpretation des Panel-Histogramms mit den Servicedetails können Sie die Werte betrachten, die am stärksten von den meisten Werten im Histogramm abweichen. Diese Ausreißer sind als Spitzen im Histogramm zu sehen und Sie können die Ablaufverfolgungen für einen bestimmten Bereich untersuchen, um zu sehen, was vor sich geht.

Wählen Sie zur Anzeige nach Latenz gefilterter Ablaufverfolgungen einen Bereich im Histogramm aus. Klicken Sie an die Stelle, an der die Auswahl beginnen soll, und ziehen Sie von links nach rechts, um einen Bereich von Latenzen hervorzuheben, der in den Ablaufverfolgungsfilter eingeschlossen werden soll.



Nach Auswahl eines Bereichs können Sie den Zoom wählen, um nur diesen Teil des Histogramms anzuzeigen, und Ihre Auswahl verfeinern.



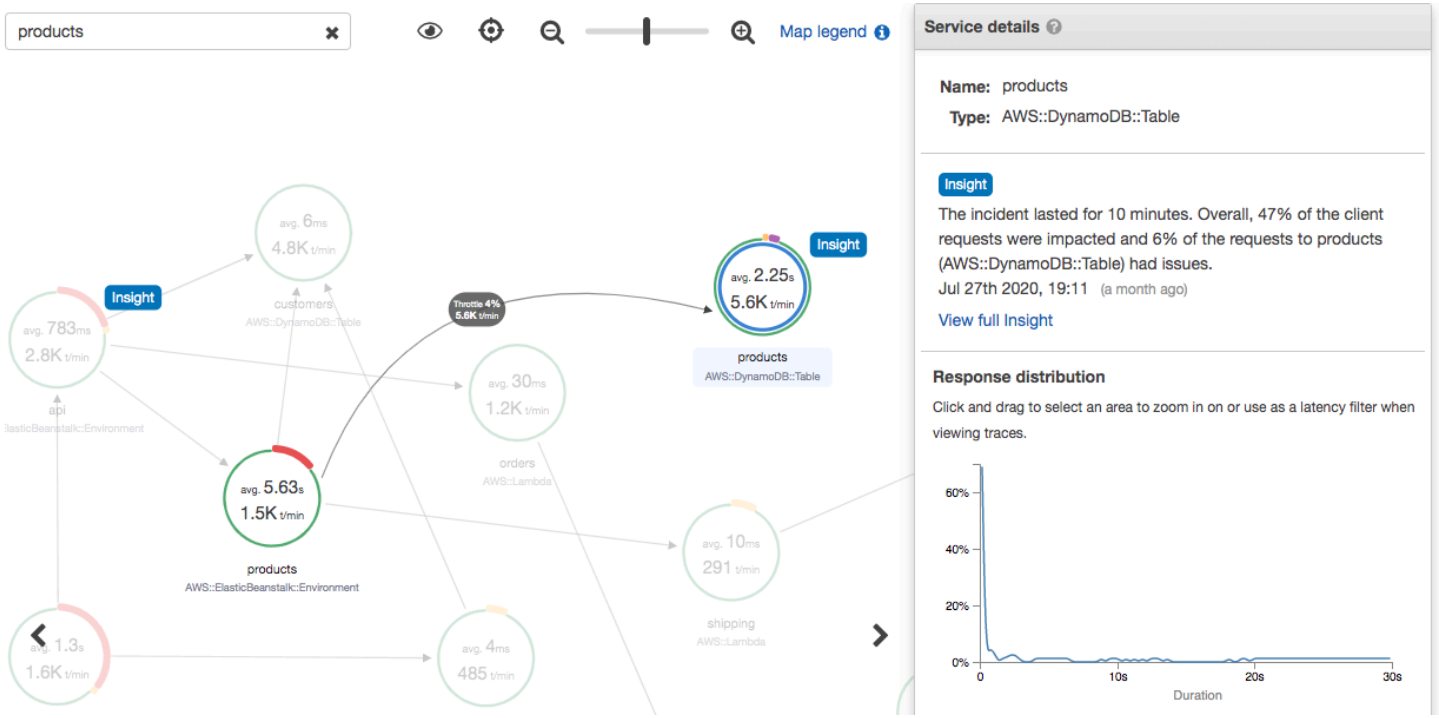

Sobald Sie den Fokus auf den interessierenden Bereich gesetzt haben, wählen Sie Ablaufverfolgungen anzeigen.

Verwenden Sie X-Ray Insights

AWS X-Ray analysiert kontinuierlich die Trace-Daten in Ihrem Konto, um neu auftretende Probleme in Ihren Anwendungen zu identifizieren. Wenn die Fehlerraten den erwarteten Bereich überschreiten, werden Erkenntnisse gewonnen, die das Problem aufzeichnen und seine Auswirkungen verfolgen, bis es behoben ist. Mit Insights können Sie:

- Identifizieren Sie, wo in Ihrer Anwendung Probleme auftreten, die Hauptursache des Problems und die damit verbundenen Auswirkungen. Die von Insights bereitgestellte Auswirkungsanalyse ermöglicht es Ihnen, den Schweregrad und die Priorität eines Problems abzuleiten.
- Erhalten Sie Benachrichtigungen, wenn sich das Problem im Laufe der Zeit ändert. Insights-Benachrichtigungen können mithilfe von Amazon EventBridge in Ihre Überwachungs- und Warnlösung integriert werden. Diese Integration ermöglicht es Ihnen, je nach Schweregrad des Problems automatisierte E-Mails oder Benachrichtigungen zu versenden.

Die X-Ray-Konsole identifiziert Knoten mit laufenden Vorfällen in der Trace-Map. Um eine Zusammenfassung der Erkenntnisse zu erhalten, wählen Sie den betroffenen Knoten aus. Sie können Insights auch anzeigen und filtern, indem Sie im Navigationsbereich auf der linken Seite Insights auswählen.




X-Ray generiert einen Einblick, wenn es eine Anomalie in einem oder mehreren Knoten der Service-Map erkennt. Der Service verwendet statistische Modelle, um die erwarteten Fehlerraten der Dienste in Ihrer Anwendung vorherzusagen. Im vorherigen Beispiel handelt es sich bei der Anomalie um eine Zunahme von Fehlern von AWS Elastic Beanstalk. Auf dem Elastic Beanstalk-Server kam es zu mehreren Timeouts bei API-Aufrufen, was zu einer Anomalie in den Downstream-Knoten führte.

### Insights in der X-Ray-Konsole aktivieren

Insights muss für jede Gruppe aktiviert sein, mit der Sie Insights-Funktionen verwenden möchten. Sie können Insights auf der Gruppenseite aktivieren.

1. Öffnen Sie die X-Ray-Konsole.
2. Wählen Sie eine bestehende Gruppe aus oder erstellen Sie eine neue, indem Sie Gruppe erstellen und dann Insights aktivieren auswählen. Weitere Informationen zur Konfiguration von Gruppen in der X-Ray-Konsole finden Sie unterGruppen konfigurieren.
3. Wählen Sie im Navigationsbereich auf der linken Seite Insights und dann einen Insight aus, den Sie sich ansehen möchten.


> **Note**
>
> X-Ray verwendet GetInsightSummaries GetInsight, GetInsightEvents, und GetInsightImpactGraph API-Operationen, um Daten aus Insights abzurufen. Verwenden Sie zum Anzeigen von Erkenntnissen die von AWSXrayReadOnlyAccess IAM verwaltete Richtlinie oder fügen Sie Ihrer IAM-Rolle die folgende benutzerdefinierte Richtlinie hinzu:

```
{
   "Version": "2012-10-17",
   "Statement": [
       {
           "Effect": "Allow",
           "Action": [
               "xray:GetInsightSummaries",
               "xray:GetInsight",
               "xray:GetInsightEvents",
               "xray:GetInsightImpactGraph"
           ],
           "Resource": [
               "*"
           ]
```

```
                }
            ]
        }
```

Weitere Informationen finden Sie unter Wie AWS X-Ray funktioniert mit IAM.

### Aktivieren Sie Insights-Benachrichtigungen

Bei Insights-Benachrichtigungen wird für jedes Insight-Ereignis eine Benachrichtigung erstellt, z. B. wenn ein Insight erstellt wird, sich erheblich ändert oder geschlossen wird. Kunden können diese Benachrichtigungen über EventBridge Amazon-Ereignisse erhalten und bedingte Regeln verwenden, um Aktionen wie SNS-Benachrichtigungen, Lambda-Aufrufe, das Posten von Nachrichten in eine SQS-Warteschlange oder eines der unterstützten Ziele zu ergreifen. EventBridge Insights-Benachrichtigungen werden nach bestem Wissen versendet, können aber nicht garantiert werden. Weitere Informationen zu Zielen finden Sie unter Amazon EventBridge Targets.

Auf der Gruppenseite können Sie Insights-Benachrichtigungen für jede Gruppe aktivieren, für die Insights aktiviert sind.

Um Benachrichtigungen für eine X-Ray-Gruppe zu aktivieren

1. Öffnen Sie die X-Ray-Konsole.
2. Wählen Sie eine bestehende Gruppe aus oder erstellen Sie eine neue, indem Sie Gruppe erstellen wählen. Vergewissern Sie sich, dass Insights aktivieren ausgewählt ist, und wählen Sie dann Benachrichtigungen aktivieren aus. Weitere Informationen zur Konfiguration von Gruppen in der X-Ray-Konsole finden Sie unterGruppen konfigurieren.

So konfigurieren Sie EventBridge bedingte Amazon-Regeln

1. Öffnen Sie die EventBridge Amazon-Konsole.
2. Navigieren Sie in der linken Navigationsleiste zu Regeln und wählen Sie Regel erstellen aus.
3. Geben Sie einen Namen und eine Beschreibung für die Regel ein.
4. Wählen Sie Ereignismuster und anschließend Benutzerdefiniertes Muster aus. Geben Sie ein Muster an, das `"source": [ "aws.xray" ]` und enthält`"detail-type": [ "AWS X-Ray Insight Update" ]`. Im Folgenden finden Sie einige Beispiele für mögliche Muster.
   - Ereignismuster, das allen eingehenden Ereignissen aus X-Ray Insights entspricht:

```
{
"source": [ "aws.xray" ],
"detail-type": [ "AWS X-Ray Insight Update" ]
}
```

- Ereignismuster, das einem bestimmten Wert entspricht **`state`** und**`category`**:

```
{
"source": [ "aws.xray" ],
"detail-type": [ "AWS X-Ray Insight Update" ],
"detail": {
        "State": [ "ACTIVE" ],
        "Category": [ "FAULT" ]
  }
}
```

5. Wählen und konfigurieren Sie die Ziele, die Sie aufrufen möchten, wenn ein Ereignis dieser Regel entspricht.
6. (Optional) Geben Sie Tags an, um diese Regel leichter identifizieren und auswählen zu können.
7. Wählen Sie Erstellen.

> **Note**
>
> X-Ray Insights-Benachrichtigungen senden Ereignisse an Amazon EventBridge, das derzeit keine vom Kunden verwalteten Schlüssel unterstützt. Weitere Informationen finden Sie unter Datenschutz in AWS X-Ray.

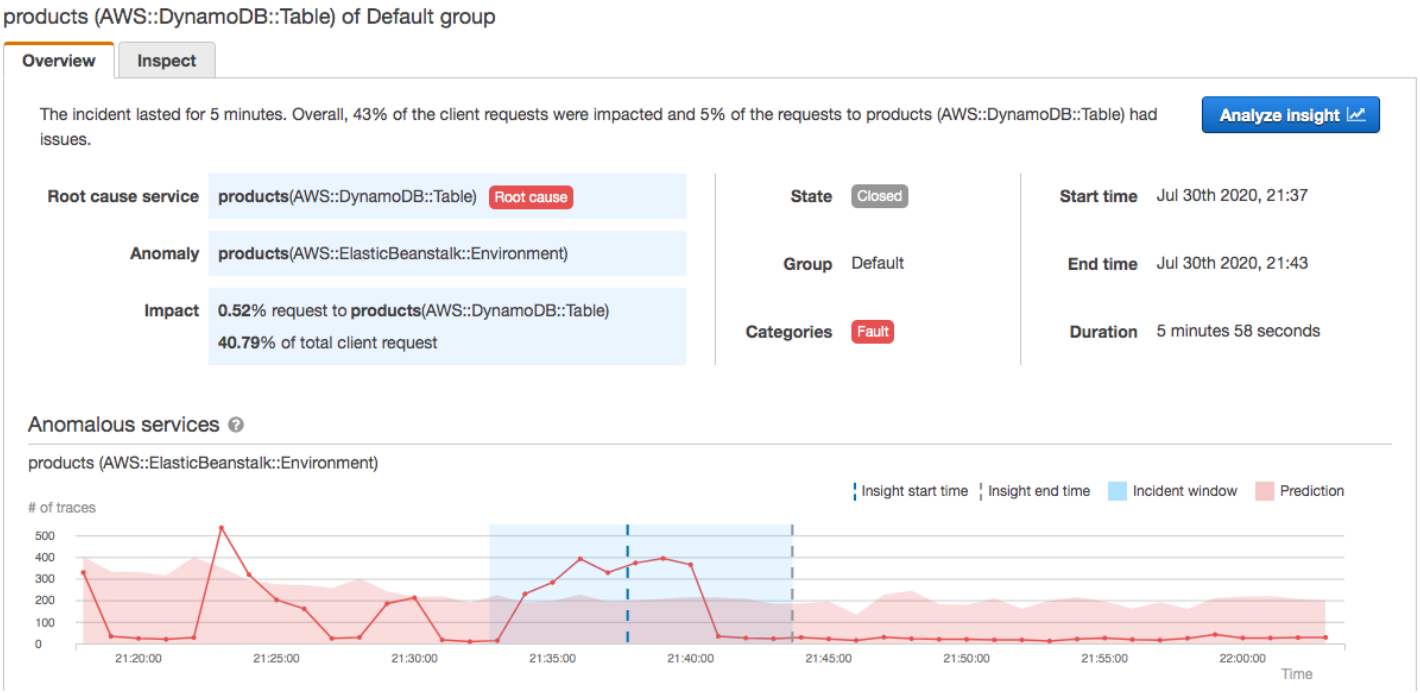
### Überblick über Insight

Die Übersichtsseite für einen Einblick versucht, drei wichtige Fragen zu beantworten:

- Was ist das zugrundeliegende Problem?
- Was ist die Hauptursache?
- Was ist die Auswirkung?

Im Abschnitt Anomale Dienste wird für jeden Service ein Zeitplan angezeigt, der die Veränderung der Fehlerraten während des Vorfalls veranschaulicht. Die Zeitleiste zeigt die Anzahl der Traces mit Fehlern auf einem durchgehenden Band, das die erwartete Anzahl von Fehlern auf der Grundlage des aufgezeichneten Datenverkehrs angibt. Die Dauer der Erkenntnisse wird im Incident-Fenster visualisiert. Das Incident-Fenster beginnt, wenn X-Ray beobachtet, dass die Metrik anomal wird, und bleibt bestehen, solange der Insight aktiv ist.

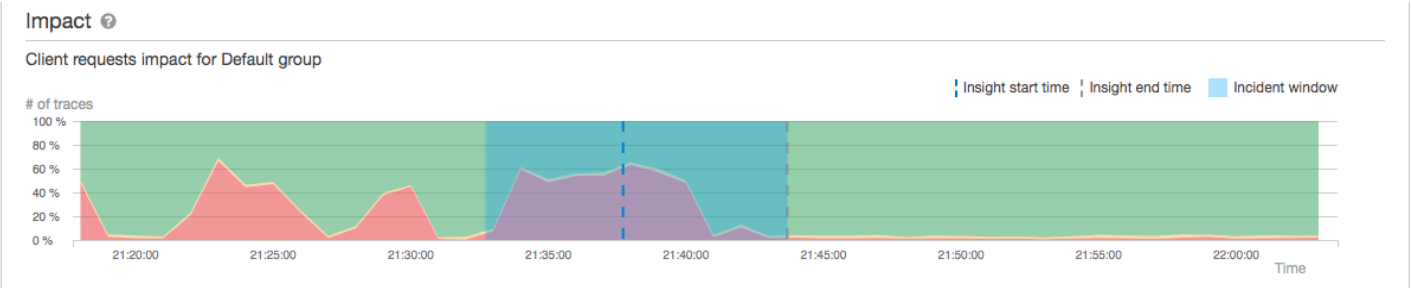
Das folgende Beispiel zeigt eine Zunahme von Fehlern, die zu einem Vorfall geführt haben:




Im Abschnitt „Ursache“ wird eine Ablaufverfolgungsübersicht angezeigt, die sich auf den Ursachendienst und den betroffenen Pfad konzentriert. Sie können die nicht betroffenen Knoten ausblenden, indem Sie auf das Augensymbol oben rechts in der Ursachenkarte klicken. Der Root Cause Service ist der am weitesten flussabwärts gelegene Knoten, an dem X-Ray eine Anomalie identifiziert hat. Dabei kann es sich um einen Service handeln, den Sie instrumentiert haben, oder um einen externen Service, den Ihr Service mit einem instrumentierten Client aufgerufen hat. Wenn Sie beispielsweise Amazon DynamoDB mit einem instrumentierten AWS SDK-Client aufrufen, führt eine Zunahme von Fehlern durch DynamoDB zu Erkenntnissen mit DynamoDB als Hauptursache.

Um die Ursache weiter zu untersuchen, wählen Sie im Ursachendiagramm die Option Ursachendetails anzeigen aus. Sie können die Analytics-Seite verwenden, um die Grundursache und zugehörige Meldungen zu untersuchen. Weitere Informationen finden Sie unter Interagieren Sie mit der Analytics-Konsole.


Fehler, die sich in der Map weiter flussaufwärts fortsetzen, können sich auf mehrere Knoten auswirken und mehrere Anomalien verursachen. Wenn ein Fehler vollständig an den Benutzer weitergegeben wird, der die Anfrage gestellt hat, ist das Ergebnis ein Client-Fehler. Dies ist ein Fehler im Stammknoten der Trace-Map. Das Impact-Diagramm bietet einen Überblick über das Kundenerlebnis für die gesamte Gruppe. Dieses Kundenerlebnis wird anhand der Prozentsätze der folgenden Zustände berechnet: Fehler, Fehler, Drosselung und Okay.




Dieses Beispiel zeigt eine Zunahme von Traces mit einem Fehler am Stammknoten während eines Vorfalls. Vorfälle in nachgelagerten Diensten entsprechen nicht immer einer Zunahme von Client-Fehlern.

Wenn Sie Analyze Insight wählen, wird die X-Ray Analytics-Konsole in einem Fenster geöffnet, in dem Sie tief in die Spuren eintauchen können, die den Einblick verursacht haben. Weitere Informationen finden Sie unter Interagieren Sie mit der Analytics-Konsole.

### Auswirkungen verstehen

AWS X-Ray misst im Rahmen der Generierung von Erkenntnissen und Benachrichtigungen die Auswirkungen, die durch ein laufendes Problem verursacht werden. Die Auswirkungen werden auf zwei Arten gemessen:

- Auswirkungen auf die X-Ray-Gruppe. Weitere Informationen finden Sie unter Gruppen konfigurieren
- Auswirkungen auf den Root-Cause-Service

Diese Auswirkung wird durch den Prozentsatz der Anfragen bestimmt, die innerhalb eines bestimmten Zeitraums fehlschlagen oder einen Fehler verursachen. Diese Auswirkungsanalyse ermöglicht es Ihnen, den Schweregrad und die Priorität des Problems anhand Ihres speziellen Szenarios abzuleiten. Diese Auswirkung ist zusätzlich zu den Insights-Benachrichtigungen als Teil des Konsolen-Erlebnisses verfügbar.

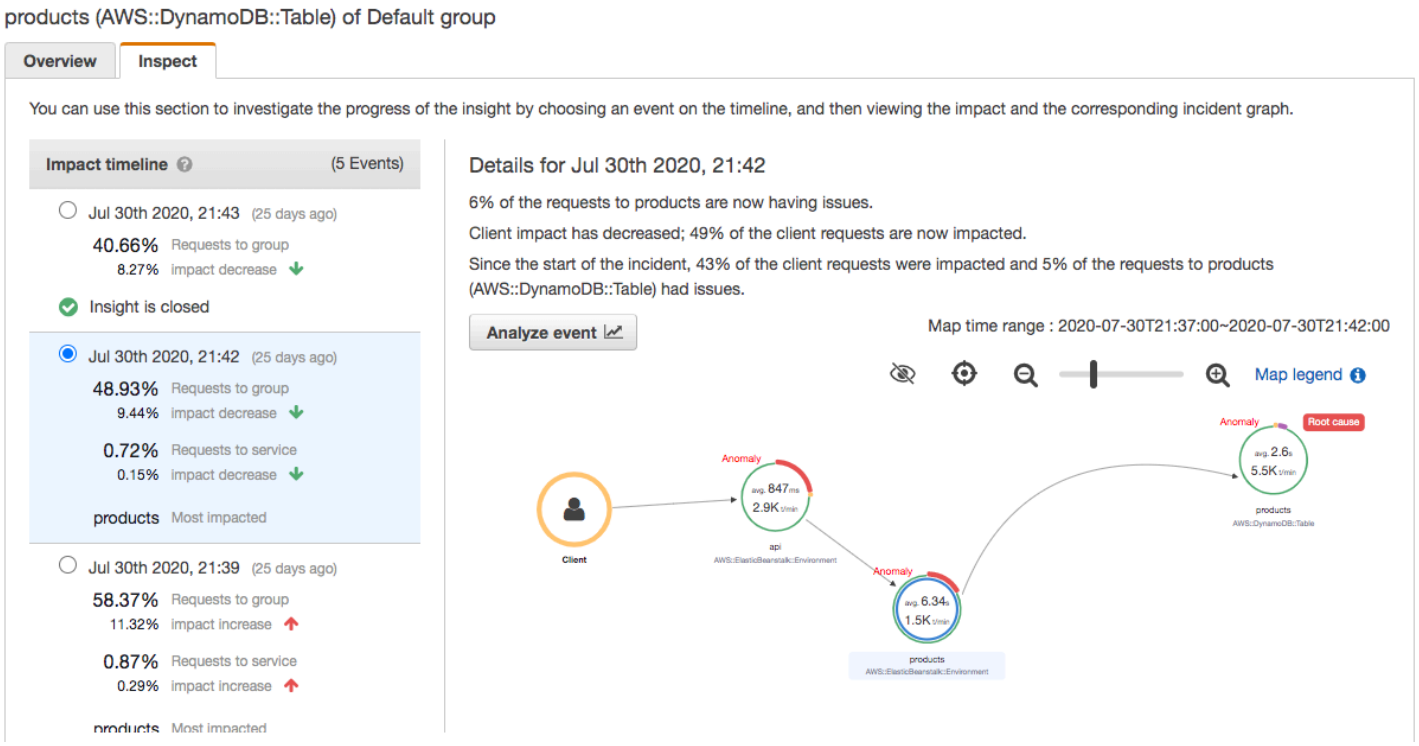
Deduplizierung

AWS X-Ray Insights dedupliziert Probleme bei mehreren Microservices. Es verwendet die Anomalieerkennung, um den Service zu ermitteln, der die Hauptursache eines Problems ist, ermittelt, ob andere verwandte Dienste aufgrund derselben Grundursache ein anomales Verhalten zeigen, und zeichnet das Ergebnis in Form eines einzigen Einblicks auf.

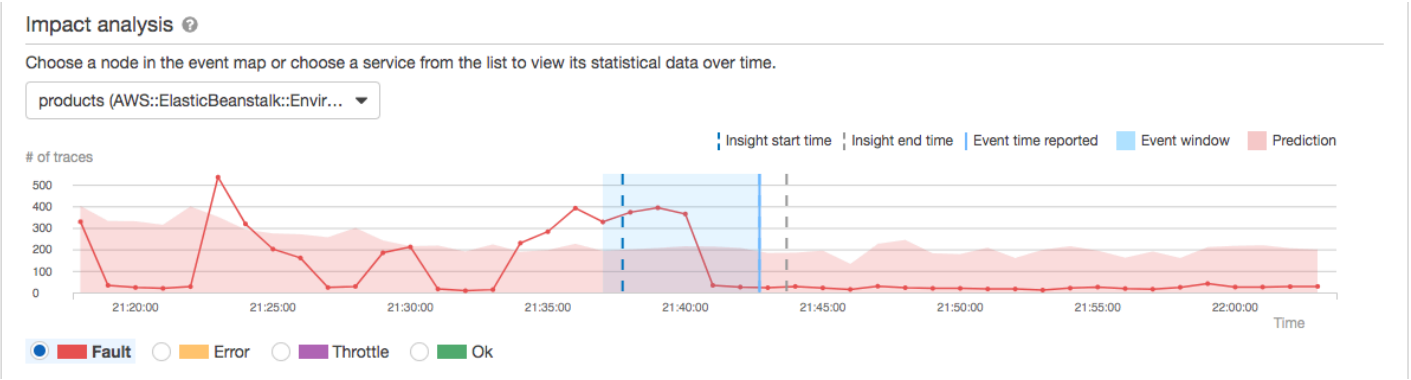
Überprüfen Sie den Fortschritt eines Insights

X-Ray bewertet die Erkenntnisse in regelmäßigen Abständen neu, bis sie behoben sind, und zeichnet jede nennenswerte zwischenzeitliche Änderung als Benachrichtigung auf, die als EventBridge Amazon-Ereignis gesendet werden kann. Auf diese Weise können Sie Prozesse und Workflows erstellen, um festzustellen, wie sich das Problem im Laufe der Zeit verändert hat, und geeignete Maßnahmen ergreifen, z. B. das Senden einer E-Mail oder die Integration in ein Warnsystem mithilfe von. EventBridge

Sie können Vorfallereignisse in der Impact-Timeline auf der Seite Inspect überprüfen. Standardmäßig zeigt die Zeitleiste den Service an, der am stärksten betroffen ist, bis Sie einen anderen Service auswählen.




Um eine Ablaufkarte und Grafiken für ein Ereignis zu sehen, wählen Sie es aus der Wirkungszeitleiste aus. Die Trace-Map zeigt die Dienste in Ihrer Anwendung, die von dem Vorfall betroffen sind. Unter Auswirkungsanalyse zeigen Diagramme die Fehlerzeitpläne für den ausgewählten Knoten und für die Clients in der Gruppe.




Wenn Sie sich die Spuren eines Vorfalls genauer ansehen möchten, wählen Sie auf der Seite Inspizieren die Option Ereignis analysieren aus. Sie können die Analytics-Seite verwenden, um die Liste der Traces zu verfeinern und die betroffenen Benutzer zu identifizieren. Weitere Informationen finden Sie unter Interagieren Sie mit der Analytics-Konsole.

### Interagieren Sie mit der Analytics-Konsole

Die AWS X-Ray Analytics-Konsole ist ein interaktives Tool zur Interpretation von Trace-Daten, mit dem Sie schnell nachvollziehen können, wie Ihre Anwendung und die zugrunde liegenden Dienste funktionieren. Die Konsole ermöglicht Ihnen das Erkunden, Analysieren und Visualisieren von Ablaufverfolgungen durch interaktive Reaktionszeit und Zeitreihen-Diagramme.

Bei der Auswahl in der Analysekonsole erstellt die Konsole Filter für die ausgewählte Teilmenge aller Ablaufverfolgungen. Sie können die aktiven Datasets mit zunehmend granularen Filtern anpassen, indem Sie auf die Diagramme und die Bereiche der Metriken und Felder klicken, die der aktuellen Ablaufverfolgungsreihe zugeordnet sind.

#### Konsolenfunktionen

Die X-Ray Analytics-Konsole verwendet die folgenden Hauptfunktionen zum Gruppieren, Filtern, Vergleichen und Quantifizieren von Trace-Daten.

#### Features

| Funktion | Beschreibung |
| --- | --- |
| Gruppen | Die anfangs ausgewählte Gruppe ist `Default`. Zum Ändern der abgerufenen Gruppe wählen Sie eine andere Gruppe aus dem Menü rechts neben der Filterausdruck-Suchleiste aus. Weitere Informationen zu Gruppen finden Sie unter Gruppen konfigurieren. |
| Abgerufene Ablaufverfolgungen | Standardmäßig generiert die Analysekonsole Diagramme auf Grundlage aller Ablaufver folgungen in der ausgewählten Gruppe. Abgerufene Ablaufverfolgungen stehen für alle Ablaufverfolgungen in Ihrem Arbeitssatz. Auf dieser Kachel finden Sie die Anzahl der Ablaufverfolgungen. Mit Filterausdrücken an der Hauptsuchleiste optimieren und aktualisi eren Sie die abgerufenen Ablaufverfolgungen. |
| In Diagrammen anzeigen/In Diagrammen ausblenden | Ein Schalter zum Vergleichen der aktiven Gruppe mit den abgerufenen Ablaufver |

| Funktion | Beschreibung |
| --- | --- |
| | folgungen. Zum Vergleichen der Daten im Zusammenhang mit der Gruppe mit sämtliche n aktiven Filtern klicken Sie auf In Diagramme n anzeigen. Zum Entfernen dieser Ansicht aus den Diagrammen wählen Sie In Diagrammen ausblenden. |
| Gefilterter Ablaufverfolgungssatz A | Wenden Sie durch Interaktionen mit den Grafiken und Tabellen Filter an, um die Kriterien für den gefilterten Ablaufverfolgungss atz A zu erstellen. Wenn die Filter angewende t werden, werden die Anzahl der zutreffenden Spuren und der Prozentsatz der abgerufen en Spuren an der Gesamtzahl der abgerufen en Spuren innerhalb dieser Kachel berechnet . Filter werden auf der Kachel Gefilterter Ablaufverfolgungssatz A als Tags angezeigt und können zudem von der Kachel entfernt werden. |
| Optimieren | Über diese Funktion wird der Satz der abgerufenen Ablaufverfolgungen basierend auf den Filtern, die auf den Ablaufverfolgungss atz A angewendet wurden, aktualisiert. Durch Optimieren des abgerufenen Ablaufver folgungssatzes wird der Arbeitssatz aller abgerufenen Ablaufverfolgungen basierend auf den Filtern für den Ablaufverfolgungssatz A abgerufen. Der Arbeitssatz der abgerufenen Ablaufverfolgungen ist eine geprüfte Teilmenge aller Ablaufverfolgungen in der Gruppe. |

| Funktion | Beschreibung |
| --- | --- |
| Gefilterter Ablaufverfolgungssatz B | Bei der Erstellung ist der gefilterte Ablaufverfolgungssatz B eine Kopie des gefilterten Ablaufverfolgungssatzes A. Um die beiden Verfolgungssätze zu vergleichen, wählen Sie einen neuen Filter aus, der für den Verfolgungssatz B gilt, während der Verfolgungssatz A unverändert bleibt. Während der Anwendung der Filter werden die Anzahl der entsprechenden Ablaufverfolgungen und der Prozentsatz der Ablaufverfolgungen von der Gesamtzahl der abgerufenen Ablaufverfolgungen auf dieser Kachel berechnet. Filter werden auf der Kachel Ablaufverfolgungssatz B als Tags angezeigt und können zudem von der Kachel entfernt werden. |
| Reaktionszeit Ursachenentitätspfade | Eine Tabelle mit aufgezeichneten Entitätspfaden. X-Ray bestimmt, welcher Pfad in Ihrer Spur die wahrscheinlichste Ursache für die Reaktionszeit ist. Das Format steht für eine Hierarchie von vorgefundenen Entitäten . Dies führt zu einer Reaktionszeit-Ursache. Verwenden Sie diese Zeilen, um nach wiederkehrenden Reaktionszeitfehlern zu filtern. Weitere Informationen zum Anpassen eines Ursachenfilters und zum Abrufen von Daten über die API finden Sie im Abschnitt Ursachenanalysen abrufen und verfeinern unter. Daten von X-Ray abrufen |
| Delta (�) | Eine Spalte, die den Metriktabellen hinzugefügt wird, wenn die Ablaufverfolgungssätze A und B beide aktiv sind. In der Delta-Spalte wird der prozentuale Unterschied zwischen Ablaufverfolgungssatz A und B berechnet |

### Reaktionszeitverteilung

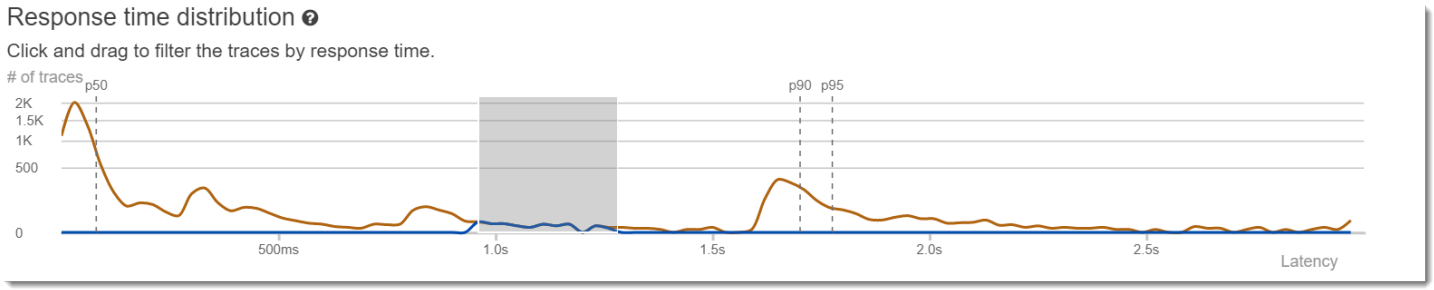
Die X-Ray Analytics-Konsole generiert zwei Hauptdiagramme, mit denen Sie Traces visualisieren können: Verteilung der Reaktionszeit und Zeitreihenaktivität. In diesem und den folgenden Abschnitten sehen Sie jeweils Beispiele dafür. Zudem werden die Grundlagen zum Lesen der Diagramme erläutert.

Im Folgenden sehen Sie die Farben, die dem Liniendiagramm „Reaktionszeit“ zugeordnet sind (das Zeitreihendiagramm verwendet dasselbe Farbschema):

- Alle Spuren in der Gruppe — grau
- Abgerufene Spuren — orange
- Gefilterter Trace-Satz A — grün
- Gefilterter Trace-Satz B — blau

#### Example — Verteilung der Antwortzeiten

Bei der Verteilung der Reaktionszeit handelt es sich um ein Diagramm, das die Anzahl der Ablaufverfolgungen innerhalb einer bestimmten Reaktionszeit zeigt. Klicken und ziehen Sie, um eine Auswahl innerhalb der Verteilung der Reaktionszeit zu treffen. Dadurch wird ein Filter im Arbeitssatz der Ablaufverfolgung mit der Bezeichnung `responseTime` für alle Ablaufverfolgungen innerhalb einer bestimmten Reaktionszeit ausgewählt und erstellt.

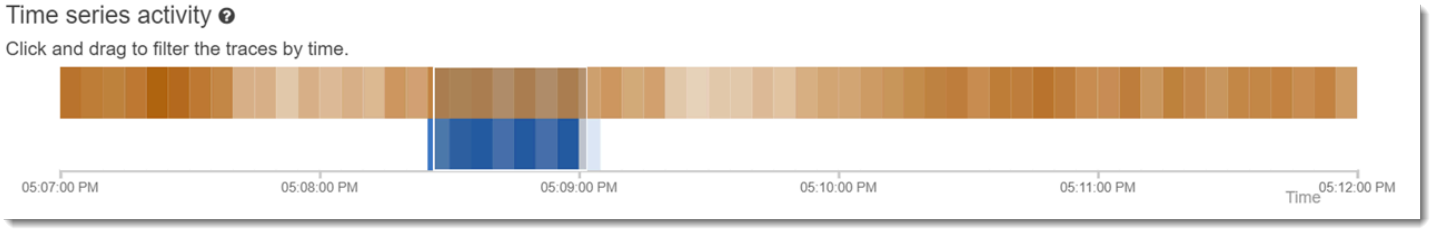



### Zeitreihenaktivität

Das Diagramm „Zeitreihenaktivität“ zeigt die Anzahl der Ablaufverfolgungen in einem bestimmten Zeitraum. Die Farbindikatoren spiegeln die Farben im Liniendiagramm der Reaktionszeitverteilung wider. Je dunkler und voller der Farbblock innerhalb der Aktivitätsreihe, desto mehr Ablaufverfolgungen werden zum angegebenen Zeitpunkt dargestellt.

Example — Aktivität in Zeitreihen

Klicken und ziehen Sie, um im Diagramm „Zeitreihenaktivität“ eine Auswahl zu treffen. Dadurch wird ein Filter mit der Bezeichnung `timerange` im Arbeitssatz der Ablaufverfolgung für alle Ablaufverfolgungen innerhalb eines bestimmten Zeitraums ausgewählt und erstellt.




Workflow-Beispiele

Die folgenden Beispiele zeigen allgemeine Anwendungsfälle für die X-Ray Analytics-Konsole. Jedes Beispiel zeigt eine wichtige Funktion der Konsolenumgebung. Die Beispiele folgen als Gruppe einem grundlegenden Fehlerbehebungsworkflow. In den Schritten wird beschrieben, wie Sie zunächst nicht betriebsbereite Knoten erkennen und dann mit der Analytics-Konsole interagieren, um vergleichende Abfragen automatisch zu generieren. Nachdem Sie den Bereich durch Abfragen eingegrenzt haben, sehen Sie sich abschließend die Details der für Sie relevanten Ablaufverfolgungen an, um zu ermitteln, was sich negativ auf die Integrität Ihres Services auswirkt.

Achten Sie auf Fehler auf im Servicediagramm.

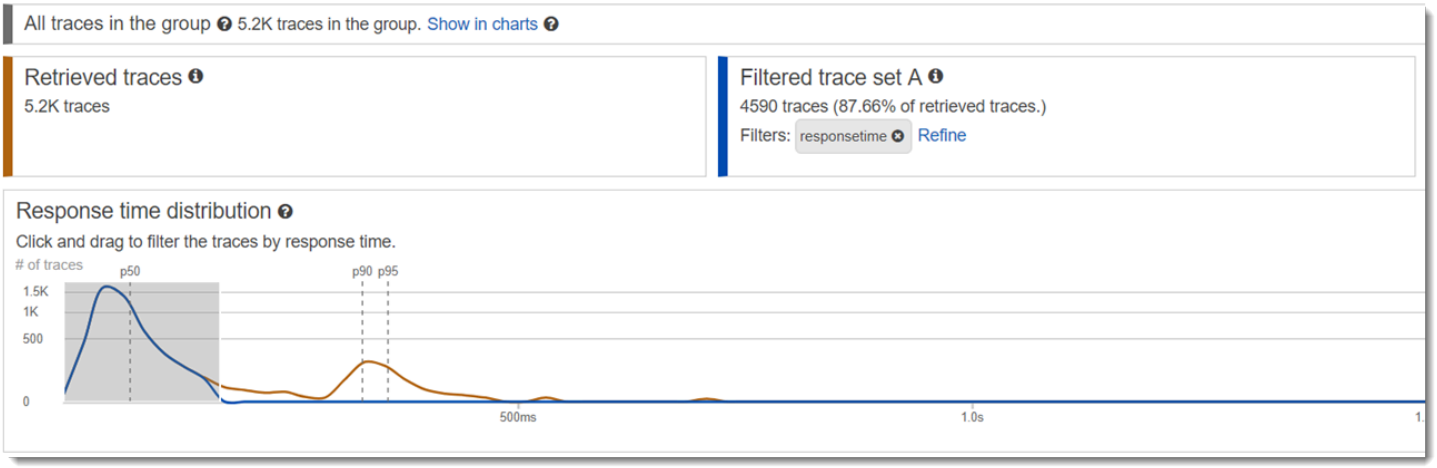
Die Trace-Map zeigt den Zustand der einzelnen Knoten an, indem sie sie anhand des Verhältnisses von erfolgreichen Aufrufen zu Fehlern und Störungen einfärbt. Ein roter Prozentsatz auf Ihrem Knoten signalisiert eine Störung. Verwenden Sie die X-Ray Analytics-Konsole, um das Problem zu untersuchen.

Weitere Informationen zum Lesen der Trace-Map finden Sie unter Verwenden der X-Ray-Trace-Map.


Spitzen der Reaktionszeit erkennen

Mithilfe der Reaktionszeitverteilung können Sie Spitzen in der Reaktionszeit beobachten. Durch die Auswahl der Spitzen in der Reaktionszeit werden die Tabellen unterhalb der Diagramme aktualisiert und enthalten dann alle zugehörigen Metriken, z. B. die Statuscodes.

Wenn Sie klicken und ziehen, wählt X-Ray einen Filter aus und erstellt ihn. Er wird in einem grauen Schatten über den grau unterlegten Linien dargestellt. Sie können diesen Schatten jetzt entlang der Verteilung nach links und rechts ziehen, um die Auswahl und den Filter zu aktualisieren.




Alle Ablaufverfolgungen mit einem Statuscode anzeigen

Sie können Ablaufverfolgungen innerhalb der ausgewählten Spitze anhand der Metriktabellen unterhalb der Diagramme detailliert anzeigen. Durch Klicken auf eine Zeile in der Tabelle HTTP STATUS CODE (HTTP-STATUSCODE können Sie automatisch einen Filter im Arbeitsdatensatz erstellen. Beispiel: Sie können alle Ablaufverfolgungen für Statuscode 500 anzeigen. Dadurch wird ein Filter-Tag auf der Kachel „Ablaufverfolgungssatz“ mit dem Namen `http.status` erstellt.

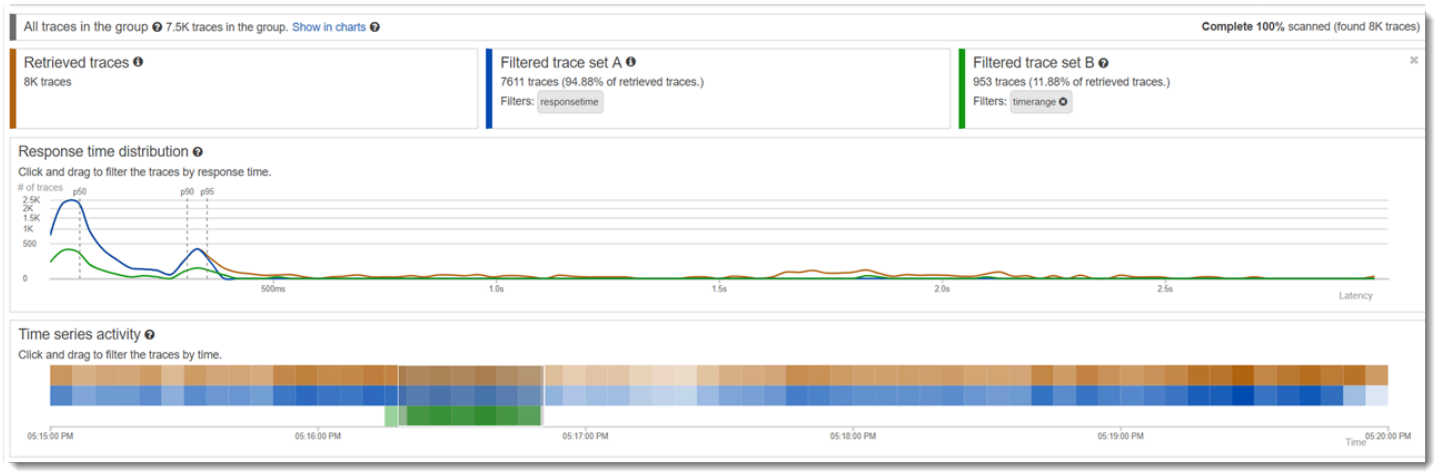
Alle Elemente in einer Untergruppe und einem Benutzer zugeordnete Elemente anzeigen

Führen Sie einen Drilldown in den Fehler auf Grundlage von Benutzer, URL, Ursache Reaktionszeit oder anderen vordefinierten Attributen durch. Wenn Sie beispielsweise den Satz der Ablaufverfolgungen mit Statuscode 500 filtern möchten, wählen Sie eine Zeile aus der Tabelle USERS (BENUTZER) aus. Dies führt zu zwei Filter-Tags auf der Kachel „Ablaufverfolgungssatz“: `http.status`, wie zuvor festgelegt, und `user`.

Vergleichen Sie zwei Ablaufverfolgungssätze mit unterschiedlichen Kriterien

Vergleichen Sie verschiedene Benutzer und ihre POST-Anfragen, um weitere Abweichungen und Korrelationen zu finden. Wenden Sie Ihren ersten Satz Filter an. Sie sind anhand einer blauen Linie in der Reaktionszeitverteilung definiert. Wählen Sie dann Compare (Vergleichen) aus. Zu Beginn wird dadurch eine Kopie der Filter im Ablaufverfolgungssatz A erstellt.

Um fortzufahren, definieren Sie einen neuen Satz Filter, die auf Ablaufverfolgungssatz B angewendet werden. Dieser zweite Satz wird durch eine grüne Linie dargestellt. Das folgende Beispiel zeigt die verschiedenen Linien entsprechend dem blauen und grünen Farbschema.



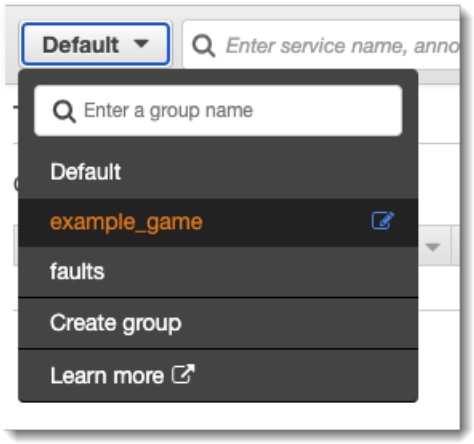

Zeigen Sie Details einer interessanten Ablaufverfolgung an

Wenn Sie mithilfe der Konsolenfilter den Umfang eingrenzen, erhält die Ablaufverfolgungsliste unterhalb der Metriktabellen mehr Bedeutung. Die Tabelle mit der Ablaufverfolgungsliste vereint Informationen über URL, USER (BENUTZER) und STATUS CODE (STATUSCODE) in einer Ansicht. Weitere Einblicke erhalten Sie, wenn Sie eine Zeile aus dieser Tabelle auswählen und so die Detailseite der Ablaufverfolgung öffnen und die Zeitleiste und Rohdaten anzeigen.

Gruppen konfigurieren

Gruppen sind eine Sammlung von Ablaufverfolgungen, die durch einen Filterausdruck definiert sind. Sie können Gruppen verwenden, um zusätzliche Servicegraphen zu erstellen und CloudWatch Amazon-Metriken bereitzustellen. Sie können die AWS X-Ray Konsole oder die X-Ray-API verwenden, um Gruppen für Ihre Dienste zu erstellen und zu verwalten. In diesem Thema wird beschrieben, wie Gruppen mithilfe der X-Ray-Konsole erstellt und verwaltet werden. Informationen zur Verwaltung von Gruppen mithilfe der X-Ray-API finden Sie unterKonfiguration von Sampling-, Gruppen- und Verschlüsselungseinstellungen mit der X-Ray-API.

Sie können Gruppen von Traces für Trace-Maps, Traces oder Analysen erstellen. Wenn Sie eine Gruppe erstellen, ist die Gruppe als Filter im Gruppen-Dropdownmenü auf allen drei Seiten verfügbar: Trace Map, Traces und Analytics.




Gruppen werden über ihren Namen oder einen Amazon-Ressourcennamen (ARN) identifiziert und enthalten einen Filterausdruck. Der Service vergleicht eingehende Ablaufverfolgungen mit dem Ausdruck und speichert sie entsprechend. Weitere Hinweise zum Erstellen eines Filterausdrucks finden Sie unterVerwenden Sie Filterausdrücke.

Durch das Aktualisieren des Filterausdrucks einer Gruppe werden die bereits aufgezeichneten Daten nicht geändert. Die Aktualisierung gilt nur für nachfolgende Ablaufverfolgungen. Dies kann dazu führen, dass im Diagramm neue und alte Ausdrücke zusammengeführt werden. Um dies zu vermeiden, löschen Sie eine aktuelle Gruppe und erstellen Sie eine neue.

> **Note**
>
> Gruppen werden anhand der Anzahl der abgerufenen Ablaufverfolgungen, die dem Filterausdruck entsprechen, in Rechnung gestellt. Weitere Informationen finden Sie unter AWS X-Ray Preise.

### Erstellen einer Gruppe

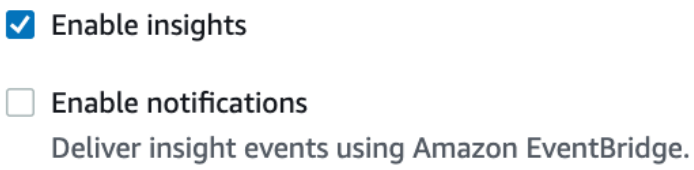
> **Note**
>
> Sie können jetzt X-Ray-Gruppen von der CloudWatch Amazon-Konsole aus konfigurieren. Sie können die X-Ray-Konsole auch weiterhin verwenden.

CloudWatch console

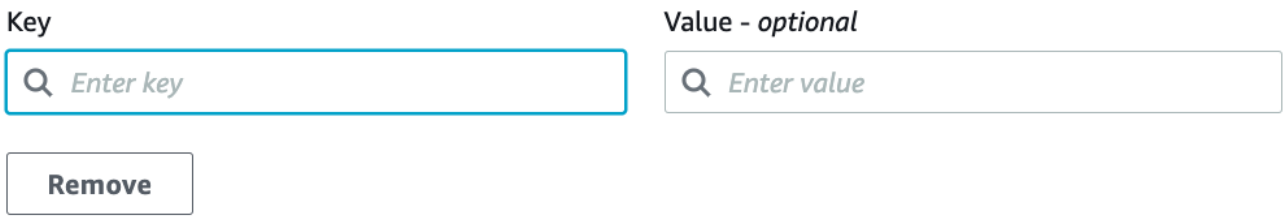
1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im linken Navigationsbereich Einstellungen aus.
3. Wählen Sie im Bereich X-Ray-Traces unter Gruppen die Option Einstellungen anzeigen.
4. Wählen Sie oberhalb der Gruppenliste die Option Gruppe erstellen aus.
5. Geben Sie auf der Seite Gruppe erstellen einen Namen für die Gruppe ein. Ein Gruppenname kann maximal 32 Zeichen lang sein und alphanumerische Zeichen und Bindestriche enthalten. Bei Gruppennamen wird zwischen Groß- und Kleinschreibung unterschieden.
6. Geben Sie einen Filterausdruck ein. Weitere Informationen zum Erstellen eines Filterausdrucks finden Sie unterVerwenden Sie Filterausdrücke. Im folgenden Beispiel filtert die Gruppe nach Fehlerspuren vom Dienst `api.example.com` und nach Anfragen an den Dienst, bei denen die Antwortzeit mindestens fünf Sekunden betrug.

   ```
   fault = true AND http.url CONTAINS "example/game" AND responsetime >= 5
   ```

7. Aktivieren oder deaktivieren Sie in Insights den Insights-Zugriff für die Gruppe. Weitere Informationen über Funde sind unter Verwenden Sie X-Ray Insights verfügbar.

8. Wählen Sie unter Tags die Option Neues Tag hinzufügen aus, um einen Tag-Schlüssel und optional einen Tag-Wert einzugeben. Fügen Sie nach Bedarf weitere Tags hinzu. Tag-Schlüssel müssen eindeutig sein. Um ein Tag zu löschen, wählen Sie unter jedem Tag die Option Entfernen aus. Weitere Informationen zu Tags erhalten Sie unter Kennzeichnen von Regeln und Gruppen für die Röntgenprobenahme.

9. Wählen Sie Create group (Gruppe erstellen) aus.

X-Ray console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die X-Ray-Konsole unter https://console.aws.amazon.com/xray/home.
2. Öffnen Sie die Seite Gruppe erstellen über die Gruppenseite im linken Navigationsbereich oder über das Gruppenmenü auf einer der folgenden Seiten: Trace Map, Traces und Analytics.
3. Geben Sie auf der Seite Gruppe erstellen einen Namen für die Gruppe ein. Ein Gruppenname kann maximal 32 Zeichen lang sein und alphanumerische Zeichen und Bindestriche enthalten. Bei Gruppennamen wird zwischen Groß- und Kleinschreibung unterschieden.
4. Geben Sie einen Filterausdruck ein. Weitere Informationen zum Erstellen eines Filterausdrucks finden Sie unterVerwenden Sie Filterausdrücke. Im folgenden Beispiel filtert die Gruppe nach Fehlerspuren vom Dienst `api.example.com` und nach Anfragen an den Dienst, bei denen die Antwortzeit mindestens fünf Sekunden betrug.

   ```
   fault = true AND http.url CONTAINS "example/game" AND responsetime >= 5
   ```

5. Aktivieren oder deaktivieren Sie in Insights den Insights-Zugriff für die Gruppe. Weitere Informationen über Funde sind unter Verwenden Sie X-Ray Insights verfügbar.

6. Geben Sie im Feld Tags einen Tag-Schlüssel und optional einen Tag-Wert ein. Wenn Sie ein Tag hinzufügen, wird eine neue Zeile angezeigt, in der Sie ein anderes Tag eingeben können. Tag-Schlüssel müssen eindeutig sein. Um ein Tag zu löschen, wählen Sie X am Ende der Tag-Zeile aus. Weitere Informationen zu Tags erhalten Sie unter Kennzeichnen von Regeln und Gruppen für die Röntgenprobenahme.

7. Wählen Sie Create group (Gruppe erstellen) aus.

### Wenden Sie eine Gruppe an

#### CloudWatch console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Öffnen Sie im Navigationsbereich unter X-Ray Traces eine der folgenden Seiten:
   - Trace-Map
   - Ablaufverfolgungen
3. Geben Sie einen Gruppennamen in den Gruppenfilter Filter by X-Ray ein. Die auf der Seite angezeigten Daten ändern sich entsprechend dem in der Gruppe festgelegten Filterausdruck.

#### X-Ray console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die X-Ray-Konsole unter https://console.aws.amazon.com/xray/home.
2. Öffnen Sie im Navigationsbereich eine der folgenden Seiten:
   - Karte verfolgen
   - Ablaufverfolgungen
   - Analysen
3. Wählen Sie im Gruppenmenü die Gruppe aus, in der Sie sie erstellt habenthe section called “Erstellen einer Gruppe”. Die auf der Seite angezeigten Daten ändern sich entsprechend dem in der Gruppe festgelegten Filterausdruck.

### Bearbeiten Sie eine Gruppe

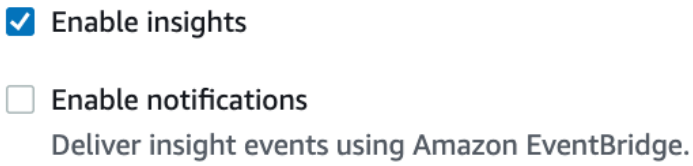
#### CloudWatch console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im linken Navigationsbereich Einstellungen aus.
3. Wählen Sie im Bereich X-Ray-Traces unter Gruppen die Option Einstellungen anzeigen.
4. Wählen Sie im Bereich Gruppen eine Gruppe aus und klicken Sie dann auf Bearbeiten.

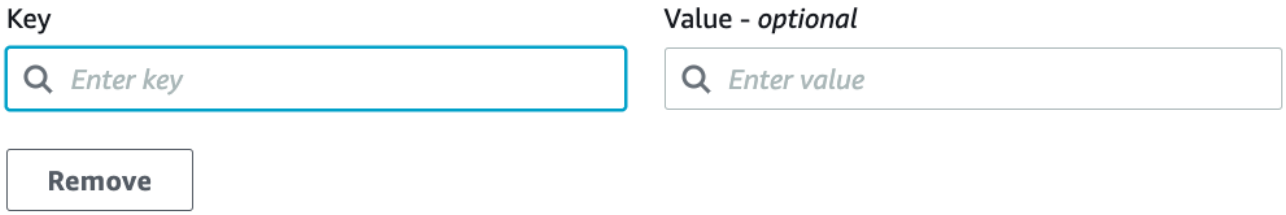
5. Sie können eine Gruppe zwar nicht umbenennen, aber Sie können den Filterausdruck aktualisieren. Weitere Informationen zum Erstellen eines Filterausdrucks finden Sie unterVerwenden Sie Filterausdrücke. Im folgenden Beispiel filtert die Gruppe nach Fehlerablaufverfolgungen des Dienstes`api.example.com`, bei dem die URL-Adresse der Anfrage Folgendes enthält`example/game`: Die Antwortzeit für Anfragen betrug mindestens fünf Sekunden.

```
fault = true AND http.url CONTAINS "example/game" AND responsetime >= 5
```

6. Aktivieren oder deaktivieren Sie in Insights den Insights-Zugriff für die Gruppe. Weitere Informationen über Funde sind unter Verwenden Sie X-Ray Insights verfügbar.




7. Wählen Sie unter Tags die Option Neues Tag hinzufügen aus, um einen Tag-Schlüssel und optional einen Tag-Wert einzugeben. Fügen Sie nach Bedarf weitere Tags hinzu. Tag-Schlüssel müssen eindeutig sein. Um ein Tag zu löschen, wählen Sie unter jedem Tag die Option Entfernen aus. Weitere Informationen zu Tags erhalten Sie unter Kennzeichnen von Regeln und Gruppen für die Röntgenprobenahme.




8. Wenn Sie mit dem Aktualisieren der Gruppe fertig sind, wählen Sie Gruppe aktualisieren.

X-Ray console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die X-Ray-Konsole unter https://console.aws.amazon.com/xray/home.
2. Gehen Sie wie folgt vor, um die Seite „Gruppe bearbeiten“ zu öffnen.

   a. Wählen Sie auf der Seite Gruppen den Namen einer Gruppe aus, um sie zu bearbeiten.

b. Zeigen Sie im Gruppenmenü auf einer der folgenden Seiten auf eine Gruppe, und wählen Sie dann Bearbeiten aus.

   - Trace-Map
   - Ablaufverfolgungen
   - Analysen

3. Sie können eine Gruppe zwar nicht umbenennen, aber Sie können den Filterausdruck aktualisieren. Weitere Informationen zum Erstellen eines Filterausdrucks finden Sie unterVerwenden Sie Filterausdrücke. Im folgenden Beispiel filtert die Gruppe nach Fehlerablaufverfolgungen des Dienstes`api.example.com`, bei dem die URL-Adresse der Anfrage Folgendes enthält`example/game`: Die Antwortzeit für Anfragen betrug mindestens fünf Sekunden.

```
fault = true AND http.url CONTAINS "example/game" AND responsetime >= 5
```

4. Aktivieren oder deaktivieren Sie in Insights Insights und Insights-Benachrichtigungen für die Gruppe. Weitere Informationen über Funde sind unter Verwenden Sie X-Ray Insights verfügbar.


5. Bearbeiten Sie unter Tags die Tag-Schlüssel und -Werte. Tag-Schlüssel müssen eindeutig sein. Tag-Werte sind optional; Sie können Werte löschen, wenn Sie möchten. Um ein Tag zu löschen, wählen Sie X am Ende der Tagzeile aus. Weitere Informationen zu Tags erhalten Sie unter Kennzeichnen von Regeln und Gruppen für die Röntgenprobenahme.


6. Wenn Sie mit dem Aktualisieren der Gruppe fertig sind, wählen Sie Gruppe aktualisieren.

### Klonen Sie eine Gruppe

Durch das Klonen einer Gruppe wird eine neue Gruppe erstellt, die den Filterausdruck und die Tags einer vorhandenen Gruppe enthält. Wenn Sie eine Gruppe klonen, hat die neue Gruppe denselben Namen wie die Gruppe, aus der sie geklont wurde, wobei der Name an den Namen `-clone` angehängt wird.

#### CloudWatch console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im linken Navigationsbereich Einstellungen aus.
3. Wählen Sie im Bereich X-Ray-Traces unter Gruppen die Option Einstellungen anzeigen.
4. Wählen Sie im Bereich Gruppen eine Gruppe aus und klicken Sie dann auf Klonen.
5. Auf der Seite Gruppe erstellen lautet der Name der Gruppe *`Gruppenname.`* `-clone` Geben Sie optional einen neuen Namen für die Gruppe ein. Ein Gruppenname kann maximal 32 Zeichen lang sein und alphanumerische Zeichen und Bindestriche enthalten. Bei Gruppennamen wird zwischen Groß- und Kleinschreibung unterschieden.
6. Sie können den Filterausdruck aus der vorhandenen Gruppe beibehalten oder optional einen neuen Filterausdruck eingeben. Weitere Informationen zum Erstellen eines Filterausdrucks finden Sie unterVerwenden Sie Filterausdrücke. Im folgenden Beispiel filtert die Gruppe nach Fehlerspuren vom Dienst `api.example.com` und nach Anfragen an den Dienst, bei denen die Antwortzeit mindestens fünf Sekunden betrug.

   ```
   service("api.example.com") { fault = true OR responsetime >= 5 }
   ```
7. Bearbeiten Sie unter Tags die Tag-Schlüssel und -Werte, falls erforderlich. Tag-Schlüssel müssen eindeutig sein. Tag-Werte sind optional; Sie können Werte löschen, wenn Sie möchten. Um ein Tag zu löschen, wählen Sie X am Ende der Tagzeile aus. Weitere Informationen zu Tags erhalten Sie unter Kennzeichnen von Regeln und Gruppen für die Röntgenprobenahme.
8. Wählen Sie Create group (Gruppe erstellen) aus.

#### X-Ray console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die X-Ray-Konsole unter https://console.aws.amazon.com/xray/home.

2. Öffnen Sie im linken Navigationsbereich die Gruppenseite und wählen Sie dann den Namen der Gruppe aus, die Sie klonen möchten.
3. Wählen Sie im Menü Aktionen die Option Gruppe klonen aus.
4. Auf der Seite Gruppe erstellen lautet der Name der Gruppe *`Gruppenname.`* `-clone` Geben Sie optional einen neuen Namen für die Gruppe ein. Ein Gruppenname kann maximal 32 Zeichen lang sein und alphanumerische Zeichen und Bindestriche enthalten. Bei Gruppennamen wird zwischen Groß- und Kleinschreibung unterschieden.
5. Sie können den Filterausdruck aus der vorhandenen Gruppe beibehalten oder optional einen neuen Filterausdruck eingeben. Weitere Informationen zum Erstellen eines Filterausdrucks finden Sie unterVerwenden Sie Filterausdrücke. Im folgenden Beispiel filtert die Gruppe nach Fehlerspuren vom Dienst `api.example.com` und nach Anfragen an den Dienst, bei denen die Antwortzeit mindestens fünf Sekunden betrug.

   ```
   service("api.example.com") { fault = true OR responsetime >= 5 }
   ```

6. Bearbeiten Sie unter Tags die Tag-Schlüssel und -Werte, falls erforderlich. Tag-Schlüssel müssen eindeutig sein. Tag-Werte sind optional; Sie können Werte löschen, wenn Sie möchten. Um ein Tag zu löschen, wählen Sie X am Ende der Tagzeile aus. Weitere Informationen zu Tags erhalten Sie unter Kennzeichnen von Regeln und Gruppen für die Röntgenprobenahme.
7. Wählen Sie Create group (Gruppe erstellen) aus.

### Löschen einer Gruppe

Folgen Sie den Schritten in diesem Abschnitt, um eine Gruppe zu löschen. Sie können die Standardgruppe nicht löschen.

#### CloudWatch console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im linken Navigationsbereich Einstellungen aus.
3. Wählen Sie im Bereich X-Ray-Traces unter Gruppen die Option Einstellungen anzeigen.
4. Wählen Sie im Bereich Gruppen eine Gruppe aus und klicken Sie dann auf Löschen.
5. Wenn Sie zur Bestätigung aufgefordert werden, wählen Sie Löschen.

X-Ray console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die X-Ray-Konsole unter https://console.aws.amazon.com/xray/home.
2. Öffnen Sie im linken Navigationsbereich die Gruppenseite und wählen Sie dann den Namen der Gruppe aus, die Sie löschen möchten.
3. Wählen Sie im Menü Aktionen die Option Gruppe löschen aus.
4. Wenn Sie zur Bestätigung aufgefordert werden, wählen Sie Löschen.

### Gruppenmetriken in Amazon anzeigen CloudWatch

Nachdem eine Gruppe erstellt wurde, werden eingehende Traces mit dem Filterausdruck der Gruppe verglichen, während sie im X-Ray-Dienst gespeichert werden. Metriken für die Anzahl der Traces, die den einzelnen Kriterien entsprechen, werden CloudWatch jede Minute auf Amazon veröffentlicht. Wenn Sie auf der Seite Gruppe bearbeiten die Option Metrik anzeigen wählen, wird in der CloudWatch Konsole die Seite Metrik geöffnet. Weitere Informationen zur Verwendung von CloudWatch Metriken finden Sie unter Using Amazon CloudWatch Metrics im  CloudWatch Amazon-Benutzerhandbuch.

CloudWatch console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im linken Navigationsbereich Einstellungen aus.
3. Wählen Sie im Bereich X-Ray-Traces unter Gruppen die Option Einstellungen anzeigen.
4. Wählen Sie im Bereich Gruppen eine Gruppe aus und klicken Sie dann auf Bearbeiten.
5. Wählen Sie auf der Seite „Gruppe bearbeiten“ die Option Metrik anzeigen aus.

   Die Seite Metriken der CloudWatch Konsole wird auf einer neuen Registerkarte geöffnet.

X-Ray console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die X-Ray-Konsole unter https://console.aws.amazon.com/xray/home.
2. Öffnen Sie im linken Navigationsbereich die Gruppenseite und wählen Sie dann den Namen einer Gruppe aus, für die Sie Messwerte anzeigen möchten.

3. Wählen Sie auf der Seite „Gruppe bearbeiten“ die Option Metrik anzeigen aus.

   Die Seite Metriken der CloudWatch Konsole wird auf einer neuen Registerkarte geöffnet.

### Konfigurieren Sie Stichprobenregeln

Sie können die AWS X-Ray Konsole verwenden, um Sampling-Regeln für Ihre Dienste zu konfigurieren. Das X-Ray SDK AWS-Services , das Active Tracing mit Sampling-Konfiguration unterstützt, verwendet Sampling-Regeln, um zu bestimmen, welche Anfragen aufgezeichnet werden sollen.

### Konfigurieren Sie Stichprobenregeln

Sie können Sampling für die folgenden Anwendungsfälle konfigurieren:

- API Gateway Entrypoint — API Gateway unterstützt Sampling und Active Tracing. Informationen zum Aktivieren der aktiven Ablaufverfolgung für eine API-Stufe finden Sie unter Unterstützung für aktives Amazon API Gateway-Tracing für AWS X-Ray.
- AWS AppSync— AWS AppSync unterstützt Sampling und aktives Tracing. Informationen zum Aktivieren der aktiven Ablaufverfolgung für AWS AppSync Anfragen finden Sie unter Tracing with AWS X-Ray.
- Instrument X-Ray SDK auf Computerplattformen — Bei der Verwendung von Rechenplattformen wie Amazon EC2, Amazon ECS oder wird Sampling unterstützt AWS Elastic Beanstalk, wenn die Anwendung mit dem neuesten X-Ray-SDK instrumentiert wurde.

### Anpassen von Samplingregeln

Durch die Anpassung der Stichprobenregeln können Sie die Menge der aufgenommenen Daten kontrollieren. Sie können auch das Sampling-Verhalten ändern, ohne Ihren Code zu ändern oder erneut bereitzustellen. Stichprobenregeln teilen dem X-Ray SDK mit, wie viele Anfragen für eine Reihe von Kriterien aufgezeichnet werden müssen. Standardmäßig zeichnet das X-Ray-SDK die erste Anfrage auf, die zu Beginn jeder Sekunde empfangen wird, und fünf Prozent aller weiteren Anfragen. Eine Anfrage pro Sekunde ist das Reservoir. Dadurch wird sichergestellt, dass jede Sekunde mindestens eine Ablaufverfolgung aufgezeichnet wird, solange der Dienst Anfragen verarbeitet. Fünf Prozent ist die Rate, mit der die über die Reservoirgröße hinausgehenden Anforderungen geprüft werden.

Sie können das X-Ray SDK so konfigurieren, dass es Sampling-Regeln aus einem JSON-Dokument liest, das Sie in Ihren Code einbinden. Wenn Sie jedoch mehrere Instances Ihres Services ausführen, führt jede Instance das Sampling unabhängig aus. Dies bewirkt, dass sich der Gesamtprozentsatz der geprüften Anforderungen erhöht, da die Reservoirs aller Instances effektiv zusammengezählt werden. Um lokale Sampling-Regeln zu aktualisieren, müssen Sie Ihren Code außerdem erneut bereitstellen.

Indem Sie Sampling-Regeln in der X-Ray-Konsole definieren und das SDK so konfigurieren, dass es Regeln aus dem X-Ray-Dienst liest, können Sie beide Probleme vermeiden. Der Service verwaltet die Reservoirs für jede Regel und weist jeder Instance Ihres Services Kontingente zu, um das Reservoir gleichmäßig zu verteilen, basierend auf der Anzahl der Instances, die ausgeführt werden. Das Reservoir-Limit wird gemäß den von Ihnen festgelegten Regeln berechnet. Da die Regeln im Dienst konfiguriert sind, können Sie Regeln verwalten, ohne zusätzliche Implementierungen vornehmen zu müssen. Weitere Informationen zum AWS SDK finden Sie unter Verwenden eines SDK.

> **Note**
>
> X-Ray verwendet bei der Anwendung von Stichprobenregeln einen Best-Effort-Ansatz, und in einigen Fällen entspricht die effektive Abtastrate möglicherweise nicht genau den konfigurierten Abtastregeln. Im Laufe der Zeit sollte die Anzahl der gesampelten Anfragen jedoch in der Nähe des konfigurierten Prozentsatzes liegen.

Sie können jetzt X-Ray-Sampling-Regeln von der CloudWatch Amazon-Konsole aus konfigurieren. Sie können die X-Ray-Konsole auch weiterhin verwenden.

CloudWatch console

Um Sampling-Regeln in der CloudWatch Konsole zu konfigurieren

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im linken Navigationsbereich Einstellungen aus.
3. Wählen Sie im Bereich X-Ray-Traces unter Sampling-Regeln die Option Einstellungen anzeigen.
4. Um eine Regel zu erstellen, wählen Sie Create sampling rule (Samplingregel erstellen) aus.

Um eine Regel zu bearbeiten, wählen Sie eine Regel aus und klicken Sie auf Bearbeiten, um sie zu bearbeiten.

Um eine Regel zu löschen, wählen Sie eine Regel aus und klicken Sie auf Löschen, um sie zu löschen.

X-Ray console

So konfigurieren Sie Sampling-Regeln in der X-Ray-Konsole

1. Öffnen Sie die X-Ray-Konsole.
2. Wählen Sie im linken Navigationsbereich Sampling.
3. Um eine Regel zu erstellen, wählen Sie Create sampling rule (Samplingregel erstellen) aus.

   Um eine Regel zu bearbeiten, wählen Sie den Namen einer Regel aus.

   Um eine Regel zu löschen, wählen Sie eine Regel aus und verwenden das Menü Actions (Aktionen), um diese zu löschen.

Optionen für Samplingregeln

Für jede Regel stehen folgende Optionen zur Verfügung. Für Zeichenfolgenwerte können Platzhalter verwendet werden, um einem einzelnen Zeichen (?) oder null oder mehreren Zeichen (*) zu entsprechen.

Optionen für Samplingregeln

- Regelname (Zeichenfolge) — Ein eindeutiger Name für die Regel.
- Priorität (Ganzzahl zwischen 1 und 9999) — Die Priorität der Stichprobenregel. Services werten Regeln in aufsteigender Reihenfolge der Priorität aus und treffen eine Sampleentscheidung mit der ersten übereinstimmenden Regel.
- Reservoir (nicht negative Ganzzahl) — Eine feste Anzahl von Abgleichsanfragen an das Instrument pro Sekunde, bevor die feste Rate angewendet wird. Das Reservoir wird nicht direkt von Services verwendet, sondern gilt für alle Services, die die Regel gemeinsam verwenden.
- Rate (Ganzzahl zwischen 0 und 100) — Der Prozentsatz der übereinstimmenden Anfragen an das Instrument, nachdem das Reservoir aufgebraucht ist. Wählen Sie bei der Konfiguration einer Stichprobenregel in der Konsole einen Prozentsatz zwischen 0 und 100. Geben Sie bei der

Konfiguration einer Stichprobenregel in einem Client-SDK mithilfe eines JSON-Dokuments einen Prozentwert zwischen 0 und 1 an.

- Dienstname (Zeichenfolge) — Der Name des instrumentierten Dienstes, wie er in der Trace-Map angezeigt wird.
  - X-Ray SDK — Der Dienstname, den Sie auf dem Rekorder konfigurieren.
  - Amazon API Gateway —*api-name*/*stage*.
- Servicetyp (Zeichenfolge) — Der Servicetyp, wie er in der Trace-Map angezeigt wird. Legen Sie für das X-Ray SDK den Diensttyp fest, indem Sie das entsprechende Plugin anwenden:
  - `AWS::ElasticBeanstalk::Environment`— Eine AWS Elastic Beanstalk Umgebung (Plugin).
  - `AWS::EC2::Instance`— Eine Amazon EC2 EC2-Instance (Plugin).
  - `AWS::ECS::Container`— Ein Amazon ECS-Container (Plugin).
  - `AWS::APIGateway::Stage`— Eine Amazon API Gateway Gateway-Phase.
  - `AWS::AppSync::GraphQLAPI` — Eine AWS AppSync API-Anfrage.
- Host (string) — Der Hostname aus dem HTTP-Host-Header.
- HTTP-Methode (Zeichenfolge) — Die Methode der HTTP-Anfrage.
- URL-Pfad (Zeichenfolge) — Der URL-Pfad der Anfrage.
  - X-Ray SDK — Der Pfadteil der HTTP-Anforderungs-URL.
- Ressourcen-ARN (Zeichenfolge) — Der ARN der AWS Ressource, auf der der Dienst ausgeführt wird.
  - X-Ray SDK — Nicht unterstützt. Das SDK kann nur Regeln verwenden, bei denen der Ressourcen-ARN auf * festgelegt ist.
  - Amazon API Gateway — Die Stufe ARN.
- (Optional) Attribute (Schlüssel und Wert) — Segmentattribute, die bekannt sind, als die Stichprobenentscheidung getroffen wurde.
  - X-Ray SDK — Nicht unterstützt. Das SDK ignoriert Regeln, die Attribute angeben.
  - Amazon API Gateway — Header aus der ursprünglichen HTTP-Anfrage.

Beispiele für Samplingregeln

Example — Standardregel ohne Reservoir und niedriger Rate

Sie können das Reservoir und die Rate der Standardregel ändern. Die Standardregel gilt für alle Anforderungen, die nicht mit einer anderen Regel übereinstimmen.

- Reservoir: **`0`**
- Rate: **`5`** (**`0.05`**falls mit einem JSON-Dokument konfiguriert)

Example — Debugging-Regel zur Nachverfolgung aller Anfragen für eine problematische Route

Eine Regel mit hoher Priorität, die vorübergehend für das Debuggen angewendet wird.

- Name der Regel: **`DEBUG - history updates`**
- Priorität: **`1`**
- Reservoir: **`1`**
- Rate: **`100`** (**`1`**falls mit einem JSON-Dokument konfiguriert)
- Name des Dienstes: **`Scorekeep`**
- Service type (Servicetyp): **`*`**
- Gastgeber: **`*`**
- HTTP-Methode: **`PUT`**
- URL-Pfad: **`/history/*`**
- Ressourcen-ARN: **`*`**

Example — Höherer Mindestsatz für POSTs

- Name der Regel: **`POST minimum`**
- Priorität: **`100`**
- Reservoir: **`10`**
- Rate: **`10`** (**`.1`**falls mit einem JSON-Dokument konfiguriert)
- Name des Dienstes: **`*`**
- Service type (Servicetyp): **`*`**
- Gastgeber: **`*`**
- HTTP-Methode: **`POST`**

- URL-Pfad: *****
- Ressourcen-ARN: *****

Konfigurieren Sie Ihren Service so, dass er Stichprobenregeln verwendet

Das X-Ray SDK erfordert eine zusätzliche Konfiguration, um Sampling-Regeln verwenden zu können, die Sie in der Konsole konfigurieren. Im Konfigurationsthema finden Sie in Ihrer Sprache Einzelheiten zur Konfiguration einer Samplingstrategie:

- Java: Samplingregeln
- Gehe zu: Samplingregeln
- Node.js: Samplingregeln
- Python: Samplingregeln
- Rubin: Samplingregeln
- .NET: Samplingregeln

Informationen zu API Gateway finden Sie unterUnterstützung für aktives Amazon API Gateway-Tracing für AWS X-Ray.

Anzeigen von Samplingergebnissen

Auf der Sampling-Seite der X-Ray-Konsole finden Sie detaillierte Informationen darüber, wie Ihre Dienste die einzelnen Sampling-Regeln verwenden.

Die Spalte Trend zeigt, wie die Regel in den letzten paar Minuten verwendet wurde. Jede Spalte zeigt Statistiken für ein 10-Sekunden-Fenster an.

Samplingstatistiken

- Gesamtzahl der übereinstimmenden Regeln: Die Anzahl der Anfragen, die dieser Regel entsprachen. Diese Zahl umfasst keine Anforderungen, die mit dieser Regel übereingestimmt hätten, aber zuvor mit einer Regel mit höherer Priorität übereingestimmt haben.
- Gesamtzahl der gesammelten Anfragen: Die Anzahl der aufgezeichneten Anfragen.
- Stichprobe mit fester Rate: Die Anzahl der Anfragen, bei denen die feste Rate der Regel angewendet wurde.
- Mit Reservoirlimit abgetastet: Die Anzahl der Anfragen, die unter Verwendung einer von X-Ray zugewiesenen Quote abgetastet wurden.

- Aus dem Reservoir ausgeliehen: Die Anzahl der Anfragen, die durch Entleihen aus dem Reservoir entnommen wurden. Wenn ein Service eine Anfrage zum ersten Mal einer Regel zuordnet, wurde ihm von X-Ray noch kein Kontingent zugewiesen. Wenn das Reservoir jedoch mindestens 1 ist, leiht sich der Dienst eine Spur pro Sekunde aus, bis X-Ray eine Quote zuweist.

Weitere Informationen zu Samplingstatistiken und dazu, wie Services Samplingregeln verwenden, finden Sie unter Verwenden von Sampling-Regeln mit der X-Ray-API.

Nächste Schritte

Sie können die X-Ray-API verwenden, um Sampling-Regeln zu verwalten. Mit der API können Sie Regeln erstellen und aktualisieren, und zwar programmgesteuert nach einem Zeitplan oder als Reaktion auf Alarme oder Benachrichtigungen. Anleitungen und weitere Regelbeispiele finden Sie unter Konfiguration von Sampling-, Gruppen- und Verschlüsselungseinstellungen mit der X-Ray-API.

Das X-Ray-SDK und die X-Ray-API verwenden AWS-Services auch, um Probenahmeregeln zu lesen, Probenahmeergebnisse zu melden und Probenahmeziele abzurufen. Services müssen verfolgen, wie oft sie die einzelnen Regeln anwenden, Regeln anhand ihrer Priorität auswerten und Daten aus dem Reservoir abrufen, wenn eine Anfrage mit einer Regel übereinstimmt, für die X-Ray dem Service noch kein Kontingent zugewiesen hat. Weitere Informationen darüber, wie ein Service die API für Sampling verwendet, finden Sie unterVerwenden von Sampling-Regeln mit der X-Ray-API.

Wenn das X-Ray-SDK Sampling-APIs aufruft, verwendet es den X-Ray-Daemon als Proxy. Wenn Sie TCP-Port 2000 bereits verwenden, können Sie den Daemon so konfigurieren, dass er den Proxy auf einem anderen Port ausführt. Details dazu finden Sie unter Konfigurieren des AWS X-Ray Daemon.

Deep Linking für Konsolen

Sie können Routen und Abfragen verwenden, um Deeplinks zu bestimmten Traces oder gefilterten Ansichten von Traces und der Trace-Map zu erstellen.

Konsolenseiten

- Willkommensseite — xray/home#/welcome
- Erste Schritte — xray/home#/getting-started
- Ablaufplan — xray/home#/service-map
- Spuren — xray/home#/traces

Ablaufverfolgungen

Sie können Links für Timeline-, Roh- und Kartenansichten einzelner Ablaufverfolgungen generieren.

Zeitleiste verfolgen — `xray/home#/traces/`*`trace-id`*

Unverarbeitete Trace-Daten — `xray/home#/traces/`*`trace-id`*`/raw`

Example — rohe Trace-Daten

```
https://console.aws.amazon.com/xray/home#/traces/1-57f5498f-d91047849216d0f2ea3b6442/
raw
```

Filterausdrücke

Link zu einer gefilterten Liste von Ablaufverfolgungsdaten.

Ansicht gefilterter Spuren — `xray/home#/traces?filter=`*`filter-expression`*

Example — Ausdruck filtern

```
https://console.aws.amazon.com/xray/home#/traces?filter=service("api.amazon.com")
 { fault = true OR responsetime > 2.5 } AND annotation.foo = "bar"
```

Example — Filterausdruck (URL-kodiert)

```
https://console.aws.amazon.com/xray/home#/traces?filter=service(%22api.amazon.com
%22)%20%7B%20fault%20%3D%20true%20OR%20responsetime%20%3E%202.5%20%7D%20AND
%20annotation.foo%20%3D%20%22bar%22
```

Weitere Informationen zu Filterausdrücken finden Sie unter Verwenden Sie Filterausdrücke.

Zeitraum

Geben Sie einen Zeitraum oder Start- und Endzeit im ISO8601-Format an. Die Zeitbereiche sind in UTC angegeben und können bis zu 6 Stunden lang sein.

Länge der Zeit — `xray/home#/`*`page`*`?timeRange=`*`range-in-minutes`*

Example — Streckenkarte für die letzte Stunde

```
https://console.aws.amazon.com/xray/home#/service-map?timeRange=PT1H
```

Start- und Endzeit — `xray/home#/page?timeRange=start~end`

Example — Sekundengenauer Zeitbereich

```
https://console.aws.amazon.com/xray/home#/traces?
timeRange=2023-7-01T16:00:00~2023-7-01T22:00:00
```

Example — minutengenauer Zeitraum

```
https://console.aws.amazon.com/xray/home#/traces?
timeRange=2023-7-01T16:00~2023-7-01T22:00
```

Region

Geben Sie AWS-Region an, ob auf Seiten in dieser Region verlinkt werden soll. Wenn Sie keine Region angeben, führt Sie die Konsole zur zuletzt besuchten Region.

Region — `xray/home?region=region#/page`

Example — Streckenkarte in den USA West (Oregon) (us-west-2)

```
https://console.aws.amazon.com/xray/home?region=us-west-2#/service-map
```

Wenn Sie eine Region mit anderen Abfrageparametern einbeziehen, steht die Regionsabfrage vor dem Hash und die röntgenspezifischen Abfragen hinter dem Seitennamen.

Example — Streckenkarte für die letzte Stunde in US West (Oregon) (us-west-2)

```
https://console.aws.amazon.com/xray/home?region=us-west-2#/service-map?timeRange=PT1H
```

Kombiniert

Example — aktuelle Spuren mit einem Dauerfilter

```
https://console.aws.amazon.com/xray/home#/traces?timeRange=PT15M&filter=duration%20%3E
%3D%205%20AND%20duration%20%3C%3D%208
```

Output

- Seite — Spuren
- Zeitraum — Letzte 15 Minuten

- Filter — Dauer >= 5 UND Dauer <= 8

# Verwenden Sie ein SDK

Verwenden Sie ein SDK, wenn Sie eine Befehlszeilenschnittstelle verwenden möchten oder mehr benutzerdefinierte Funktionen für die Ablaufverfolgung, Überwachung oder Protokollierung benötigen als die, die in einem AWS Management Console verfügbar sind. Sie können auch ein AWS SDK verwenden, um Programme zu entwickeln, die die X-Ray-APIs verwenden. Sie können entweder das AWS Distro for OpenTelemetry (ADOT) SDK oder das X-Ray SDK verwenden.

Wenn Sie ein SDK verwenden, können Sie Ihrem Workflow sowohl bei der Instrumentierung Ihrer Anwendung als auch bei der Konfiguration Ihres Collectors oder Agenten Anpassungen hinzufügen. Sie können ein SDK für die folgenden Aufgaben verwenden, die Sie mit einem AWS Management Console nicht erledigen können:

- Veröffentlichen Sie benutzerdefinierte Metriken — Stichproben von Metriken mit hoher Auflösung von bis zu 1 Sekunde, verwenden Sie mehrere Dimensionen, um Informationen zu einer Metrik hinzuzufügen, und aggregieren Sie Datenpunkte in einem Statistiksatz.
- Passen Sie Ihren Collector an — Passen Sie die Konfiguration für jeden Teil eines Collectors an, einschließlich Empfänger, Prozessor, Exporter und Konnektor.
- Passen Sie Ihre Instrumentierung an — Passen Sie Segmente und Untersegmente an, fügen Sie benutzerdefinierte Schlüssel-Wert-Paare als Attribute hinzu und erstellen Sie benutzerdefinierte Metriken.
- Erstellen und aktualisieren Sie Stichprobenregeln programmgesteuert.

Verwenden Sie das ADOT SDK, wenn Sie die Flexibilität eines standardisierten OpenTelemetry SDK mit zusätzlichen AWS Sicherheits- und Optimierungsebenen nutzen möchten. Das AWS Distro for OpenTelemetry (ADOT) SDK ist ein herstellerunabhängiges Paket, das die Integration mit Backends von anderen Anbietern und AWS Nichtdiensten ermöglicht, ohne dass Sie Ihren Code neu instrumentieren müssen.

Verwenden Sie das X-Ray-SDK, wenn Sie das X-Ray-SDK bereits verwenden, nur in AWS Backends integrieren und die Art und Weise, wie Sie mit X-Ray oder Ihrem Anwendungscode interagieren, nicht ändern möchten.

Weitere Informationen zu den einzelnen Funktionen finden Sie unterWahl zwischen den AWS SDKs Distro for OpenTelemetry und X-Ray.

# Verwenden Sie das ADOT SDK

Das ADOT SDK besteht aus einer Reihe von Open-Source-APIs, -Bibliotheken und -Agenten, die Daten an Backend-Dienste senden. ADOTwird von mehreren Backends und Agenten unterstützt AWS, lässt sich in mehrere Backends und Agenten integrieren und bietet eine große Anzahl von Open-Source-Bibliotheken, die von der OpenTelemetry Community verwaltet werden. Verwenden Sie das ADOT SDK, um Ihre Anwendung zu instrumentieren und Protokolle, Metadaten, Metriken und Traces zu sammeln. Sie können es auch verwendenADOT, um Dienste zu überwachen und einen Alarm auf der Grundlage Ihrer Messwerte in einzustellen CloudWatch.

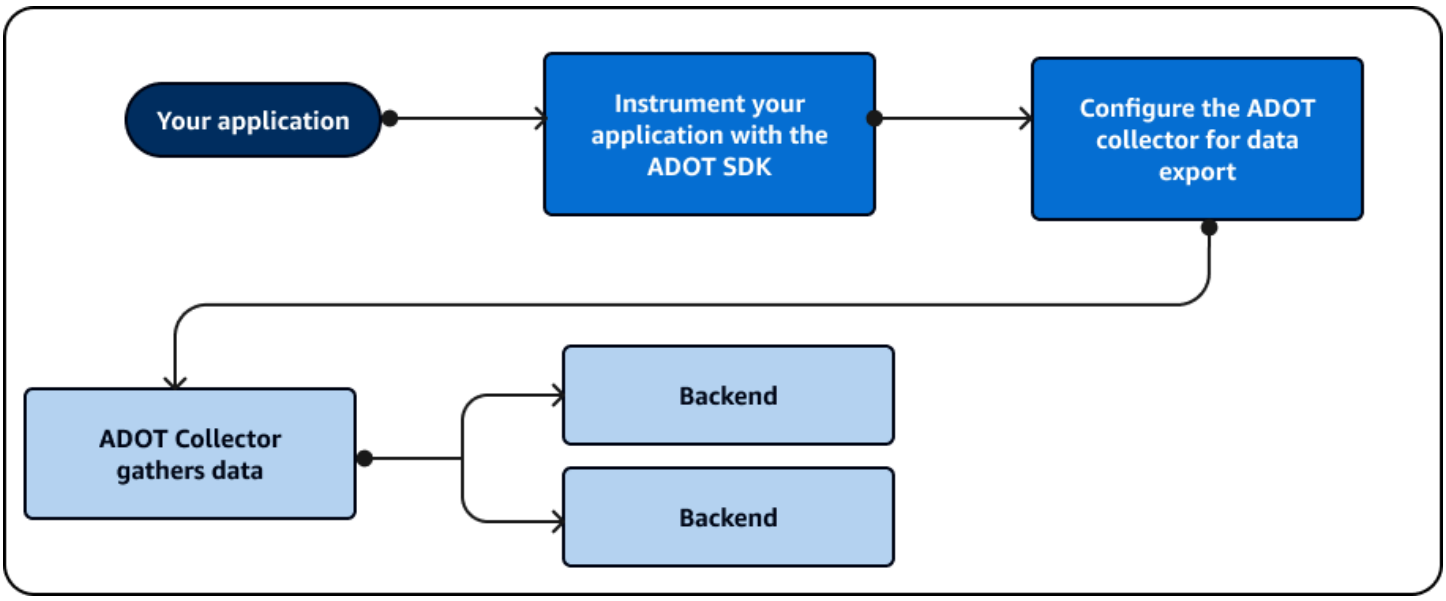
Wenn Sie das ADOT SDK verwenden, haben Sie in Kombination mit einem Agenten die folgenden Optionen:

- Verwenden Sie das ADOT SDK mit dem CloudWatch Agenten — empfohlen.
- Verwenden Sie das ADOT SDK mit dem ADOTCollector — empfohlen, wenn Sie herstellerunabhängige Software mit Sicherheits- und AWS Optimierungsebenen verwenden möchten.

Gehen Sie wie folgt vor, um das ADOT SDK zu verwenden:

- Instrumentieren Sie Ihre Anwendung mithilfe des ADOT SDK. Weitere Informationen finden Sie in der Dokumentation zu Ihrer Programmiersprache in der technischen Dokumentation von ADOT.
- Konfigurieren Sie einen ADOT Collector so, dass er ihm mitteilt, wohin die gesammelten Daten gesendet werden sollen.

Nachdem der ADOT Collector Ihre Daten empfangen hat, sendet er sie an das Backend, das Sie in der ADOT Konfiguration angeben. ADOTkann Daten an mehrere Backends senden, auch an Anbieter außerhalb von AWS, wie in der folgenden Abbildung dargestellt:




AWS wird regelmäßig aktualisiertADOT, um Funktionen hinzuzufügen und sich an das OpenTelemetryFramework anzupassen. Aktualisierungen und future Entwicklungspläne ADOT sind Teil einer Roadmap, die der Öffentlichkeit zugänglich ist. ADOTunterstützt mehrere Programmiersprachen, darunter die folgenden:

- Go
- Java
- JavaScript
- Python
- .NET
- Ruby
- PHP

Wenn Sie Python verwenden, ADOT kann Ihre Anwendung automatisch instrumentieren. Informationen zu den ersten Schritten ADOT finden Sie unter Einführung und Erste Schritte mit der AWS Distribution for OpenTelemetry Collector.

## Verwenden Sie das SDK X-Ray

Das X-Ray SDK besteht aus einer Reihe von AWS APIs und Bibliotheken, die Daten an AWS Backend-Dienste senden. Verwenden Sie das X-Ray SDK, um Ihre Anwendung zu instrumentieren und Trace-Daten zu sammeln. Sie können das X-Ray SDK nicht zum Sammeln von Protokoll- oder Metrikdaten verwenden.

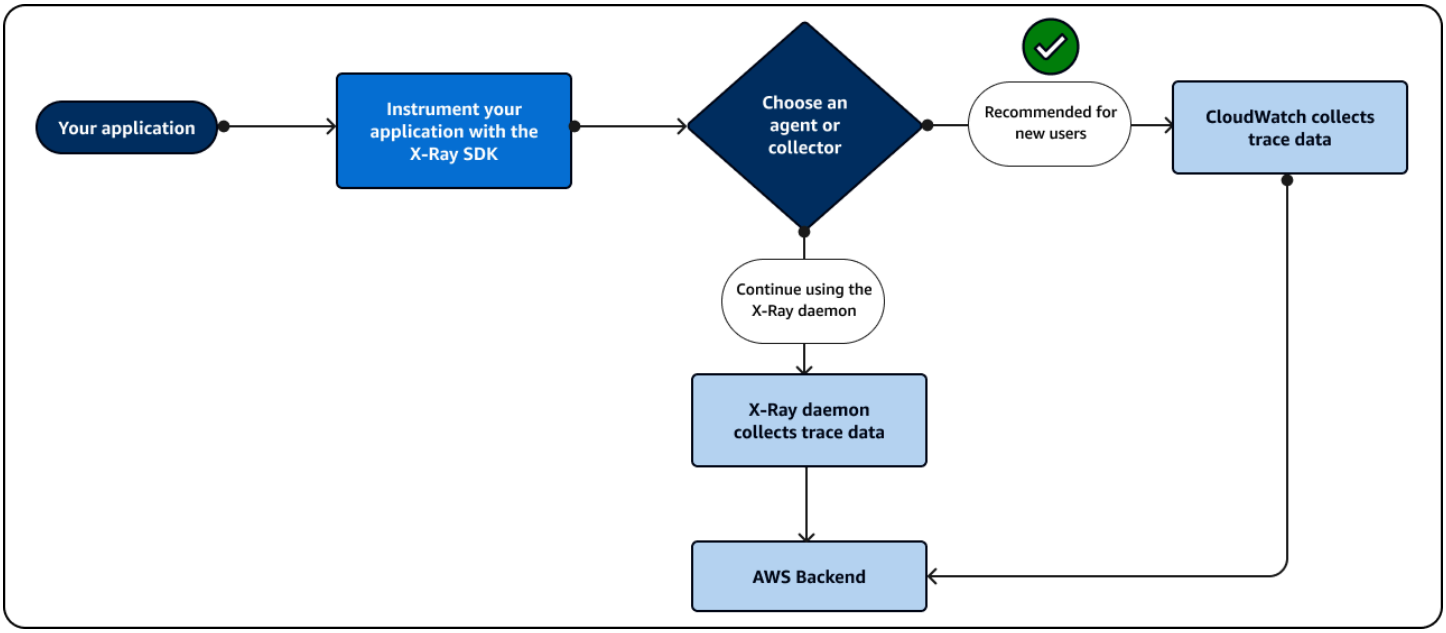
Wenn Sie das X-Ray SDK verwenden, haben Sie in Kombination mit einem Agenten die folgenden Optionen:

- Verwenden Sie das X-Ray-SDK mit AWS X-Ray Dämon — Verwenden Sie dies, wenn Sie Ihren Anwendungscode nicht aktualisieren möchten.
- Verwenden Sie das X-Ray-SDK mit dem CloudWatch Agenten — (empfohlen) Der CloudWatch Agent ist mit dem X-Ray-SDK kompatibel.

Gehen Sie wie folgt vor, um das X-Ray SDK zu verwenden:

- Instrumentieren Sie Ihre Anwendung mit dem X-Ray SDK.
- Konfigurieren Sie einen Collector so, dass er ihm mitteilt, wohin er die gesammelten Daten senden soll. Sie können entweder den CloudWatch Agenten oder den X-Ray-Daemon verwenden, um Ihre Trace-Informationen zu sammeln.

Nachdem der Collector oder Agent Ihre Daten empfangen hat, sendet er sie an ein AWS Backend, das Sie in der Agentenkonfiguration angeben. Das X-Ray SDK kann nur Daten an ein AWS Backend senden, wie in der folgenden Abbildung dargestellt:




Wenn Sie verwendenJava, können Sie das X-Ray SDK verwenden, um Ihre Anwendung automatisch zu instrumentieren. Informationen zu den ersten Schritten mit dem X-Ray-SDK finden Sie in den Bibliotheken, die den folgenden Programmiersprachen zugeordnet sind:

- Go
- Java
- Node.js
- Python
- .NET
- Ruby

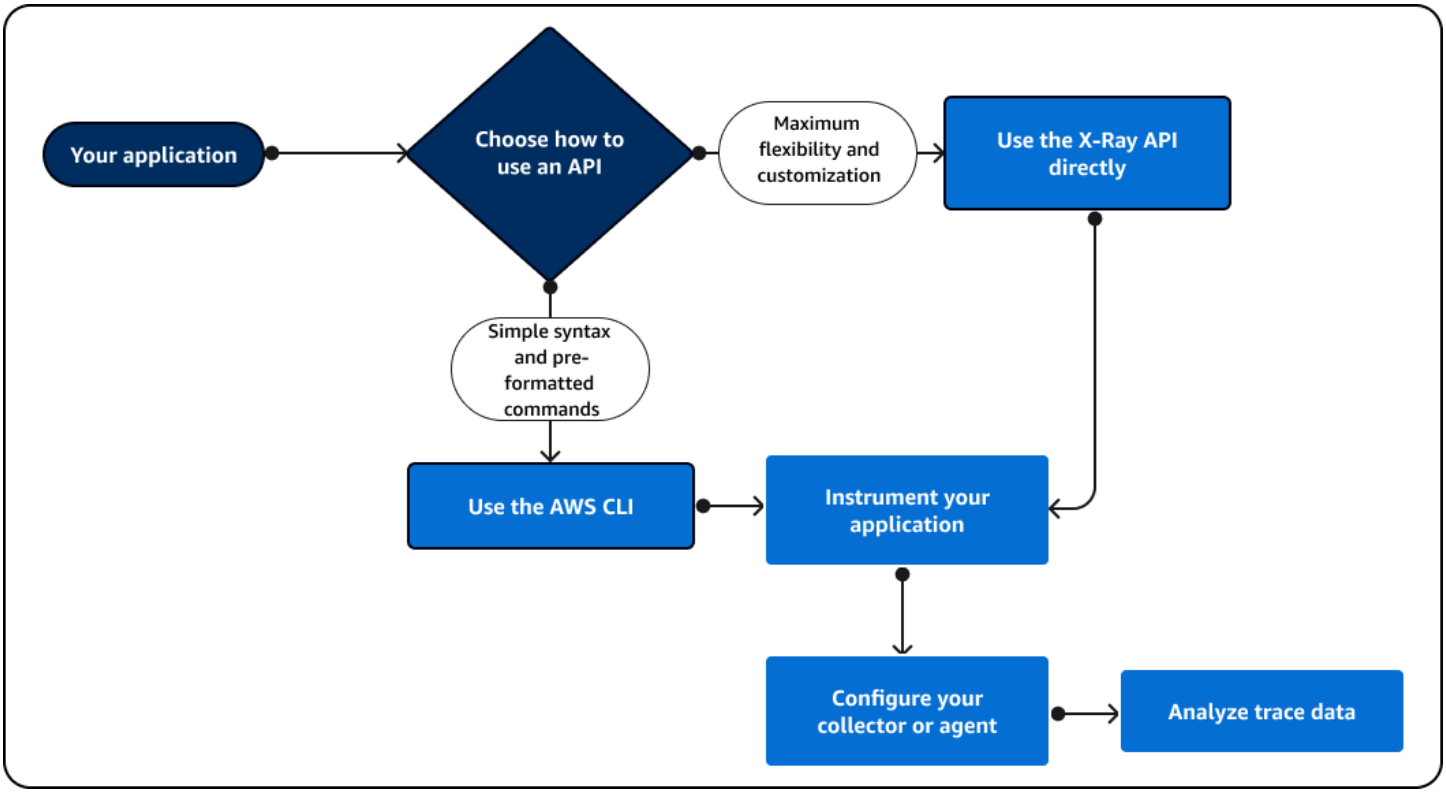
## Verwenden Sie die X-Ray-API

Wenn das X-Ray-SDK Ihre Programmiersprache nicht unterstützt, können Sie entweder die X-Ray-APIs direkt oder die AWS Command Line Interface (AWS CLI) verwenden, um X-Ray-API-Befehle aufzurufen. Wählen Sie anhand der folgenden Anleitung aus, wie Sie mit der API interagieren möchten:

- Verwenden Sie die AWS CLI einfachere Syntax mit vorformatierten Befehlen oder mit Optionen in Ihrer Anfrage.
- Verwenden Sie die X-Ray-API direkt für maximale Flexibilität und Anpassung bei Anfragen, die Sie an X-Ray stellen.

Wenn Sie die X-Ray-API direkt anstelle von verwenden AWS CLI, müssen Sie Ihre Anfrage im richtigen Datenformat parametrisieren und möglicherweise auch die Authentifizierung und Fehlerbehandlung konfigurieren.

Das folgende Diagramm zeigt eine Anleitung zur Auswahl der Interaktion mit der X-Ray-API:




Verwenden Sie die X-Ray-API, um Trace-Daten direkt an X-Ray zu senden. Die X-Ray-API unterstützt alle im X-Ray-SDK verfügbaren Funktionen, einschließlich der folgenden allgemeinen Aktionen:

- PutTraceSegments— Lädt Segmentdokumente auf X-Ray hoch.
- BatchGetTraces— Ruft eine Liste von Traces in einer Liste von Trace-IDs ab. Jeder abgerufene Trace ist eine Sammlung von Segmentdokumenten aus einer einzigen Anfrage.
- GetTraceSummaries— Ruft IDs und Anmerkungen für Traces ab. Sie können a angeben`FilterExpression`, um eine Teilmenge der Trace-Zusammenfassungen abzurufen.
- GetTraceGraph— Ruft ein Service-Diagramm für eine bestimmte Trace-ID ab.
- GetServiceGraph— Ruft ein JSON formatiertes Dokument ab, das Dienste beschreibt, die eingehende Anfragen verarbeiten und nachgelagerte Anfragen aufrufen.

Sie können auch das AWS Command Line Interface (AWS CLI) in Ihrem Anwendungscode verwenden, um programmgesteuert mit X-Ray zu interagieren. Das AWS CLI unterstützt alle Funktionen, die im X-Ray SDK verfügbar sind, einschließlich der Funktionen für andere AWS-Services. Die folgenden Funktionen sind Versionen der zuvor aufgeführten API-Operationen mit einem einfacheren Format:

- put-trace-segments— Lädt Segmentdokumente auf X-Ray hoch.
- batch-get-traces— Ruft eine Liste von Traces in einer Liste von Trace-IDs ab. Jeder abgerufene Trace ist eine Sammlung von Segmentdokumenten aus einer einzigen Anfrage.
- get-trace-summaries— Ruft IDs und Anmerkungen für Traces ab. Sie können a angeben`FilterExpression`, um eine Teilmenge der Trace-Zusammenfassungen abzurufen.
- get-trace-graph— Ruft ein Service-Diagramm für eine bestimmte Trace-ID ab.
- get-service-graph— Ruft ein `JSON` formatiertes Dokument ab, das Dienste beschreibt, die eingehende Anfragen verarbeiten und nachgelagerte Anfragen aufrufen.

Um zu beginnen, müssen Sie das AWS CLIfür Ihr Betriebssystem installieren. AWS LinuxUnterstützungen macOS und Windows Betriebssysteme. Weitere Informationen zur Liste der X-Ray-Befehle finden Sie in der AWS CLI Befehlsreferenz für X-Ray.

## Erkunden Sie die X-Ray-API

Die X-Ray-API bietet Zugriff auf alle X-Ray-Funktionen über das AWS SDK oder direkt über HTTPS. AWS Command Line Interface Die X-Ray-API-Referenz dokumentiert Eingabeparameter für jede API-Aktion sowie die Felder und Datentypen, die sie zurückgeben.

Sie können das AWS SDK verwenden, um Programme zu entwickeln, die die X-Ray-API verwenden. Die X-Ray-Konsole und der X-Ray-Daemon verwenden beide das AWS SDK, um mit X-Ray zu kommunizieren. Das AWS SDK für jede Sprache enthält ein Referenzdokument für Klassen und Methoden, die X-Ray-API-Aktionen und -Typen zugeordnet sind.

AWS SDK-Referenzen

- Java — AWS SDK for Java
- JavaScript – AWS SDK for JavaScript
- .NET – AWS SDK for .NET
- Ruby – AWS SDK for Ruby
- Go — AWS SDK for Go
- PHP – AWS SDK for PHP
- Python – AWS SDK for Python (Boto)

Das AWS Command Line Interface ist ein Befehlszeilentool, das das SDK für Python verwendet, um AWS APIs aufzurufen. Wenn Sie sich zum ersten Mal mit einer AWS API vertraut AWS CLI machen,

können Sie auf einfache Weise die verfügbaren Parameter erkunden und die Serviceausgabe in JSON- oder Textform anzeigen.

Einzelheiten zu `aws xray` Unterbefehlen finden Sie in der AWS CLI Befehlsreferenz.

Verwenden der X-Ray-API mit der AWS CLI

Über die AWS CLI können Sie direkt auf den X-Ray-Dienst zugreifen und dieselben APIs verwenden, die die X-Ray-Konsole zum Abrufen des Service-Graphen und der Trace-Rohdaten verwendet. Die Beispielanwendung enthält Skripts, die zeigen, wie diese APIs mit der AWS CLI verwendet werden.

Voraussetzungen

In diesem Tutorial werden die Scorekeep-Beispielanwendung und darin enthaltene Skripts verwendet, um Ablaufverfolgungsdaten und eine Service-Übersicht zu erstellen. Folgen Sie den Anweisungen im Tutorial zur Beispielanwendung, um die Anwendung zu starten.

In diesem Tutorial wird AWS CLI die grundlegende Verwendung der X-Ray-API veranschaulicht. Die AWS CLI, verfügbar für Windows, Linux und OS-X, bietet Befehlszeilenzugriff auf die öffentlichen APIs für alle AWS-Services.

> **Note**
>
> Sie müssen sicherstellen, dass Ihre für dieselbe Region konfiguriert AWS CLI ist, in der Ihre Beispielanwendung erstellt wurde.

Die enthaltenen Skripts zum Testen der Beispielanwendung verwenden `cURL`, um Datenverkehr zur API zu senden, und `jq` zum Analysieren der Ausgabe. Sie können die ausführbare `jq`-Datei von stedolan.github.io und die ausführbare `curl`-Datei von https://curl.haxx.se/download.html herunterladen. Die meisten Linux- und OS X-Installationen umfassen cURL.

Generieren von Ablaufverfolgungsdaten

Die Web-App generiert weiterhin alle paar Sekunden Datenverkehr zur API, während das Spiel fortgesetzt wird, es wird aber nur eine Art der Anforderung generiert. Verwenden Sie das Skript `test-api.sh`, um End-to-End-Szenarien auszuführen und um während der Prüfung der API vielfältigere Ablaufverfolgungsdaten zu generieren.

So verwenden Sie das **`test-api.sh`**-Skript

1. In der Elastic-Beanstalk-Konsole öffnen.
2. Navigieren Sie zur Managementkonsole für Ihre Umgebung.
3. Kopieren Sie die Umgebungs-URL aus dem Header der Seite.
4. Öffnen Sie `bin/test-api.sh` und ersetzen Sie den Wert für die API mit der URL Ihrer Umgebung.

```
#!/bin/bash
API=scorekeep.9hbtbm23t2.us-west-2.elasticbeanstalk.com/api
```

5. Führen Sie das Skript aus, um Datenverkehr zur API zu generieren.

```
~/debugger-tutorial$ ./bin/test-api.sh
Creating users,
session,
game,
configuring game,
playing game,
ending game,
game complete.
{"id":"MTBP8BAS","session":"HUF6IT64","name":"tic-tac-toe-test","users":
["QFF3HBGM","KL6JR98D"],"rules":"102","startTime":1476314241,"endTime":1476314245,"states":
["JQVLEOM2","D67QLPIC","VF9BM9NC","OEAA6GK9","2A705O73","1U2LFTLJ","HUKIDD70","BAN1C8FI","G
["BS8F8LQ","4MTTSPKP","463OETES","SVEBCL3N","N7CQ1GHP","O84ONEPD","EG4BPROQ","V4BLIDJ3","9R
```

Verwenden Sie die X-Ray-API

Die AWS CLI stellt Befehle für alle API-Aktionen bereit, die X-Ray bereitstellt, einschließlich `GetServiceGraph`und `GetTraceSummaries`. Weitere Informationen über alle unterstützten Aktionen und die von ihnen verwendeten Datentypen finden Sie in der AWS X-Ray -API-Referenz.

Example bin/service-graph.sh

```
EPOCH=$(date +%s)
aws xray get-service-graph --start-time $(($EPOCH-600)) --end-time $EPOCH
```

Das Skript ruft ein Service-Diagramm für die letzten 10 Minuten ab.

```
~/eb-java-scorekeep$ ./bin/service-graph.sh | less
```

```
{
    "StartTime": 1479068648.0,
    "Services": [
        {
            "StartTime": 1479068648.0,
            "ReferenceId": 0,
            "State": "unknown",
            "EndTime": 1479068651.0,
            "Type": "client",
            "Edges": [
                {
                    "StartTime": 1479068648.0,
                    "ReferenceId": 1,
                    "SummaryStatistics": {
                        "ErrorStatistics": {
                            "ThrottleCount": 0,
                            "TotalCount": 0,
                            "OtherCount": 0
                        },
                        "FaultStatistics": {
                            "TotalCount": 0,
                            "OtherCount": 0
                        },
                        "TotalCount": 2,
                        "OkCount": 2,
                        "TotalResponseTime": 0.054000139236450195
                    },
                    "EndTime": 1479068651.0,
                    "Aliases": []
                }
            ]
        },
        {
            "StartTime": 1479068648.0,
            "Names": [
                "scorekeep.elasticbeanstalk.com"
            ],
            "ReferenceId": 1,
            "State": "active",
            "EndTime": 1479068651.0,
            "Root": true,
            "Name": "scorekeep.elasticbeanstalk.com",
...
```

Example bin/trace-urls.sh

```
EPOCH=$(date +%s)
aws xray get-trace-summaries --start-time $(($EPOCH-120)) --end-time $(($EPOCH-60)) --
query 'TraceSummaries[*].Http.HttpURL'
```

Das Skript ruft die URLs der Ablaufverfolgungen ab, die ein bis zwei Minuten zuvor generiert wurden.

```
~/eb-java-scorekeep$ ./bin/trace-urls.sh
[
    "http://scorekeep.elasticbeanstalk.com/api/game/6Q0UE1DG/5FGLM9U3/
endtime/1479069438",
    "http://scorekeep.elasticbeanstalk.com/api/session/KH4341QH",
    "http://scorekeep.elasticbeanstalk.com/api/game/GLQBJ3K5/153AHDIA",
    "http://scorekeep.elasticbeanstalk.com/api/game/VPDL672J/G2V41HM6/
endtime/1479069466"
]
```

Example bin/full-traces.sh

```
EPOCH=$(date +%s)
TRACEIDS=$(aws xray get-trace-summaries --start-time $(($EPOCH-120)) --end-time
 $(($EPOCH-60)) --query 'TraceSummaries[*].Id' --output text)
aws xray batch-get-traces --trace-ids $TRACEIDS --query 'Traces[*]'
```

Das Skript ruft die vollständigen Ablaufverfolgungsdaten ab, die ein bis zwei Minuten zuvor generiert wurden.

```
~/eb-java-scorekeep$ ./bin/full-traces.sh | less
[
    {
        "Segments": [
            {
                "Id": "3f212bc237bafd5d",
                "Document": "{\"id\":\"3f212bc237bafd5d\",\"name\":\"DynamoDB\",
\"trace_id\":\"1-5828d9f2-a90669393f4343211bc1cf75\",\"start_time\":1.479072242459E9,
\"end_time\":1.479072242477E9,\"parent_id\":\"72a08dcf87991ca9\",\"http\":
{\"response\":{\"content_length\":60,\"status\":200}},\"inferred\":true,\"aws\":
{\"consistent_read\":false,\"table_name\":\"scorekeep-session-xray\",\"operation\":
\"GetItem\",\"request_id\":\"QAKE0S8DD0LJM245KAOPMA746BVV4KQNSO5AEMVJF66Q9ASUAAJG\",
\"resource_names\":[\"scorekeep-session-xray\"]},\"origin\":\"AWS::DynamoDB::Table\"}"
```

```
            },
            {
                "Id": "309e355f1148347f",
                "Document": "{\"id\":\"309e355f1148347f\",\"name\":\"DynamoDB\",
\"trace_id\":\"1-5828d9f2-a90669393f4343211bc1cf75\",\"start_time\":1.479072242477E9,
\"end_time\":1.479072242494E9,\"parent_id\":\"37f14ef837f00022\",\"http\":
{\"response\":{\"content_length\":606,\"status\":200}},\"inferred\":true,\"aws\":
{\"table_name\":\"scorekeep-game-xray\",\"operation\":\"UpdateItem\",\"request_id
\":\"388GEROC4PCA6D59ED3CTI5EEJVV4KQNSO5AEMVJF66Q9ASUAAJG\",\"resource_names\":
[\"scorekeep-game-xray\"]},\"origin\":\"AWS::DynamoDB::Table\"}"
            }
        ],
        "Id": "1-5828d9f2-a90669393f4343211bc1cf75",
        "Duration": 0.05099987983703613
    }
...
```


Bereinigen

Beenden Sie Ihre Elastic Beanstalk Beanstalk-Umgebung, um die Amazon EC2 EC2-Instances, DynamoDB-Tabellen und andere Ressourcen herunterzufahren.

So beenden Sie die Elastic Beanstalk-Umgebung

1. In der Elastic-Beanstalk-Konsole öffnen.
2. Navigieren Sie zur Management-Konsole für Ihre Umgebung.
3. Wählen Sie Aktionen.
4. Wählen Sie Terminate Environment.
5. Wähen Sie Beenden.

Trace-Daten werden nach 30 Tagen automatisch aus X-Ray gelöscht.

Trace-Daten an X-Ray senden

Sie können Trace-Daten in Form von Segmentdokumenten an X-Ray senden. Ein Segmentdokument ist eine JSON-formatierte Zeichenfolge mit Informationen über die Arbeit, die Ihre Anwendung im Dienste einer Anforderung verrichtet. Ihre Anwendung kann Daten über die Arbeit, die sie selbst in Segmenten verrichtet, oder über Arbeit, die Downstream-Services und -Ressourcen in Untersegmenten verrichtet, aufzeichnen.

Segmente zeichnen Informationen über die Arbeit auf, die Ihre Anwendung verrichtet. Ein Segment zeichnet mindestens die für eine Aufgabe aufgewendete zeit, ihren Namen sowie zwei IDs auf. Die Nachverfolgungs-ID zeichnet die Anforderung auf Ihrem Weg zwischen den Services auf. Die Segment-ID verfolgt die für die Anforderung durch einen einzelnen Service verrichtete Arbeit.

Example Minimales vollständiges Segment

```
{
  "name" : "Scorekeep",
  "id" : "70de5b6f19ff9a0a",
  "start_time" : 1.478293361271E9,
  "trace_id" : "1-581cf771-a006649127e371903a2de979",
  "end_time" : 1.478293361449E9
}
```

Wenn eine Anfrage empfangen wird, können Sie eine In-Progress-Segment als Platzhalter senden, bis die Anforderung abgeschlossen ist.

Example In Bearbeitung befindliches Segment

```
{
  "name" : "Scorekeep",
  "id" : "70de5b6f19ff9a0b",
  "start_time" : 1.478293361271E9,
  "trace_id" : "1-581cf771-a006649127e371903a2de979",
  “in_progress”: true
}
```

Sie können Segmente direkt, mit oder über den `PutTraceSegments`X-Ray-Daemon an X-Ray senden.

Die meisten Anwendungen rufen mit dem AWS SDK andere Dienste auf oder greifen auf Ressourcen zu. Zeichnen Sie Informationen über Downstream-Aufrufe in Untersegmenten auf. X-Ray verwendet Untersegmente, um Downstream-Services zu identifizieren, die keine Segmente senden, und Einträge für sie im Service-Graph zu erstellen.

Ein Untersegment kann in einem vollständigen Segmentdokument eingebettet oder separat gesendet werden. Senden Sie Untersegmente separat, um Downstream-Aufrufe für lang andauernde Anfragen asynchron zu verfolgen oder um zu verhindern, dass die maximale Segmentdokumentgröße (64 kB) überschritten wird.

Example Untersegment

Ein Untersegment hat den `type` `subsegment` und eine `parent_id`, mit der das übergeordnete Segment identifiziert wird.

```
{
  "name" : "www2.example.com",
  "id" : "70de5b6f19ff9a0c",
  "start_time" : 1.478293361271E9,
  "trace_id" : "1-581cf771-a006649127e371903a2de979"
  “end_time” : 1.478293361449E9,
  “type” : “subsegment”,
  “parent_id” : “70de5b6f19ff9a0b”
}
```

Weitere Informationen über die Felder und Werte, die in Segmente und Untersegmente eingegeben werden können, finden Sie unter Dokumente für Röntgensegmente.

Erzeugen von Ablaufverfolgungs-IDs

Um Daten an X-Ray zu senden, müssen Sie für jede Anfrage eine eindeutige Trace-ID generieren.

Röntgen-Trace-ID-Format

Ein X-Ray `trace_id` besteht aus drei Zahlen, die durch Bindestriche getrennt sind. z. B. `1-58406520-a006649127e371903a2de979`. Dies umfasst:

- Die Versionsnummer, die ist. 1
- Die Uhrzeit der ursprünglichen Anfrage in Unix-Epochenzeit unter Verwendung von 8 Hexadezimalziffern.

  Zum Beispiel ist 10:00 Uhr am 1. Dezember 2016 PST in Epochenzeit `1480615200` Sekunden oder `58406520` in Hexadezimalziffern.
- Eine weltweit eindeutige 96-Bit-ID für den Trace mit 24 Hexadezimalziffern.

> **Note**
>
> X-Ray unterstützt jetzt Trace-IDs, die mit OpenTelemetry und jedem anderen Framework erstellt wurden, das der W3C Trace Context-Spezifikation entspricht. Eine W3C-

Trace-ID muss beim Senden an X-Ray im X-Ray-Trace-ID-Format formatiert werden. Beispielsweise `4efaaf4d1e8720b39541901950019ee5` sollte die W3C-Trace-ID wie `1-4efaaf4d-1e8720b39541901950019ee5` beim Senden an X-Ray formatiert werden. X-Ray-Trace-IDs enthalten den ursprünglichen Anforderungszeitstempel in der Unix-Epochenzeit, dies ist jedoch nicht erforderlich, wenn W3C-Trace-IDs im X-Ray-Format gesendet werden.

Sie können ein Skript schreiben, um X-Ray-Trace-IDs für Tests zu generieren. Nachfolgend finden Sie zwei Beispiele.

Python

```
import time
import os
import binascii

START_TIME = time.time()
HEX=hex(int(START_TIME))[2:]
TRACE_ID="1-{}-{}".format(HEX, binascii.hexlify(os.urandom(12)).decode('utf-8'))
```

Bash

```
START_TIME=$(date +%s)
HEX_TIME=$(printf '%x\n' $START_TIME)
GUID=$(dd if=/dev/random bs=12 count=1 2>/dev/null | od -An -tx1 | tr -d ' \t\n')
TRACE_ID="1-$HEX_TIME-$GUID"
```

In der Beispielanwendung Scorekeep finden Sie Skripte, die Trace-IDs erstellen und Segmente an den X-Ray-Daemon senden.

- Python – `xray_start.py`
- Bash — `xray_start.sh`

### Benutzen PutTraceSegments

Sie können Segmentdokumente mit der `PutTraceSegments`-API hochladen. Die API hat einen einzelnen Parameter, `TraceSegmentDocuments`, der eine Liste von JSON-Segmentdokumenten aufnimmt.

Verwenden Sie mit der AWS-CLI den `aws xray put-trace-segments` Befehl, um Segmentdokumente direkt an X-Ray zu senden.

```
$ DOC='{"trace_id": "1-5960082b-ab52431b496add878434aa25", "id": "6226467e3f845502",
 "start_time": 1498082657.37518, "end_time": 1498082695.4042, "name":
 "test.elasticbeanstalk.com"}'
$ aws xray put-trace-segments --trace-segment-documents "$DOC"
{
    "UnprocessedTraceSegments": []
}
```

> **Note**
>
> Windows Command Processor und Windows PowerShell haben unterschiedliche Anforderungen für das Anführungszeichen und das Maskieren von Anführungszeichen in JSON-Zeichenketten. Weitere Details finden Sie unter Setzen von Anführungszeichen für Zeichenfolgen im AWS CLI -Benutzerhandbuch.

Die Ausgabe führt alle Segmente auf, deren Verarbeitung fehlgeschlagen ist. Beispiel: Wenn das Datum in der Nachverfolgungs-ID zu weit in der Vergangenheit liegt, sehen Sie eine Fehlermeldung wie die folgende.

```
{
    "UnprocessedTraceSegments": [
        {
            "ErrorCode": "InvalidTraceId",
            "Message": "Invalid segment. ErrorCode: InvalidTraceId",
            "Id": "6226467e3f845502"
        }
    ]
}
```

Sie können mehrere Segmentdokumente gleichzeitig, durch Leerräume getrennt, weitergeben.

```
$ aws xray put-trace-segments --trace-segment-documents "$DOC1" "$DOC2"
```

### Segmentdokumente an den X-Ray-Daemon senden

Anstatt Segmentdokumente an die X-Ray-API zu senden, können Sie Segmente und Untersegmente an den X-Ray-Daemon senden, der sie zwischenspeichert und stapelweise auf die X-Ray-API hochlädt. Das X-Ray-SDK sendet Segmentdokumente an den Daemon, um AWS direkte Aufrufe zu vermeiden.

> **Note**
>
> Informationen zur Ausführung des Daemons finden Sie unter Lokales Ausführen des X-Ray-Daemons.

Senden Sie das Segment in JSON über den UDP-Port 2000 mit dem vorangestellten Daemon-Kopf `{"format": "json", "version": 1}\n`.

```
{"format": "json", "version": 1}\n{"trace_id": "1-5759e988-bd862e3fe1be46a994272793",
 "id": "defdfd9912dc5a56", "start_time": 1461096053.37518, "end_time": 1461096053.4042,
 "name": "test.elasticbeanstalk.com"}
```

Unter Linux können Sie Segmentdokumente von einem Bash-Terminal aus an den Daemon senden. Speichern Sie den Kopf und das Segmentdokument in einer Textdatei, und senden Sie sie an `/dev/udp` mit `cat`.

```
$ cat segment.txt > /dev/udp/127.0.0.1/2000
```

Example segment.txt

```
{"format": "json", "version": 1}
{"trace_id": "1-594aed87-ad72e26896b3f9d3a27054bb", "id": "6226467e3f845502",
 "start_time": 1498082657.37518, "end_time": 1498082695.4042, "name":
 "test.elasticbeanstalk.com"}
```

Überprüfen Sie das Daemon-Protokoll, um sicherzustellen, dass das Segment an X-Ray gesendet wurde.

```
2017-07-07T01:57:24Z [Debug] processor: sending partial batch
2017-07-07T01:57:24Z [Debug] processor: segment batch size: 1. capacity: 50
2017-07-07T01:57:24Z [Info] Successfully sent batch of 1 segments (0.020 seconds)
```

### Daten von X-Ray abrufen

X-Ray verarbeitet die Trace-Daten, die Sie an X-Ray senden, um vollständige Traces, Trace-Zusammenfassungen und Service-Diagramme in JSON zu generieren. Sie können die generierten Daten mit der AWS CLI direkt von der API abrufen.

#### Abrufen des Service-Diagramms

Sie können die API `GetServiceGraph` verwenden, um das JSON-Service-Diagramm abzurufen. Die API erfordert eine Start- und Endzeit. Sie können diese mit dem `date`-Befehl an einem Linux-Terminal berechnen.

```
$ date +%s
1499394617
```

`date +%s` druckt ein Datum in Sekunden. Verwenden Sie diese Zahl als Endzeit. Ziehen Sie Zeit ab, um eine Startzeit zu erhalten.

Example Script, um ein Service-Diagramm für die letzten 10 Minuten abzurufen

```
EPOCH=$(date +%s)
aws xray get-service-graph --start-time $(($EPOCH-600)) --end-time $EPOCH
```

Das folgende Beispiel zeigt ein Service-Diagramm mit 4 Knoten, darunter einen Client-Knoten, eine EC2-Instance, eine DynamoDB-Tabelle und ein Amazon SNS SNS-Thema.

Example GetServiceGraph Ausgabe

```
{
    "Services": [
        {
            "ReferenceId": 0,
            "Name": "xray-sample.elasticbeanstalk.com",
            "Names": [
                "xray-sample.elasticbeanstalk.com"
            ],
            "Type": "client",
            "State": "unknown",
            "StartTime": 1528317567.0,
            "EndTime": 1528317589.0,
            "Edges": [
                {
```

```
                    "ReferenceId": 2,
                    "StartTime": 1528317567.0,
                    "EndTime": 1528317589.0,
                    "SummaryStatistics": {
                        "OkCount": 3,
                        "ErrorStatistics": {
                            "ThrottleCount": 0,
                            "OtherCount": 1,
                            "TotalCount": 1
                        },
                        "FaultStatistics": {
                            "OtherCount": 0,
                            "TotalCount": 0
                        },
                        "TotalCount": 4,
                        "TotalResponseTime": 0.273
                    },
                    "ResponseTimeHistogram": [
                        {
                            "Value": 0.005,
                            "Count": 1
                        },
                        {
                            "Value": 0.015,
                            "Count": 1
                        },
                        {
                            "Value": 0.157,
                            "Count": 1
                        },
                        {
                            "Value": 0.096,
                            "Count": 1
                        }
                    ],
                    "Aliases": []
                }
            ]
        },
        {
            "ReferenceId": 1,
            "Name": "awseb-e-dixzws4s9p-stack-StartupSignupsTable-4IMSMHAYX2BA",
            "Names": [
                "awseb-e-dixzws4s9p-stack-StartupSignupsTable-4IMSMHAYX2BA"
```

```
            ],
            "Type": "AWS::DynamoDB::Table",
            "State": "unknown",
            "StartTime": 1528317583.0,
            "EndTime": 1528317589.0,
            "Edges": [],
            "SummaryStatistics": {
                "OkCount": 2,
                "ErrorStatistics": {
                    "ThrottleCount": 0,
                    "OtherCount": 0,
                    "TotalCount": 0
                },
                "FaultStatistics": {
                    "OtherCount": 0,
                    "TotalCount": 0
                },
                "TotalCount": 2,
                "TotalResponseTime": 0.12
            },
            "DurationHistogram": [
                {
                    "Value": 0.076,
                    "Count": 1
                },
                {
                    "Value": 0.044,
                    "Count": 1
                }
            ],
            "ResponseTimeHistogram": [
                {
                    "Value": 0.076,
                    "Count": 1
                },
                {
                    "Value": 0.044,
                    "Count": 1
                }
            ]
        },
        {
            "ReferenceId": 2,
            "Name": "xray-sample.elasticbeanstalk.com",
```

```
            "Names": [
                "xray-sample.elasticbeanstalk.com"
            ],
            "Root": true,
            "Type": "AWS::EC2::Instance",
            "State": "active",
            "StartTime": 1528317567.0,
            "EndTime": 1528317589.0,
            "Edges": [
                {
                    "ReferenceId": 1,
                    "StartTime": 1528317567.0,
                    "EndTime": 1528317589.0,
                    "SummaryStatistics": {
                        "OkCount": 2,
                        "ErrorStatistics": {
                            "ThrottleCount": 0,
                            "OtherCount": 0,
                            "TotalCount": 0
                        },
                        "FaultStatistics": {
                            "OtherCount": 0,
                            "TotalCount": 0
                        },
                        "TotalCount": 2,
                        "TotalResponseTime": 0.12
                    },
                    "ResponseTimeHistogram": [
                        {
                            "Value": 0.076,
                            "Count": 1
                        },
                        {
                            "Value": 0.044,
                            "Count": 1
                        }
                    ],
                    "Aliases": []
                },
                {
                    "ReferenceId": 3,
                    "StartTime": 1528317567.0,
                    "EndTime": 1528317589.0,
                    "SummaryStatistics": {
```

```
                    "OkCount": 2,
                    "ErrorStatistics": {
                        "ThrottleCount": 0,
                        "OtherCount": 0,
                        "TotalCount": 0
                    },
                    "FaultStatistics": {
                        "OtherCount": 0,
                        "TotalCount": 0
                    },
                    "TotalCount": 2,
                    "TotalResponseTime": 0.125
                },
                "ResponseTimeHistogram": [
                    {
                        "Value": 0.049,
                        "Count": 1
                    },
                    {
                        "Value": 0.076,
                        "Count": 1
                    }
                ],
                "Aliases": []
            }
        ],
        "SummaryStatistics": {
            "OkCount": 3,
            "ErrorStatistics": {
                "ThrottleCount": 0,
                "OtherCount": 1,
                "TotalCount": 1
            },
            "FaultStatistics": {
                "OtherCount": 0,
                "TotalCount": 0
            },
            "TotalCount": 4,
            "TotalResponseTime": 0.273
        },
        "DurationHistogram": [
            {
                "Value": 0.005,
                "Count": 1
```

```
                },
                {
                    "Value": 0.015,
                    "Count": 1
                },
                {
                    "Value": 0.157,
                    "Count": 1
                },
                {
                    "Value": 0.096,
                    "Count": 1
                }
            ],
            "ResponseTimeHistogram": [
                {
                    "Value": 0.005,
                    "Count": 1
                },
                {
                    "Value": 0.015,
                    "Count": 1
                },
                {
                    "Value": 0.157,
                    "Count": 1
                },
                {
                    "Value": 0.096,
                    "Count": 1
                }
            ]
        },
        {
            "ReferenceId": 3,
            "Name": "SNS",
            "Names": [
                "SNS"
            ],
            "Type": "AWS::SNS",
            "State": "unknown",
            "StartTime": 1528317583.0,
            "EndTime": 1528317589.0,
            "Edges": [],
```

```
            "SummaryStatistics": {
                "OkCount": 2,
                "ErrorStatistics": {
                    "ThrottleCount": 0,
                    "OtherCount": 0,
                    "TotalCount": 0
                },
                "FaultStatistics": {
                    "OtherCount": 0,
                    "TotalCount": 0
                },
                "TotalCount": 2,
                "TotalResponseTime": 0.125
            },
            "DurationHistogram": [
                {
                    "Value": 0.049,
                    "Count": 1
                },
                {
                    "Value": 0.076,
                    "Count": 1
                }
            ],
            "ResponseTimeHistogram": [
                {
                    "Value": 0.049,
                    "Count": 1
                },
                {
                    "Value": 0.076,
                    "Count": 1
                }
            ]
        }
    ]
}
```

Abrufen des Service-Diagramms nach Gruppe

Zum Abrufen eines Service-Diagramms basierend auf dem Inhalt einer Gruppe geben Sie einen `groupName` oder `groupARN` an. Das folgende Beispiel zeigt einen Service-Diagrammaufruf einer Gruppe mit dem Namen Example1.

Example Skript zum Abrufen eines Service-Diagramms nach Name für die Gruppe Example1

```
aws xray get-service-graph --group-name "Example1"
```

Abrufen von Ablaufverfolgungen

Sie können die API `GetTraceSummaries` verwenden, um eine Liste von Ablaufverfolgungsübersichten abzurufen. Ablaufverfolgungsübersichten enthalten Informationen, mit denen Sie Ablaufverfolgungen identifizieren können, die Sie vollständig herunterladen möchten, einschließlich Anmerkungen, Informationen zu Anforderung und Reaktion sowie IDs.

Beim Aufruf von `aws xray get-trace-summaries` stehen zwei `TimeRangeType`- Flags zur Verfügung:

- TraceId— Die `GetTraceSummaries` Standardsuche verwendet die TraceID-Zeit und gibt Traces zurück, die innerhalb des berechneten `[start_time, end_time)` Bereichs gestartet wurden. Dieser Bereich von Zeitstempeln wird auf der Grundlage der Kodierung des Zeitstempels innerhalb von berechnet oder kann TraceId manuell definiert werden.
- Ereigniszeit — Um nach Ereignissen zu suchen, die sich im Laufe der Zeit ereignen, ermöglicht AWS X-Ray die Suche nach Spuren mithilfe von Ereigniszeitstempeln. Die Ereigniszeit gibt Traces zurück, die während des `[start_time, end_time)`-Bereichs aktiv sind, unabhängig davon, wann der Trace begonnen hat.

Mit dem Befehl `aws xray get-trace-summaries` können Sie eine Liste von Ablaufverfolgungsübersichten abrufen. Die folgenden Befehle rufen unter Verwendung der TraceId Standardzeit eine Liste von Trace-Zusammenfassungen ab, die zwischen 1 und 2 Minuten zurückliegen.

Example Skript zum Abrufen von Ablaufverfolgungsübersichten

```
EPOCH=$(date +%s)
aws xray get-trace-summaries --start-time $(($EPOCH-120)) --end-time $(($EPOCH-60))
```

Example GetTraceSummaries Ausgabe

```
{
    "TraceSummaries": [
        {
            "HasError": false,
```

```
            "Http": {
                "HttpStatus": 200,
                "ClientIp": "205.255.255.183",
                "HttpURL": "http://scorekeep.elasticbeanstalk.com/api/session",
                "UserAgent": "Mozilla/5.0 (Windows NT 6.1; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/59.0.3071.115 Safari/537.36",
                "HttpMethod": "POST"
            },
            "Users": [],
            "HasFault": false,
            "Annotations": {},
            "ResponseTime": 0.084,
            "Duration": 0.084,
            "Id": "1-59602606-a43a1ac52fc7ee0eea12a82c",
            "HasThrottle": false
        },
        {
            "HasError": false,
            "Http": {
                "HttpStatus": 200,
                "ClientIp": "205.255.255.183",
                "HttpURL": "http://scorekeep.elasticbeanstalk.com/api/user",
                "UserAgent": "Mozilla/5.0 (Windows NT 6.1; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/59.0.3071.115 Safari/537.36",
                "HttpMethod": "POST"
            },
            "Users": [
                {
                    "UserName": "5M388M1E"
                }
            ],
            "HasFault": false,
            "Annotations": {
                "UserID": [
                    {
                        "AnnotationValue": {
                            "StringValue": "5M388M1E"
                        }
                    }
                ],
                "Name": [
                    {
                        "AnnotationValue": {
                            "StringValue": "Ola"
```

```
                        }
                    }
                ]
            },
            "ResponseTime": 3.232,
            "Duration": 3.232,
            "Id": "1-59602603-23fc5b688855d396af79b496",
            "HasThrottle": false
        }
    ],
    "ApproximateTime": 1499473304.0,
    "TracesProcessedCount": 2
}
```

Mithilfe der Ablaufverfolgungs-ID aus der Ausgabe können Sie mit der API `BatchGetTraces` eine vollständige Ablaufverfolgung abrufen.

Example BatchGetTraces Befehl

```
$ aws xray batch-get-traces --trace-ids 1-596025b4-7170afe49f7aa708b1dd4a6b
```

Example BatchGetTraces Ausgabe

```
{
    "Traces": [
        {
            "Duration": 3.232,
            "Segments": [
                {
                    "Document": "{\"id\":\"1fb07842d944e714\",\"name\":
\"random-name\",\"start_time\":1.499473411677E9,\"end_time\":1.499473414572E9,
\"parent_id\":\"0c544c1b1bbff948\",\"http\":{\"response\":{\"status\":200}},
\"aws\":{\"request_id\":\"ac086670-6373-11e7-a174-f31b3397f190\"},\"trace_id\":
\"1-59602603-23fc5b688855d396af79b496\",\"origin\":\"AWS::Lambda\",\"resource_arn\":
\"arn:aws:lambda:us-west-2:123456789012:function:random-name\"}",
                    "Id": "1fb07842d944e714"
                },
                {
                    "Document": "{\"id\":\"194fcc8747581230\",\"name\":\"Scorekeep
\",\"start_time\":1.499473411562E9,\"end_time\":1.499473414794E9,\"http\":{\"request
\":{\"url\":\"http://scorekeep.elasticbeanstalk.com/api/user\",\"method\":\"POST\",
\"user_agent\":\"Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/59.0.3071.115 Safari/537.36\",\"client_ip\":\"205.251.233.183\"},
```

\"response\":{\"status\":200}},\"aws\":{\"elastic_beanstalk\":{\"version_label\":\"app-abb9-170708_002045\",\"deployment_id\":406,\"environment_name\":\"scorekeep-dev\"},\"ec2\":{\"availability_zone\":\"us-west-2c\",\"instance_id\":\"i-0cd9e448944061b4a\"},\"xray\":{\"sdk_version\":\"1.1.2\",\"sdk\":\"X-Ray for Java\"}},\"service\":{},\"trace_id\":\"1-59602603-23fc5b688855d396af79b496\",\"user\":\"5M388M1E\",\"origin\":\"AWS::ElasticBeanstalk::Environment\",\"subsegments\":[{\"id\":\"0c544c1b1bbff948\",\"name\":\"Lambda\",\"start_time\":1.499473411629E9,\"end_time\":1.499473414572E9,\"http\":{\"response\":{\"status\":200,\"content_length\":14}},\"aws\":{\"log_type\":\"None\",\"status_code\":200,\"function_name\":\"random-name\",\"invocation_type\":\"RequestResponse\",\"operation\":\"Invoke\",\"request_id\":\"ac086670-6373-11e7-a174-f31b3397f190\",\"resource_names\":[\"random-name\"]},\"namespace\":\"aws\"},{\"id\":\"071684f2e555e571\",\"name\":\"## UserModel.saveUser\",\"start_time\":1.499473414581E9,\"end_time\":1.499473414769E9,\"metadata\":{\"debug\":{\"test\":\"Metadata string from UserModel.saveUser\"}},\"subsegments\":[{\"id\":\"4cd3f10b76c624b4\",\"name\":\"DynamoDB\",\"start_time\":1.49947341469E9,\"end_time\":1.499473414769E9,\"http\":{\"response\":{\"status\":200,\"content_length\":57}},\"aws\":{\"table_name\":\"scorekeep-user\",\"operation\":\"UpdateItem\",\"request_id\":\"MFQ8CGJ3JTDDVVVASUAAJGQ6NJ82F738BOB4KQNSO5AEMVJF66Q9\",\"resource_names\":[\"scorekeep-user\"]},\"namespace\":\"aws\"}]}]}",
                    "Id": "194fcc8747581230"
                },
                {
                    "Document": "{\"id\":\"00f91aa01f4984fd\",\"name\":\"random-name\",\"start_time\":1.49947341283E9,\"end_time\":1.49947341457E9,\"parent_id\":\"1fb07842d944e714\",\"aws\":{\"function_arn\":\"arn:aws:lambda:us-west-2:123456789012:function:random-name\",\"resource_names\":[\"random-name\"],\"account_id\":\"123456789012\"},\"trace_id\":\"1-59602603-23fc5b688855d396af79b496\",\"origin\":\"AWS::Lambda::Function\",\"subsegments\":[{\"id\":\"e6d2fe619f827804\",\"name\":\"annotations\",\"start_time\":1.499473413012E9,\"end_time\":1.499473413069E9,\"annotations\":{\"UserID\":\"5M388M1E\",\"Name\":\"Ola\"}},{\"id\":\"b29b548af4d54a0f\",\"name\":\"SNS\",\"start_time\":1.499473413112E9,\"end_time\":1.499473414071E9,\"http\":{\"response\":{\"status\":200}},\"aws\":{\"operation\":\"Publish\",\"region\":\"us-west-2\",\"request_id\":\"a2137970-f6fc-5029-83e8-28aadeb99198\",\"retries\":0,\"topic_arn\":\"arn:aws:sns:us-west-2:123456789012:awseb-e-ruag3jyweb-stack-NotificationTopic-6B829NT9V5O9\"},\"namespace\":\"aws\"},{\"id\":\"2279c0030c955e52\",\"name\":\"Initialization\",\"start_time\":1.499473412064E9,\"end_time\":1.499473412819E9,\"aws\":{\"function_arn\":\"arn:aws:lambda:us-west-2:123456789012:function:random-name\"}}]}",
                    "Id": "00f91aa01f4984fd"
                },
                {
                    "Document": "{\"id\":\"17ba309b32c7fbaf\",\"name\":\"DynamoDB\",\"start_time\":1.49947341469E9,\"end_time\":1.499473414769E9,\"parent_id\":\"4cd3f10b76c624b4\",\"inferred\":true,\"http\":{\"response

```
\":{\"status\":200,\"content_length\":57}},\"aws\":{\"table_name
\":\"scorekeep-user\",\"operation\":\"UpdateItem\",\"request_id\":
\"MFQ8CGJ3JTDDVVVASUAAJGQ6NJ82F738BOB4KQNSO5AEMVJF66Q9\",\"resource_names\":
[\"scorekeep-user\"]},\"trace_id\":\"1-59602603-23fc5b688855d396af79b496\",\"origin\":
\"AWS::DynamoDB::Table\"}",
                    "Id": "17ba309b32c7fbaf"
                },
                {
                    "Document": "{\"id\":\"1ee3c4a523f89ca5\",\"name\":\"SNS
\",\"start_time\":1.499473413112E9,\"end_time\":1.499473414071E9,\"parent_id\":
\"b29b548af4d54a0f\",\"inferred\":true,\"http\":{\"response\":{\"status\":200}},\"aws
\":{\"operation\":\"Publish\",\"region\":\"us-west-2\",\"request_id\":\"a2137970-
f6fc-5029-83e8-28aadeb99198\",\"retries\":0,\"topic_arn\":\"arn:aws:sns:us-
west-2:123456789012:awseb-e-ruag3jyweb-stack-NotificationTopic-6B829NT9V5O9\"},
\"trace_id\":\"1-59602603-23fc5b688855d396af79b496\",\"origin\":\"AWS::SNS\"}",
                    "Id": "1ee3c4a523f89ca5"
                }
            ],
            "Id": "1-59602603-23fc5b688855d396af79b496"
        }
    ],
    "UnprocessedTraceIds": []
}
```


Die vollständige Ablaufverfolgung umfasst ein Dokument für jedes Segment, das aus allen Segmentdokumenten kompiliert wurde, die mit der gleichen Ablaufverfolgungs-ID erhalten wurden. Diese Dokumente stellen nicht die Daten dar, wie sie durch Ihre Anwendung an X-Ray gesendet wurden. Stattdessen stellen sie die verarbeiteten Dokumente dar, die vom X-Ray-Dienst generiert wurden. X-Ray erstellt das vollständige Trace-Dokument, indem es die von Ihrer Anwendung gesendeten Segmentdokumente kompiliert und Daten entfernt, die nicht dem Segmentdokumentschema entsprechen. Weitere Informationen finden Sie unter Dokumente für Röntgensegmente.

X-Ray erstellt auch abgeleitete Segmente für Downstream-Aufrufe an Dienste, die selbst keine Segmente senden. Wenn Sie DynamoDB beispielsweise mit einem instrumentierten Client aufrufen, zeichnet das X-Ray-SDK ein Untersegment mit Details zum Aufruf aus seiner Sicht auf. DynamoDB sendet jedoch kein entsprechendes Segment. X-Ray verwendet die Informationen im Untersegment, um ein abgeleitetes Segment zu erstellen, das die DynamoDB-Ressource in der Trace-Map darstellt, und fügt es dem Trace-Dokument hinzu.

Um mehrere Traces von der API abzurufen, benötigen Sie eine Liste von Trace-IDs, die Sie mit einer Abfrage aus der `get-trace-summaries` Ausgabe extrahieren können.AWS CLI Leiten Sie die Liste zur Eingabe von `batch-get-traces` um, um vollständige Traces für einen bestimmten Zeitraum zu erhalten.

Example Script, um vollständige Ablaufverfolgungen für den Zeitraum von einer Minute zu erhalten

```
EPOCH=$(date +%s)
TRACEIDS=$(aws xray get-trace-summaries --start-time $(($EPOCH-120)) --end-time
 $(($EPOCH-60)) --query 'TraceSummaries[*].Id' --output text)
aws xray batch-get-traces --trace-ids $TRACEIDS --query 'Traces[*]'
```

Abrufen und Anpassen der Ursachenanalyse

Nach der Generierung einer Trace-Zusammenfassung mit der GetTraceSummaries API können unvollständige Trace-Zusammenfassungen in ihrem JSON-Format wiederverwendet werden, um einen verfeinerten Filterausdruck zu erstellen, der auf den Grundursachen basiert. In den nachstehenden Beispielen finden Sie eine exemplarische Vorgehensweise für die Optimierungsschritte.

Example GetTraceSummaries Beispielausgabe — Abschnitt zur Hauptursache bei der Antwortzeit

```
{
  "Services": [
    {
      "Name": "GetWeatherData",
      "Names": ["GetWeatherData"],
      "AccountId": 123456789012,
      "Type": null,
      "Inferred": false,
      "EntityPath": [
        {
          "Name": "GetWeatherData",
          "Coverage": 1.0,
          'Remote": false
        },
        {
          "Name": "get_temperature",
          "Coverage": 0.8,
          "Remote": false
        }
      ]
```

```
    },
    {
      "Name": "GetTemperature",
      "Names": ["GetTemperature"],
      "AccountId": 123456789012,
      "Type": null,
      "Inferred": false,
      "EntityPath": [
        {
          "Name": "GetTemperature",
          "Coverage": 0.7,
          "Remote": false
        }
      ]
    }
  ]
}
```

Durch das Bearbeiten und Vornehmen von Auslassungen der oben genannten Ausgaben kann dieses JSON-Objekt ein Filter für übereinstimmende Ursachenentitäten werden. Für jedes Feld im JSON-Objekt müssen alle Bewerberabgleiche exakt erfolgen. Andernfalls wird die Nachverfolgung nicht zurückgegeben. Entfernte Felder werden zu Platzhalterwerten, ein Format, das mit der Abfragestruktur des Filterausdrucks kompatibel ist.

Example Reaktionszeit „Ursache“ neu formatiert

```
{
  "Services": [
    {
      "Name": "GetWeatherData",
      "EntityPath": [
        {
          "Name": "GetWeatherData"
        },
        {
          "Name": "get_temperature"
        }
      ]
    },
    {
      "Name": "GetTemperature",
      "EntityPath": [
        {
```

```
          "Name": "GetTemperature"
        }
      ]
    }
  ]
}
```

Dieses JSON-Objekt wird dann als Teil eines Filterausdrucks durch einen Aufruf von `rootcause.json = #[{}]` verwendet. Weitere Informationen zur Abfrage mit Filterausdrücken finden Sie im Abschnitt Verwenden von Filterausdrücken unter Erkunden der X-Ray-Konsole.

Example Beispiel für JSON-Filter

```
rootcause.json = #[{ "Services": [ { "Name": "GetWeatherData", "EntityPath": [{ "Name":
 "GetWeatherData" }, { "Name": "get_temperature" } ] }, { "Name": "GetTemperature",
 "EntityPath": [ { "Name": "GetTemperature" } ] } ] }]
```

Konfiguration von Sampling-, Gruppen- und Verschlüsselungseinstellungen mit der X-Ray-API

X-Ray bietet APIs für die Konfiguration von Sampling-Regeln, Gruppenregeln und Verschlüsselungseinstellungen.

Verschlüsselungseinstellungen

Dient `PutEncryptionConfig`zur Angabe eines AWS Key Management Service (AWS KMS) -Schlüssels, der für die Verschlüsselung verwendet werden soll.

> **Note**
>
> X-Ray unterstützt keine asymmetrischen KMS-Schlüssel.

```
$ aws xray put-encryption-config --type KMS --key-id alias/aws/xray
{
    "EncryptionConfig": {
        "KeyId": "arn:aws:kms:us-east-2:123456789012:key/c234g4e8-39e9-4gb0-84e2-
b0ea215cbba5",
        "Status": "UPDATING",
        "Type": "KMS"
    }
}
```

Für die Schlüssel-ID können Sie einen Alias verwenden (wie im Beispiel gezeigt), eine Schlüssel-ID oder einen Amazon-Ressourcennamen (ARN).

Verwenden Sie `GetEncryptionConfig`, um die aktuelle Konfiguration abzurufen. Wenn X-Ray die Übernahme Ihrer Einstellungen abgeschlossen hat, ändert sich der Status von `UPDATING` zu`ACTIVE`.

```
$ aws xray get-encryption-config
{
    "EncryptionConfig": {
        "KeyId": "arn:aws:kms:us-east-2:123456789012:key/c234g4e8-39e9-4gb0-84e2-
b0ea215cbba5",
        "Status": "ACTIVE",
        "Type": "KMS"
    }
}
```

Um die Verwendung eines KMS-Schlüssels zu beenden und die Standardverschlüsselung zu verwenden, setzen Sie den Verschlüsselungstyp auf`NONE`.

```
$ aws xray put-encryption-config --type NONE
{
    "EncryptionConfig": {
        "Status": "UPDATING",
        "Type": "NONE"
    }
}
```

Samplingregeln

Sie können die Sampling-Regeln in Ihrem Konto mit der X-Ray-API verwalten. Weitere Informationen zur Probenahme finden Sie unterKonfigurieren Sie Stichprobenregeln. Weitere Informationen zum Hinzufügen und Verwalten von Tags finden Sie unterKennzeichnen von Regeln und Gruppen für die Röntgenprobenahme.

Sie können alle Samplingregeln mit `GetSamplingRules` abrufen.

```
$ aws xray get-sampling-rules
{
    "SamplingRuleRecords": [
        {
            "SamplingRule": {
```

```
                "RuleName": "Default",
                "RuleARN": "arn:aws:xray:us-east-2:123456789012:sampling-rule/Default",
                "ResourceARN": "*",
                "Priority": 10000,
                "FixedRate": 0.05,
                "ReservoirSize": 1,
                "ServiceName": "*",
                "ServiceType": "*",
                "Host": "*",
                "HTTPMethod": "*",
                "URLPath": "*",
                "Version": 1,
                "Attributes": {}
            },
            "CreatedAt": 0.0,
            "ModifiedAt": 1529959993.0
        }
    ]
}
```

Die Standardregel gilt für alle Anfragen, die nicht mit einer anderen Regel übereinstimmen. Es ist die Regel mit der niedrigsten Priorität, die nicht gelöscht werden kann. Sie können jedoch die Rate und die Reservoirgröße mit `UpdateSamplingRule` ändern.

Example API-Eingabe für **`UpdateSamplingRule`**— 10000-default.json

```
{
    "SamplingRuleUpdate": {
        "RuleName": "Default",
        "FixedRate": 0.01,
        "ReservoirSize": 0
    }
}
```

Das folgende Beispiel verwendet die vorherige Datei als Eingabe, um die Standardregel in ein Prozent ohne Reservoir zu ändern. Tags sind optional. Wenn Sie Tags hinzufügen möchten, ist ein Tag-Schlüssel erforderlich, und Tag-Werte sind optional. Um vorhandene Tags aus einer Stichprobenregel zu entfernen, verwenden Sie UntagResource.

```
$ aws xray update-sampling-rule --cli-input-json file://1000-default.json --tags
 [{"Key": "key_name","Value": "value"},{"Key": "key_name","Value": "value"}]
{
```

```
    "SamplingRuleRecords": [
        {
            "SamplingRule": {
                "RuleName": "Default",
                "RuleARN": "arn:aws:xray:us-east-2:123456789012:sampling-rule/Default",
                "ResourceARN": "*",
                "Priority": 10000,
                "FixedRate": 0.01,
                "ReservoirSize": 0,
                "ServiceName": "*",
                "ServiceType": "*",
                "Host": "*",
                "HTTPMethod": "*",
                "URLPath": "*",
                "Version": 1,
                "Attributes": {}
            },
            "CreatedAt": 0.0,
            "ModifiedAt": 1529959993.0
        },
```

Erstellen Sie zusätzliche Samplingregeln mit `CreateSamplingRule`. Wenn Sie eine Regel erstellen, sind die meisten Regelfelder erforderlich. Das folgende Beispiel erstellt zwei Regeln. Diese erste Regel legt einen Basissatz für die Scorekeep-Beispielanwendung fest. Er entspricht allen Anforderungen, die von der API bereitgestellt werden, die nicht mit einer höheren Priorität übereinstimmen.

Example API-Eingabe für **`UpdateSamplingRule`**— 9000-base-scorekeep.json

```
{
    "SamplingRule": {
        "RuleName": "base-scorekeep",
        "ResourceARN": "*",
        "Priority": 9000,
        "FixedRate": 0.1,
        "ReservoirSize": 5,
        "ServiceName": "Scorekeep",
        "ServiceType": "*",
        "Host": "*",
        "HTTPMethod": "*",
        "URLPath": "*",
        "Version": 1
    }
```

```
}
```

Die zweite Regel gilt auch für Scorekeep, hat aber eine höhere Priorität und ist spezifischer. Diese Regel enthält eine sehr niedrige Samplingrate für Polling-Anfragen. Dies sind GET-Anfragen, die der Client alle paar Sekunden sendet, um eine Prüfung auf Änderungen am Spielstatus durchzuführen.

Example API-Eingabe für **UpdateSamplingRule**— 5000-polling-scorekeep.json

```
{
    "SamplingRule": {
        "RuleName": "polling-scorekeep",
        "ResourceARN": "*",
        "Priority": 5000,
        "FixedRate": 0.003,
        "ReservoirSize": 0,
        "ServiceName": "Scorekeep",
        "ServiceType": "*",
        "Host": "*",
        "HTTPMethod": "GET",
        "URLPath": "/api/state/*",
        "Version": 1
    }
}
```

Tags sind optional. Wenn Sie Tags hinzufügen möchten, ist ein Tag-Schlüssel erforderlich, und Tag-Werte sind optional.

```
$ aws xray create-sampling-rule --cli-input-json file://5000-polling-scorekeep.json --
tags [{"Key": "key_name","Value": "value"},{"Key": "key_name","Value": "value"}]
{
    "SamplingRuleRecord": {
        "SamplingRule": {
            "RuleName": "polling-scorekeep",
            "RuleARN": "arn:aws:xray:us-east-1:123456789012:sampling-rule/polling-
scorekeep",
            "ResourceARN": "*",
            "Priority": 5000,
            "FixedRate": 0.003,
            "ReservoirSize": 0,
            "ServiceName": "Scorekeep",
            "ServiceType": "*",
            "Host": "*",
```

```
            "HTTPMethod": "GET",
            "URLPath": "/api/state/*",
            "Version": 1,
            "Attributes": {}
        },
        "CreatedAt": 1530574399.0,
        "ModifiedAt": 1530574399.0
    }
}
$ aws xray create-sampling-rule --cli-input-json file://9000-base-scorekeep.json
{
    "SamplingRuleRecord": {
        "SamplingRule": {
            "RuleName": "base-scorekeep",
            "RuleARN": "arn:aws:xray:us-east-1:123456789012:sampling-rule/base-
scorekeep",
            "ResourceARN": "*",
            "Priority": 9000,
            "FixedRate": 0.1,
            "ReservoirSize": 5,
            "ServiceName": "Scorekeep",
            "ServiceType": "*",
            "Host": "*",
            "HTTPMethod": "*",
            "URLPath": "*",
            "Version": 1,
            "Attributes": {}
        },
        "CreatedAt": 1530574410.0,
        "ModifiedAt": 1530574410.0
    }
}
```

Um eine Samplingregel zu löschen, verwenden Sie [DeleteSamplingRule](#).

```
$ aws xray delete-sampling-rule --rule-name polling-scorekeep
{
    "SamplingRuleRecord": {
        "SamplingRule": {
            "RuleName": "polling-scorekeep",
            "RuleARN": "arn:aws:xray:us-east-1:123456789012:sampling-rule/polling-
scorekeep",
            "ResourceARN": "*",
```

```
            "Priority": 5000,
            "FixedRate": 0.003,
            "ReservoirSize": 0,
            "ServiceName": "Scorekeep",
            "ServiceType": "*",
            "Host": "*",
            "HTTPMethod": "GET",
            "URLPath": "/api/state/*",
            "Version": 1,
            "Attributes": {}
        },
        "CreatedAt": 1530574399.0,
        "ModifiedAt": 1530574399.0
    }
}
```

Gruppen

Sie können die X-Ray-API verwenden, um Gruppen in Ihrem Konto zu verwalten. Gruppen sind eine Sammlung von Ablaufverfolgungen, die durch einen Filterausdruck definiert sind. Sie können Gruppen verwenden, um zusätzliche Servicegraphen zu erstellen und CloudWatch Amazon-Metriken bereitzustellen. Weitere Informationen Daten von X-Ray abrufen zur Arbeit mit Servicediagrammen und Metriken über die X-Ray-API finden Sie unter. Weitere Informationen zu Gruppen finden Sie unterGruppen konfigurieren. Weitere Informationen zum Hinzufügen und Verwalten von Stichwörtern finden Sie unterKennzeichnen von Regeln und Gruppen für die Röntgenprobenahme.

Erstellen Sie eine Gruppe mit `CreateGroup`. Tags sind optional. Wenn Sie Tags hinzufügen möchten, ist ein Tag-Schlüssel erforderlich, und Tag-Werte sind optional.

```
$ aws xray create-group --group-name "TestGroup" --filter-expression
 "service(\"example.com\") {fault}" --tags [{"Key": "key_name","Value": "value"},
{"Key": "key_name","Value": "value"}]
{
    "GroupName": "TestGroup",
    "GroupARN": "arn:aws:xray:us-east-2:123456789012:group/TestGroup/UniqueID",
    "FilterExpression": "service(\"example.com\") {fault OR error}"
}
```

Rufen Sie alle vorhandenen Gruppen mit `GetGroups` ab.

```
$ aws xray get-groups
{
```

```
    "Groups": [
        {
            "GroupName": "TestGroup",
            "GroupARN": "arn:aws:xray:us-east-2:123456789012:group/TestGroup/UniqueID",
            "FilterExpression": "service(\"example.com\") {fault OR error}"
        },
  {
            "GroupName": "TestGroup2",
            "GroupARN": "arn:aws:xray:us-east-2:123456789012:group/TestGroup2/
UniqueID",
            "FilterExpression": "responsetime > 2"
        }
    ],
 "NextToken": "tokenstring"
}
```

Aktualisieren Sie eine Gruppe mit `UpdateGroup`. Tags sind optional. Wenn Sie Tags hinzufügen möchten, ist ein Tag-Schlüssel erforderlich, und Tag-Werte sind optional. Um vorhandene Tags aus einer Gruppe zu entfernen, verwenden Sie UntagResource.

```
$ aws xray update-group --group-name "TestGroup" --group-arn "arn:aws:xray:us-
east-2:123456789012:group/TestGroup/UniqueID" --filter-expression
 "service(\"example.com\") {fault OR error}" --tags [{"Key": "Stage","Value": "Prod"},
{"Key": "Department","Value": "QA"}]
{
    "GroupName": "TestGroup",
    "GroupARN": "arn:aws:xray:us-east-2:123456789012:group/TestGroup/UniqueID",
    "FilterExpression": "service(\"example.com\") {fault OR error}"
}
```

Löschen Sie eine Gruppe mit `DeleteGroup`.

```
$ aws xray delete-group --group-name "TestGroup" --group-arn "arn:aws:xray:us-
east-2:123456789012:group/TestGroup/UniqueID"
    {
    }
```

Verwenden von Sampling-Regeln mit der X-Ray-API

Das X-Ray SDK verwendet die X-Ray-API, um Probenahmeregeln abzurufen, Probenahmeergebnisse zu melden und Kontingente abzurufen. Sie können diese APIs verwenden,

um besser zu verstehen, wie Sampling-Regeln funktionieren, oder um Sampling in einer Sprache zu implementieren, die das X-Ray SDK nicht unterstützt.

Rufen Sie zunächst alle Samplingregeln mit `GetSamplingRules` ab.

```
$ aws xray get-sampling-rules
{
    "SamplingRuleRecords": [
        {
            "SamplingRule": {
                "RuleName": "Default",
                "RuleARN": "arn:aws:xray:us-east-1::sampling-rule/Default",
                "ResourceARN": "*",
                "Priority": 10000,
                "FixedRate": 0.01,
                "ReservoirSize": 0,
                "ServiceName": "*",
                "ServiceType": "*",
                "Host": "*",
                "HTTPMethod": "*",
                "URLPath": "*",
                "Version": 1,
                "Attributes": {}
            },
            "CreatedAt": 0.0,
            "ModifiedAt": 1530558121.0
        },
        {
            "SamplingRule": {
                "RuleName": "base-scorekeep",
                "RuleARN": "arn:aws:xray:us-east-1::sampling-rule/base-scorekeep",
                "ResourceARN": "*",
                "Priority": 9000,
                "FixedRate": 0.1,
                "ReservoirSize": 2,
                "ServiceName": "Scorekeep",
                "ServiceType": "*",
                "Host": "*",
                "HTTPMethod": "*",
                "URLPath": "*",
                "Version": 1,
                "Attributes": {}
            },
            "CreatedAt": 1530573954.0,
```

```
            "ModifiedAt": 1530920505.0
        },
        {
            "SamplingRule": {
                "RuleName": "polling-scorekeep",
                "RuleARN": "arn:aws:xray:us-east-1::sampling-rule/polling-scorekeep",
                "ResourceARN": "*",
                "Priority": 5000,
                "FixedRate": 0.003,
                "ReservoirSize": 0,
                "ServiceName": "Scorekeep",
                "ServiceType": "*",
                "Host": "*",
                "HTTPMethod": "GET",
                "URLPath": "/api/state/*",
                "Version": 1,
                "Attributes": {}
            },
            "CreatedAt": 1530918163.0,
            "ModifiedAt": 1530918163.0
        }
    ]
}
```

Die Ausgabe enthält die Standardregel und benutzerdefinierte Regeln. Weitere Informationen finden Sie unter Konfiguration von Sampling-, Gruppen- und Verschlüsselungseinstellungen mit der X-Ray-API, wenn Sie noch keine Samplingregeln erstellt haben.

Bewerten Sie Regeln anhand eingehender Anfragen in aufsteigender Reihenfolge der Priorität. Wenn eine Regel übereinstimmt, verwenden Sie die feste Rate und eine feste Reservoirgröße, um eine Samplingentscheidung zu treffen. Zeichnen Sie die per Stichprobe geprüften Anforderungen auf und ignorieren Sie (für die Nachverfolgung) nicht geprüfte Anfragen. Beenden Sie die Bewertung der Regeln, wenn eine Samplingentscheidung getroffen wird.

Eine Regel-Reservoirgröße ist die Zielanzahl der Ablaufverfolgungen pro Sekunde, bevor Sie die feste Rate anwenden. Das Reservoir gilt kumulativ für alle Services, sodass Sie es nicht direkt verwenden können. Wenn sie jedoch ungleich Null ist, können Sie eine Spur pro Sekunde aus dem Reservoir ausleihen, bis X-Ray eine Quote zuweist. Vor dem Empfang eines Kontingents zeichnen Sie die erste Anfrage pro Sekunde auf und wenden die feste Rate auf zusätzliche Anfragen an. Die feste Rate ist ein Dezimalwert zwischen 0 und 1,00 (100 %).

Das folgende Beispiel zeigt einen Aufruf von GetSamplingTargets mit Details zu Samplingentscheidungen in den letzten 10 Sekunden.

```
$ aws xray get-sampling-targets --sampling-statistics-documents '[
    {
        "RuleName": "base-scorekeep",
        "ClientID": "ABCDEF1234567890ABCDEF10",
        "Timestamp": "2018-07-07T00:20:06",
        "RequestCount": 110,
        "SampledCount": 20,
        "BorrowCount": 10
    },
    {
        "RuleName": "polling-scorekeep",
        "ClientID": "ABCDEF1234567890ABCDEF10",
        "Timestamp": "2018-07-07T00:20:06",
        "RequestCount": 10500,
        "SampledCount": 31,
        "BorrowCount": 0
    }
]'
{
    "SamplingTargetDocuments": [
        {
            "RuleName": "base-scorekeep",
            "FixedRate": 0.1,
            "ReservoirQuota": 2,
            "ReservoirQuotaTTL": 1530923107.0,
            "Interval": 10
        },
        {
            "RuleName": "polling-scorekeep",
            "FixedRate": 0.003,
            "ReservoirQuota": 0,
            "ReservoirQuotaTTL": 1530923107.0,
            "Interval": 10
        }
    ],
    "LastRuleModification": 1530920505.0,
    "UnprocessedStatistics": []
}
```

Die Antwort von X-Ray beinhaltet ein Kontingent, das verwendet werden kann, anstatt Kredite aus dem Reservoir aufzunehmen. In diesem Beispiel hat sich der Service aus dem Reservoir 10 Ablaufverfolgungen über 10 Sekunden geliehen und die feste Rate von 10 % auf den anderen 100 Anfragen angewendet. Daraus ergeben sich 20 geprüfte Anfragen. Das Kontingent ist für fünf Minuten gültig (angegeben durch die Gültigkeitsdauer) oder bis ein neues Kontingent zugewiesen wird. X-Ray weist möglicherweise auch ein längeres Berichtsintervall als das Standardintervall zu, obwohl dies hier nicht der Fall war.

> **Note**
>
> Die Antwort von X-Ray enthält möglicherweise kein Kontingent, wenn Sie sie zum ersten Mal aufrufen. Leihen Sie weiterhin vom Reservoir, bis Sie ein Kontingent zugewiesen bekommen.

Die anderen zwei Felder in der Antwort können möglicherweise Konflikte mit der Eingabe anzeigen. Vergleichen Sie `LastRuleModification` mit dem letzten Aufruf von `GetSamplingRules`. Wenn er neuer ist, rufen Sie eine Kopie der Regeln ab. `UnprocessedStatistics` kann Fehler enthalten, die angeben, dass eine Regel gelöscht wurde, dass das Statistikdokument in der Eingabe zu alt war oder Berechtigungsfehler aufgetreten sind.

Dokumente für Röntgensegmente

Ein Ablaufverfolgungssegment ist eine JSON-Darstellung einer Anfrage, die Ihrer Anwendung dient. Ein Trace-Segment zeichnet Informationen über die ursprüngliche Anfrage, Informationen über die Arbeit auf lokaler Ebene und Untersegmente mit Informationen über Downstream-Aufrufe auf, die Ihre Anwendung an AWS Ressourcen, HTTP-APIs und SQL-Datenbanken durchführt.

Ein Segmentdokument übermittelt Informationen über ein Segment an X-Ray. Ein Segmentdokument kann bis zu 64 kB groß sein und ein ganzes Segment mit Untersegmenten, ein Fragment eines Segments, das angibt, dass eine Anfrage bearbeitet wird, oder ein einzelnes Untersegment, das separat gesendet wird, enthalten. Mithilfe der `PutTraceSegments`API können Sie Segmentdokumente direkt an X-Ray senden.

X-Ray kompiliert und verarbeitet Segmentdokumente, um abfragbare Trace-Zusammenfassungen und vollständige Traces zu generieren, auf die Sie mithilfe der APIs bzw. der `GetTraceSummaries`APIs zugreifen können. `BatchGetTraces` Zusätzlich zu den Segmenten und Untersegmenten, die Sie an X-Ray senden, verwendet der Service Informationen in Untersegmenten, um abgeleitete Segmente zu generieren, und fügt sie dem vollständigen Trace hinzu. Abgeleitete Segmente stellen nachgelagerte Dienste und Ressourcen in der Trace-Map dar.

X-Ray bietet ein JSON-Schema für Segmentdokumente. Sie können das Schema hier herunterladen: xray-segmentdocument-schema-v1.0.0. Die im Schema angegebenen Felder und Objekte werden in den folgenden Abschnitten genauer beschrieben.

Eine Teilmenge von Segmentfeldern wird von X-Ray für die Verwendung mit Filterausdrücken indexiert. Wenn Sie beispielsweise das `user` Feld in einem Segment auf eine eindeutige Kennung festlegen, können Sie in der X-Ray-Konsole oder mithilfe der `GetTraceSummaries` API nach Segmenten suchen, die bestimmten Benutzern zugeordnet sind. Weitere Informationen finden Sie unter Verwenden Sie Filterausdrücke.

Wenn Sie Ihre Anwendung mit dem X-Ray SDK instrumentieren, generiert das SDK Segmentdokumente für Sie. Anstatt Segmentdokumente direkt an X-Ray zu senden, überträgt das SDK sie über einen lokalen UDP-Port an den X-Ray-Daemon. Weitere Informationen finden Sie unter Segmentdokumente an den X-Ray-Daemon senden.

Segmentfelder

Ein Segment zeichnet Ablaufverfolgungsinformationen über eine Anforderung auf, die Ihrer Anwendung dient. Ein Segment zeichnet mindestens den Namen, die ID, die Anfangszeit, die Ablaufverfolgungs-ID und die Endzeit der Anforderung auf.

Example Minimales vollständiges Segment

```
{
  "name" : "example.com",
  "id" : "70de5b6f19ff9a0a",
  "start_time" : 1.478293361271E9,
  "trace_id" : "1-581cf771-a006649127e371903a2de979",
  "end_time" : 1.478293361449E9
}
```

Die folgenden Felder sind für Segmente erforderlich oder in einigen Fällen notwendig.

> **Note**
>
> Der Wert muss aus Zeichenfolgen (bis zu 250 Zeichen) bestehen, soweit nicht anders angegeben.

Erforderliche Segmentfelder

- `name`— Der logische Name des Dienstes, der die Anfrage bearbeitet hat, bis zu 200 Zeichen. Beispielsweise der Name der Anwendung oder Domäne. Namen dürfen Unicode-Buchstaben, -Ziffern und -Leerzeichen enthalten sowie die folgenden Symbole: `_`, `.`, `:`, `/`, `%`, `&`, `#`, `=`, `+`, `\`, `-`, `@`
- `id`— Eine 64-Bit-ID für das Segment, die unter den Segmenten in derselben Spur eindeutig ist, mit 16 Hexadezimalziffern.
- `trace_id`— Eine eindeutige Kennung, die alle Segmente und Untersegmente verbindet, die aus einer einzigen Client-Anfrage stammen.

  Röntgen-Trace-ID-Format

  Ein X-Ray `trace_id` besteht aus drei Zahlen, die durch Bindestriche getrennt sind. z. B. `1-58406520-a006649127e371903a2de979`. Dies umfasst:

  - Die Versionsnummer, die ist. `1`
  - Die Uhrzeit der ursprünglichen Anfrage in Unix-Epochenzeit unter Verwendung von 8 Hexadezimalziffern.

    Zum Beispiel ist 10:00 Uhr am 1. Dezember 2016 PST in Epochenzeit `1480615200` Sekunden oder `58406520` in Hexadezimalziffern.
  - Eine weltweit eindeutige 96-Bit-ID für den Trace mit 24 Hexadezimalziffern.

  > **Note**
  >
  > X-Ray unterstützt jetzt Trace-IDs, die mit OpenTelemetry und jedem anderen Framework erstellt wurden, das der W3C Trace Context-Spezifikation entspricht. Eine W3C-Trace-ID muss beim Senden an X-Ray im X-Ray-Trace-ID-Format formatiert werden. Beispielsweise `4efaaf4d1e8720b39541901950019ee5` sollte die W3C-Trace-ID wie `1-4efaaf4d-1e8720b39541901950019ee5` beim Senden an X-Ray formatiert werden. X-Ray-Trace-IDs enthalten den ursprünglichen Anforderungszeitstempel in der Unix-Epochenzeit, dies ist jedoch nicht erforderlich, wenn W3C-Trace-IDs im X-Ray-Format gesendet werden.

> **Sicherheit der Ablaufverfolgungs-ID**
>
> Ablaufverfolgungs-IDs sind in Antwortheadern sichtbar. Generieren Sie Ablaufverfolgungs-IDs mit einem sicheren Random-Algorithmus, um sicherzustellen, dass Angreifer zukünftige Ablaufverfolgungs-IDs nicht berechnen und keine Anforderungen mit diesen IDs an Ihre Anwendung senden können.

- `start_time`— Zahl, die die Zeit angibt, zu der das Segment erstellt wurde, in Fließkommasekunden in Epochenzeit. Zum Beispiel `1480615200.010` oder `1.480615200010E9`. Verwenden Sie so viele Dezimalstellen, wie Sie benötigen. Mikrosekundenauflösung ist empfohlen, wenn verfügbar.
- `end_time`— Zahl, die die Zeit angibt, zu der das Segment geschlossen wurde. Zum Beispiel `1480615200.090` oder `1.480615200090E9`. Geben Sie entweder `end_time` oder `in_progress` an.
- `in_progress`— boolescher Wert, auf gesetzt, `true` anstatt anzugebenend_time, dass ein Segment gestartet, aber noch nicht abgeschlossen ist. Wenn Ihre Anwendung eine Anforderung empfängt, die lange dauern wird, senden Sie ein in Bearbeitung befindliches Segment, um den Empfang zu bestätigen. Wenn die Antwort gesendet wird, senden Sie das vollständige Segment zum Überschreiben des in Bearbeitung befindlichen Segments. Senden Sie pro Anforderung nur ein vollständiges Segment und ein oder kein angefangenes Segment.

> **Namen der Dienste**
>
> Der eines Segments `name` sollte mit dem Domainnamen oder dem logischen Namen des Dienstes übereinstimmen, der das Segment generiert. Dies wird jedoch nicht erzwungen. Jede Anwendung, die über die entsprechende Berechtigung verfügt, `PutTraceSegments`kann Segmente mit einem beliebigen Namen senden.

Die folgenden Felder sind für Segmente optional.

Optionale Segmentfelder

- `service`— Ein Objekt mit Informationen zu Ihrer Anwendung.

  - `version`— Eine Zeichenfolge, die die Version Ihrer Anwendung identifiziert, die die Anfrage bearbeitet hat.
- `user`— Eine Zeichenfolge, die den Benutzer identifiziert, der die Anfrage gesendet hat.
- `origin`— Der Typ der AWS Ressource, auf der Ihre Anwendung ausgeführt wird.

  Unterstützte Werte

  - `AWS::EC2::Instance`— Eine Amazon EC2 EC2-Instance.
  - `AWS::ECS::Container`— Ein Amazon ECS-Container.
  - `AWS::ElasticBeanstalk::Environment`— Eine Elastic Beanstalk Beanstalk-Umgebung.

  Wenn mehrere Werte für Ihre Anwendung gelten, verwenden Sie den spezifischsten. In einer Multicontainer Docker Elastic Beanstalk Beanstalk-Umgebung wird Ihre Anwendung beispielsweise auf einem Amazon ECS-Container ausgeführt, der wiederum auf einer Amazon EC2 EC2-Instance läuft. In diesem Fall müssen Sie den Ursprung auf `AWS::ElasticBeanstalk::Environment` festlegen, da die Umgebung das übergeordnete Element der beiden anderen Ressourcen ist.
- `parent_id`— Eine Subsegment-ID, die Sie angeben, wenn die Anfrage von einer instrumentierten Anwendung stammt. Das X-Ray-SDK fügt die ID des übergeordneten Untersegments zum Tracing-Header für Downstream-HTTP-Aufrufe hinzu. Bei verschachtelten Untersegmenten kann ein Untersegment ein Segment oder ein Untersegment als übergeordnetes Element haben.
- `http`— `http`Objekte mit Informationen über die ursprüngliche HTTP-Anfrage.
- `aws`— `aws`Objekt mit Informationen über die AWS Ressource, auf der Ihre Anwendung die Anfrage bearbeitet hat.
- `error`, `throttle`, `fault`, und `cause` — Fehlerfelder, die darauf hinweisen, dass ein Fehler aufgetreten ist, und die Informationen über die Ausnahme enthalten, die den Fehler verursacht hat.
- `annotations`— `annotations`Objekt mit Schlüssel-Wert-Paaren, die X-Ray für die Suche indexieren soll.
- `metadata`— `metadata`Objekt mit allen zusätzlichen Daten, die Sie in dem Segment speichern möchten.
- `subsegments`— Anordnung von `subsegment`Objekten.

### Untersegmente

Sie können Untersegmente erstellen, um Aufrufe AWS-Services und Ressourcen, die Sie mit dem AWS SDK tätigen, Aufrufe interner oder externer HTTP-Web-APIs oder SQL-Datenbankabfragen

aufzuzeichnen. Sie können auch Untersegmente erstellen, um Code-Blöcke in Ihrer Anwendung zu debuggen oder zu kommentieren. Untersegmente können weitere Untersegmente enthalten, damit ein individuelles Untersegment, das Metadaten über einen internen Funktionsaufruf aufzeichnet, weitere individuelle Untersegmente sowie Untersegmente nachgelagerter Aufrufe umfassen kann.

Ein Untersegment zeichnet einen Downstream-Aufruf aus der Sicht des Dienstes auf, der ihn aufruft. X-Ray verwendet Untersegmente, um Downstream-Services zu identifizieren, die keine Segmente senden, und Einträge für sie im Service-Graph zu erstellen.

Ein Untersegment kann in einem vollständigen Segmentdokument eingebettet oder separat gesendet werden. Senden Sie Untersegmente separat, um nachgelagerte Aufrufe für lange andauernde Anforderungen asynchron nachzuverfolgen oder um zu verhindern, dass die maximale Größe des Segmentdokuments überschritten wird.

Example Segment mit eingebettetem Untersegment

Ein unabhängiges Untersegment hat einen `type` vom `subsegment` und einen `parent_id`, der das übergeordnete Segment identifiziert.

```
{
  "trace_id"   : "1-5759e988-bd862e3fe1be46a994272793",
  "id"         : "defdfd9912dc5a56",
  "start_time" : 1461096053.37518,
  "end_time"   : 1461096053.4042,
  "name"       : "www.example.com",
  "http"       : {
    "request"  : {
      "url"        : "https://www.example.com/health",
      "method"     : "GET",
      "user_agent" : "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6)
 AppleWebKit/601.7.7",
      "client_ip"  : "11.0.3.111"
    },
    "response" : {
      "status"         : 200,
      "content_length" : 86
    }
  },
  "subsegments" : [
    {
      "id"         : "53995c3f42cd8ad8",
      "name"       : "api.example.com",
```

```
      "start_time" : 1461096053.37769,
      "end_time"   : 1461096053.40379,
      "namespace"  : "remote",
      "http"       : {
        "request"  : {
          "url"    : "https://api.example.com/health",
          "method" : "POST",
          "traced" : true
        },
        "response" : {
          "status"         : 200,
          "content_length" : 861
        }
      }
    }
  ]
}
```

Bei Anfragen mit langer Laufzeit können Sie ein Segment in Bearbeitung senden, um X-Ray darüber zu informieren, dass die Anfrage eingegangen ist, und dann Untersegmente separat senden, um sie zu verfolgen, bevor die ursprüngliche Anfrage abgeschlossen wird.

Example In Bearbeitung befindliches Segment

```
{
  "name" : "example.com",
  "id" : "70de5b6f19ff9a0b",
  "start_time" : 1.478293361271E9,
  "trace_id" : "1-581cf771-a006649127e371903a2de979",
  "in_progress": true
}
```

Example Unabhängiges Untersegment

Ein unabhängiges Untersegment hat einen `type-subsegment`, eine `trace_id` und eine `parent_id`, die das übergeordnete Segment identifiziert.

```
{
  "name" : "api.example.com",
  "id" : "53995c3f42cd8ad8",
  "start_time" : 1.478293361271E9,
  "end_time" : 1.478293361449E9,
```

```
  "type" : "subsegment",
  "trace_id" : "1-581cf771-a006649127e371903a2de979"
  "parent_id" : "defdfd9912dc5a56",
  "namespace"  : "remote",
  "http"       : {
      "request"  : {
          "url"    : "https://api.example.com/health",
          "method" : "POST",
          "traced" : true
      },
      "response" : {
          "status"         : 200,
          "content_length" : 861
      }
  }
}
```

Schließen Sie das Segment nach Abschluss der Anforderung mit einer `end_time`. Das vollständige Segment überschreibt das sich in Bearbeitung befindliche Segment.

Sie können außerdem Untersegmente für abgeschlossene Anforderungen, die asynchrone Workflows auslösen, getrennt senden. Beispielsweise kann eine Web-API eine `OK 200`-Antwort, unmittelbar bevor die vom Benutzer angeforderte Arbeit beginnt, zurücksenden. Sie können ein vollständiges Segment an X-Ray senden, sobald die Antwort gesendet wurde, gefolgt von Untersegmenten für später abgeschlossene Arbeiten. Wie bei Segmenten können Sie auch einen Untersegmentsfragment senden, um den Beginn des Untersegments aufzuzeichnen, und es anschließend mit einem vollständigen Segment überschreiben, sobald ein nachgelagerter Aufruf abgeschlossen ist.

Die folgenden Felder sind für Untersegmente erforderlich oder in einigen Fällen notwendig.

> **Note**
>
> Werte sind Zeichenfolgen, die aus bis zu 250 Zeichen bestehen, soweit nicht anders angegeben.

Erforderliche Untersegmentsfelder

- `id`— Eine 64-Bit-ID für das Untersegment, die unter den Segmenten in derselben Spur eindeutig ist und aus 16 Hexadezimalziffern besteht.

- `name`— Der logische Name des Untersegments. Benennen Sie bei nachgelagerten Aufrufen das Untersegment, nachdem die Ressource oder der Service aufgerufen wurde. Benennen Sie bei benutzerdefinierten Untersegmenten das Untersegment nach dem genutzten Code (z. B. einem Funktionsnamen).
- `start_time`— Zahl, die die Zeit angibt, zu der das Untersegment erstellt wurde, in Gleitkommasekunden in der Epochenzeit, auf Millisekunden genau. Zum Beispiel `1480615200.010` oder `1.480615200010E9`.
- `end_time`— Zahl, die die Zeit angibt, zu der das Untersegment geschlossen wurde. Zum Beispiel `1480615200.090` oder `1.480615200090E9`. Geben Sie eine `end_time` oder `in_progress` ein.
- `in_progress`— boolescher Wert, der auf gesetzt ist, `true` anstatt anzugeben, dass ein `end_time` Untersegment zwar gestartet, aber noch nicht abgeschlossen ist. Senden Sie pro nachgelagerter Anfrage nur ein vollständiges Untersegment und ein oder kein angefangenes Untersegment.
- `trace_id`— Trace-ID des übergeordneten Segments des Untersegments. Nur erforderlich, wenn ein Untersegment separat gesendet wird.

  Röntgen-Trace-ID-Format

  Ein X-Ray `trace_id` besteht aus drei Zahlen, die durch Bindestriche getrennt sind. z. B. `1-58406520-a006649127e371903a2de979`. Dies umfasst:

  - Die Versionsnummer, die ist. 1
  - Die Uhrzeit der ursprünglichen Anfrage in Unix-Epochenzeit unter Verwendung von 8 Hexadezimalziffern.

    Zum Beispiel ist 10:00 Uhr am 1. Dezember 2016 PST in Epochenzeit `1480615200` Sekunden oder `58406520` in Hexadezimalziffern.
  - Eine weltweit eindeutige 96-Bit-ID für den Trace mit 24 Hexadezimalziffern.

  > **Note**
  >
  > X-Ray unterstützt jetzt Trace-IDs, die mit OpenTelemetry und jedem anderen Framework erstellt wurden, das der W3C Trace Context-Spezifikation entspricht. Eine W3C-Trace-ID muss beim Senden an X-Ray im X-Ray-Trace-ID-Format formatiert werden. Beispielsweise `4efaaf4d1e8720b39541901950019ee5` sollte die W3C-Trace-ID wie `1-4efaaf4d-1e8720b39541901950019ee5` beim Senden an X-Ray formatiert werden.

X-Ray-Trace-IDs enthalten den ursprünglichen Anforderungszeitstempel in der Unix-Epochenzeit, dies ist jedoch nicht erforderlich, wenn W3C-Trace-IDs im X-Ray-Format gesendet werden.

- `parent_id`— Segment-ID des übergeordneten Segments des Untersegments. Nur erforderlich, wenn ein Untersegment separat gesendet wird. Bei verschachtelten Untersegmenten kann ein Untersegment ein Segment oder ein Untersegment als übergeordnetes Element haben.
- `type`—`subsegment`. Nur erforderlich, wenn ein Untersegment separat gesendet wird.

Die folgenden Felder sind für Untersegmente optional.

Optionale Untersegmentsfelder

- `namespace`— `aws` für AWS-SDK-Aufrufe; `remote` für andere Downstream-Aufrufe.
- `http`— `http`Objekt mit Informationen über einen ausgehenden HTTP-Aufruf.
- `aws`— `aws`Objekt mit Informationen über die AWS Downstream-Ressource, die Ihre Anwendung aufgerufen hat.
- `error`, `throttle`, `fault`, und `cause` — Fehlerfelder, die darauf hinweisen, dass ein Fehler aufgetreten ist, und die Informationen über die Ausnahme enthalten, die den Fehler verursacht hat.
- `annotations`— `annotations`Objekt mit Schlüssel-Wert-Paaren, die X-Ray für die Suche indexieren soll.
- `metadata`— `metadata`Objekt mit allen zusätzlichen Daten, die Sie in dem Segment speichern möchten.
- `subsegments`— Anordnung von `subsegment`Objekten.
- `precursor_ids`— Array von Untersegment-IDs, das Untersegmente identifiziert, denen dasselbe übergeordnete Objekt zugewiesen wurde, das vor diesem Untersegment abgeschlossen wurde.

HTTP-Anfragedaten

Verwenden Sie einen HTTP-Block, um Details zu einer HTTP-Anforderung aufzuzeichnen, die Ihrer Anwendung (in einem Segment) dient oder die Ihre Anwendung (in einem Untersegment) an eine nachgelagerte HTTP-API gestellt hat. Die meisten Felder in dieser Objektübersicht gehören zu Informationen von HTTP-Anforderungen und -Antworten.

**`http`**

Alle Felder sind optional.

- `request`— Informationen zu einer Anfrage.
  - `method`— Die Anforderungsmethode. z. B. `GET`.
  - `url`— Die vollständige URL der Anfrage, zusammengestellt aus dem Protokoll, dem Hostnamen und dem Pfad der Anfrage.
  - `user_agent`— Die User-Agent-Zeichenfolge vom Client des Anforderers.
  - `client_ip`— Die IP-Adresse des Anforderers. Kann im IP-Paket `Source Address` oder, für weitergeleitete Anforderungen, in einem `X-Forwarded-For`-Header eingesehen werden.
  - `x_forwarded_for`— (nur Segmente) boolescher Wert, der angibt, dass der aus einem `X-Forwarded-For` Header gelesen `client_ip` wurde und nicht zuverlässig ist, da er gefälscht worden sein könnte.
  - `traced`— (nur Untersegmente) boolescher Wert, der angibt, dass der Downstream-Aufruf an einen anderen verfolgten Dienst gerichtet ist. Wenn dieses Feld auf gesetzt ist`true`, betrachtet X-Ray den Trace als unterbrochen, bis der Downstream-Service ein Segment hochlädt`parent_id`, dessen a dem `id` des Untersegments entspricht, das diesen Block enthält.
- `response`— Informationen über eine Antwort.
  - `status`— Ganzzahl, die den HTTP-Status der Antwort angibt.
  - `content_length`— Ganzzahl, die die Länge des Antworttextes in Byte angibt.

Wenn Sie einen Aufruf einer Downstream-Web-API instrumentieren, zeichnen Sie ein Untersegment mit Informationen zur HTTP-Anfrage und -Antwort auf. X-Ray verwendet das Untersegment, um ein abgeleitetes Segment für die Remote-API zu generieren.

Example Segment für einen HTTP-Aufruf, der von einer Anwendung, die über Amazon EC2 ausgeführt wird, bereitgestellt wird.

```
{
  "id": "6b55dcc497934f1a",
  "start_time": 1484789387.126,
  "end_time": 1484789387.535,
  "trace_id": "1-5880168b-fd5158284b67678a3bb5a78c",
  "name": "www.example.com",
  "origin": "AWS::EC2::Instance",
  "aws": {
```

```
    "ec2": {
      "availability_zone": "us-west-2c",
      "instance_id": "i-0b5a4678fc325bg98"
    },
    "xray": {
        "sdk_version": "2.11.0 for Java"
    },
  },
  "http": {
    "request": {
      "method": "POST",
      "client_ip": "78.255.233.48",
      "url": "http://www.example.com/api/user",
      "user_agent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:45.0) Gecko/20100101
 Firefox/45.0",
      "x_forwarded_for": true
    },
    "response": {
      "status": 200
    }
  }
```

Example Untersegment für einen nachgelagerten HTTP-Aufruf

```
{
  "id": "004f72be19cddc2a",
  "start_time": 1484786387.131,
  "end_time": 1484786387.501,
  "name": "names.example.com",
  "namespace": "remote",
  "http": {
    "request": {
      "method": "GET",
      "url": "https://names.example.com/"
    },
    "response": {
      "content_length": -1,
      "status": 200
    }
  }
}
```

Example Abgeleitetes Segment für einen nachgelagerten HTTP-Anruf

```
{
  "id": "168416dc2ea97781",
  "name": "names.example.com",
  "trace_id": "1-62be1272-1b71c4274f39f122afa64eab",
  "start_time": 1484786387.131,
  "end_time": 1484786387.501,
  "parent_id": "004f72be19cddc2a",
  "http": {
    "request": {
      "method": "GET",
      "url": "https://names.example.com/"
    },
    "response": {
      "content_length": -1,
      "status": 200
    }
  },
  "inferred": true
}
```

Anmerkungen

Segmente und Untersegmente können ein `annotations` Objekt enthalten, das ein oder mehrere Felder enthält, die X-Ray für die Verwendung mit Filterausdrücken indexiert. Felder können eine Zeichenfolge, Zahl oder einen booleschen Werte (keine Objekte oder Arrays) umfassen. X-Ray indexiert bis zu 50 Anmerkungen pro Spur.

Example Segment für HTTP-Aufrufe mit Anmerkungen

```
{
  "id": "6b55dcc497932f1a",
  "start_time": 1484789187.126,
  "end_time": 1484789187.535,
  "trace_id": "1-5880168b-fd515828bs07678a3bb5a78c",
  "name": "www.example.com",
  "origin": "AWS::EC2::Instance",
  "aws": {
    "ec2": {
      "availability_zone": "us-west-2c",
      "instance_id": "i-0b5a4678fc325bg98"
    },
```

```
    "xray": {
        "sdk_version": "2.11.0 for Java"
    },
  },
  "annotations": {
    "customer_category" : 124,
    "zip_code" : 98101,
    "country" : "United States",
    "internal" : false
  },
  "http": {
    "request": {
      "method": "POST",
      "client_ip": "78.255.233.48",
      "url": "http://www.example.com/api/user",
      "user_agent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:45.0) Gecko/20100101
 Firefox/45.0",
      "x_forwarded_for": true
    },
    "response": {
      "status": 200
    }
  }
```

Die Schlüssel müssen alphanumerisch sein, um mit Filtern funktionieren zu können. Unterstriche sind zulässig. Andere Zeichen und Leerzeichen sind nicht zulässig.

Metadaten

Segmente und Untersegmente können ein `metadata` Objekt enthalten, das ein oder mehrere Felder mit Werten beliebigen Typs, einschließlich Objekten und Arrays, enthält. X-Ray indexiert keine Metadaten, und Werte können beliebig groß sein, solange das Segmentdokument die maximale Größe (64 kB) nicht überschreitet. Sie können Metadaten im vollständigen Segmentdokument, das von der `BatchGetTraces`-API zurückgesendet wurde, einsehen. Feldschlüssel (`debug`im folgenden Beispiel), die mit beginnen, `AWS.` sind für die Verwendung durch von AWS-bereitgestellte SDKs und Clients reserviert.

Example Individuelles Untersegment mit Metadaten

```
{
  "id": "0e58d2918e9038e8",
  "start_time": 1484789387.502,
```

```
  "end_time": 1484789387.534,
  "name": "## UserModel.saveUser",
  "metadata": {
    "debug": {
      "test": "Metadata string from UserModel.saveUser"
    }
  },
  "subsegments": [
    {
      "id": "0f910026178b71eb",
      "start_time": 1484789387.502,
      "end_time": 1484789387.534,
      "name": "DynamoDB",
      "namespace": "aws",
      "http": {
        "response": {
          "content_length": 58,
          "status": 200
        }
      },
      "aws": {
        "table_name": "scorekeep-user",
        "operation": "UpdateItem",
        "request_id": "3AIENM5J4ELQ3SPODHKBIRVIC3VV4KQNSO5AEMVJF66Q9ASUAAJG",
        "resource_names": [
          "scorekeep-user"
        ]
      }
    }
  ]
}
```

### AWS Ressourcendaten

Bei Segmenten enthält das `aws`-Objekt Informationen zu den Ressourcen, auf denen Ihre Anwendung ausgeführt wird. Mehrere Felder können für eine einzelne Ressource zutreffen. Beispielsweise könnte eine Anwendung, die in einer Docker-Umgebung mit mehreren Containern auf Elastic Beanstalk ausgeführt wird, Informationen über die Amazon EC2 EC2-Instance, den Amazon ECS-Container, der auf der Instance ausgeführt wird, und die Elastic Beanstalk Beanstalk-Umgebung selbst enthalten.

**`aws`** (Segmente)

Alle Felder sind optional.

- `account_id`— Wenn Ihre Anwendung Segmente an eine andere sendet AWS-Konto, notieren Sie sich die ID des Kontos, auf dem Ihre Anwendung ausgeführt wird.
- `cloudwatch_logs`— Array von Objekten, die eine einzelne CloudWatch Protokollgruppe beschreiben.
  - `log_group`— Der Name der CloudWatch Protokollgruppe.
  - `arn`— Die CloudWatch Protokollgruppe ARN.
- `ec2`— Informationen über eine Amazon EC2 EC2-Instance.
  - `instance_id`— Die Instance-ID der EC2-Instance.
  - `instance_size`— Der Typ der EC2-Instance.
  - `ami_id`— Die Amazon Machine Image-ID.
  - `availability_zone`— Die Availability Zone, in der die Instance ausgeführt wird.
- `ecs`— Informationen über einen Amazon ECS-Container.
  - `container`— Der Hostname Ihres Containers.
  - `container_id`— Die vollständige Container-ID Ihres Containers.
  - `container_arn`— Der ARN Ihrer Container-Instance.
- `eks`— Informationen über einen Amazon EKS-Cluster.
  - pod— Der Hostname Ihres EKS-Pods.
  - `cluster_name`— Der Name des EKS-Clusters.
  - `container_id`— Die vollständige Container-ID Ihres Containers.
- `elastic_beanstalk`— Informationen über eine Elastic Beanstalk Beanstalk-Umgebung. Sie finden diese Informationen in einer Datei mit dem Namen `/var/elasticbeanstalk/xray/environment.conf` auf den neuesten Elastic Beanstalk-Plattformen.
  - `environment_name` – Der Name der Umgebung.
  - `version_label`— Der Name der Anwendungsversion, die derzeit auf der Instance bereitgestellt wird, die die Anfrage bedient hat.
  - `deployment_id`— Zahl, die die ID der letzten erfolgreichen Bereitstellung auf der Instance angibt, die die Anfrage bedient hat.
- `xray`— Metadaten über den Typ und die Version der verwendeten Instrumentierung.

- `auto_instrumentation`— Boolescher Wert, der angibt, ob die automatische Instrumentierung verwendet wurde (z. B. der Java-Agent).
- `sdk_version`— Die Version des verwendeten SDK oder Agenten.
- `sdk`— Der Typ des SDK.

Example AWS Block mit Plugins

```
"aws":{
   "elastic_beanstalk":{
      "version_label":"app-5a56-170119_190650-stage-170119_190650",
      "deployment_id":32,
      "environment_name":"scorekeep"
   },
   "ec2":{
      "availability_zone":"us-west-2c",
      "instance_id":"i-075ad396f12bc325a",
      "ami_id":
   },
   "cloudwatch_logs":[
      {
         "log_group":"my-cw-log-group",
         "arn":"arn:aws:logs:us-west-2:012345678912:log-group:my-cw-log-group"
      }
   ],
   "xray":{
      "auto_instrumentation":false,
      "sdk":"X-Ray for Java",
      "sdk_version":"2.8.0"
   }
}
```

Erfassen Sie für Untersegmente Informationen über die Ressourcen AWS-Services und die Ressourcen, auf die Ihre Anwendung zugreift. X-Ray verwendet diese Informationen, um abgeleitete Segmente zu erstellen, die die Downstream-Services in Ihrer Service-Map darstellen.

**`aws`** (Untersegmente)

Alle Felder sind optional.

- `operation`— Der Name der API-Aktion, die für eine AWS-Service OR-Ressource aufgerufen wurde.

- `account_id`— Wenn Ihre Anwendung auf Ressourcen in einem anderen Konto zugreift oder Segmente an ein anderes Konto sendet, notieren Sie sich die ID des Kontos, dem die AWS Ressource gehört, auf die Ihre Anwendung zugegriffen hat.
- `region`— Wenn sich die Ressource in einer anderen Region als Ihre Anwendung befindet, notieren Sie die Region. z. B. `us-west-2`.
- `request_id`— Eindeutiger Bezeichner für die Anfrage.
- `queue_url`— Für Operationen in einer Amazon SQS SQS-Warteschlange die URL der Warteschlange.
- `table_name`— Für Operationen an einer DynamoDB-Tabelle der Name der Tabelle.

Example Untersegment für einen Aufruf von DynamoDB zum Speichern eines Elements

```
{
  "id": "24756640c0d0978a",
  "start_time": 1.480305974194E9,
  "end_time": 1.4803059742E9,
  "name": "DynamoDB",
  "namespace": "aws",
  "http": {
    "response": {
      "content_length": 60,
      "status": 200
    }
  },
  "aws": {
    "table_name": "scorekeep-user",
    "operation": "UpdateItem",
    "request_id": "UBQNSO5AEM8T4FDA4RQDEB94OVTDRVV4K4HIRGVJF66Q9ASUAAJG",
  }
}
```

Fehler und Ausnahmen

Wenn ein Fehler auftritt, können Sie die Einzelheiten zum Fehler und den Ausnahmen, die er generiert, aufzeichnen. Zeichnen Sie Fehler in Segmenten auf, wenn Ihre Anwendung einen Fehler an den Benutzer zurückgibt, sowie in Untersegmenten, wenn ein nachgelagerter Aufruf einen Fehler ausgibt.

Fehlertypen

Stellen Sie ein oder mehrere der folgenden Felder auf `true` ein, um anzuzeigen, dass ein Fehler aufgetreten ist. Bei schwerwiegenden Fehlern können mehrere Typen ausgewählt werden. Ein `429 Too Many Requests`-Fehler von einem nachgelagerten Aufruf kann beispielsweise dazu führen, dass Ihre Anwendung zu einem `500 Internal Server Error` zurückkehrt. In diesem Fall treffen alle drei Typen zu.

- `error`— Boolescher Wert, der angibt, dass ein Client-Fehler aufgetreten ist (der Antwortstatuscode lautete 4XX Client Error).
- `throttle`— boolescher Wert, der angibt, dass eine Anfrage gedrosselt wurde (der Antwortstatuscode lautete 429 Too Many Requests).
- `fault`— boolescher Wert, der angibt, dass ein Serverfehler aufgetreten ist (der Antwortstatuscode lautete 5XX Server Error).

Geben Sie die Fehlerursache an, indem Sie im Segment oder Untersegment ein Ursachenobjekt einschließen.

**`cause`**

Eine Ursache kann entweder eine 16-stellige Ausnahmen-ID oder ein Objekt mit den folgenden Feldern sein:

- `working_directory`— Der vollständige Pfad des Arbeitsverzeichnisses, als die Ausnahme auftrat.
- `paths`— Das Array von Pfaden zu Bibliotheken oder Modulen, die verwendet wurden, als die Ausnahme auftrat.
- `exceptions`— Das Array von Ausnahmeobjekten.

Geben Sie detaillierte Informationen über die Fehler in einem oder mehreren Ausnahmenobjekten an.

**`exception`**

Alle Felder sind optional.

- `id`— Eine 64-Bit-ID für die Ausnahme, die unter den Segmenten derselben Spur eindeutig ist und aus 16 Hexadezimalziffern besteht.
- `message`— Die Ausnahmemeldung.

- `type`— Der Ausnahmetyp.
- `remote`— Boolescher Wert, der angibt, dass die Ausnahme durch einen Fehler verursacht wurde, der von einem nachgelagerten Dienst zurückgegeben wurde.
- `truncated`— Ganzzahl, die die Anzahl der Stack-Frames angibt, die in der weggelassen wurden. `stack`
- `skipped`— Ganzzahl, die die Anzahl der Ausnahmen angibt, die zwischen dieser Ausnahme und ihrem untergeordneten Element, d. h. der von ihr verursachten Ausnahme, übersprungen wurden.
- `cause`— Ausnahme-ID des der Ausnahme übergeordneten Elements, d. h. der Ausnahme, die diese Ausnahme verursacht hat.
- `stack`— Array von StackFrame-Objekten.

Falls verfügbar, zeichnen Sie Informationen zu dem Aufruf-Stack in stackFrame-Objekten auf.

**`stackFrame`**

Alle Felder sind optional.

- `path`— Der relative Pfad zur Datei.
- `line`— Die Zeile in der Datei.
- `label`— Der Funktions- oder Methodenname.

SQL-Abfragen

Für Anfragen, die Ihre Anwendung in eine SQL-Datenbank umwandeln, können Sie Untersegmente erstellen.

**`sql`**

Alle Felder sind optional.

- `connection_string`— Für SQL Server- oder andere Datenbankverbindungen, die keine URL-Verbindungszeichenfolgen verwenden, notieren Sie sich die Verbindungszeichenfolge, ausgenommen Kennwörter.
- `url`— Notieren Sie für eine Datenbankverbindung, die eine URL-Verbindungszeichenfolge verwendet, die URL ohne Kennwörter.
- `sanitized_query`— Die Datenbankabfrage, bei der alle vom Benutzer angegebenen Werte entfernt oder durch einen Platzhalter ersetzt wurden.

- `database_type`— Der Name der Datenbank-Engine.
- `database_version`— Die Versionsnummer der Datenbank-Engine.
- `driver_version`— Der Name und die Versionsnummer des Datenbank-Engine-Treibers, den Ihre Anwendung verwendet.
- `user`— Der Datenbank-Benutzername.
- `preparation`— `call` ob die Abfrage eine verwendet hat`PreparedCall`; `statement` wenn die Abfrage eine verwendet hat`PreparedStatement`.

Example Untersegment mit einer SQL-Abfrage

```
{
  "id": "3fd8634e78ca9560",
  "start_time": 1484872218.696,
  "end_time": 1484872218.697,
  "name": "ebdb@aawijb5u25wdoy.cpamxznpdoq8.us-west-2.rds.amazonaws.com",
  "namespace": "remote",
  "sql" : {
    "url": "jdbc:postgresql://aawijb5u25wdoy.cpamxznpdoq8.us-
west-2.rds.amazonaws.com:5432/ebdb",
    "preparation": "statement",
    "database_type": "PostgreSQL",
    "database_version": "9.5.4",
    "driver_version": "PostgreSQL 9.4.1211.jre7",
    "user" : "dbuser",
    "sanitized_query" : "SELECT  *  FROM  customers  WHERE  customer_id=?;"
  }
}
```

# AWS X-Ray Dämon

> **Note**
>
> Sie können den CloudWatch Agenten jetzt verwenden, um Metriken, Protokolle und Traces von Amazon EC2 EC2-Instances und lokalen Servern zu sammeln. CloudWatch Agentenversion 1.300025.0 und höher können Traces von OpenTelemetryoder X-Ray-Client-SDKs sammeln und an X-Ray senden. Wenn Sie den CloudWatch Agenten anstelle des AWS Distro for OpenTelemetry (ADOT) Collector oder des X-Ray-Daemons zum Sammeln von Traces verwenden, können Sie die Anzahl der verwalteten Agenten reduzieren. Weitere Informationen finden Sie unter dem Thema CloudWatch Agent im CloudWatch Benutzerhandbuch.

Der AWS X-Ray Daemon ist eine Softwareanwendung, die den Datenverkehr auf dem UDP-Port 2000 überwacht, Rohsegmentdaten sammelt und sie an die API weiterleitet. AWS X-Ray Der Daemon arbeitet mit den AWS X-Ray SDKs zusammen und muss laufen, damit die von den SDKs gesendeten Daten den X-Ray-Dienst erreichen können. Der X-Ray-Daemon ist ein Open-Source-Projekt. Du kannst dem Projekt folgen und Issues und Pull-Requests einreichen unter GitHub: github.com/aws/ aws-xray-daemon

Aktivieren AWS Lambda und AWS Elastic Beanstalk verwenden Sie die Integration dieser Dienste mit X-Ray, um den Daemon auszuführen. Lambda führt den Daemon jedes Mal automatisch aus, wenn eine Funktion für eine Stichprobenanforderung aufgerufen wird. Verwenden Sie auf Elastic Beanstalk die `XRayEnabled` Konfigurationsoption, um den Daemon auf den Instances in Ihrer Umgebung auszuführen. Weitere Informationen finden Sie unter

Um den X-Ray-Daemon lokal, lokal oder auf einem anderen Server auszuführen, laden Sie ihn herunter AWS-Services, führen Sie ihn aus und geben Sie ihm dann die Erlaubnis, Segmentdokumente auf X-Ray hochzuladen.

## Herunterladen des Daemons

Sie können den Daemon von Amazon S3, Amazon ECR oder Docker Hub herunterladen und ihn dann lokal ausführen oder ihn beim Start auf einer Amazon EC2 EC2-Instance installieren.

### Amazon S3

X-Ray-Daemon-Installationsprogramme und ausführbare Dateien

- Linux (ausführbar) — (sig) **aws-xray-daemon-linux-3.x.zip**
- Linux (RPM-Installationsprogramm) — aws-xray-daemon-3.x.rpm
- Linux (DEB-Installationsprogramm) — aws-xray-daemon-3.x.deb
- Linux (ARM64, ausführbar) — aws-xray-daemon-linux-arm64-3.x.zip(sig)
- Linux (ARM64, RPM-Installationsprogramm) — aws-xray-daemon-arm64-3.x.rpm
- Linux (ARM64, DEB-Installationsprogramm) — aws-xray-daemon-arm64-3.x.deb
- OS X (ausführbar) — aws-xray-daemon-macos-3.x.zip(sig)
- Windows (ausführbar) — aws-xray-daemon-windows-process-3.x.zip(sig)
- Windows (Dienst) — aws-xray-daemon-windows-service-3.x.zip(sig)

Diese Links verweisen immer auf die neueste 3.x-Version des Daemons. Gehen Sie wie folgt vor, um eine bestimmte Version herunterzuladen:

- Wenn Sie eine Version vor der Version herunterladen möchten`3.3.0`, `3.x` ersetzen Sie sie durch die Versionsnummer. z. B. `2.1.0`. Vor der Version `3.3.0` ist die einzig verfügbare Architektur`arm64`.
- Wenn Sie eine Version nach der anderen herunterladen möchten`3.3.0`, `3.x` ersetzen Sie diese durch die Versionsnummer und `arch` den Architekturtyp. Beispiel: `2.1.0` und `arm64`.

X-Ray-Assets werden in jeder unterstützten Region in Buckets repliziert. Um den Bucket zu verwenden, der Ihnen oder Ihren AWS Ressourcen am nächsten ist, ersetzen Sie die Region in den obigen Links durch Ihre Region.

```
https://s3.us-west-2.amazonaws.com/aws-xray-assets.us-west-2/xray-daemon/aws-xray-daemon-3.x.rpm
```

### Amazon ECR

Ab Version 3.2.0 ist der Daemon auf Amazon ECR zu finden. Bevor Sie ein Image abrufen, sollten Sie Ihren Docker-Client bei der öffentlichen Registrierung von Amazon ECR authentifizieren.

Rufen Sie das neueste veröffentlichte 3.x-Versions-Tag ab, indem Sie den folgenden Befehl ausführen:

```
docker pull public.ecr.aws/xray/aws-xray-daemon:3.x
```

Frühere Versionen oder Alpha-Versionen können heruntergeladen werden, indem sie durch eine `alpha` oder eine bestimmte Versionsnummer ersetzt `3.x` werden.

Wir empfehlen nicht, ein Daemon-Image mit einem Alpha-Tag in einer Produktionsumgebung zu verwenden.

Docker Hub

Der Daemon ist auf Docker Hub zu finden. Führen Sie den folgenden Befehl aus, um die neueste veröffentlichte 3.x-Version herunterzuladen:

```
docker pull amazon/aws-xray-daemon:3.x
```

Frühere Versionen des Daemons können veröffentlicht werden, indem sie durch die gewünschte `3.x` Version ersetzt werden.

## Überprüfen der Signatur des Daemon-Archivs

GPG-Signatur-Dateien sind für Daemon-Komponenten in komprimierten ZIP-Archiven eingeschlossen. Der öffentliche Schlüssel befindet sich hier: `aws-xray.gpg`.

Sie können den öffentlichen Schlüssel verwenden, um zu überprüfen, ob das Daemon-ZIP-Archiv ein Original und unverändert ist. Importieren Sie zunächst den öffentlichen Schlüssel mit GnuPG.

So importieren Sie den öffentlichen Schlüssel

1. Laden Sie den öffentlichen Schlüssel herunter.

   ```
   $ BUCKETURL=https://s3.us-east-2.amazonaws.com/aws-xray-assets.us-east-2
   $ wget $BUCKETURL/xray-daemon/aws-xray.gpg
   ```

2. Importieren Sie den öffentlichen Schlüssel in Ihren Schlüsselbund.

   ```
   $ gpg --import aws-xray.gpg
   gpg: /Users/me/.gnupg/trustdb.gpg: trustdb created
   ```

```
gpg: key 7BFE036BFE6157D3: public key "AWS X-Ray <aws-xray@amazon.com>" imported
gpg: Total number processed: 1
gpg:               imported: 1
```

Verwenden Sie den importierten Schlüssel, um die Signatur des Daemon-ZIP-Archivs zu überprüfen.

So überprüfen Sie die Signatur eines Archivs

1. Laden Sie das Archiv und die Signaturdatei herunter.

```
$ BUCKETURL=https://s3.us-east-2.amazonaws.com/aws-xray-assets.us-east-2
$ wget $BUCKETURL/xray-daemon/aws-xray-daemon-linux-3.x.zip
$ wget $BUCKETURL/xray-daemon/aws-xray-daemon-linux-3.x.zip.sig
```

2. Führen Sie `gpg --verify` aus, um die Signatur zu überprüfen.

```
$ gpg --verify aws-xray-daemon-linux-3.x.zip.sig aws-xray-daemon-linux-3.x.zip
gpg: Signature made Wed 19 Apr 2017 05:06:31 AM UTC using RSA key ID FE6157D3
gpg: Good signature from "AWS X-Ray <aws-xray@amazon.com>"
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: EA6D 9271 FBF3 6990 277F  4B87 7BFE 036B FE61 57D3
```

Beachten Sie die Warnung zu vertrauenswürdigen Inhalten. Ein Schlüssel ist nur vertrauenswürdig, wenn Sie oder eine Person Ihres Vertrauens ihn signiert hat. Das bedeutet nicht, dass die Signatur ungültig ist, sondern nur, dass Sie den öffentlichen Schlüssel nicht überprüft haben.

## Ausführen des Daemons

Führen Sie den Daemon lokal über eine Befehlszeile aus. Verwenden Sie die Option `-o` zur Ausführung im lokalen Modus und `-n` zum Festlegen der Region.

```
~/Downloads$ ./xray -o -n us-east-2
```

Detaillierte plattformspezifische Anweisungen finden Sie in den folgenden Themen:

- Linux (lokal) — Den X-Ray-Daemon unter Linux ausführen
- Windows (lokal) — Den X-Ray-Daemon unter Windows ausführen

- Elastic Beanstalk — Ausführen des X-Ray-DaemonsAWS Elastic Beanstalk
- Amazon EC2 — Ausführen des X-Ray-Daemons auf Amazon EC2
- Amazon ECS — Ausführen von X-Ray-Daemon auf Amazon ECs

Sie können mithilfe von Befehlszeilenoptionen oder einer Konfigurationsdatei das Verhalten des Daemons anpassen. Details dazu finden Sie unter Konfigurieren des AWS X-Ray Daemon.

## Dem Daemon die Erlaubnis geben, Daten an X-Ray zu senden

Der X-Ray-Daemon verwendet das AWS SDK, um Trace-Daten auf X-Ray hochzuladen, und benötigt dazu AWS Zugangsdaten mit entsprechender Genehmigung.

Auf Amazon EC2 verwendet der Daemon automatisch die Instance-Profilrolle der Instance. Informationen zu den Anmeldeinformationen, die für die lokale Ausführung des Daemons erforderlich sind, finden Sie unter Lokales Ausführen Ihrer Anwendung.

Wenn Sie an mehreren Orten (Anmeldeinformationsdatei, Instance-Profil oder Umgebungsvariablen) Anmeldeinformationen angeben, bestimmt die SDK-Anbieterkette, welche Anmeldeinformationen verwendet werden. Weitere Informationen zur Bereitstellung von Anmeldeinformationen für das SDK finden Sie unter Spezifying Credentials im AWS SDK for Go Developer Guide.

Die IAM-Rolle oder der Benutzer, zu der bzw. dem die Anmeldeinformationen des Daemons gehören, benötigt die Berechtigung zum Schreiben von Daten in den Service in Ihrem Namen.

- Um den Daemon auf Amazon EC2 zu verwenden, erstellen Sie eine neue Instance-Profilrolle oder fügen Sie die verwaltete Richtlinie zu einer vorhandenen hinzu.
- Um den Daemon auf Elastic Beanstalk zu verwenden, fügen Sie die verwaltete Richtlinie der Standard-Instance-Profilrolle von Elastic Beanstalk hinzu.
- Informationen zum lokalen Ausführen des Daemons finden Sie unter Lokales Ausführen Ihrer Anwendung.

Weitere Informationen finden Sie unter Identity and Access Management für AWS X-Ray.

## X-Ray-Daemon-Protokolle

Der Daemon gibt Informationen über seine aktuelle Konfiguration und die Segmente aus, an die er sendet. AWS X-Ray

```
2016-11-24T06:07:06Z [Info] Initializing AWS X-Ray daemon 2.1.0
2016-11-24T06:07:06Z [Info] Using memory limit of 49 MB
2016-11-24T06:07:06Z [Info] 313 segment buffers allocated
2016-11-24T06:07:08Z [Info] Successfully sent batch of 1 segments (0.123 seconds)
2016-11-24T06:07:09Z [Info] Successfully sent batch of 1 segments (0.006 seconds)
```

Der Daemon gibt Protokolle standardmäßig an STDOUT aus. Wenn Sie den Daemon im Hintergrund ausführen, verwenden Sie zum Festlegen des Protokolldateipfades die Befehlszeilenoption `--log-file` oder eine Konfigurationsdatei. Sie können auch die Protokollierungsebene festlegen und die Protokoll-Rotation deaktivieren. Detaillierte Anweisungen finden Sie unter Konfigurieren des AWS X-Ray Daemon.

Auf Elastic Beanstalk legt die Plattform den Speicherort der Daemon-Logs fest. Details dazu finden Sie unter Ausführen des X-Ray-DaemonsAWS Elastic Beanstalk.

## Konfigurieren des AWS X-Ray Daemon

Sie können Befehlszeilenoptionen oder eine Konfigurationsdatei verwenden, um das Verhalten des X-Ray-Daemons anzupassen. Die meisten Optionen sind bei beiden Methoden verfügbar. Manche sind jedoch nur in Form von Konfigurationsdateien oder nur in der Befehlszeile verfügbar.

Um zu beginnen, müssen Sie `-n` nur oder kennen`--region`, mit der Sie die Region festlegen, die der Daemon zum Senden von Ablaufverfolgungsdaten an X-Ray verwendet.

```
~/xray-daemon$ ./xray -n us-east-2
```

Wenn Sie den Daemon lokal ausführen, d. h. nicht auf Amazon EC2, können Sie die `-o` Option hinzufügen, die Überprüfung auf Instance-Profilanmeldeinformationen zu überspringen, damit der Daemon schneller bereit wird.

```
~/xray-daemon$ ./xray -o -n us-east-2
```

Mithilfe der restlichen Befehlszeilenoptionen können Sie die Protokollierung konfigurieren, einen anderen Port abhören, die vom Daemon nutzbare Speichermenge begrenzen oder eine Rolle annehmen, um Ablaufverfolgungsdaten an ein anderes Konto zu senden.

Sie können eine Konfigurationsdatei an den Daemon übergeben, um auf erweiterte Konfigurationsoptionen zuzugreifen und beispielsweise die Anzahl gleichzeitiger Aufrufe an X-Ray zu begrenzen, die Protokollrotation zu deaktivieren und Datenverkehr an einen Proxy zu senden.

Sections



## Unterstützte Umgebungsvariablen

Der X-Ray-Daemon unterstützt die folgenden Umgebungsvariablen:

- `AWS_REGION` – Gibt die AWS-Region des X-Ray-Service-Endpunkts an.
- `HTTPS_PROXY` – Gibt eine Proxy-Adresse an, über die der Daemon Segmente hochladen kann. Dies können entweder die DNS-Domänennamen oder die IP-Adressen und Portnummern sein, die von Ihren Proxy-Servern verwendet werden.

## Verwenden von Befehlszeilenoptionen

Übergeben Sie diese Optionen an den Daemon, wenn Sie ihn lokal oder mit einem Benutzerdatenskript ausführen.

Befehlszeilenoptionen

- `-b`, `--bind` – Achten Sie auf Segmentdokumente auf einem anderen UDP-Port.

  ```
  --bind "127.0.0.1:3000"
  ```

  Standard – `2000`.
- `-t`, `--bind-tcp` – Horcht auf Aufrufe des X-Ray-Service auf einem anderen TCP-Port.

  ```
  -bind-tcp "127.0.0.1:3000"
  ```

  Standard – `2000`.
- `-c`, `--config` – Laden Sie eine Konfigurationsdatei aus dem angegebenen Pfad.

  ```
  --config "/home/ec2-user/xray-daemon.yaml"
  ```
- `-f`, `--log-file` – Ausgabeprotokolle an den angegebenen Dateipfad.

```
--log-file "/var/log/xray-daemon.log"
```

- `-l`, `--log-level` – Protokollebene, vom ausführlichsten zum geringsten: dev, debug, info, warningn, error, prod.

  ```
  --log-level warn
  ```

  Standard – `prod`
- `-m`, `--buffer-memory` – Ändern Sie die Speichermenge in MB, die Puffer verwenden können (mindestens 3).

  ```
  --buffer-memory 50
  ```

  Standard – 1 % des verfügbaren Speichers.
- `-o`, `--local-mode` – Überprüfen Sie nicht auf EC2-Instance-Metadaten.
- `-r`, `--role-arn` – Übernehmen Sie die angegebene IAM-Rolle, um Segmente in ein anderes Konto hochzuladen.

  ```
  --role-arn "arn:aws:iam::123456789012:role/xray-cross-account"
  ```
- `-a`, `--resource-arn` – Amazon-Ressourcenname (ARN) der AWS Ressource, auf der der Daemon ausgeführt wird.
- `-p`, `--proxy-address` – Laden Sie Segmente AWS X-Ray über einen Proxy in hoch. Das Protokoll des Proxyservers muss angegeben werden.

  ```
  --proxy-address "http://192.0.2.0:3000"
  ```
- `-n`, `--region` – Senden Sie Segmente an den X-Ray-Service in einer bestimmten Region.
- `-v`, `--version` – Zeigt die AWS X-Ray Daemon-Version an.
- `-h`, `--help` – Zeigt den Hilfebildschirm an.

## Verwendung einer Konfigurationsdatei

Sie können auch eine Datei im YAML-Format zur Konfiguration des Daemons verwenden. Übergeben Sie die Konfigurationsdatei mit der `-c`-Option an den Daemon.

```
~$ ./xray -c ~/xray-daemon.yaml
```

Konfigurationsdateioptionen

- `TotalBufferSizeMB` – Maximale Puffergröße in MB (mindestens 3). Wählen Sie 0, um 1 % des Host-Speichers zu verwenden.
- `Concurrency` – Maximale Anzahl gleichzeitiger Aufrufe an AWS X-Ray , um Segmentdokumente hochzuladen.
- `Region` – Senden Sie Segmente an den AWS X-Ray Service in einer bestimmten Region.
- `Socket` – Konfigurieren Sie die Bindung des Daemon.
  - `UDPAddress` – Ändern Sie den Port, den der Daemon überwacht.
  - `TCPAddress` – Horcht auf Aufrufe des X-Ray-Service auf einem anderen TCP-Port.
- `Logging` – Konfigurieren Sie das Protokollierungsverhalten.
  - `LogRotation` – Setzen Sie diesen Wert auf `false`, um die Protokollrotation zu deaktivieren.
  - `LogLevel` – Ändern Sie die Protokollebene von am ausführlichsten zu am wenigsten: `dev`debug, `info` oder `prod`, `warn`, `error`, , `prod`. Der Standardwert ist `prod`, was entsprichtinfo.
  - `LogPath` – Ausgabeprotokolle an den angegebenen Dateipfad.
- `LocalMode` – Setzen Sie diesen Wert auf `true`, um die Überprüfung auf EC2-Instance-Metadaten zu überspringen.
- `ResourceARN` – Amazon-Ressourcenname (ARN) der AWS Ressource, auf der der Daemon ausgeführt wird.
- `RoleARN` – Übernehmen Sie die angegebene IAM-Rolle, um Segmente in ein anderes Konto hochzuladen.
- `ProxyAddress` – Laden Sie Segmente AWS X-Ray über einen Proxy in hoch.
- `Endpoint` – Ändern Sie den X-Ray-Service-Endpunkt, an den der Daemon Segmentdokumente sendet.
- `NoVerifySSL` – Deaktivieren Sie die Verifizierung des TLS-Zertifikats.
- `Version` – Version des Daemon-Konfigurationsdateiformats. Die Dateiformatversion ist ein Pflichtfeld.

Example Xray-daemon.yaml

In dieser Konfigurationsdatei wird der Listening-Port des Daemons in 3000 geändert. Dazu werden die Prüfungen auf Instance-Metadaten deaktiviert, eine Rolle für den Upload von Segmenten eingerichtet und die Optionen für Region und Protokollierung geändert.

```
Socket:
  UDPAddress: "127.0.0.1:3000"
  TCPAddress: "127.0.0.1:3000"
Region: "us-west-2"
Logging:
  LogLevel: "warn"
  LogPath: "/var/log/xray-daemon.log"
LocalMode: true
RoleARN: "arn:aws:iam::123456789012:role/xray-cross-account"
Version: 2
```

# Lokales Ausführen des X-Ray-Daemons

Sie können den AWS X-Ray Daemon lokal unter Linux, macOS, Windows oder in einem Docker-Container ausführen. Führen Sie den Daemon aus, um Trace-Daten an X-Ray weiterzuleiten, wenn Sie Ihre instrumentierte Anwendung entwickeln und testen. Laden Sie den Daemon herunter und extrahieren Sie ihn. Diesbezügliche Anweisungen finden Sie hier.

Wenn der Daemon lokal ausgeführt wird, kann er Anmeldeinformationen aus einer AWS SDK-Anmeldeinformationsdatei (`.aws/credentials`in Ihrem Benutzerverzeichnis) oder aus Umgebungsvariablen lesen. Weitere Informationen finden Sie unter Dem Daemon die Erlaubnis geben, Daten an X-Ray zu senden.

Der Daemon lauscht an Port 2000 auf Daten. Sie können den Port und andere Optionen mithilfe einer Konfigurationsdatei und von Befehlszeilenoptionen ändern. Weitere Informationen finden Sie unter Konfigurieren des AWS X-Ray Daemon.

## Den X-Ray-Daemon unter Linux ausführen

Sie können die ausführbare Datei des Daemons an der Befehlszeile ausführen. Verwenden Sie die Option `-o` zur Ausführung im lokalen Modus und `-n` zum Festlegen der Region.

```
~/xray-daemon$ ./xray -o -n us-east-2
```

Zum Ausführen des Daemons im Hintergrund verwenden Sie &.

```
~/xray-daemon$ ./xray -o -n us-east-2 &
```

Beenden Sie einen Daemonprozess, der im Hintergrund ausgeführt wird, mit `pkill`.

```
~$ pkill xray
```

## Den X-Ray-Daemon in einem Docker-Container ausführen

Um den Daemon lokal in einem Docker-Container auszuführen, speichern Sie den folgenden Text in einer Datei mit dem Namen `Dockerfile`. Laden Sie das vollständige Beispielbild auf Amazon ECR herunter. Weitere Informationen finden Sie unter Den Daemon herunterladen.

Example Dockerfile — Amazon Linux

```
FROM amazonlinux
RUN yum install -y unzip
RUN curl -o daemon.zip https://s3.us-east-2.amazonaws.com/aws-xray-assets.us-east-2/
xray-daemon/aws-xray-daemon-linux-3.x.zip
RUN unzip daemon.zip && cp xray /usr/bin/xray
ENTRYPOINT ["/usr/bin/xray", "-t", "0.0.0.0:2000", "-b", "0.0.0.0:2000"]
EXPOSE 2000/udp
EXPOSE 2000/tcp
```

Entwickeln Sie das Container-Image mit `docker build`.

```
~/xray-daemon$ docker build -t xray-daemon .
```

Führen Sie das Abbild in einem Container mit `docker run` aus.

```
~/xray-daemon$ docker run \
      --attach STDOUT \
      -v ~/.aws/:/root/.aws/:ro \
      --net=host \
      -e AWS_REGION=us-east-2 \
      --name xray-daemon \
      -p 2000:2000/udp \
      xray-daemon -o
```

Dieser Befehl verwendet die folgenden Optionen:

- `--attach STDOUT`— Zeigt die Ausgabe des Daemons im Terminal an.
- `-v ~/.aws/:/root/.aws/:ro`— Gewähren Sie dem Container nur Lesezugriff auf das `.aws` Verzeichnis, damit er Ihre AWS SDK-Anmeldeinformationen lesen kann.
- `AWS_REGION=`*`us-east-2`*— Setzt die `AWS_REGION` Umgebungsvariable, um dem Daemon mitzuteilen, welche Region er verwenden soll.
- `--net=host`— Hängt den Container an das `host` Netzwerk an. Container im Hostnetzwerk können miteinander kommunizieren, ohne Ports zu veröffentlichen.
- `-p 2000:2000/udp`— Ordnen Sie den UDP-Port 2000 auf Ihrem Computer demselben Port auf dem Container zu. Dies ist für die Kommunikation von Containern in demselben Netzwerk nicht erforderlich. Sie können mit ihr aber Segmente über die Befehlszeile oder Anwendung, die nicht in Docker ausgeführt wird, zum Daemon senden.
- `--name xray-daemon`— Benennen Sie den Container, `xray-daemon` anstatt einen zufälligen Namen zu generieren.
- `-o`(nach dem Imagenamen) — Hängen Sie die `-o` Option an den Einstiegspunkt an, der den Daemon innerhalb des Containers ausführt. Diese Option weist den Daemon an, im lokalen Modus zu laufen, um zu verhindern, dass er versucht, Amazon EC2 EC2-Instance-Metadaten zu lesen.

Um den Daemon zu beenden, verwenden Sie `docker stop`. Wenn Sie nach der Vornahme von Änderungen am `Dockerfile` ein neues Abbild erstellen, müssen Sie den vorhandenen Container löschen, damit Sie einen anderen mit demselben Namen erstellen können. Verwenden Sie `docker rm` zum Löschen des Containers.

```
$ docker stop xray-daemon
$ docker rm xray-daemon
```

## Den X-Ray-Daemon unter Windows ausführen

Sie können die ausführbare Datei des Daemons an der Befehlszeile ausführen. Verwenden Sie die Option `-o` zur Ausführung im lokalen Modus und `-n` zum Festlegen der Region.

```
> .\xray_windows.exe -o -n us-east-2
```

Verwenden Sie ein PowerShell Skript, um einen Dienst für den Daemon zu erstellen und auszuführen.

Example PowerShell Skript — Windows

```
if ( Get-Service "AWSXRayDaemon" -ErrorAction SilentlyContinue ){
    sc.exe stop AWSXRayDaemon
    sc.exe delete AWSXRayDaemon
}
if ( Get-Item -path aws-xray-daemon -ErrorAction SilentlyContinue ) {
    Remove-Item -Recurse -Force aws-xray-daemon
}

$currentLocation = Get-Location
$zipFileName = "aws-xray-daemon-windows-service-3.x.zip"
$zipPath = "$currentLocation\$zipFileName"
$destPath = "$currentLocation\aws-xray-daemon"
$daemonPath = "$destPath\xray.exe"
$daemonLogPath = "C:\inetpub\wwwroot\xray-daemon.log"
$url = "https://s3.dualstack.us-west-2.amazonaws.com/aws-xray-assets.us-west-2/xray-
daemon/aws-xray-daemon-windows-service-3.x.zip"

Invoke-WebRequest -Uri $url -OutFile $zipPath
Add-Type -Assembly "System.IO.Compression.Filesystem"
[io.compression.zipfile]::ExtractToDirectory($zipPath, $destPath)

sc.exe create AWSXRayDaemon binPath= "$daemonPath -f $daemonLogPath"
sc.exe start AWSXRayDaemon
```

## Den X-Ray-Daemon auf OS X ausführen

Sie können die ausführbare Datei des Daemons an der Befehlszeile ausführen. Verwenden Sie die Option `-o` zur Ausführung im lokalen Modus und `-n` zum Festlegen der Region.

```
~/xray-daemon$ ./xray_mac -o -n us-east-2
```

Zum Ausführen des Daemons im Hintergrund verwenden Sie `&`.

```
~/xray-daemon$ ./xray_mac -o -n us-east-2 &
```

Verwenden Sie `nohup`, um zu verhindern, dass der Daemon vorzeitig beendet wird, wenn das Terminal geschlossen wird.

```
~/xray-daemon$ nohup ./xray_mac &
```

# Ausführen des X-Ray-DaemonsAWS Elastic Beanstalk

So leiten Sie Trace-Daten von Ihrer Anwendung anAWS X-Rayverwenden, können Sie den X-Ray-Daemon auf den Amazon EC2 EC2-Instances Ihrer Elastic Beanstalk-Umgebung ausführen. Eine Liste der unterstützten Plattformen finden Sie unterKonfigurierenAWS X-RayDebuggingimAWS Elastic BeanstalkEntwicklerhandbuchaus.

> **Note**
>
> Der Daemon verwendet das Instance-Profil Ihrer Umgebung für Berechtigungen. Anweisungen zum Hinzufügen von Berechtigungen zum Elastic Beanstalk-Instance-Profil finden Sie unterDem Daemon die Erlaubnis geben, Daten an X-Ray zu sendenaus.

Elastic Beanstalk-Plattformen bieten eine Konfigurationsoption, die Sie festlegen können, um den Daemon automatisch auszuführen. Sie können den Daemon in einer Konfigurationsdatei in Ihrem Quellcode oder durch Wahl einer Option in der Elastic Beanstalk-Konsole aktivieren. Wenn Sie die Konfigurationsoption aktivieren, wird der Daemon in der Instance installiert und als Service ausgeführt.

Die auf Elastic Beanstalk-Plattformen enthaltene Version ist möglicherweise nicht die neueste Version. Dem Thema "Unterstützte Plattformen" können Sie entnehmen, welche Version des Daemons für Ihre Plattformkonfiguration verfügbar ist.

Elastic Beanstalk stellt den X-Ray-Daemon nicht auf der Multicontainer Docker (Amazon ECS) - Plattform zur Verfügung.

## Verwenden der Elastic Beanstalk-X-Ray-Daemon

Verwenden Sie die Konsole, um die X-Ray-Integration zu aktivieren, oder konfigurieren Sie sie in Ihrem Anwendungsquellcode mit einer Konfigurationsdatei.

So aktivieren Sie den X-Ray-Daemon in der Elastic Beanstalk-Konsole

1. Öffnen SieElastic Beanstalk-Konsoleaus.
2. Navigieren Sie zur-Managementkonsolefür Ihre Umgebung.
3. Wählen Sie Konfiguration.
4. Wählen Sie Software Settings (Softwareeinstellungen).

5. Wählen Sie für X-Ray daemon Enabled.
6. Wählen Sie Apply (Anwenden) aus.

Sie können eine Konfigurationsdatei in Ihren Quellcode aufnehmen, um die Konfiguration zwischen verschiedenen Umgebungen übertragbar zu gestalten.

Example .ebextensions/xray-daemon.config

```
option_settings:
  aws:elasticbeanstalk:xray:
    XRayEnabled: true
```

Elastic Beanstalk übergibt eine Konfigurationsdatei an den Daemon und gibt Protokolle an einem Standardstandort aus.

Auf Windows Server-Plattformen

- Konfigurationsdatei–`C:\Program Files\Amazon\XRay\cfg.yaml`
- Protokolle–`c:\Program Files\Amazon\XRay\logs\xray-service.log`

Auf Linux-Plattformen

- Konfigurationsdatei–`/etc/amazon/xray/cfg.yaml`
- Protokolle–`/var/log/xray/xray.log`

Elastic Beanstalk bietet Tools zum Ziehen von Instance-Protokollen aus demAWS Management Consoleoder Befehlszeile Sie können Elastic Beanstalk anweisen, die X-Ray-Daemon-Protokolle einzuschließen, indem Sie eine Aufgabe mit einer Konfigurationsdatei hinzufügen.

Example .ebextensions/xray-logs.config – Linux

```
files:
  "/opt/elasticbeanstalk/tasks/taillogs.d/xray-daemon.conf" :
    mode: "000644"
    owner: root
    group: root
    content: |
      /var/log/xray/xray.log
```

Example .ebextensions/xray-logs.config – Windows Server

```
files:
  "c:/Program Files/Amazon/ElasticBeanstalk/config/taillogs.d/xray-daemon.conf" :
    mode: "000644"
    owner: root
    group: root
    content: |
      c:\Progam Files\Amazon\XRay\logs\xray-service.log
```

Siehe .Anzeigen von Protokollen aus den Amazon EC2 EC2-Instances Ihrer Elastic Beanstalk-UmgebungimAWS Elastic BeanstalkEntwicklerhandbuchWeitere Informationen finden Sie unter.

## Manuelles Herunterladen und Ausführen des X-Ray-Daemons (Erweitert)

Wenn der X-Ray-Daemon für Ihre Plattformkonfiguration nicht verfügbar ist, können Sie sie von Amazon S3 herunterladen und mit einer Konfigurationsdatei ausführen.

Verwenden Sie eine Elastic Beanstalk-Konfigurationsdatei, um den Daemon herunterzuladen und auszuführen.

Example .ebextensions/xray.config – Linux

```
commands:
  01-stop-tracing:
    command: yum remove -y xray
    ignoreErrors: true
  02-copy-tracing:
    command: curl https://s3.us-east-2.amazonaws.com/aws-xray-assets.us-east-2/xray-
daemon/aws-xray-daemon-3.x.rpm -o /home/ec2-user/xray.rpm
  03-start-tracing:
    command: yum install -y /home/ec2-user/xray.rpm

files:
  "/opt/elasticbeanstalk/tasks/taillogs.d/xray-daemon.conf" :
    mode: "000644"
    owner: root
    group: root
    content: |
      /var/log/xray/xray.log
  "/etc/amazon/xray/cfg.yaml" :
    mode: "000644"
```

```
    owner: root
    group: root
    content: |
      Logging:
        LogLevel: "debug"
      Version: 2
```

Example .ebextensions/xray.config – Windows Server

```
container_commands:
  01-execute-config-script:
    command: Powershell.exe -ExecutionPolicy Bypass -File c:\\temp\\installDaemon.ps1
    waitAfterCompletion: 0

files:
  "c:/temp/installDaemon.ps1":
    content: |
      if ( Get-Service "AWSXRayDaemon" -ErrorAction SilentlyContinue ) {
          sc.exe stop AWSXRayDaemon
          sc.exe delete AWSXRayDaemon
      }

      $targetLocation = "C:\Program Files\Amazon\XRay"
      if ((Test-Path $targetLocation) -eq 0) {
          mkdir $targetLocation
      }

      $zipFileName = "aws-xray-daemon-windows-service-3.x.zip"
      $zipPath = "$targetLocation\$zipFileName"
      $destPath = "$targetLocation\aws-xray-daemon"
      if ((Test-Path $destPath) -eq 1) {
          Remove-Item -Recurse -Force $destPath
      }

      $daemonPath = "$destPath\xray.exe"
      $daemonLogPath = "$targetLocation\xray-daemon.log"
      $url = "https://s3.dualstack.us-west-2.amazonaws.com/aws-xray-assets.us-west-2/
xray-daemon/aws-xray-daemon-windows-service-3.x.zip"

      Invoke-WebRequest -Uri $url -OutFile $zipPath
      Add-Type -Assembly "System.IO.Compression.Filesystem"
      [io.compression.zipfile]::ExtractToDirectory($zipPath, $destPath)
```

```
      New-Service -Name "AWSXRayDaemon" -StartupType Automatic -BinaryPathName
 "`"$daemonPath`" -f `"$daemonLogPath`""
      sc.exe start AWSXRayDaemon
    encoding: plain
  "c:/Program Files/Amazon/ElasticBeanstalk/config/taillogs.d/xray-daemon.conf" :
    mode: "000644"
    owner: root
    group: root
    content: |
      C:\Program Files\Amazon\XRay\xray-daemon.log
```

In diesen Beispielen wird die Protokolldatei des Daemons außerdem der Elastic Beanstalkfragmentaufgabe hinzugefügt, um sie einzubeziehen, wenn Sie Protokolle mit der Konsole oder Elastic Beanstalk-Befehlszeilenschnittstelle (EB CLI) anfordern.

## Ausführen des X-Ray-Daemons auf Amazon EC2

Sie können den X-Ray-Daemon unter folgenden Betriebssystemen in Amazon EC2 ausführen:

- Amazon Linux
- Ubuntu
- Windows Server (2012 R2 und neuer)

Verwenden Sie ein Instance-Profil, um dem Daemon die Berechtigung zum Hochladen von Ablaufverfolgungsdaten auf X-Ray Weitere Informationen finden Sie unter Dem Daemon die Erlaubnis geben, Daten an X-Ray zu senden .

Verwenden Sie ein Benutzerdatenskript, um den Daemon automatisch auszuführen, wenn Sie die Instance starten.

Example Benutzerdatenskript – Linux

```
#!/bin/bash
curl https://s3.us-east-2.amazonaws.com/aws-xray-assets.us-east-2/xray-daemon/aws-xray-
daemon-3.x.rpm -o /home/ec2-user/xray.rpm
yum install -y /home/ec2-user/xray.rpm
```

Example Benutzerdatenskript – Windows Server

```
<powershell>
```

```
if ( Get-Service "AWSXRayDaemon" -ErrorAction SilentlyContinue ) {
    sc.exe stop AWSXRayDaemon
    sc.exe delete AWSXRayDaemon
}

$targetLocation = "C:\Program Files\Amazon\XRay"
if ((Test-Path $targetLocation) -eq 0) {
    mkdir $targetLocation
}

$zipFileName = "aws-xray-daemon-windows-service-3.x.zip"
$zipPath = "$targetLocation\$zipFileName"
$destPath = "$targetLocation\aws-xray-daemon"
if ((Test-Path $destPath) -eq 1) {
    Remove-Item -Recurse -Force $destPath
}

$daemonPath = "$destPath\xray.exe"
$daemonLogPath = "$targetLocation\xray-daemon.log"
$url = "https://s3.dualstack.us-west-2.amazonaws.com/aws-xray-assets.us-west-2/xray-
daemon/aws-xray-daemon-windows-service-3.x.zip"

Invoke-WebRequest -Uri $url -OutFile $zipPath
Add-Type -Assembly "System.IO.Compression.Filesystem"
[io.compression.zipfile]::ExtractToDirectory($zipPath, $destPath)

New-Service -Name "AWSXRayDaemon" -StartupType Automatic -BinaryPathName
 "`"$daemonPath`" -f `"$daemonLogPath`""
sc.exe start AWSXRayDaemon
</powershell>
```

# Ausführen von X-Ray-Daemon auf Amazon ECs

Erstellen Sie in Amazon ECS ein Docker-Image, das den X-Ray-Daemon ausführt, laden Sie es in ein Docker-Image-Repository hoch und stellen Sie es dann in Ihrem Amazon ECS-Cluster bereit. Sie können Ihrer Anwendung über Port-Zuordnungen und Netzwerkmodus-Einstellungen in Ihrer Aufgabendefinitionsdatei die Kommunikation mit dem Daemon-Container ermöglichen.

## Verwenden des offiziellen -Docker-Image

X-Ray stellt ein Docker-Container-Image auf Amazon ECR bereit, das Sie zusammen mit Ihrer Anwendung bereitstellen können. Weitere Informationen finden Sie unter Herunterladen des Daemons.

Example Aufgabendefinition

```
{
  "name": "xray-daemon",
  "image": "amazon/aws-xray-daemon",
  "cpu": 32,
  "memoryReservation": 256,
  "portMappings" : [
      {
          "hostPort": 0,
          "containerPort": 2000,
          "protocol": "udp"
      }
   ]
}
```

## Entwickeln und Erstellen eines Docker-Images

Bei einer benutzerdefinierten Konfiguration müssen Sie möglicherweise ein eigenes Docker-Image definieren.

Fügen Sie Ihrer Aufgabenrolle verwaltete Richtlinien zu X-Ray hinzu, um dem Daemon die Berechtigung zum Hochladen von Trace-Daten zu X-Ray. Weitere Informationen finden Sie unter Dem Daemon die Erlaubnis geben, Daten an X-Ray zu senden.

Erstellen Sie mit einer der folgenden Dockerfiles ein Abbild, mit dem der Daemon ausführt wird.

Example Dockerfile-Datei — Amazon Linux

```
FROM amazonlinux
RUN yum install -y unzip
RUN curl -o daemon.zip https://s3.us-east-2.amazonaws.com/aws-xray-assets.us-east-2/
xray-daemon/aws-xray-daemon-linux-3.x.zip
```

```
RUN unzip daemon.zip && cp xray /usr/bin/xray
ENTRYPOINT ["/usr/bin/xray", "-t", "0.0.0.0:2000", "-b", "0.0.0.0:2000"]
EXPOSE 2000/udp
EXPOSE 2000/tcp
```

> **Note**
>
> Die Flags `-t` und `-b` sind erforderlich, um eine Bindungsadresse anzugeben, die den Loopback einer Multi-Container-Umgebung überwacht.

Example Dockerfile-Datei — Ubuntu

Für Debian-Derivate müssen Sie auch Zertifikate installieren, die von einer Zertifizierungsstelle (Certificate Authority, CA) ausgegeben wurden, um Probleme beim Herunterladen des Installationsprogramms zu vermeiden.

```
FROM ubuntu:16.04
RUN apt-get update && apt-get install -y --force-yes --no-install-recommends apt-
transport-https curl ca-certificates wget && apt-get clean && apt-get autoremove && rm
 -rf /var/lib/apt/lists/*
RUN wget https://s3.us-east-2.amazonaws.com/aws-xray-assets.us-east-2/xray-daemon/aws-
xray-daemon-3.x.deb
RUN dpkg -i aws-xray-daemon-3.x.deb
ENTRYPOINT ["/usr/bin/xray", "--bind=0.0.0.0:2000", "--bind-tcp=0.0.0.0:2000"]
EXPOSE 2000/udp
EXPOSE 2000/tcp
```

In Ihrer Aufgabendefinition hängt die Konfiguration von dem Netzwerkmodus ab, den Sie tatsächlich nutzen. Brückennetzwerke sind die Standardeinstellung und können in Ihrer Standard-VPC verwendet werden. Stellen Sie in einem Bridge-Netzwerk die`AWS_XRAY_DAEMON_ADDRESS` Umgebungsvariable ein, um dem X-Ray SDK mitzuteilen, auf welchen Container-Port verwiesen werden soll, und legen Sie den Host-Port fest. Sie können beispielsweise UDP-Port 2000 veröffentlichen und einen Link von Ihrem Anwendungs-Container zum Daemon-Container erstellen.

Example Aufgabendefinition

```
    {
      "name": "xray-daemon",
```

```
      "image": "123456789012.dkr.ecr.us-east-2.amazonaws.com/xray-daemon",
      "cpu": 32,
      "memoryReservation": 256,
      "portMappings" : [
          {
              "hostPort": 0,
              "containerPort": 2000,
              "protocol": "udp"
          }
       ]
    },
    {
      "name": "scorekeep-api",
      "image": "123456789012.dkr.ecr.us-east-2.amazonaws.com/scorekeep-api",
      "cpu": 192,
      "memoryReservation": 512,
      "environment": [
          { "name" : "AWS_REGION", "value" : "us-east-2" },
          { "name" : "NOTIFICATION_TOPIC", "value" : "arn:aws:sns:us-
east-2:123456789012:scorekeep-notifications" },
          { "name" : "AWS_XRAY_DAEMON_ADDRESS", "value" : "xray-daemon:2000" }
      ],
      "portMappings" : [
          {
              "hostPort": 5000,
              "containerPort": 5000
          }
      ],
      "links": [
        "xray-daemon"
      ]
    }
```

Wenn Sie Ihren Cluster im privaten Subnetz einer VPC ausführen, können Sie Ihren Containern mit dem awsvpc-Netzwerkmodus einer Elastic Network-Schnittstelle (ENI) zuweisen. Auf diese Weise können Sie die Nutzung von Links vermeiden. Lassen Sie den Host-Port in den Port-Zuordnungen, dem Link und der Umgebungsvariable `AWS_XRAY_DAEMON_ADDRESS` weg.

Example VPC-Aufgabendefinition

```
{
    "family": "scorekeep",
    "networkMode":"awsvpc",
```

```
    "containerDefinitions": [
        {
          "name": "xray-daemon",
          "image": "123456789012.dkr.ecr.us-east-2.amazonaws.com/xray-daemon",
          "cpu": 32,
          "memoryReservation": 256,
          "portMappings" : [
              {
                  "containerPort": 2000,
                  "protocol": "udp"
              }
          ]
        },
        {
            "name": "scorekeep-api",
            "image": "123456789012.dkr.ecr.us-east-2.amazonaws.com/scorekeep-api",
            "cpu": 192,
            "memoryReservation": 512,
            "environment": [
                { "name" : "AWS_REGION", "value" : "us-east-2" },
                { "name" : "NOTIFICATION_TOPIC", "value" : "arn:aws:sns:us-
east-2:123456789012:scorekeep-notifications" }
            ],
            "portMappings" : [
                {
                    "containerPort": 5000
                }
            ]
        }
    ]
}
```

## Konfigurieren Sie die Befehlszeilenoptionen in der Amazon ECS-Konsole

Befehlszeilenoptionen überschreiben Konflikte in der Konfigurationsdatei Ihres Abbildes. Befehlszeilenoptionen werden normalerweise für lokale Tests verwendet, können aber auch aus praktischen Gründen beim Festlegen von Umgebungsvariablen oder zur Steuerung des Startvorgangs verwendet werden.

Durch Hinzufügen von Befehlszeilenoptionen aktualisieren Sie den Docker `CMD`, der an den Container übergeben wird. Weitere Informationen finden Sie in der Referenz zu docker run.

So legen Sie eine Befehlszeilenoption fest

1. Öffnen Sie die klassische Amazon-EC-Amazon-EC-Amazon-EC-Amazon-EC https://console.aws.amazon.com/ecs/
2. Wählen Sie auf der Navigationsleiste die Region aus, in der Ihre Aufgabendefinition enthalten ist.
3. Wählen Sie im Navigationsbereich Task Definitions aus.
4. Aktivieren Sie auf der Seite Task Definitions das Kontrollkästchen links neben der Aufgabendefinition, die Sie überarbeiten möchten, und wählen Sie dann Create new revision aus.
5. Wählen Sie auf der Seite Create new revision of Task Definition (Neue Revision der Aufgabendefinition erstellen) den Container aus.
6. Fügen Sie im Abschnitt ENVIRONMENT Ihre durch Komma getrennte Liste von Befehlszeilenoptionen zum Feld Command hinzu.
7. Wählen Sie Update (Aktualisieren).
8. Überprüfen Sie die Informationen und wählen Sie dann Create aus.

Das folgende Beispiel zeigt, wie eine durch Kommas getrennte Befehlszeilenoption für die Option `RoleARN` geschrieben wird. Die`RoleARN` Option nimmt die angegebene IAM-Rolle an, um Segmente auf ein anderes Konto hochzuladen.

Example

```
--role-arn, arn:aws:iam::123456789012:role/xray-cross-account
```

Weitere Informationen zu den verfügbaren Befehlszeilenoptionen in X-Ray finden Sie unter Configuring theAWS X-Ray Daemon.

# Instrumentieren Sie Ihre Bewerbung für AWS X-Ray

Die Instrumentierung Ihrer Anwendung umfasst das Senden von Trace-Daten für eingehende und ausgehende Anfragen und andere Ereignisse innerhalb Ihrer Anwendung sowie Metadaten zu jeder Anfrage. Es gibt verschiedene Instrumentierungsoptionen, aus denen Sie je nach Ihren speziellen Anforderungen wählen oder diese kombinieren können:

- auto Instrumentierung — Instrumentierung Ihrer Anwendung ohne Codeänderungen, typischerweise durch Konfigurationsänderungen, Hinzufügen eines Agenten für automatische Instrumentierung oder andere Mechanismen.
- Bibliotheksinstrumentierung — Nehmen Sie minimale Änderungen am Anwendungscode vor, um vorgefertigte Instrumentierung hinzuzufügen, die auf bestimmte Bibliotheken oder Frameworks wie das AWS SDK, Apache HTTP-Clients oder SQL-Clients abzielt.
- Manuelle Instrumentierung — fügen Sie Ihrer Anwendung an jeder Stelle, an die Sie Trace-Informationen senden möchten, Instrumentierungscode hinzu.

Es gibt mehrere SDKs, Agenten und Tools, mit denen Sie Ihre Anwendung für das X-Ray-Tracing instrumentieren können.

Themen

# Instrumentierung Ihrer Anwendung mit der Distro für AWS OpenTelemetry

The AWS Distro for OpenTelemetry (ADOT) ist eine AWS Distribution, die auf dem Projekt Cloud Native Computing Foundation (CNCF) basiert. OpenTelemetry OpenTelemetry bietet einen einzigen Satz von Open-Source-APIs, -Bibliotheken und -Agenten zur Erfassung verteilter Traces und Metriken. Bei diesem Toolkit handelt es sich um eine Distribution von OpenTelemetry Upstream-Komponenten wie SDKs, Agenten für automatische Instrumentierung und Collectors, die von getestet, optimiert, gesichert und unterstützt werden. AWS

Mit ADOT können Techniker ihre Anwendungen einmal instrumentieren und korrelierte Metriken und Traces an mehrere AWS Überwachungslösungen wie Amazon CloudWatch und Amazon AWS X-Ray OpenSearch Service senden.

Die Verwendung von X-Ray mit ADOT erfordert zwei Komponenten: ein OpenTelemetry SDK, das für die Verwendung mit X-Ray aktiviert ist, und das AWS Distro for OpenTelemetry Collector, das für die Verwendung mit X-Ray aktiviert ist. Weitere Informationen zur Verwendung der AWS Distribution für OpenTelemetry AWS X-Ray und andere finden Sie in der AWS-Services Dokumentation zur AWS Distribution. OpenTelemetry

Weitere Informationen zur Sprachunterstützung und -verwendung finden Sie unter AWS Observability on. GitHub

> **Note**
>
> Sie können den CloudWatch Agenten jetzt verwenden, um Metriken, Protokolle und Traces von Amazon EC2 EC2-Instances und lokalen Servern zu sammeln. CloudWatch Agentenversion 1.300025.0 und höher können Traces von OpenTelemetryoder X-Ray-Client-SDKs sammeln und an X-Ray senden. Wenn Sie den CloudWatch Agenten anstelle des AWS Distro for OpenTelemetry (ADOT) Collector oder des X-Ray-Daemons zum Sammeln von Traces verwenden, können Sie die Anzahl der verwalteten Agenten reduzieren. Weitere Informationen finden Sie unter dem Thema CloudWatch Agent im CloudWatch Benutzerhandbuch.

ADOT umfasst Folgendes:

- AWS Distro für Go OpenTelemetry

- AWS Distro für Java OpenTelemetry
- AWS Distro für OpenTelemetry JavaScript
- AWS Distro für Python OpenTelemetry
- AWS Distro für .NET OpenTelemetry

ADOT bietet derzeit Unterstützung für automatische Instrumentierung für Java und Python. Darüber hinaus ermöglicht ADOT die automatische Instrumentierung von AWS Lambda-Funktionen und ihren Downstream-Anfragen mithilfe von Java-, Node.js- und Python-Laufzeiten über ADOT Managed Lambda Layers.

ADOT SDKs für Java und Go unterstützen zentralisierte X-Ray-Sampling-Regeln. Wenn Sie Unterstützung für X-Ray-Sampling-Regeln in anderen Sprachen benötigen, sollten Sie die Verwendung eines AWS X-Ray SDK in Betracht ziehen.

> **Note**
>
> Sie können jetzt W3C-Trace-IDs an X-Ray senden. Standardmäßig OpenTelemetry haben Traces, die mit erstellt wurden, ein Trace-ID-Format, das auf der W3C Trace Context-Spezifikation basiert. Dies unterscheidet sich von dem Format für Trace-IDs, die mit einem X-Ray-SDK oder durch AWS Dienste, die in X-Ray integriert sind, erstellt werden. Um sicherzustellen, dass Trace-IDs im W3C-Format von X-Ray akzeptiert werden, müssen Sie AWS X-Ray Exporter Version 0.86.0 oder höher verwenden, die in ADOT Collector Version 0.34.0 und höher enthalten ist. Frühere Versionen des Exporters validieren Trace-ID-Zeitstempel, was dazu führen kann, dass W3C-Trace-IDs zurückgewiesen werden.

# Instrumentierung Ihrer Anwendung mit SDKs AWS X-Ray

AWS X-Ray enthält eine Reihe sprachspezifischer SDKs, mit denen Sie Ihre Anwendung so ausstatten können, dass sie Traces an X-Ray sendet. Jedes X-Ray-SDK bietet Folgendes:

- Interceptors, die Sie Ihrem Code hinzufügen können, um eingehende HTTP-Anforderungen nachzuverfolgen.
- Client-Handler zur Instrumentierung von AWS SDK-Clients, die Ihre Anwendung verwendet, um andere aufzurufen AWS-Services
- Ein HTTP-Client zur Instrumentierung von Aufrufen an andere interne und externe HTTP-Webdienste

X-Ray-SDKs unterstützen auch die Instrumentierung von Aufrufen von SQL-Datenbanken, automatische AWS SDK-Client-Instrumentierung und andere Funktionen. Anstatt Trace-Daten direkt an X-Ray zu senden, sendet das SDK JSON-Segmentdokumente an einen Daemon-Prozess, der auf UDP-Verkehr wartet. Der X-Ray-Daemon puffert Segmente in einer Warteschlange und lädt sie stapelweise auf X-Ray hoch.

Die folgenden sprachspezifischen SDKs werden bereitgestellt:

- AWS X-Ray SDK for Go
- AWS X-Ray SDK für Java
- AWS X-Ray SDK für Node.js
- AWS X-Ray SDK für Python
- AWS X-Ray SDK for .NET
- AWS X-Ray SDK for Ruby

X-Ray bietet derzeit Unterstützung für automatische Instrumentierung für Java.

## Wahl zwischen den AWS SDKs Distro for OpenTelemetry und X-Ray

Die in X-Ray enthaltenen SDKs sind Teil einer eng integrierten Instrumentierungslösung von AWS. Das AWS Distro for OpenTelemetry ist Teil einer umfassenderen Branchenlösung, bei der X-Ray nur eine von vielen Tracing-Lösungen ist. Sie können die end-to-end Ablaufverfolgung in X-Ray mit beiden Methoden implementieren, aber es ist wichtig, die Unterschiede zu verstehen, um den für Sie nützlichsten Ansatz zu finden.

Wir empfehlen, Ihre Anwendung mit der AWS Distribution zu instrumentieren, OpenTelemetry wenn Sie Folgendes benötigen:

- Die Möglichkeit, Traces an mehrere verschiedene Tracing-Backends zu senden, ohne Ihren Code erneut instrumentieren zu müssen
- Support für eine große Anzahl von Bibliotheksinstrumenten für jede Sprache, verwaltet von der Community OpenTelemetry
- Vollständig verwaltete Lambda-Ebenen, die alles zusammenfassen, was Sie für die Erfassung von Telemetriedaten benötigen, ohne dass Codeänderungen erforderlich sind, wenn Sie Java, Python oder Node.js verwenden

> **Note**
>
> AWS Distro for OpenTelemetry bietet einen einfacheren Einstieg in die Instrumentierung Ihrer Lambda-Funktionen. Aufgrund der gebotenen Flexibilität OpenTelemetry benötigt Ihre Lambda-Funktion jedoch zusätzlichen Speicher, und bei Aufrufen kann es zu einer Erhöhung der Kaltstart-Latenz kommen, was zu zusätzlichen Kosten führen kann. Wenn Sie für niedrige Latenzzeiten optimieren und keine erweiterten Funktionen wie dynamisch konfigurierbare Back-End-Ziele benötigen OpenTelemetry, sollten Sie das AWS X-Ray-SDK zur Instrumentierung Ihrer Anwendung verwenden.

Wir empfehlen, ein X-Ray-SDK für die Instrumentierung Ihrer Anwendung zu wählen, wenn Sie Folgendes benötigen:

- Eine eng integrierte Lösung aus einem einzigen Anbieter
- Integration mit zentralisierten X-Ray-Probenregeln, einschließlich der Möglichkeit, Sampling-Regeln von der X-Ray-Konsole aus zu konfigurieren und sie automatisch auf mehreren Hosts zu verwenden, wenn Node.js, Python, Ruby oder .NET verwendet werden

## Instrumentieren Sie Ihre Bewerbung mit Go

Es gibt zwei Möglichkeiten, Ihre Go Anwendung so zu instrumentieren, dass sie Spuren an X-Ray sendet:

- AWS Distro for OpenTelemetry Go — Eine AWS Distribution, die eine Reihe von Open-Source-Bibliotheken zum Senden korrelierter Metriken und Traces über AWS Distro for Collector an mehrere AWS Überwachungslösungen wie Amazon CloudWatch und Amazon OpenSearch Service bereitstellt. AWS X-Ray OpenTelemetry
- AWS X-Ray SDK für Go — Eine Reihe von Bibliotheken zum Generieren und Senden von Traces an X-Ray über den X-Ray-Daemon.

Weitere Informationen finden Sie unter Wahl zwischen den AWS SDKs Distro for OpenTelemetry und X-Ray.

## AWSDistro fürOpenTelemetryGeh

Mit demAWSDistribution fürOpenTelemetryGo, Sie können Ihre Anwendungen einmal instrumentieren und korrelierte Metriken und Traces an mehrere sendenAWSÜberwachungslösungen wie AmazonCloudWatch,AWS X-Ray, und AmazonOpenSearchBedienung. X-Ray verwenden mitAWSDistribution fürOpenTelemetrybenötigt zwei Komponenten: eineOpenTelemetrySDKaktiviert für die Verwendung mit X-Ray und demAWSDistribution fürOpenTelemetrySammlerfür die Verwendung mit X-Ray aktiviert.

Informationen zu den ersten Schritten finden Sie unterAWSDistribution fürOpenTelemetryZur Dokumentation.

Weitere Informationen zur Verwendung desAWSDistribution fürOpenTelemetrymitAWS X-Rayund andereAWS-Services, siheAWSDistribution fürOpenTelemetryoder dieAWSDistribution fürOpenTelemetryDokumentation.

Weitere Informationen zur Sprachunterstützung und Sprachverwendung finden Sie unterAWSBeobachtbarkeit aufGitHub.

## AWS X-Ray-SDK für Go

Das X-Ray-SDK SDK for Go besteht aus einer Reihe von Bibliotheken für Go-Anwendungen, die Klassen und Methoden zum Generieren und Senden von Ablaufverfolgungsdaten an den X-Ray-Daemon bereitstellen. Ablaufverfolgungsdaten umfassen Informationen zu eingehenden HTTP-Anforderungen, die von der Anwendung beantwortet werden, sowie Aufrufe, die Anwendung an nachgelagerte Services mithilfe desAWSSDK, HTTP-Clients oder ein SQL-Datenbankkonnektor. Sie können Segmente auch manuell erstellen und Debug-Informationen Anmerkungen und Metadaten hinzufügen.

Laden Sie das SDK aus seinem GitHub-Repository mit `go get` herunter:

```
$ go get -u github.com/aws/aws-xray-sdk-go/...
```

Für Webanwendungen beginnen Sie damit, die Funktion `xray.Handler` zu verwenden, um eingehende Anforderungen nachzuverfolgen. Der Message Handler erstellt für jede rückverfolgte Anforderung ein Segment und vervollständigt das Segment, nachdem die Antwort gesendet wurde. Während das Segment geöffnet ist, können Sie die SDK-Client-Methoden nutzen, um dem Segment Informationen hinzuzufügen, Untersegmente zu erstellen und nachgelagerte Aufrufe rückzuverfolgen.

Das SDK erfasst auch automatisch Ausnahmen, die Ihre Anwendung ausgibt, während das Segment geöffnet ist.

Für Lambda-Funktionen, die von einer instrumentierten Anwendung oder einem Dienst aufgerufen werden, liest Lambda dieAblaufverfolgungs-Headerund verfolgt gesampelte Anfragen automatisch. Für andere Funktionen können SieKonfigurieren Sie LambdaAblaufverfolgungs-eingehende Anforderungen. In beiden Fällen erstellt Lambda das Segment und stellt es dem X-Ray SDK zur Verfügung.

> **Note**
>
> Auf Lambda ist das X-Ray-SDK optional. Wenn Sie es nicht in Ihrer Funktion verwenden, enthält Ihre Service-Map immer noch einen Knoten für den Lambda-Dienst und einen für jede Lambda-Funktion. Durch Hinzufügen des SDK können Sie Ihren Funktionscode so programmieren, dass Untersegmente zum von Lambda aufgezeichneten Funktionssegment hinzugefügt werden. Weitere Informationen finden Sie unter AWS Lambda und AWS X-Ray.

Als Nächstes umschließen Sie den Client mit einem Aufruf der AWS-Funktion. Dieser Schritt stellt sicher, dass X-Ray Aufrufe jeder Client-Methode instrumentiert. Sie können auch Aufrufe von SQL-Datenbanken instrumentieren.

Nachdem Sie die Verwendung des SDK begonnen haben, können Sie das Verhalten durchKonfigurieren des Recorders und der Middlewareaus. Sie können Plugins zum Festhalten von Daten über die Datenverarbeitungsressourcen, auf denen Ihre Anwendung ausgeführt wird, hinzufügen, das Samplingverhalten durch Samplingregeln anpassen und Protokollebenen einrichten, um mehr oder weniger Informationen von dem SDK in Ihren Anwendungsprotokollen zu sehen.

Zeichnen Sie zusätzliche Informationen zu Anforderungen und den Aufgaben, die Ihre Anwendung ausführt, in Anmerkungen und Metadaten auf. Anmerkungen sind einfache Schlüsselwertpaare, die für die Verwendung mit Filterausdrücken indiziert werden, damit Sie nach Ablaufverfolgen mit bestimmten Daten suchen können. Metadateneinträge sind weniger einschränkend und können ganze Objekte und Arrays aufzeichnen – alle Daten, die in eine JSON zusammengefasst werden können.

> **Anmerkungen und Metadaten**
>
> Anmerkungen und Metadaten sind beliebiger Text, den Sie Segmenten mit dem X-Ray-SDK hinzufügen. Anmerkung werden für die Verwendung mit Filterausdrücken indiziert.

Metadaten werden nicht indiziert, können aber im Rohsegment mit der X-Ray-Konsole oder API angezeigt werden. Jeder, dem Sie Lesezugriff auf X-Ray gewähren, kann diese Daten anzeigen.

Wenn Sie viele instrumentierten Clients in Ihrem Code haben, kann ein einzelnes Anforderungssegmente viele Untersegmente enthalten, eines für jeden Aufruf mit einem instrumentierten Client. Sie können Untersegmente organisieren und gruppieren, indem Sie Client-Aufrufe in benutzerdefinierten Untersegmenten zusammenfassen. Sie können ein benutzerdefiniertes Untersegment für eine ganze Funktion oder eine Code-Abschnitt erstellen und Metadaten und Anmerkungen im Untersegment festhalten, anstatt alles im übergeordneten Segment aufzuzeichnen.

## Voraussetzungen

Das X-Ray-SDK SDK for Go Go benötigt Go 1.9 oder höher.

Das SDK hängt von den folgenden Bibliotheken zur Kompilierungs- und Laufzeit ab:

- AWSSDK for Go Version 1.10.0 oder höher

Diese Abhängigkeiten werden in der `README.md`-Datei des SDK angegeben.

## Referenzdokumentation

Nachdem Sie das SDK heruntergeladen haben, erstellen und hosten Sie die Dokumentation lokal, um sie in einem Webbrowser anzuzeigen.

So zeigen Sie die Referenzdokumentation an

1. Navigieren Sie zum (Linux- oder Mac-)Verzeichnis `$GOPATH/src/github.com/aws/aws-xray-sdk-go` oder zum (Windows-)Ordner `%GOPATH%\src\github.com\aws\aws-xray-sdk-go`
2. Führen Sie den Befehl `godoc` aus.

   ```
   $ godoc -http=:6060
   ```

3. Öffnen Sie einen Browser unter `http://localhost:6060/pkg/github.com/aws/aws-xray-sdk-go/`.

# Konfigurieren des X-Ray-SDK für Go

Sie können die Konfiguration für das X-Ray SDK for Go über Umgebungsvariablen angeben, indem Sie sie `Configure` mit einem `Config` Objekt aufrufen oder Standardwerte annehmen. Umgebungsvariablen haben Vorrang vor `Config`-Werten, die wiederum Vorrang vor Standardwerten haben.

Sections

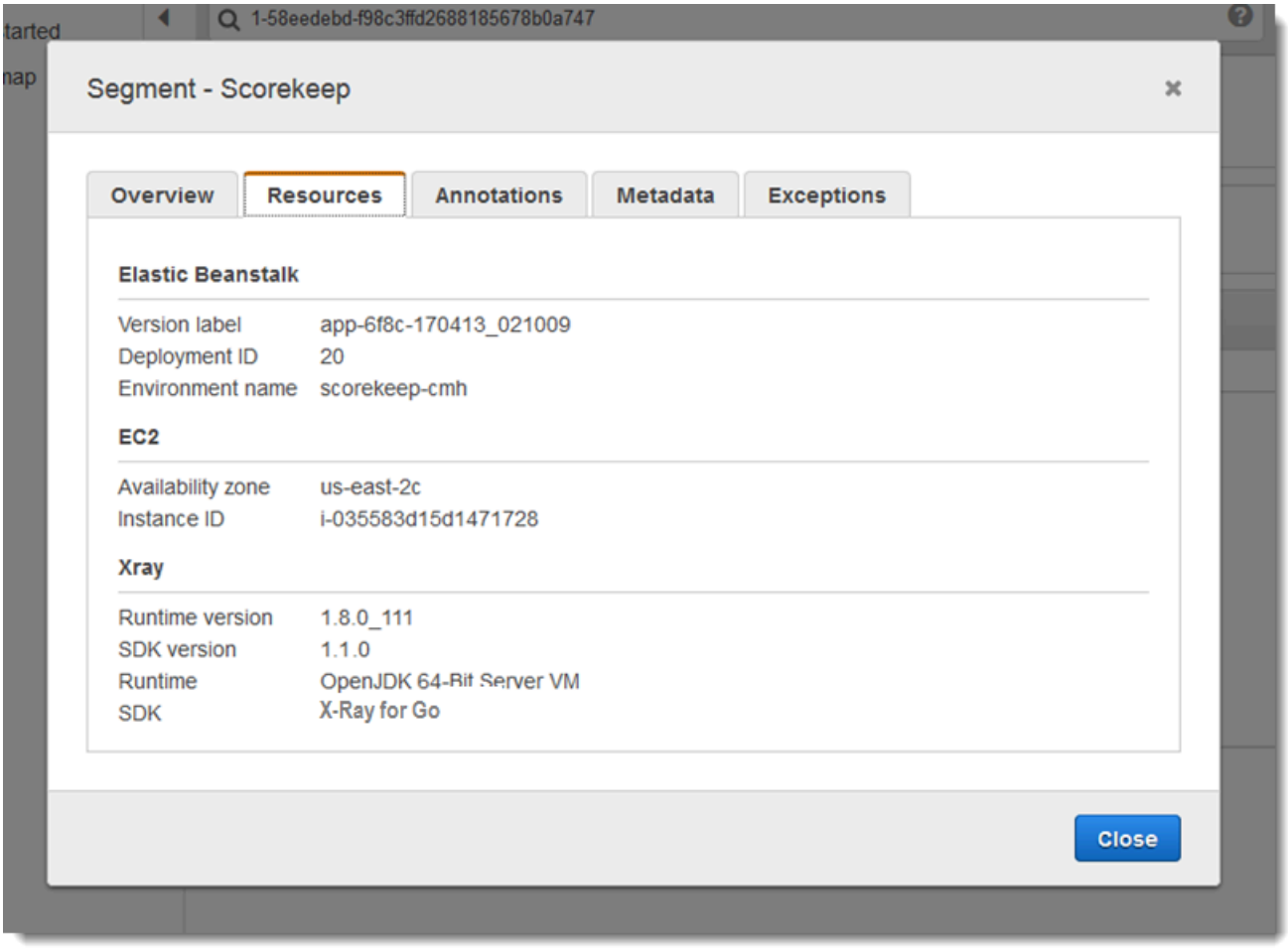


## Service-Plugins

Wird verwendet`plugins`, um Informationen über den Dienst aufzuzeichnen, der Ihre Anwendung hostet.

Plug-ins

- Amazon EC2 — `EC2Plugin` fügt die Instance-ID, die Availability Zone und die CloudWatch Logs-Gruppe hinzu.
- Elastic Beanstalk — `ElasticBeanstalkPlugin` fügt den Umgebungsnamen, die Versionsbezeichnung und die Bereitstellungs-ID hinzu.
- Amazon ECS — `ECSPlugin` fügt die Container-ID hinzu.




Um ein Plugin zu verwenden, importieren Sie eines der folgenden Pakete.

```
"github.com/aws/aws-xray-sdk-go/awsplugins/ec2"
"github.com/aws/aws-xray-sdk-go/awsplugins/ecs"
"github.com/aws/aws-xray-sdk-go/awsplugins/beanstalk"
```

Jedes Plugin hat einen expliziten `Init()`-Funktionsaufruf, der das Plugin lädt.

Example ec2.Init()

```
import (
 "os"

 "github.com/aws/aws-xray-sdk-go/awsplugins/ec2"
 "github.com/aws/aws-xray-sdk-go/xray"
)

func init() {
```

```
  // conditionally load plugin
  if os.Getenv("ENVIRONMENT") == "production" {
    ec2.Init()
  }

  xray.Configure(xray.Config{
    ServiceVersion: "1.2.3",
  })
}
```

Das SDK verwendet auch Plugin-Einstellungen, um das `origin` Feld im Segment festzulegen. Dies gibt den AWS Ressourcentyp an, auf dem Ihre Anwendung ausgeführt wird. Wenn Sie mehrere Plugins verwenden, verwendet das SDK die folgende Auflösungsreihenfolge, um den Ursprung zu bestimmen: ElasticBeanstalk > EKS > ECS > EC2.

Samplingregeln

Das SDK verwendet die Sampling-Regeln, die Sie in der X-Ray-Konsole definieren, um zu bestimmen, welche Anfragen aufgezeichnet werden sollen. Die Standardregel verfolgt die erste Anfrage jede Sekunde und fünf Prozent aller weiteren Anfragen aller Dienste, die Traces an X-Ray senden. Erstellen Sie zusätzliche Regeln in der X-Ray-Konsole, um die Menge der aufgezeichneten Daten für jede Ihrer Anwendungen anzupassen.

Das SDK wendet benutzerdefinierte Regeln in der Reihenfolge an, in der sie definiert sind. Wenn eine Anfrage mehreren benutzerdefinierten Regeln entspricht, wendet das SDK nur die erste Regel an.

> **Note**
>
> Wenn das SDK X-Ray nicht erreichen kann, um Sampling-Regeln abzurufen, kehrt es zu einer lokalen Standardregel zurück, die die erste zu Beginn jeder Sekunde empfangene Anfrage und fünf Prozent aller zusätzlichen Anfragen pro Host festlegt. Dies kann passieren, wenn der Host nicht berechtigt ist, Sampling-APIs aufzurufen, oder wenn er keine Verbindung zum X-Ray-Daemon herstellen kann, der als TCP-Proxy für API-Aufrufe durch das SDK fungiert.

Sie können das SDK auch so konfigurieren, dass Sampling-Regeln aus einem JSON-Dokument geladen werden. Das SDK kann lokale Regeln als Backup für Fälle verwenden, in denen X-Ray Sampling nicht verfügbar ist, oder ausschließlich lokale Regeln verwenden.

Example sampling-rules.json

```
{
  "version": 2,
  "rules": [
    {
      "description": "Player moves.",
      "host": "*",
      "http_method": "*",
      "url_path": "/api/move/*",
      "fixed_target": 0,
      "rate": 0.05
    }
  ],
  "default": {
    "fixed_target": 1,
    "rate": 0.1
  }
}
```

In diesem Beispiel werden eine benutzerdefinierte Regel und eine Standardregel definiert. Die benutzerdefinierte Regel wendet eine Stichprobenrate von fünf Prozent an, ohne dass eine Mindestanzahl von Anfragen für Pfade verfolgt werden muss, unter `/api/move/` denen Pfade verfolgt werden müssen. Die Standardregel verfolgt die erste Anfrage jede Sekunde und 10 Prozent der weiteren Anfragen.

Der Nachteil der lokalen Definition von Regeln besteht darin, dass das feste Ziel von jeder Instanz des Rekorders unabhängig angewendet wird, anstatt vom X-Ray-Dienst verwaltet zu werden. Wenn Sie mehr Hosts bereitstellen, wird die feste Rate vervielfacht, wodurch es schwieriger wird, die Menge der aufgezeichneten Daten zu kontrollieren.

Wenn aktiviert AWS Lambda, können Sie die Samplerate nicht ändern. Wenn Ihre Funktion von einem instrumentierten Dienst aufgerufen wird, werden Aufrufe, die Anfragen generierten, die von diesem Dienst abgetastet wurden, von Lambda aufgezeichnet. Wenn aktives Tracing aktiviert ist und kein Tracing-Header vorhanden ist, trifft Lambda die Stichprobenentscheidung.

Um Sicherungsregeln bereitzustellen, verweisen Sie mit `NewCentralizedStrategyWithFilePath` auf die lokale Sampling-JSON-Datei.

Example main.go — Lokale Stichprobenregel

```
s, _ := sampling.NewCentralizedStrategyWithFilePath("sampling.json") // path to local
 sampling json
xray.Configure(xray.Config{SamplingStrategy: s})
```

Um nur lokale Regeln zu verwenden, verweisen Sie mit `NewLocalizedStrategyFromFilePath` auf die lokale Sampling-JSON-Datei.

Example main.go — Sampling deaktivieren

```
s, _ := sampling.NewLocalizedStrategyFromFilePath("sampling.json") // path to local
 sampling json
xray.Configure(xray.Config{SamplingStrategy: s})
```

Protokollierung

> **Note**
>
> Die `xray.Config{}`-Felder `LogLevel` und `LogFormat` sind ab Version 1.0.0-rc.10 veraltet.

X-Ray verwendet die folgende Schnittstelle für die Protokollierung. Der Standardlogger schreibt in `stdout` bei `LogLevelInfo` und höher.

```
type Logger interface {
 Log(level LogLevel, msg fmt.Stringer)
}

const (
 LogLevelDebug LogLevel = iota + 1
 LogLevelInfo
 LogLevelWarn
 LogLevelError
)
```

Example Schreibvorgänge in **`io.Writer`**

```
xray.SetLogger(xraylog.NewDefaultLogger(os.Stderr, xraylog.LogLevelError))
```

Umgebungsvariablen

Sie können Umgebungsvariablen verwenden, um das X-Ray SDK for Go zu konfigurieren. Das SDK unterstützt die folgenden Variablen.

- `AWS_XRAY_CONTEXT_MISSING`— Stellen Sie diese Option ein`RUNTIME_ERROR`, um Ausnahmen auszulösen, wenn Ihr instrumentierter Code versucht, Daten aufzuzeichnen, obwohl kein Segment geöffnet ist.

  Zulässige Werte

  - `RUNTIME_ERROR`— Löst eine Laufzeitausnahme aus.
  - `LOG_ERROR`— Einen Fehler protokollieren und fortfahren (Standard).
  - `IGNORE_ERROR`— Fehler ignorieren und fortfahren.

  Fehler im Zusammenhang mit fehlenden Segmenten oder Untersegmenten können auftreten, wenn Sie versuchen, einen instrumentierten Client in Startcode zu verwenden, der ausgeführt wird, wenn keine Anfrage geöffnet ist, oder in Code, der einen neuen Thread erzeugt.
- `AWS_XRAY_TRACING_NAME`— Legen Sie den Dienstnamen fest, den das SDK für Segmente verwendet.
- `AWS_XRAY_DAEMON_ADDRESS`— Legt den Host und den Port des X-Ray-Daemon-Listeners fest. Standardmäßig sendet das SDK Trace-Daten an. `127.0.0.1:2000` Verwenden Sie diese Variable, wenn Sie den Daemon so konfiguriert haben, dass er auf einem anderen Port lauscht, oder wenn er auf einem anderen Host läuft.

Umgebungsvariablen überschreiben äquivalente Werte im Code.

Verwenden von „Configure“ (konfigurieren)

Sie können das X-Ray SDK for Go auch mit `Configure` dieser Methode konfigurieren. `Configure`akzeptiert ein Argument, ein `Config` Objekt, mit den folgenden optionalen Feldern.

DaemonAddr

Diese Zeichenfolge gibt den Host und den Port des X-Ray-Daemon-Listeners an. Falls nicht angegeben, verwendet X-Ray den Wert der `AWS_XRAY_DAEMON_ADDRESS` Umgebungsvariablen. Wenn dieser Wert nicht festgelegt ist, wird "127.0.0.1:2000" verwendet.

ServiceVersion

Diese Zeichenfolge legt die Service-Version fest. Falls nicht angegeben, verwendet X-Ray die leere Zeichenfolge („“).

SamplingStrategy

Das `SamplingStrategy`-Objekt legt fest, welche Ihrer Anwendungsaufrufe verfolgt werden. Falls nicht angegeben, verwendet X-Ray a`LocalizedSamplingStrategy`, was die Strategie wie unter definiert verwendet`xray/resources/DefaultSamplingRules.json`.

StreamingStrategy

Dieses `StreamingStrategy` Objekt gibt an, ob ein Segment gestreamt werden soll, wenn der RequiresStreamingWert true zurückgegeben wird. Falls nicht angegeben, verwendet X-Ray eine, `DefaultStreamingStrategy` die ein abgetastetes Segment streamt, wenn die Anzahl der Untersegmente größer als 20 ist.

ExceptionFormattingStrategy

Dieses `ExceptionFormattingStrategy`-Objekt legt fest, auf welche Weise verschiedene Ausnahmen gehandhabt werden sollen. Falls nicht angegeben, verwendet X-Ray a `DefaultExceptionFormattingStrategy` mit einem `XrayError` of-Type`error`, die Fehlermeldung und Stack-Trace.

## Instrumentieren von eingehenden HTTP-Anforderungen mit dem X-Ray-SDK SDK for Go

Sie können mit dem X-Ray-SDK eingehende HTTP-Anforderungen verfolgen, die Ihre Anwendung auf einer EC2 Instance in Amazon EC2 verarbeitet.AWS Elastic Beanstalkoder Amazon ECS.

Verwenden Sie `xray.Handler`, um eingehende HTTP-Anforderungen zu instrumentieren. Das X-Ray-SDK SDK for Go implementiert die Standard-Go-Bibliothek`http.Handler`-Schnittstelle im`xray.Handler`Klasse zum Abfangen von Webanfragen. Die `xray.Handler`-Klasse umschließt den bereitgestellten `http.Handler` mit `xray.Capture`, wobei der Anfragekontext genutzt, eingehende Header analysiert und bei Bedarf Antwort-Header hinzugefügt werden, und legt HTTP-spezifische Ablaufverfolgungsfelder fest.

Wenn Sie diese Klasse zur Verarbeitung von HTTP-Anforderungen und -Antworten verwenden, erzeugt das X-Ray-SDK SDK for Go ein Segment für jede gesampelte Anfrage. Dieses Segment

umfasst Dauer, Methode und Status der HTTP-Anforderung. Die zusätzliche Instrumentierung schafft Untersegmente zu diesem Segment.

> **Note**
>
> FürAWS Lambdafür jede gesampelte Anfrage erzeugt Lambda ein Segment. Weitere Informationen finden Sie unter AWS Lambda und AWS X-Ray.

Das folgende Beispiel fängt Anfragen auf Port 8000 ab und gibt "Hello!" als Antwort zurück. Es erzeugt das Segment `myApp` und instrumentiert Aufrufe über jede beliebige Anwendung.

Example main.go

```
func main() {
  http.Handle("/", xray.Handler(xray.NewFixedSegmentNamer("MyApp"),
 http.HandlerFunc(func(w http.ResponseWriter, r *http.Request) {
    w.Write([]byte("Hello!"))
  })))

  http.ListenAndServe(":8000", nil)
}
```

Jedes Segment hat einen Namen, der Ihre Anwendung in der Service-Map identifiziert. Das Segment kann statisch benannt werden, oder Sie können das SDK so konfigurieren, dass es basierend auf dem Host-Header in der eingehenden Anforderung dynamisch benennt. Mit der dynamischen Benennung können Sie Traces basierend auf dem Domänennamen in der Anforderung gruppieren und einen Standardnamen anwenden, wenn der Name nicht mit einem erwarteten Muster übereinstimmt (z. B. wenn der Host-Header gefälscht ist).

> **Weitergeleitete Anfragen**
>
> Wenn ein Load Balancer oder ein anderer Vermittler eine Anfrage an Ihre Anwendung weiterleitet, übernimmt X-Ray die Client-IP von der`X-Forwarded-For`Header in der Anforderung statt von der Quell-IP im IP-Paket. Die Client-IP, die für eine weitergeleitete Anfrage aufgezeichnet wird, kann gefälscht werden und sollte daher nicht vertrauenswürdig sein.

Wenn eine Anforderung weitergeleitet wird, legt das SDK ein zusätzliches Feld im Segment fest, um dies anzugeben. Wenn das Segment das Feld enthält`x_forwarded_for`eingestellt auf`true`wurde die Client-IP von der`X-Forwarded-For`Header in der HTTP-Anforderung.

Der Handler erzeugt für jede eingehende Anfrage ein Segment mit einem `http`-Block mit folgenden Informationen:

- HTTP method (HTTP-Methode)— HOLEN, POSTEN, PUTEN, LÖSCHEN usw.
- Client-Adresse— Die IP-Adresse des Clients, der die Anforderung gesendet hat.
- Antwortcode— Der HTTP-Antwortcode für die abgeschlossene Anfrage.
- Timing— Die Startzeit (zu der die Anforderung empfangen wurde) und die Endzeit (zu der die Antwort gesendet wurde).
- Benutzer-Agent— Die`user-agent`aus der Anfrage.
- Inhaltslänge— Die`content-length`aus der Antwort.

### Konfiguration einer Segmentbenennungsstrategie

AWS X-Rayverwendet einService-Nameum Ihre Anwendung zu identifizieren und von den anderen Anwendungen, Datenbanken, externen APIs zu unterscheiden undAWS-Ressourcen, die Ihre Anwendung verwendet. Wenn das X-Ray SDK Segmente für eingehende Anfragen generiert, zeichnet es den Dienstnamen Ihrer Anwendung im Segment aufName-Feldaus.

Das X-Ray SDK kann Segmente nach dem Hostnamen im HTTP-Anforderungsheader benennen. Dieser Header kann jedoch gefälscht werden, was zu unerwarteten Knoten in Ihrer Service-Map führen kann. Um zu verhindern, dass das SDK Segmente aufgrund von Anforderungen mit gefälschten Host-Headern falsch benennt, müssen Sie einen Standardnamen für eingehende Anfragen angeben.

Wenn Ihre Anwendung Anfragen für mehrere Domänen bereitstellt, können Sie das SDK so konfigurieren, dass es eine dynamische Benennungsstrategie verwendet, die dies in Segmentnamen widerspiegelt. Eine dynamische Benennungsstrategie ermöglicht es dem SDK, den Hostnamen für Anfragen zu verwenden, die einem erwarteten Muster entsprechen, und den Standardnamen auf Anfragen anzuwenden, die dies nicht tun.

Sie haben beispielsweise möglicherweise eine einzige Anwendung, die Anforderungen an drei Subdomains verarbeitet.`www.example.com`,`api.example.com`, und`static.example.com`aus. Sie können eine dynamische Benennungsstrategie mit dem Muster verwenden`*.example.com`um

Segmente für jede Subdomain mit einem anderen Namen zu identifizieren, was zu drei Serviceknoten auf der Service-Map führt. Wenn Ihre Anwendung Anfragen mit einem Hostnamen erhält, der nicht mit dem Muster übereinstimmt, wird auf der Service-Map ein vierter Knoten mit einem von Ihnen angegebenen Fallback-Namen angezeigt.

Wenn Sie denselben Namen für alle Anfragesegmente verwenden möchten, geben Sie bei der Erstellung des Handlers den Namen Ihrer Anwendung ein, wie im vorherigen Abschnitt gezeigt.

> **Note**
>
> Sie können den mit der `AWS_XRAY_TRACING_NAME`-Umgebungsvariablen in Code definierten standardmäßigen Dienstnamen überschreiben.

Eine dynamische Benennungsstrategie definiert ein Muster, dem Hostnamen entsprechen sollten, sowie einen Standardnamen, der verwendet wird, wenn der Hostname in der HTTP-Anforderung nicht mit diesem Muster übereinstimmt. Um Segmente dynamisch zu benennen, verwenden Sie `NewDynamicSegmentNamer` zur Konfiguration des Standardnamens und Musters zum Abgleich.

Example main.go

Wenn der Hostname in der Anforderung dem Muster `*.example.com` entspricht, verwenden Sie den Hostnamen. Verwenden Sie andernfalls `MyApp`.

```
func main() {
  http.Handle("/", xray.Handler(xray.NewDynamicSegmentNamer("MyApp", "*.example.com"),
 http.HandlerFunc(func(w http.ResponseWriter, r *http.Request) {
    w.Write([]byte("Hello!"))
  })))

  http.ListenAndServe(":8000", nil)
}
```

## Nachverfolgen von AWS SDK-Aufrufen mit dem X-Ray SDK for Go

Wenn Ihre Anwendung Aufrufe an tätigt, AWS-Services um Daten zu speichern, in eine Warteschlange zu schreiben oder Benachrichtigungen zu senden, verfolgt das X-Ray SDK for Go die Aufrufe nachgelagert in Teilsegmenten . Nachverfolgte AWS-Services und Ressourcen, auf die Sie innerhalb dieser Services zugreifen (z. B. ein Amazon S3-Bucket oder eine Amazon SQS-

Warteschlange), werden auf der Ablaufverfolgungskarte in der X-Ray-Konsole als nachgelagerte Knoten angezeigt.

Um AWS SDK-Clients zu verfolgen, umschließen Sie das Client-Objekt mit dem `xray.AWS()`-Aufruf, wie im folgenden Beispiel dargestellt.

Example main.go

```
var dynamo *dynamodb.DynamoDB
func main() {
  dynamo = dynamodb.New(session.Must(session.NewSession()))
  xray.AWS(dynamo.Client)
}
```

Wenn Sie dann den AWS SDK-Client nutzen, verwenden Sie die Version `withContext` der Aufrufmethode und übergeben Sie den `context` aus dem `http.Request`-Objekt, das an den Handler übergeben wurde.

Example main.go – AWS SDK-Aufruf

```
func listTablesWithContext(ctx context.Context) {
  output := dynamo.ListTablesWithContext(ctx, &dynamodb.ListTablesInput{})
  doSomething(output)
}
```

Für alle Services können Sie den Namen der API mit dem Namen in der X-Ray-Konsole sehen. Für eine Teilmenge von Services fügt das X-Ray-SDK dem Segment Informationen hinzu, um eine höhere Granularität in der Service-Übersicht zu gewährleisten.

Wenn Sie beispielsweise einen Aufruf mit einem instrumentierten DynamoDB-Client tätigen, fügt das SDK dem Segment den Tabellennamen für Aufrufe hinzu, die auf eine Tabelle abzielen. In der Konsole wird jede Tabelle als separater Knoten in der Service-Übersicht angezeigt, mit einem generischen DynamoDB-Knoten für Aufrufe, die keine Tabelle anvisieren.

Example Untersegment für einen Aufruf an DynamoDB zum Speichern eines Elements

```
{
  "id": "24756640c0d0978a",
  "start_time": 1.480305974194E9,
  "end_time": 1.4803059742E9,
  "name": "DynamoDB",
```

```
  "namespace": "aws",
  "http": {
    "response": {
      "content_length": 60,
      "status": 200
    }
  },
  "aws": {
    "table_name": "scorekeep-user",
    "operation": "UpdateItem",
    "request_id": "UBQNSO5AEM8T4FDA4RQDEB94OVTDRVV4K4HIRGVJF66Q9ASUAAJG",
  }
}
```

Wenn Sie auf benannte Ressourcen zugreifen, werden durch Aufrufe der folgenden Services weitere Knoten in der Service-Übersicht erstellt. Durch Aufrufe, die keinen bestimmten Ressourcen gelten, wird ein generischer Knoten für den Service erstellt.

- Amazon DynamoDB – Tabellenname
- Amazon Simple Storage Service – Bucket- und Schlüsselname
- Amazon Simple Queue Service – Name der Warteschlange

## Nachverfolgen von Aufrufen nachgelagerter HTTP-Web-Services mit X-Ray-SDK SDK for Go

Wenn Ihre Anwendung Microservices oder öffentliche HTTP-APIs aufruft, können Sie den `xray.Client` zum Instrumentieren dieser Aufrufe als Untersegmente Ihrer Go-Anwendung verwenden, wie im folgenden Beispiel dargestellt, bei dem http-client ein HTTP-Client ist.

Der Client erstellt eine flache Kopie des bereitgestellten HTTP-Clients, standardmäßig`http.DefaultClient`, mit Roundtripper umwickelt mit`xray.RoundTripper`aus.

Example

<caption>main.go — HTTP-Client</caption>

```
myClient := xray.Client(http-client)
```

<caption>main.go — Verfolgen Sie den nachgelagerten HTTP-Aufruf mit der ctxhttp-Bibliothek</caption>

Im folgenden Beispiel wird der ausgehende HTTP-Aufruf mit der ctxhttp Library unter`xray.Client`aus`ctx`kann vom Upstream-Aufruf übergeben werden. Dies stellt sicher, dass der vorhandene Segmentkontext verwendet wird. X-Ray lässt beispielsweise nicht zu, dass ein neues Segment innerhalb einer Lambda-Funktion erstellt wird, daher sollte der vorhandene Lambda-Segmentkontext verwendet werden.

```
resp, err := ctxhttp.Get(ctx, xray.Client(nil), url)
```

## Ablaufverfolgen von SQL-Abfragen mit dem X-Ray-SDK SDK for Go

So können Sie SQL-Aufrufe von PostgreSQL oder MySQL rückverfolgen, die `sql.Open`-Aufrufe von `xray.SQLContext` ersetzen, wie im folgenden Beispiel dargestellt. Verwenden Sie URLs anstelle von Konfigurationszeichenfolgen, wenn möglich.

Example main.go

```
func main() {
  db, err := xray.SQLContext("postgres", "postgres://user:password@host:port/db")
  row, err := db.QueryRowContext(ctx, "SELECT 1") // Use as normal
}
```

## Erstellen benutzerdefinierter Untersegmente mit dem X-Ray-SDK SDK for Go

Teilsegmente erweitern ein TraceAbschnittmit Details über die geleistete Arbeit, um eine Anfrage zu stellen. Jedes Mal, wenn Sie einen Aufruf mit einem instrumentierten Client erstellen, zeichnet X-Ray-SDK die in einem Untersegment generierten Informationen auf. Sie können zusätzliche Teilsegmente erstellen, um andere Teilsegmente zu gruppieren, die Leistung eines Codeabschnitts zu messen oder Anmerkungen und Metadaten aufzuzeichnen.

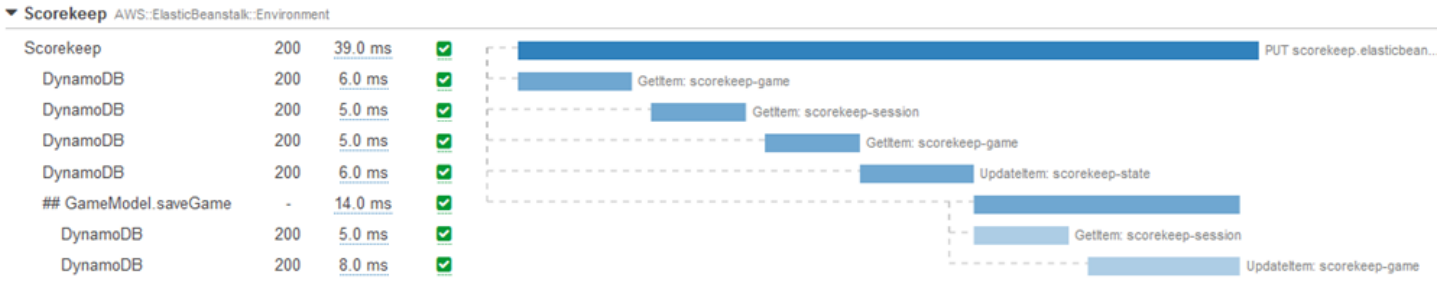
Mit der `Capture`-Methode legen Sie ein Untersegment um eine Funktion herum an.

Example main.go — benutzerdefiniertes Untersegment

```
func criticalSection(ctx context.Context) {
  //this is an example of a subsegment
  xray.Capture(ctx, "GameModel.saveGame", func(ctx1 context.Context) error {
    var err error

    section.Lock()
```

```
    result := someLockedResource.Go()
    section.Unlock()

    xray.AddMetadata(ctx1, "ResourceResult", result)
  })
```

Der folgende Screenshot zeigt ein Beispiel dafür, wie das `saveGame`-Untersegment in Ablaufverfolgungen für die Anwendung `Scorekeep` aussehen kann.




## Fügen Sie mit dem X-Ray SDK for Go Anmerkungen und Metadaten zu Segmenten hinzu

Sie können Anmerkungen und Metadaten verwenden, um zusätzliche Informationen über Anfragen, die Umgebung oder Ihre Anwendung aufzuzeichnen. Sie können Anmerkungen und Metadaten zu den Segmenten hinzufügen, die das X-Ray SDK erstellt, oder zu benutzerdefinierten Untersegmenten, die Sie erstellen.

Anmerkungen sind Schlüssel-Wert-Paare mit Zeichenfolgen-, Zahlen- oder booleschen Werten. Anmerkungen sind für die Verwendung mit Filterausdrücken indexiert. Berücksichtigen Sie Anmerkungen, um Daten zur Gruppierung von Ablaufverfolgungen in der Konsole zu verwenden, oder wenn Sie die `GetTraceSummaries`-API aufrufen.

Metadaten sind Schlüssel-Wert-Paare, die Werte beliebigen Typs enthalten können, einschließlich Objekte und Listen, aber nicht für die Verwendung mit Filterausdrücken indexiert sind. Verwenden Sie Metadaten, um zusätzliche Daten aufzuzeichnen, die Sie im Trace speichern möchten, aber nicht für die Suche verwenden müssen.

Zusätzlich zu Anmerkungen und Metadaten können Sie auch Benutzer-ID-Zeichenfolgen in Segmenten aufzeichnen. Benutzer-IDs werden in einem eigenen Feld aufgezeichnet und zur Suche indiziert.

Sections

### Anmerkungen mit dem X-Ray SDK for Go aufnehmen

Verwenden Sie Anmerkungen, um Informationen zu Segmenten, die zur Suche indiziert werden sollten, aufzuzeichnen.

#### Anmerkung zu Anforderungen

- Schlüssel — Der Schlüssel für eine X-Ray-Anmerkung kann bis zu 500 alphanumerische Zeichen enthalten. Sie können keine anderen Leerzeichen oder Symbole als den Unterstrich (_) verwenden.
- Werte — Der Wert für eine X-Ray-Anmerkung kann bis zu 1.000 Unicode-Zeichen enthalten.
- Die Anzahl der Anmerkungen — Sie können bis zu 50 Anmerkungen pro Spur verwenden.

Um Anmerkungen aufzuzeichnen, rufen Sie `AddAnnotation` mit einer Zeichenfolge auf, die die Metadaten enthält, die Sie dem Segment zuordnen möchten.

```
xray.AddAnnotation(key string, value interface{})
```

Das SDK zeichnet Anmerkungen als Schlüssel-Wert-Paare in einem `annotations`-Objekt im Segmentdokument auf. Wenn `AddAnnotation` zweimal mit demselben Schlüssel aufgerufen wird, werden zuvor aufgezeichnete Werte im selben Segment überschrieben.

Nutzen Sie das `annotations.`*`key`*-Schlüsselwort in einem Filterausdruck, um Ablaufverfolgungen durch Anmerkungen mit bestimmten Werten zu finden.

### Metadaten mit dem X-Ray SDK for Go aufnehmen

Verwenden Sie Metadaten, um Segmentinformationen aufzuzeichnen, die nicht zur Suche indiziert werden müssen.

Um Metadaten aufzuzeichnen, rufen Sie `AddMetadata` mit einer Zeichenfolge auf, die die Metadaten enthält, die Sie dem Segment zuordnen möchten.

```
xray.AddMetadata(key string, value interface{})
```

Aufzeichnen von Benutzer-IDs mit dem X-Ray SDK for Go

Zeichnen Sie Benutzer-IDs in Anforderungssegmenten auf, um den Benutzer zu identifizieren, der die Anforderung gesendet hat.

So zeichnen Sie Benutzer-IDs auf

1. Eine Referenz des aktuellen Segments finden Sie unter `AWSXRay`.

```
import (
  "context"
  "github.com/aws/aws-xray-sdk-go/xray"
)

mySegment := xray.GetSegment(context)
```

2. Rufen Sie `setUser` mit einer Zeichenfolgen-ID des Benutzers auf, der die Anforderung gesendet hat.

```
mySegment.User = "U12345"
```

Nutzen Sie das `user`-Schlüsselwort in einem Filterausdruck, um Ablaufverfolgungen einer Benutzer-ID zu finden.

## Instrumentieren Sie Ihre Bewerbung mit Java

Es gibt zwei Möglichkeiten, Ihre Java Anwendung so zu instrumentieren, dass sie Spuren an X-Ray sendet:

- AWS Distro for OpenTelemetry Java — Eine AWS Distribution, die eine Reihe von Open-Source-Bibliotheken zum Senden korrelierter Metriken und Traces über AWS Distro for Collector an mehrere AWS Überwachungslösungen CloudWatch AWS X-Ray, darunter Amazon und Amazon OpenSearch Service, bereitstellt. OpenTelemetry
- AWS X-Ray SDK für Java — Eine Reihe von Bibliotheken zum Generieren und Senden von Traces an X-Ray über den X-Ray-Daemon.

Weitere Informationen finden Sie unter Wahl zwischen den AWS SDKs Distro for OpenTelemetry und X-Ray.

## AWSDistro fürOpenTelemetryJava

Mit demAWSDistro fürOpenTelemetry(ADOT) Java, Sie können Ihre Anwendungen einmal instrumentieren und korrelierte Metriken und Traces an mehrere sendenAWSMonitoring-Lösungen, einschließlich AmazonCloudWatch,AWS X-Ray, und AmazonOpenSearchBedienung. Die Verwendung von X-Ray mit ADOT erfordert zwei Komponenten: eineOpenTelemetrySDKaktiviert für die Verwendung mit X-Ray und demAWSDistribution fürOpenTelemetrySammlerfür die Verwendung mit X-Ray aktiviert. ADOT Java bietet Unterstützung für automatische Instrumentierung, sodass Ihre Anwendung Traces ohne Codeänderungen senden kann.

Informationen zu den ersten Schritten finden Sie imAWSDistribution fürOpenTelemetryJava-Dokumentation.

Weitere Informationen zur Verwendung desAWSDistribution fürOpenTelemetrymitAWS X-Rayund andereAWS-Services, siheAWSDistribution fürOpenTelemetryoder dieAWSDistribution fürOpenTelemetryDokumentation.

Weitere Informationen zur Sprachunterstützung und Sprachverwendung finden Sie unterAWSBeobachtbarkeit aufGitHub.

## AWS X-Ray SDK für Java

Das X-Ray SDK for Java besteht aus einer Reihe von Bibliotheken für Java Webanwendungen, die Klassen und Methoden zum Generieren und Senden von Trace-Daten an den X-Ray-Daemon bereitstellen. Zu den Trace-Daten gehören Informationen über eingehende HTTP-Anfragen, die von der Anwendung bedient werden, sowie über Aufrufe, die die Anwendung mithilfe des AWS SDK, HTTP-Clients oder eines SQL-Datenbank-Connectors an Downstream-Dienste sendet. Sie können Segmente auch manuell erstellen und Debug-Informationen Anmerkungen und Metadaten hinzufügen.

Das X-Ray SDK for Java ist ein Open-Source-Projekt. Sie können das Projekt verfolgen und Issues und Pull-Requests einreichen unter GitHub: github.com/aws/ aws-xray-sdk-java

Fügen Sie zunächst `AWSXRayServletFilter` als Servlet-Filter hinzu, um eingehende Anforderungen zu verfolgen. Ein Servlet-Filter erstellt ein Segment. Während das Segment geöffnet ist, können Sie die SDK-Client-Methoden nutzen, um dem Segment Informationen hinzuzufügen, Untersegmente zu erstellen und nachgelagerte Aufrufe rückzuverfolgen. Das SDK erfasst auch automatisch Ausnahmen, die Ihre Anwendung ausgibt, während das Segment geöffnet ist.

Ab Version 1.3 können Sie Ihre Anwendung mit aspekt-orientierter Programmierung (AOP) in Spring instrumentieren. Das bedeutet, dass Sie Ihre Anwendung instrumentieren können, während sie läuft AWS, ohne der Laufzeit Ihrer Anwendung Code hinzufügen zu müssen.

Verwenden Sie als Nächstes das X-Ray-SDK SDK for Java, um Ihre AWS SDK for Java Clients zu instrumentieren, indem Sie das SDK Instrumentor-Untermodul in Ihre Build-Konfiguration aufnehmen. Immer wenn Sie mit einem instrumentierten Client einen Downstream AWS-Service oder eine Ressource aufrufen, zeichnet das SDK Informationen über den Anruf in einem Untersegment auf. AWS-Services und die Ressourcen, auf die Sie innerhalb der Services zugreifen, werden in der Trace-Map als Downstream-Knoten angezeigt, sodass Sie Fehler und Drosselungsprobleme bei einzelnen Verbindungen leichter identifizieren können.

Wenn Sie nicht alle Downstream-Aufrufe instrumentieren möchten AWS-Services, können Sie das Instrumentor-Untermodul weglassen und auswählen, welche Clients instrumentiert werden sollen. Instrumentieren Sie einzelne Clients, indem Sie einem `TracingHandler` AWS SDK-Serviceclient einen hinzufügen.

Andere Submodule des X-Ray SDK for Java bieten Instrumentierung für Downstream-Aufrufe von HTTP-Web-APIs und SQL-Datenbanken. Sie können das X-Ray-SDK SDK for Java Java-Versionen von `HTTPClient` und `HTTPClientBuilder` im Apache HTTP-Submodul verwenden, um Apache HTTP-Clients zu instrumentieren. Um SQL-Anfragen zu instrumentieren, fügen Sie der Datenquelle den SDK-Interceptor hinzu.

Nachdem Sie das SDK verwendet haben, passen Sie sein Verhalten an, indem Sie den Rekorder und den Servlet-Filter konfigurieren. Sie können Plugins zum Festhalten von Daten über die Datenverarbeitungsressourcen, auf denen Ihre Anwendung ausgeführt wird, hinzufügen, das Samplingverhalten durch Samplingregeln anpassen und Protokollebenen einrichten, um mehr oder weniger Informationen von dem SDK in Ihren Anwendungsprotokollen zu sehen.

Zeichnen Sie zusätzliche Informationen zu Anforderungen und den Aufgaben, die Ihre Anwendung ausführt, in Anmerkungen und Metadaten auf. Anmerkungen sind einfache Schlüsselwertpaare, die für die Verwendung mit Filterausdrücken indiziert werden, damit Sie nach Ablaufverfolgen mit bestimmten Daten suchen können. Metadateneinträge sind weniger einschränkend und können ganze Objekte und Arrays aufzeichnen – alle Daten, die in eine JSON zusammengefasst werden können.

> **Anmerkungen und Metadaten**
>
> Anmerkungen und Metadaten sind beliebiger Text, den Sie Segmenten mit dem X-Ray SDK hinzufügen. Anmerkungen werden für die Verwendung mit Filterausdrücken indexiert. Metadaten werden nicht indexiert, können aber im Rohsegment mit der X-Ray-Konsole oder API angezeigt werden. Jeder, dem Sie Lesezugriff auf X-Ray gewähren, kann diese Daten einsehen.

Wenn Sie viele instrumentierte Clients in Ihrem Code haben, kann ein einzelnes Anfragesegment viele Untersegmente enthalten, eines für jeden Aufruf mit einem instrumentierten Client. Sie können Untersegmente organisieren und gruppieren, indem Sie Client-Aufrufe in benutzerdefinierten Untersegmenten zusammenfassen. Sie können ein benutzerdefiniertes Untersegment für eine ganze Funktion oder eine Code-Abschnitt erstellen und Metadaten und Anmerkungen im Untersegment festhalten, anstatt alles im übergeordneten Segment aufzuzeichnen.

## Untermodule

Sie können das X-Ray SDK for Java von Maven herunterladen. Das X-Ray-SDK SDK for Java ist je nach Anwendungsfall in Untermodule aufgeteilt und enthält eine Materialliste für die Versionsverwaltung:

- `aws-xray-recorder-sdk-core`(erforderlich) — Grundlegende Funktionen für die Erstellung von Segmenten und die Übertragung von Segmenten. Umfasst `AWSXRayServletFilter` für die Instrumentierung eingehender Anforderungen.
- `aws-xray-recorder-sdk-aws-sdk`— Instrumentiert Aufrufe, die mit AWS SDK for Java Clients AWS-Services getätigt wurden, indem ein Tracing-Client als Request-Handler hinzugefügt wird.
- `aws-xray-recorder-sdk-aws-sdk-v2`— Instrumentiert Aufrufe, die mit Clients der Version AWS SDK for Java 2.2 und höher AWS-Services getätigt wurden, indem ein Tracing-Client als Request-Interzeptor hinzugefügt wird.
- `aws-xray-recorder-sdk-aws-sdk-instrumentor`— Instrumentiert alle `aws-xray-recorder-sdk-aws-sdk` Clients automatisch. AWS SDK for Java
- `aws-xray-recorder-sdk-aws-sdk-v2-instrumentor`— `aws-xray-recorder-sdk-aws-sdk-v2` Instrumentiert alle Clients ab Version AWS SDK for Java 2.2 automatisch.
- `aws-xray-recorder-sdk-apache-http`— Instrumentiert ausgehende HTTP-Aufrufe, die mit Apache HTTP-Clients getätigt wurden.

- `aws-xray-recorder-sdk-spring`— Stellt Interzeptoren für Spring AOP Framework-Anwendungen bereit.
- `aws-xray-recorder-sdk-sql-postgres`— Instrumentiert ausgehende Aufrufe an eine PostgreSQL-Datenbank, die mit JDBC getätigt wurden.
- `aws-xray-recorder-sdk-sql-mysql`— Instrumentiert ausgehende Aufrufe an eine MySQL-Datenbank, die mit JDBC getätigt wurden.
- `aws-xray-recorder-sdk-bom`— Stellt eine Stückliste bereit, anhand derer Sie die Version angeben können, die für alle Submodule verwendet werden soll.
- `aws-xray-recorder-sdk-metrics`— Veröffentlichen Sie CloudWatch Amazon-Metriken ohne Stichproben aus Ihren gesammelten X-Ray-Segmenten.

Wenn Sie Maven oder Gradle verwenden, um Ihre Anwendung zu erstellen, fügen Sie das X-Ray SDK for Java zu Ihrer Build-Konfiguration hinzu.

Eine Referenzdokumentation der Klassen und Methoden des SDK finden Sie unter AWS X-Ray SDK for Java API Reference.

## Voraussetzungen

Das X-Ray SDK for Java benötigt Java 8 oder höher, Servlet API 3, das AWS SDK und Jackson.

Das SDK hängt von den folgenden Bibliotheken zur Kompilierungs- und Laufzeit ab:

- AWS SDK für Java Version 1.11.398 oder höher
- Servlet API 3.1.0

Diese Abhängigkeiten werden in der `pom.xml`-Datei des SDK deklariert und automatisch eingefügt, wenn Sie Maven oder Gradle für die Build-Erstellung verwenden.

Wenn Sie eine Bibliothek verwenden, die im X-Ray SDK for Java enthalten ist, müssen Sie die mitgelieferte Version verwenden. Wenn Sie beispielsweise bereits zur Laufzeit von Jackson abhängen und für diese Abhängigkeit JAR-Dateien in Ihre Bereitstellung aufnehmen, müssen Sie diese JAR-Dateien entfernen, da das SDK JAR eigene Versionen von Jackson-Bibliotheken umfasst.

## Abhängigkeitsmanagement

Das X-Ray SDK for Java ist bei Maven erhältlich:

- Gruppe — `com.amazonaws`
- Artifact — `aws-xray-recorder-sdk-bom`
- Version – `2.11.0`

Wenn Sie Ihre Anwendung mit Maven erstellen, fügen Sie das SDK als Abhängigkeit in Ihrer `pom.xml`-Datei hinzu.

Example pom.xml - Abhängigkeiten

```
<dependencyManagement>
  <dependencies>
    <dependency>
      <groupId>com.amazonaws</groupId>
      <artifactId>aws-xray-recorder-sdk-bom</artifactId>
      <version>2.11.0</version>
      <type>pom</type>
      <scope>import</scope>
    </dependency>
  </dependencies>
</dependencyManagement>
<dependencies>
  <dependency>
    <groupId>com.amazonaws</groupId>
    <artifactId>aws-xray-recorder-sdk-core</artifactId>
  </dependency>
  <dependency>
    <groupId>com.amazonaws</groupId>
    <artifactId>aws-xray-recorder-sdk-apache-http</artifactId>
  </dependency>
  <dependency>
    <groupId>com.amazonaws</groupId>
    <artifactId>aws-xray-recorder-sdk-aws-sdk</artifactId>
  </dependency>
  <dependency>
    <groupId>com.amazonaws</groupId>
    <artifactId>aws-xray-recorder-sdk-aws-sdk-instrumentor</artifactId>
  </dependency>
  <dependency>
    <groupId>com.amazonaws</groupId>
    <artifactId>aws-xray-recorder-sdk-sql-postgres</artifactId>
  </dependency>
  <dependency>
```

```
    <groupId>com.amazonaws</groupId>
    <artifactId>aws-xray-recorder-sdk-sql-mysql</artifactId>
  </dependency>
</dependencies>
```

Für Gradle fügen Sie das SDK als Kompilierungszeitabhängigkeit in Ihrer `build.gradle`-Datei hinzu.

Example build.gradle – Abhängigkeiten

```
dependencies {
  compile("org.springframework.boot:spring-boot-starter-web")
  testCompile("org.springframework.boot:spring-boot-starter-test")
  compile("com.amazonaws:aws-java-sdk-dynamodb")
  compile("com.amazonaws:aws-xray-recorder-sdk-core")
  compile("com.amazonaws:aws-xray-recorder-sdk-aws-sdk")
  compile("com.amazonaws:aws-xray-recorder-sdk-aws-sdk-instrumentor")
  compile("com.amazonaws:aws-xray-recorder-sdk-apache-http")
  compile("com.amazonaws:aws-xray-recorder-sdk-sql-postgres")
  compile("com.amazonaws:aws-xray-recorder-sdk-sql-mysql")
  testCompile("junit:junit:4.11")
}
dependencyManagement {
    imports {
        mavenBom('com.amazonaws:aws-java-sdk-bom:1.11.39')
        mavenBom('com.amazonaws:aws-xray-recorder-sdk-bom:2.11.0')
    }
}
```

Wenn Sie Elastic Beanstalk für die Bereitstellung Ihrer Anwendung verwenden, können Sie Maven oder Gradle verwenden, um bei jeder Bereitstellung eine On-Instance zu erstellen, anstatt ein großes Archiv mit all Ihren Abhängigkeiten zu erstellen und hochzuladen. Für ein Beispiel, bei dem Gradle verwendet wird, sehen Sie sich die Beispielanwendung an.

## AWS X-Ray Agent für automatische Instrumentierung für Java

Der AWS X-Ray Autoinstrumentation-Agent für Java ist eine Tracing-Lösung, die Ihre Java-Webanwendungen mit minimalem Entwicklungsaufwand instrumentiert. Der Agent ermöglicht die Ablaufverfolgung für Servlet-basierte Anwendungen und alle Downstream-Anfragen des Agenten, die mit unterstützten Frameworks und Bibliotheken gestellt wurden. Dazu gehören Downstream-HTTP-Anfragen von Apache, AWS SDK-Anfragen und SQL-Abfragen, die mithilfe eines JDBC-Treibers

gestellt wurden. Der Agent verbreitet den X-Ray-Kontext, einschließlich aller aktiven Segmente und Untersegmente, über Threads hinweg. Alle Konfigurationen und die Vielseitigkeit des X-Ray-SDK sind weiterhin mit dem Java-Agenten verfügbar. Es wurden geeignete Standardeinstellungen ausgewählt, um sicherzustellen, dass der Agent mit minimalem Aufwand arbeitet.

Die X-Ray-Agentenlösung eignet sich am besten für Servlet-basierte Java-Webanwendungsserver mit Request-Response. Wenn Ihre Anwendung ein asynchrones Framework verwendet oder nicht gut als Request-Response-Service modelliert ist, sollten Sie stattdessen eine manuelle Instrumentierung mit dem SDK in Betracht ziehen.

Der X-Ray-Agent wird mit dem Distributed Systems Comprehension Toolkit (DISCO) erstellt. DisCo ist ein Open-Source-Framework zum Erstellen von Java-Agenten, die in verteilten Systemen verwendet werden können. Es ist zwar nicht notwendig, DisCo zu verstehen, um den X-Ray-Agenten verwenden zu können, aber Sie können mehr über das Projekt erfahren, indem Sie die Homepage unter besuchen GitHub. Der X-Ray-Agent ist ebenfalls vollständig als Open Source verfügbar. Um den Quellcode einzusehen, Beiträge zu leisten oder Probleme mit dem Agenten zu äußern, besuchen Sie das entsprechende Repository unter GitHub.

### Beispielanwendung

Die eb-java-scorekeepProbenapplikation ist so angepasst, dass sie mit dem Röntgenmittel instrumentiert werden kann. Dieser Zweig enthält keine Servlet-Filter- oder Rekorderkonfiguration, da diese Funktionen vom Agenten ausgeführt werden. Um die Anwendung lokal oder mithilfe von AWS Ressourcen auszuführen, folgen Sie den Schritten in der Readme-Datei der Beispielanwendung. Die Anweisungen zur Verwendung der Beispiel-App zum Generieren von Röntgenspuren finden Sie im Tutorial der Beispiel-App.

### Erste Schritte

Gehen Sie wie folgt vor, um mit dem X-Ray-Java-Agenten für automatische Instrumentierung in Ihrer eigenen Anwendung zu beginnen.

1. Führen Sie den X-Ray-Daemon in Ihrer Umgebung aus. Weitere Informationen finden Sie unter X-Ray-Daemon.
2. Laden Sie die neueste Version des Agenten herunter. Entpacken Sie das Archiv und notieren Sie sich seinen Speicherort in Ihrem Dateisystem. Sein Inhalt sollte wie folgt aussehen.

```
disco
### disco-java-agent.jar
### disco-plugins
```

```
    ### aws-xray-agent-plugin.jar
    ### disco-java-agent-aws-plugin.jar
    ### disco-java-agent-sql-plugin.jar
    ### disco-java-agent-web-plugin.jar
```

3. Ändern Sie die JVM-Argumente Ihrer Anwendung so, dass sie Folgendes enthalten, wodurch der Agent aktiviert wird. Stellen Sie sicher, dass das `-javaagent` Argument vor dem `-jar` Argument steht, falls zutreffend. Der Prozess zum Ändern von JVM-Argumenten hängt von den Tools und Frameworks ab, die Sie zum Starten Ihres Java-Servers verwenden. Spezifische Anleitungen finden Sie in der Dokumentation Ihres Server-Frameworks.

   ```
   -javaagent:/<path-to-disco>/disco-java-agent.jar=pluginPath=/<path-to-disco>/disco-plugins
   ```

4. Um anzugeben, wie der Name Ihrer Anwendung auf der X-Ray-Konsole angezeigt wird, legen Sie die `AWS_XRAY_TRACING_NAME` Umgebungsvariable oder die `com.amazonaws.xray.strategy.tracingName` Systemeigenschaft fest. Wenn kein Name angegeben wird, wird ein Standardname verwendet.
5. Starten Sie Ihren Server oder Container neu. Eingehende Anfragen und ihre Downstream-Aufrufe werden jetzt verfolgt. Wenn Sie nicht die erwarteten Ergebnisse sehen, finden Sie weitere Informationen unterthe section called “Fehlerbehebung”.

### Konfiguration

Der X-Ray-Agent wird durch eine externe, vom Benutzer bereitgestellte JSON-Datei konfiguriert. Standardmäßig wird diese Datei im Stammverzeichnis des Klassenpfads des Benutzers (z. B. in seinem `resources` Verzeichnis) benannt. `xray-agent.json` Sie können einen benutzerdefinierten Speicherort für die Konfigurationsdatei konfigurieren, indem Sie die `com.amazonaws.xray.configFile` Systemeigenschaft auf den absoluten Dateisystempfad Ihrer Konfigurationsdatei setzen.

Als Nächstes wird ein Beispiel für eine Konfigurationsdatei angezeigt.

```
{
    "serviceName": "XRayInstrumentedService",
    "contextMissingStrategy": "LOG_ERROR",
    "daemonAddress": "127.0.0.1:2000",
    "tracingEnabled": true,
    "samplingStrategy": "CENTRAL",
    "traceIdInjectionPrefix": "prefix",
```

```
    "samplingRulesManifest": "/path/to/manifest",
    "awsServiceHandlerManifest": "/path/to/manifest",
    "awsSdkVersion": 2,
    "maxStackTraceLength": 50,
    "streamingThreshold": 100,
    "traceIdInjection": true,
    "pluginsEnabled": true,
    "collectSqlQueries": false
}
```

Konfigurationsspezifikation

In der folgenden Tabelle werden gültige Werte für jede Eigenschaft beschrieben. Bei Eigenschaftsnamen wird zwischen Groß- und Kleinschreibung unterschieden, bei ihren Schlüsseln jedoch nicht. Bei Eigenschaften, die durch Umgebungsvariablen und Systemeigenschaften außer Kraft gesetzt werden können, ist die Prioritätsreihenfolge immer die Umgebungsvariable, dann die Systemeigenschaft und dann die Konfigurationsdatei. Informationen zu Eigenschaften, die Sie überschreiben können, finden Sie in den Umgebungsvariablen. Alle Felder sind optional.

| Name der Eigenschaft | Typ | Zulässige Werte | Beschreibung | Umgebungsvariable | Systemeigenschaft | Standard |
|---|---|---|---|---|---|---|
| serviceName | String | Jede Zeichenfolge | Der Name Ihres instrumentierten Dienstes, so wie er in der X-Ray-Konsole angezeigt wird. | AWS_XRAY_TRACING_NAME | com.amazonaws.xray.strategy.tracingName | XRayInstrumentedService |
| contextMissingStrategy | String | LOG_ERROR, IGNORE_ERROR | Die Aktion, die der Agent ausführt, wenn er | AWS_XRAY_CONTEXT_MISSING | com.amazonaws.xray.strategy.contextMi | LOG_ERROR |

| Name der Eigenschaft | Typ | Zulässige Werte | Beschreibung | Umgebungsvariable | Systemeigenschaft | Standard |
|---|---|---|---|---|---|---|
| | | | versucht, den X-Ray-Segmentkontext zu verwenden, aber keiner vorhanden ist. | | ssingStrategy | |
| Daemon-Adresse | String | Formatierte IP-Adresse und Port oder Liste von TCP- und UDP-Adressen | Die Adresse, die der Agent für die Kommunikation mit dem X-Ray-Daemon verwendet. | AWS_XRAY_DAEMON_ADDRESS | com.amazonaws.xray.emitter.DaemonAddresse | 127.0.0.1:2000 |
| Ablaufverfolgung aktiviert | Boolesch | Wahr, falsch | Ermöglicht die Instrumentierung durch den X-Ray-Agenten. | AWS_XRAY_TRACING_ENABLED | com.amazonaws.xray.TracingEnabled | TRUE |

| Name der Eigenschaft | Typ | Zulässige Werte | Beschreibung | Umgebungsvariable | Systemeigenschaft | Standard |
|---|---|---|---|---|---|---|
| Strategie für die Probenahme | String | ZENTRAL, LOKAL, KEINER, ALLES | Die vom Agenten verwendete Probenahmestrategie. ALL erfasst alle Anfragen, NONE erfasst keine Anfragen. Siehe Regeln für die Probenahme. | N/A | N/A | ZENTRALE |
| traceIdInjectionPräfix | String | Jede Zeichenfolge | Schließt das angegebene Präfix vor den eingegebenen Trace-IDs in die Protokolle ein. | N/A | N/A | Keine (leere Zeichenfolge) |

| Name der Eigenschaft | Typ | Zulässige Werte | Beschreibung | Umgebungsvariable | Systemeigenschaft | Standard |
|---|---|---|---|---|---|---|
| samplingRulesManifest | String | Ein absoluter Dateipfad | Der Pfad zu einer Datei mit benutzerdefinierten Stichprobenregeln, die als Quelle für Stichprobenregeln für die lokale Stichprobenstrategie oder als Ausweichregeln für die zentrale Strategie verwendet werden soll. | N/A | N/A | DefaultSamplingRules.json |

| Name der Eigenschaft | Typ | Zulässige Werte | Beschreibung | Umgebungsvariable | Systemeigenschaft | Standard |
|---|---|---|---|---|---|---|
| awsServiceHandlerManifest | String | Ein absoluter Dateipfad | Der Pfad zu einer Zulassungsliste für benutzerdefinierte Parameter, die zusätzliche Informationen von AWS SDK-Clients erfasst. | N/A | N/A | DefaultOperationParameterWhitelist.json |
| awsSdkVersion | Ganzzahl | 1, 2 | Version des AWS SDK for Java, das Sie verwenden. Wird ignoriert, wenn `awsServiceHandlerManifest` es nicht auch gesetzt ist. | N/A | N/A | 2 |

| Name der Eigenschaft | Typ | Zulässige Werte | Beschreibung | Umgebungsvariable | Systemeigenschaft | Standard |
|---|---|---|---|---|---|---|
| maxStackTraceLänge | Ganzzahl | Nichtnegative Ganzzahlen | Die maximale Anzahl von Zeilen eines Stack-Trace, die in einem Trace aufgezeichnet werden sollen. | N/A | N/A | 50 |
| Streaming Threshold | Ganzzahl | Nichtnegative Ganzzahlen | Nachdem mindestens so viele Untersegmente geschlossen wurden, werden sie an den Daemon gestreamt, um zu verhindern, dass die Blöcke out-of-band zu groß werden. | N/A | N/A | 100 |

| Name der Eigenschaft | Typ | Zulässige Werte | Beschreibung | Umgebungsvariable | Systemeigenschaft | Standard |
|---|---|---|---|---|---|---|
| traceIdInjection | Boolesch | Wahr, falsch | Aktiviert die X-Ray-Trace-ID-Injektion in Protokolle, wenn die in der Protokollierungskonfiguration beschriebenen Abhängigkeiten und die Konfiguration ebenfalls hinzugefügt werden. Sonst tut nichts. | N/A | N/A | TRUE |

| Name der Eigenschaft | Typ | Zulässige Werte | Beschreibung | Umgebungsvariable | Systemeigenschaft | Standard |
|---|---|---|---|---|---|---|
| Plugins aktiviert | Boolesch | Wahr, falsch | Aktiviert Plugins, die Metadaten über die AWS Umgebungen aufzeichnen, in denen Sie arbeiten. Siehe Plugins. | N/A | N/A | TRUE |
| collectSqlQueries | Boolesch | Wahr, Falsch | Zeichnet SQL-Abfragezeichenfolgen nach bestem Wissen und Gewissen in SQL-Untersegmenten auf. | N/A | N/A | FALSE |

| Name der Eigenschaft | Typ | Zulässige Werte | Beschreibung | Umgebungsvariable | Systemeigenschaft | Standard |
|---|---|---|---|---|---|---|
| Kontext-Propagierung | Boolesch | Wahr, falsch | Gibt automatisch den X-Ray-Kontext zwischen Threads weiter, falls der Wert wahr ist. Andernfalls verwendet Thread Local, um den Kontext zu speichern, und eine manuelle Weitergabe zwischen Threads ist erforderlich. | N/A | N/A | TRUE |

Konfiguration der Protokollierung

Das Log Level des X-Ray-Agenten kann auf die gleiche Weise wie das X-Ray SDK for Java konfiguriert werden. Weitere Informationen Protokollierung zur Konfiguration der Protokollierung mit dem X-Ray SDK for Java finden Sie unter.

### Manuelle Instrumentierung

Wenn Sie zusätzlich zur automatischen Instrumentierung des Agenten eine manuelle Instrumentierung durchführen möchten, fügen Sie das X-Ray SDK als Abhängigkeit zu Ihrem Projekt hinzu. Beachten Sie, dass die unter Tracing Incoming Requests erwähnten benutzerdefinierten Servlet-Filter des SDK nicht mit dem X-Ray-Agenten kompatibel sind.

> **Note**
>
> Sie müssen die neueste Version des X-Ray-SDK verwenden, um die manuelle Instrumentierung durchzuführen und gleichzeitig den Agenten zu verwenden.

Wenn Sie in einem Maven-Projekt arbeiten, fügen Sie Ihrer `pom.xml` Datei die folgenden Abhängigkeiten hinzu.

```
<dependencies>
  <dependency>
    <groupId>com.amazonaws</groupId>
    <artifactId>aws-xray-recorder-sdk-core</artifactId>
    <version>2.11.0</version>
  </dependency>
  </dependencies>
```

Wenn Sie in einem Gradle-Projekt arbeiten, fügen Sie Ihrer `build.gradle` Datei die folgenden Abhängigkeiten hinzu.

```
implementation 'com.amazonaws:aws-xray-recorder-sdk-core:2.11.0'
```

Sie können während der Verwendung des Agenten zusätzlich zu Anmerkungen, Metadaten und Benutzer-IDs benutzerdefinierte Untersegmente hinzufügen, genau wie beim normalen SDK. Der Agent verteilt den Kontext automatisch zwischen Threads, sodass bei der Arbeit mit Multithread-Anwendungen keine Behelfslösungen für die Übertragung des Kontexts erforderlich sein sollten.

### Fehlerbehebung

Da der Agent über eine vollautomatische Instrumentierung verfügt, kann es schwierig sein, die Ursache eines Problems zu ermitteln, wenn Probleme auftreten. Wenn der X-Ray-Agent bei Ihnen nicht wie erwartet funktioniert, überprüfen Sie die folgenden Probleme und Lösungen. Der X-Ray-Agent und das SDK verwenden Jakarta Commons Logging (JCL). Um die Logging-Ausgabe zu

sehen, stellen Sie sicher, dass sich eine Bridge, die JCL mit Ihrem Logging-Backend verbindet, im Klassenpfad befindet, wie im folgenden Beispiel: oder. `log4j-jcl` `jcl-over-slf4j`

Problem: Ich habe den Java-Agenten in meiner Anwendung aktiviert, sehe aber nichts auf der X-Ray-Konsole

Läuft der X-Ray-Daemon auf demselben Computer?

Falls nicht, lesen Sie in der Dokumentation zum X-Ray-Daemon nach, um ihn einzurichten.

Sehen Sie in Ihren Anwendungsprotokollen eine Meldung wie „Initialisierung des X-Ray Agent Recorders“?

Wenn Sie den Agenten korrekt zu Ihrer Anwendung hinzugefügt haben, wird diese Meldung auf der INFO-Ebene protokolliert, wenn Ihre Anwendung gestartet wird, bevor sie Anfragen entgegennimmt. Wenn diese Meldung nicht angezeigt wird, läuft der Java-Agent nicht zusammen mit Ihrem Java-Prozess. Stellen Sie sicher, dass Sie alle Einrichtungsschritte korrekt und ohne Tippfehler ausgeführt haben.

Sehen Sie in Ihren Anwendungsprotokollen mehrere Fehlermeldungen mit der Aufschrift „Ausnahme fehlt AWS X-Ray im Kontext unterdrücken“?

Diese Fehler treten auf, weil der Agent versucht, Downstream-Anfragen wie AWS SDK-Anfragen oder SQL-Abfragen zu instrumentieren, der Agent aber nicht in der Lage war, automatisch ein Segment zu erstellen. Wenn Sie viele dieser Fehler sehen, ist der Agent möglicherweise nicht das beste Tool für Ihren Anwendungsfall und Sie sollten stattdessen eine manuelle Instrumentierung mit dem X-Ray-SDK in Betracht ziehen. Alternativ können Sie die X-Ray SDK-Debug-Logs aktivieren, um den Stack-Trace zu sehen, wo die kontextlosen Ausnahmen auftreten. Sie können diese Teile Ihres Codes mit benutzerdefinierten Segmenten umschließen, wodurch diese Fehler behoben werden sollten. Ein Beispiel für das Umschließen von Downstream-Anfragen mit benutzerdefinierten Segmenten finden Sie im Beispielcode unter Instrumentieren von Startcode.

Problem: Einige der Segmente, die ich erwarte, erscheinen nicht auf der X-Ray-Konsole

Verwendet Ihre Anwendung Multithreading?

Wenn einige Segmente, von denen Sie erwarten, dass sie erstellt werden, nicht in Ihrer Konsole erscheinen, könnte dies an Hintergrund-Threads in Ihrer Anwendung liegen. Wenn Ihre Anwendung Aufgaben mithilfe von Hintergrund-Threads ausführt, die „Fire and Forget“ sind, wie z. B. einen einmaligen Aufruf einer Lambda-Funktion mit dem AWS SDK oder das regelmäßige Abfragen einiger HTTP-Endpunkte, kann dies den Agenten verwirren, während er den Kontext zwischen Threads

verbreitet. Um zu überprüfen, ob dies Ihr Problem ist, aktivieren Sie die X-Ray-SDK-Debug-Logs und suchen Sie nach Meldungen wie: Segment wird nicht ausgegeben, da es Untersegmente in Bearbeitung übergeordnet <NAME >hat. Um dieses Problem zu umgehen, können Sie versuchen, den Hintergrund-Threads beizutreten, bevor Ihr Server zurückkehrt, um sicherzustellen, dass die gesamte in ihnen geleistete Arbeit aufgezeichnet wird. Oder Sie können die `contextPropagation` Konfiguration des Agenten so einstellen, dass die `false` Kontextweiterleitung in Hintergrund-Threads deaktiviert wird. In diesem Fall müssen Sie diese Threads manuell mit benutzerdefinierten Segmenten instrumentieren oder die Ausnahmen ignorieren, die sie erzeugen, wenn der Kontext fehlt.

Haben Sie Stichprobenregeln eingerichtet?

Wenn scheinbar zufällige oder unerwartete Segmente auf der X-Ray-Konsole erscheinen oder die Segmente, von denen Sie erwarten, dass sie sich auf der Konsole befinden, nicht, liegt möglicherweise ein Sampling-Problem vor. Der X-Ray-Agent wendet zentralisierte Stichproben auf alle von ihm erstellten Segmente an und verwendet dabei die Regeln der X-Ray-Konsole. Die Standardregel lautet, dass 1 Segment pro Sekunde plus 5% der nachfolgenden Segmente abgetastet werden. Das bedeutet, dass Segmente, die schnell mit dem Agenten erstellt werden, möglicherweise nicht gesampelt werden. Um dieses Problem zu lösen, sollten Sie auf der X-Ray-Konsole benutzerdefinierte Sampling-Regeln erstellen, die die gewünschten Segmente entsprechend abtasten. Weitere Informationen finden Sie unter Sampling-Regeln konfigurieren unterErkunden Sie die X-Ray-Konsole.

## Konfiguration des X-Ray SDK for Java

Das X-Ray-SDK SDK for Java enthält eine Klasse mit dem Namen`AWSXRay`, die den globalen Rekorder bereitstellt. Dies ist ein `TracingHandler`, mit dem Sie Ihren Code instrumentieren können. Sie können die globale Aufzeichnung so konfigurieren, dass der `AWSXRayServletFilter`, der Segmente für eingehende HTTP-Aufrufe erstellt, angepasst wird.

Sections

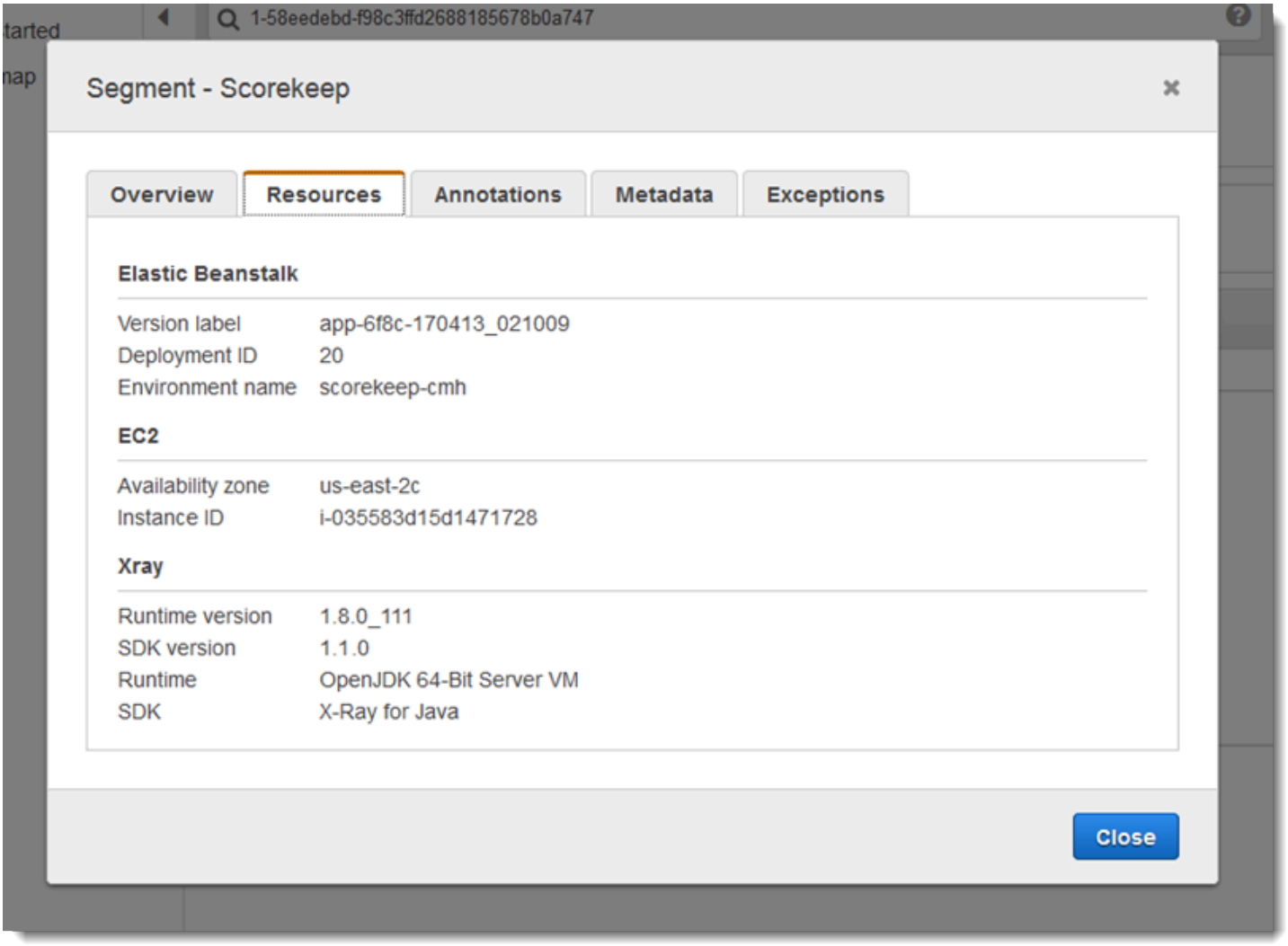
- Service-Plugins
- Samplingregeln
- Protokollierung
- Segment-Listeners
- Umgebungsvariablen
- Systemeigenschaften

Service-Plugins

Wird verwendet`plugins`, um Informationen über den Dienst aufzuzeichnen, der Ihre Anwendung hostet.

Plug-ins

- Amazon EC2 — `EC2Plugin` fügt die Instance-ID, die Availability Zone und die CloudWatch Logs-Gruppe hinzu.
- Elastic Beanstalk — `ElasticBeanstalkPlugin` fügt den Umgebungsnamen, die Versionsbezeichnung und die Bereitstellungs-ID hinzu.
- Amazon ECS — `ECSPlugin` fügt die Container-ID hinzu.
- Amazon EKS — `EKSPlugin` fügt die Container-ID, den Clusternamen, die Pod-ID und die CloudWatch Logs-Gruppe hinzu.




Um ein Plugin zu verwenden, rufen Sie `withPlugin` im `AWSXRayRecorderBuilder` auf.

Example src/main/java/scorekeep/ .java WebConfig — Rekorder

```
import com.amazonaws.xray.AWSXRay;
import com.amazonaws.xray.AWSXRayRecorderBuilder;
import com.amazonaws.xray.plugins.EC2Plugin;
import com.amazonaws.xray.plugins.ElasticBeanstalkPlugin;
import com.amazonaws.xray.strategy.sampling.LocalizedSamplingStrategy;

@Configuration
public class WebConfig {
...
  static {
    AWSXRayRecorderBuilder builder = AWSXRayRecorderBuilder.standard().withPlugin(new
 EC2Plugin()).withPlugin(new ElasticBeanstalkPlugin());
```

```
    URL ruleFile = WebConfig.class.getResource("/sampling-rules.json");
    builder.withSamplingStrategy(new LocalizedSamplingStrategy(ruleFile));

    AWSXRay.setGlobalRecorder(builder.build());
  }
}
```

Das SDK verwendet auch Plugin-Einstellungen, um das Feld für das Segment festzulegen. `origin` Dies gibt den AWS Ressourcentyp an, auf dem Ihre Anwendung ausgeführt wird. Wenn Sie mehrere Plugins verwenden, verwendet das SDK die folgende Auflösungsreihenfolge, um den Ursprung zu bestimmen: ElasticBeanstalk > EKS > ECS > EC2.

Samplingregeln

Das SDK verwendet die Sampling-Regeln, die Sie in der X-Ray-Konsole definieren, um zu bestimmen, welche Anfragen aufgezeichnet werden sollen. Die Standardregel verfolgt die erste Anfrage jede Sekunde und fünf Prozent aller weiteren Anfragen aller Dienste, die Traces an X-Ray senden. Erstellen Sie zusätzliche Regeln in der X-Ray-Konsole, um die Menge der aufgezeichneten Daten für jede Ihrer Anwendungen anzupassen.

Das SDK wendet benutzerdefinierte Regeln in der Reihenfolge an, in der sie definiert sind. Wenn eine Anfrage mehreren benutzerdefinierten Regeln entspricht, wendet das SDK nur die erste Regel an.

> **Note**
>
> Wenn das SDK X-Ray nicht erreichen kann, um Sampling-Regeln abzurufen, kehrt es zu einer lokalen Standardregel zurück, die die erste zu Beginn jeder Sekunde empfangene Anfrage und fünf Prozent aller zusätzlichen Anfragen pro Host festlegt. Dies kann passieren, wenn der Host nicht berechtigt ist, Sampling-APIs aufzurufen, oder wenn er keine Verbindung zum X-Ray-Daemon herstellen kann, der als TCP-Proxy für API-Aufrufe durch das SDK fungiert.

Sie können das SDK auch so konfigurieren, dass Sampling-Regeln aus einem JSON-Dokument geladen werden. Das SDK kann lokale Regeln als Backup für Fälle verwenden, in denen X-Ray Sampling nicht verfügbar ist, oder ausschließlich lokale Regeln verwenden.

Example sampling-rules.json

```
{
  "version": 2,
  "rules": [
    {
      "description": "Player moves.",
      "host": "*",
      "http_method": "*",
      "url_path": "/api/move/*",
      "fixed_target": 0,
      "rate": 0.05
    }
  ],
  "default": {
    "fixed_target": 1,
    "rate": 0.1
  }
}
```

Dieses Beispiel definiert eine benutzerdefinierte Regel und eine Standardregel. Die benutzerdefinierte Regel wendet eine Stichprobenrate von fünf Prozent an, ohne dass eine Mindestanzahl von Anfragen für Pfade verfolgt werden muss, unter `/api/move/` denen Pfade verfolgt werden müssen. Die Standardregel verfolgt die erste Anfrage jede Sekunde und 10 Prozent der weiteren Anfragen.

Der Nachteil der lokalen Definition von Regeln besteht darin, dass das feste Ziel von jeder Instanz des Rekorders unabhängig angewendet wird, anstatt vom X-Ray-Dienst verwaltet zu werden. Wenn Sie mehr Hosts bereitstellen, wird die feste Rate vervielfacht, wodurch es schwieriger wird, die Menge der aufgezeichneten Daten zu kontrollieren.

Bei AWS Lambda aktivierter Option können Sie die Samplerate nicht ändern. Wenn Ihre Funktion von einem instrumentierten Dienst aufgerufen wird, werden Aufrufe, die Anfragen generierten, die von diesem Dienst abgetastet wurden, von Lambda aufgezeichnet. Wenn aktives Tracing aktiviert ist und kein Tracing-Header vorhanden ist, trifft Lambda die Stichprobenentscheidung.

Um Sicherungsregeln in Spring bereitzustellen, konfigurieren Sie den globalen Recorder mit einer `CentralizedSamplingStrategy` in einer Konfigurationsklasse.

Example src/main/java/myapp/ .java WebConfig — Rekorder-Konfiguration

```
import com.amazonaws.xray.AWSXRay;
import com.amazonaws.xray.AWSXRayRecorderBuilder;
```

```
import com.amazonaws.xray.javax.servlet.AWSXRayServletFilter;
import com.amazonaws.xray.plugins.EC2Plugin;
import com.amazonaws.xray.strategy.sampling.LocalizedSamplingStrategy;

@Configuration
public class WebConfig {

  static {
  AWSXRayRecorderBuilder builder = AWSXRayRecorderBuilder.standard().withPlugin(new
 EC2Plugin());

  URL ruleFile = WebConfig.class.getResource("/sampling-rules.json");
  builder.withSamplingStrategy(new CentralizedSamplingStrategy(ruleFile));

  AWSXRay.setGlobalRecorder(builder.build());
}
```

Für Tomcat fügen Sie einen Listener hinzu, der `ServletContextListener` erweitert, und registrieren den Listener in der Bereitstellungsbeschreibung.

Example src/com/myapp/web/Startup.java

```
import com.amazonaws.xray.AWSXRay;
import com.amazonaws.xray.AWSXRayRecorderBuilder;
import com.amazonaws.xray.plugins.EC2Plugin;
import com.amazonaws.xray.strategy.sampling.LocalizedSamplingStrategy;

import java.net.URL;
import javax.servlet.ServletContextEvent;
import javax.servlet.ServletContextListener;

public class Startup implements ServletContextListener {

    @Override
    public void contextInitialized(ServletContextEvent event) {
        AWSXRayRecorderBuilder builder =
 AWSXRayRecorderBuilder.standard().withPlugin(new EC2Plugin());

        URL ruleFile = Startup.class.getResource("/sampling-rules.json");
        builder.withSamplingStrategy(new CentralizedSamplingStrategy(ruleFile));

        AWSXRay.setGlobalRecorder(builder.build());
    }
```

```
    @Override
    public void contextDestroyed(ServletContextEvent event) { }
}
```

Example WEB-INF/web.xml

```
...
  <listener>
    <listener-class>com.myapp.web.Startup</listener-class>
  </listener>
```

Um nur lokale Regeln zu verwenden, ersetzen Sie die `CentralizedSamplingStrategy` durch eine `LocalizedSamplingStrategy`.

```
builder.withSamplingStrategy(new LocalizedSamplingStrategy(ruleFile));
```

Protokollierung

Standardmäßig gibt das SDK Meldungen auf -Ebene in Ihre Anwendungsprotokolle aus. `ERROR` Sie können die Protokollierung auf Debug-Ebene im SDK aktivieren, um detailliertere Protokolle in Ihrer Anwendungsprotokolldatei auszugeben. Gültige Protokollebenen sind`DEBUG`,, `INFO`, `WARN` `ERROR` und. `FATAL` `FATAL`Bei der Protokollebene werden alle Protokollmeldungen stummgeschaltet, da das SDK nicht auf fataler Ebene protokolliert.

Example application.properties

Legen Sie die Protokollierungsebene mit der `logging.level.com.amazonaws.xray`-Eigenschaft fest.

```
logging.level.com.amazonaws.xray = DEBUG
```

Verwenden Sie Debug-Protokolle, um Probleme wie nicht geschlossene Untersegmente zu identifizieren, wenn Sie Untersegmente manuell generieren.

Trace-ID-Injection in Protokolle

Wenn Sie den Protokollanweisungen Ihre aktuelle vollqualifizierte Trace-ID zur Verfügung stellen möchten, können Sie die ID in den zugeordneten Diagnosekontext (MDC) einfügen. Über die `SegmentListener`-Schnittstelle werden Methoden während der Ereignisse des Segmentlebenszyklus aus dem X-Ray-Recorder aufgerufen. Wenn ein Segment oder Teilsegment

beginnt, wird die qualifizierte Trace-ID mit dem Schlüssel `AWS-XRAY-TRACE-ID` in den MDC injiziert. Wenn dieses Segment endet, wird der Schlüssel aus dem MDC entfernt. Dadurch wird die Trace-ID der verwendeten Protokollierungsbibliothek verfügbar gemacht. Wenn ein Teilsegment endet, wird seine übergeordnete ID in den MDC injiziert.

Example vollqualifizierte Trace-ID

Die vollqualifizierte ID wird als `TraceID@EntityID` dargestellt.

```
1-5df42873-011e96598b447dfca814c156@541b3365be3dafc3
```

Diese Funktion funktioniert mit Java-Anwendungen, die mit dem AWS X-Ray SDK for Java instrumentiert sind, und unterstützt die folgenden Logging-Konfigurationen:

- SLF4J Frontend-API mit Logback-Backend
- SLF4J Frontend-API mit Log4J2-Backend
- Log4J2 Frontend-API mit Log4J2-Backend

In den folgenden Registerkarten finden Sie die Anforderungen jedes Frontends und jedes Backends.

SLF4J Frontend

1. Fügen Sie Ihrem Projekt die folgende Maven-Abhängigkeit hinzu.

   ```
   <dependency>
       <groupId>com.amazonaws</groupId>
       <artifactId>aws-xray-recorder-sdk-slf4j</artifactId>
       <version>2.11.0</version>
   </dependency>
   ```

2. Schließen Sie beim Erstellen von `AWSXRayRecorder` die `withSegmentListener`-Methode ein. Dadurch wird eine `SegmentListener`-Klasse hinzugefügt, die automatisch neue Trace-IDs in das SLF4J MDC einfügt.

   Der `SegmentListener` akzeptiert eine optionale Zeichenfolge als Parameter, um das Präfix der Log-Anweisung zu konfigurieren. Das Präfix kann auf folgende Weise konfiguriert werden:

   - Keine — Verwendet das `AWS-XRAY-TRACE-ID` Standardpräfix.
   - Leer — Verwendet eine leere Zeichenfolge (z. B.`""`).

- Benutzerdefiniert — Verwendet ein benutzerdefiniertes Präfix, wie es in der Zeichenfolge definiert ist.

Example **`AWSXRayRecorderBuilder`**-Anweisung

```
AWSXRayRecorderBuilder builder = AWSXRayRecorderBuilder
        .standard().withSegmentListener(new SLF4JSegmentListener("CUSTOM-
PREFIX"));
```

Log4J2 front end

1. Fügen Sie Ihrem Projekt die folgende Maven-Abhängigkeit hinzu.

```
<dependency>
    <groupId>com.amazonaws</groupId>
    <artifactId>aws-xray-recorder-sdk-log4j</artifactId>
    <version>2.11.0</version>
</dependency>
```

2. Schließen Sie beim Erstellen von `AWSXRayRecorder` die `withSegmentListener`-Methode ein. Auf diese Weise wird eine `SegmentListener`-Klasse hinzugefügt, die automatisch neue vollqualifizierte Trace-IDs in das SLF4J MDC einfügt.

   Der `SegmentListener` akzeptiert eine optionale Zeichenfolge als Parameter, um das Präfix der Log-Anweisung zu konfigurieren. Das Präfix kann auf folgende Weise konfiguriert werden:

   - Keine — Verwendet das `AWS-XRAY-TRACE-ID` Standardpräfix.
   - Leer — Verwendet eine leere Zeichenfolge (z. B.`""`) und entfernt das Präfix.
   - Benutzerdefiniert — Verwendet das in der Zeichenfolge definierte benutzerdefinierte Präfix.

   Example **`AWSXRayRecorderBuilder`**-Anweisung

```
AWSXRayRecorderBuilder builder = AWSXRayRecorderBuilder
        .standard().withSegmentListener(new Log4JSegmentListener("CUSTOM-
PREFIX"));
```

Logback backend

Um die Trace-ID in die Protokollereignisse einzufügen, müssen Sie das `PatternLayout` des Loggers ändern, der jede Protokollierungsanweisung formatiert.

1. Suchen Sie, wo das `patternLayout` konfiguriert ist. Sie können dies programmgesteuert oder über eine XML-Konfigurationsdatei erreichen. Weitere Informationen finden Sie unter Logback-Konfiguration.
2. Fügen Sie `%X{AWS-XRAY-TRACE-ID}` an einer beliebigen Stelle in `patternLayout` ein, um die Trace-ID in zukünftige Protokollierungsanweisungen einzufügen. `%X{}` gibt an, dass Sie mit dem aus dem MDC bereitgestellten Schlüssel einen Wert abrufen. Weitere Informationen zu PatternLayouts In Logback finden Sie unter PatternLayout.

Log4J2 backend

1. Suchen Sie, wo das `patternLayout` konfiguriert ist. Sie können dies programmgesteuert oder über eine im XML-, JSON-, YAML- oder Eigenschaftenformat geschriebene Konfigurationsdatei erreichen.

   Weitere Informationen zum Konfigurieren von Log4J2 über eine Konfigurationsdatei finden Sie unter Konfiguration.

   Weitere Informationen zur programmgesteuerten Konfiguration von Log4J2 finden Sie unter Programmatische Konfiguration.
2. Fügen Sie `%X{AWS-XRAY-TRACE-ID}` an einer beliebigen Stelle in `PatternLayout` ein, um die Trace-ID in zukünftige Protokollierungsanweisungen einzufügen. `%X{}` gibt an, dass Sie mit dem aus dem MDC bereitgestellten Schlüssel einen Wert abrufen. Weitere Informationen PatternLayouts zu Log4J2 finden Sie unter Pattern Layout.

Beispiel für Trace-ID-Injection

Im Folgenden wird eine `PatternLayout`-Zeichenfolge angezeigt, die geändert wurde, sodass die Trace-ID enthalten ist. Die Trace-ID wird nach dem Thread-Namen (`%t`) und vor der Protokollebene (`%-5p`) ausgegeben.

Example **`PatternLayout`** mit ID-Injection

```
%d{HH:mm:ss.SSS} [%t] %X{AWS-XRAY-TRACE-ID} %-5p %m%n
```

AWS X-Ray druckt automatisch den Schlüssel und die Trace-ID in der Protokollanweisung aus, um das Parsen zu vereinfachen. Im Folgenden wird eine Protokollanweisung dargestellt, die das modifizierte `PatternLayout` verwendet.

Example Protokollanweisung mit ID-Injection

```
2019-09-10 18:58:30.844 [nio-5000-exec-4]  AWS-XRAY-TRACE-ID:
 1-5d77f256-19f12e4eaa02e3f76c78f46a@1ce7df03252d99e1 WARN 1 - Your logging message
 here
```

Die Protokollierungsnachricht selbst ist im Muster `%m` eingebettet und wird beim Aufruf des Loggers festgelegt.

Segment-Listeners

Segment-Listeners sind eine Schnittstelle zum Abfangen von Lebenszyklusereignissen, wie Anfang und Ende von Segmenten, die von `AWSXRayRecorder` erstellt werden. Die Implementierung einer Segment-Listener-Ereignisfunktion könnte darin bestehen, allen Teilsegmenten dieselbe Anmerkung hinzuzufügen, wenn sie mit `onBeginSubsegment` erstellt werden, eine Meldung zu protokollieren, nachdem jedes Segment mit `afterEndSegment` an den Daemon gesendet wurde, oder von SQL Interceptors mit `beforeEndSubsegment` gesendete Abfragen aufzuzeichnen, um zu überprüfen, ob das Teilsegment eine SQL-Abfrage darstellt, wobei zusätzliche Metadaten hinzugefügt werden, falls dies der Fall ist.

Die vollständige Liste der `SegmentListener` Funktionen finden Sie in der Dokumentation für das AWS X-Ray Recorder SDK for Java API.

Das folgende Beispiel zeigt, wie Sie allen Teilsegmenten bei der Erstellung eine konsistente Anmerkung mit `onBeginSubsegment` hinzufügen und mit `afterEndSegment` eine Protokollmeldung am Ende jedes Segments drucken.

Example MySegmentListener.java

```
import com.amazonaws.xray.entities.Segment;
import com.amazonaws.xray.entities.Subsegment;
import com.amazonaws.xray.listeners.SegmentListener;

public class MySegmentListener implements SegmentListener {
    .....
```

```
    @Override
    public void onBeginSubsegment(Subsegment subsegment) {
        subsegment.putAnnotation("annotationKey", "annotationValue");
    }

    @Override
    public void afterEndSegment(Segment segment) {
        // Be mindful not to mutate the segment
        logger.info("Segment with ID " + segment.getId());
    }
}
```

Dieser benutzerdefinierte Segment-Listener wird dann beim Erstellen des `AWSXRayRecorder` referenziert.

Example AWSXRayRecorderBuilder Aussage

```
AWSXRayRecorderBuilder builder = AWSXRayRecorderBuilder
        .standard().withSegmentListener(new MySegmentListener());
```

Umgebungsvariablen

Sie können Umgebungsvariablen verwenden, um das X-Ray SDK for Java zu konfigurieren. Das SDK unterstützt die folgenden Variablen.

- `AWS_XRAY_CONTEXT_MISSING`— Legt fest, `RUNTIME_ERROR` dass Ausnahmen ausgelöst werden, wenn Ihr instrumentierter Code versucht, Daten aufzuzeichnen, obwohl kein Segment geöffnet ist.

  Zulässige Werte

  - `RUNTIME_ERROR`— Löst eine Laufzeitausnahme aus.
  - `LOG_ERROR`— Fehler protokollieren und fortfahren (Standard).
  - `IGNORE_ERROR`— Fehler ignorieren und fortfahren.

  Fehler im Zusammenhang mit fehlenden Segmenten oder Untersegmenten können auftreten, wenn Sie versuchen, einen instrumentierten Client in Startcode zu verwenden, der ausgeführt wird, wenn keine Anfrage geöffnet ist, oder in Code, der einen neuen Thread erzeugt.
- `AWS_XRAY_DAEMON_ADDRESS`— Legt den Host und den Port des X-Ray-Daemon-Listeners fest. Standardmäßig verwendet `127.0.0.1:2000` das SDK sowohl Trace-Daten (UDP) als auch

Sampling-Daten (TCP). Verwenden Sie diese Variable, wenn Sie den Daemon so konfiguriert haben, dass er auf einem anderen Port lauscht oder wenn er auf einem anderen Host läuft.

Format

- Derselbe Port — *address*:*port*
- Verschiedene Anschlüsse — tcp:*address*:*port* udp:*address*:*port*

- AWS_LOG_GROUP— Legen Sie den Namen einer Protokollgruppe auf eine Protokollgruppe fest, die Ihrer Anwendung zugeordnet ist. Wenn Ihre Protokollgruppe dasselbe AWS Konto und dieselbe Region wie Ihre Anwendung verwendet, sucht X-Ray anhand dieser angegebenen Protokollgruppe automatisch nach den Segmentdaten Ihrer Anwendung. Weitere Informationen zu Protokollgruppen finden Sie unter Arbeiten mit Protokollgruppen und Streams.
- AWS_XRAY_TRACING_NAME— Legen Sie einen Dienstnamen fest, den das SDK für Segmente verwendet. Überschreibt den für die Segmentbenennungsstrategie des Servlet-Filters festgelegten Dienstnamen.

Umgebungsvariablen überschreiben äquivalente Systemeigenschaften und Werte in Code.

Systemeigenschaften

Sie können Systemeigenschaften als JVM-spezifische Alternative zu Umgebungsvariablen verwenden. Das SDK unterstützt die folgenden Eigenschaften:

- com.amazonaws.xray.strategy.tracingName— EntsprichtAWS_XRAY_TRACING_NAME.
- com.amazonaws.xray.emitters.daemonAddress— EntsprichtAWS_XRAY_DAEMON_ADDRESS.
- com.amazonaws.xray.strategy.contextMissingStrategy— EntsprichtAWS_XRAY_CONTEXT_MISSING.

Wenn eine Systemeigenschaft und die entsprechende Umgebungsvariable eingerichtet sind, wird der Wert der Umgebungsvariablen verwendet. Bei beiden Methoden werden Werte in Code überschrieben.

## Nachverfolgung eingehender Anfragen mit dem X-Ray SDK for Java

Sie können das X-Ray SDK verwenden, um eingehende HTTP-Anfragen zu verfolgen, die Ihre Anwendung auf einer EC2-Instance in Amazon EC2 oder Amazon ECS bearbeitet. AWS Elastic Beanstalk

Verwenden Sie einen `Filter`, um eingehende HTTP-Anforderungen zu instrumentieren. Wenn Sie den X-Ray-Servlet-Filter zu Ihrer Anwendung hinzufügen, erstellt das X-Ray SDK for Java ein Segment für jede gesampelte Anfrage. Dieses Segment umfasst Dauer, Methode und Status der HTTP-Anforderung. Die zusätzliche Instrumentierung schafft Untersegmente zu diesem Segment.

> **Note**
>
> Für AWS Lambda Funktionen erstellt Lambda für jede abgetastete Anfrage ein Segment. Weitere Informationen finden Sie unter AWS Lambda und AWS X-Ray.

Jedes Segment hat einen Namen, der Ihre Anwendung in der Service Map identifiziert. Das Segment kann statisch benannt werden, oder Sie können das SDK so konfigurieren, dass es dynamisch auf der Grundlage des Host-Headers in der eingehenden Anfrage benannt wird. Mit der dynamischen Benennung können Sie Traces auf der Grundlage des Domainnamens in der Anfrage gruppieren und einen Standardnamen anwenden, wenn der Name nicht einem erwarteten Muster entspricht (z. B. wenn der Host-Header gefälscht ist).

> **Weitergeleitete Anfragen**
>
> Wenn ein Load Balancer oder ein anderer Vermittler eine Anfrage an Ihre Anwendung weiterleitet, nimmt X-Ray die Client-IP aus dem `X-Forwarded-For` Header in der Anfrage und nicht aus der Quell-IP im IP-Paket. Die Client-IP, die für eine weitergeleitete Anfrage aufgezeichnet wird, kann gefälscht sein und sollte daher nicht als vertrauenswürdig eingestuft werden.

Wenn eine Anfrage weitergeleitet wird, legt das SDK ein zusätzliches Feld im Segment fest, um dies anzuzeigen. Wenn das Segment das Feld enthält, das auf `x_forwarded_for` gesetzt ist`true`, wurde die Client-IP aus dem `X-Forwarded-For` Header in der HTTP-Anfrage übernommen.

Der Meldungshandler erzeugt für jede eingehende Anforderung ein Segment mit einem `http`-Block, der die folgenden Informationen enthält:

- HTTP-Methode — GET, POST, PUT, DELETE usw.
- Client-Adresse — Die IP-Adresse des Clients, der die Anfrage gesendet hat.
- Antwortcode — Der HTTP-Antwortcode für die abgeschlossene Anfrage.

- Timing — Die Startzeit (als die Anfrage empfangen wurde) und die Endzeit (als die Antwort gesendet wurde).
- Benutzeragent — Der `user-agent` aus der Anfrage.
- Länge des Inhalts — Die Länge `content-length` der Antwort.

Sections



### Hinzufügen eines Ablaufverfolgungsfilters zu Ihrer Anwendung (Tomcat)

Im Fall von Tomcat fügen Sie einen `<filter>` zur `web.xml`-Datei Ihres Projekts hinzu. Legen Sie mit dem `fixedName` Parameter einen Servicenamen fest, damit die erstellten Segmente auf eingehende Anforderungen angewendet werden.

Example WEB-INF/web.xml – Tomcat

```
<filter>
  <filter-name>AWSXRayServletFilter</filter-name>
  <filter-class>com.amazonaws.xray.javax.servlet.AWSXRayServletFilter</filter-class>
  <init-param>
    <param-name>fixedName</param-name>
    <param-value>MyApp</param-value>
  </init-param>
</filter>
<filter-mapping>
  <filter-name>AWSXRayServletFilter</filter-name>
  <url-pattern>*</url-pattern>
</filter-mapping>
```

### Hinzufügen eines Ablaufverfolgungsfilters zu Ihrer Anwendung (Spring)

Für Spring fügen Sie einen `Filter` zu Ihrer `WebConfig`-Klasse hinzu. Übermitteln Sie den Segmentnamen als Zeichenfolge an den `AWSXRayServletFilter`-Konstruktor.

Example src/main/java/myapp/ .java WebConfig — Frühling

```
package myapp;
```

```
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.Bean;
import javax.servlet.Filter;
import com.amazonaws.xray.javax.servlet.AWSXRayServletFilter;

@Configuration
public class WebConfig {

  @Bean
  public Filter TracingFilter() {
    return new AWSXRayServletFilter("Scorekeep");
  }
}
```

Konfiguration einer Segmentbenennungsstrategie

AWS X-Ray verwendet einen Dienstnamen, um Ihre Anwendung zu identifizieren und sie von den anderen Anwendungen, Datenbanken, externen APIs und Ressourcen zu unterscheiden, die Ihre Anwendung verwendet. AWS Wenn das X-Ray SDK Segmente für eingehende Anfragen generiert, zeichnet es den Dienstnamen Ihrer Anwendung im Namensfeld des Segments auf.

Das X-Ray SDK kann Segmente nach dem Hostnamen im HTTP-Anforderungsheader benennen. Dieser Header kann jedoch gefälscht sein, was zu unerwarteten Knoten in Ihrer Service Map führen kann. Um zu verhindern, dass das SDK Segmente aufgrund von Anfragen mit gefälschten Host-Headern falsch benennt, müssen Sie einen Standardnamen für eingehende Anfragen angeben.

Wenn Ihre Anwendung Anfragen für mehrere Domänen bearbeitet, können Sie das SDK so konfigurieren, dass es eine dynamische Benennungsstrategie verwendet, um dies in Segmentnamen widerzuspiegeln. Eine dynamische Benennungsstrategie ermöglicht es dem SDK, den Hostnamen für Anfragen zu verwenden, die einem erwarteten Muster entsprechen, und den Standardnamen auf Anfragen anzuwenden, bei denen dies nicht der Fall ist.

Beispielsweise könnten Sie eine einzige Anwendung haben, die Anfragen an drei Subdomänen — `www.example.com`api.example.com, und — bedient. `static.example.com` Sie können eine dynamische Benennungsstrategie mit dem Muster verwenden`*.example.com`, um Segmente für jede Subdomain mit einem anderen Namen zu identifizieren, was zu drei Dienstknoten auf der Service-Map führt. Wenn Ihre Anwendung Anfragen mit einem Hostnamen empfängt, der nicht dem Muster entspricht, wird auf der Service Map ein vierter Knoten mit einem von Ihnen angegebenen Fallback-Namen angezeigt.

Wenn Sie denselben Namen für alle Segmente verwenden möchten, geben Sie bei der Initialisierung des Servlet-Filters den Namen Ihrer Anwendung, wie im vorherigen Abschnitt gezeigt, ein. Dies hat den gleichen Effekt wie das Erstellen eines Fixed SegmentNamingStrategydurch Aufrufen `SegmentNamingStrategy.fixed()` und Übergeben an den `AWSXRayServletFilter`Konstruktor.

> **Note**
>
> Sie können den mit der `AWS_XRAY_TRACING_NAME`-Umgebungsvariablen in Code definierten standardmäßigen Dienstnamen überschreiben.

Eine dynamische Benennungsstrategie definiert ein Muster, dem Hostnamen entsprechen sollten, sowie einen Standardnamen, der verwendet wird, wenn der Hostname in der HTTP-Anforderung nicht mit diesem Muster übereinstimmt. Alternativ können Sie `dynamicNamingRecognizedHosts` und `dynamicNamingFallbackName` nutzen, um das Muster und den Standardnamen zu bestimmen und Segmente in Tomcat dynamisch zu benennen.

Example WEB-INF/web.xml – Servlet-Filter mit dynamischer Benennung

```
<filter>
  <filter-name>AWSXRayServletFilter</filter-name>
  <filter-class>com.amazonaws.xray.javax.servlet.AWSXRayServletFilter</filter-class>
  <init-param>
    <param-name>dynamicNamingRecognizedHosts</param-name>
    <param-value>*.example.com</param-value>
  </init-param>
  <init-param>
    <param-name>dynamicNamingFallbackName</param-name>
    <param-value>MyApp</param-value>
  </init-param>
</filter>
<filter-mapping>
  <filter-name>AWSXRayServletFilter</filter-name>
  <url-pattern>*</url-pattern>
</filter-mapping>
```

Erstellen Sie für Spring eine Dynamik, SegmentNamingStrategyindem Sie sie aufrufen`SegmentNamingStrategy.dynamic()`, und übergeben Sie sie an den `AWSXRayServletFilter` Konstruktor.

Example WebConfigsrc/main/java/myapp/ .java — Servlet-Filter mit dynamischer Benennung

```
package myapp;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.Bean;
import javax.servlet.Filter;
import com.amazonaws.xray.javax.servlet.AWSXRayServletFilter;
import com.amazonaws.xray.strategy.SegmentNamingStrategy;

@Configuration
public class WebConfig {

  @Bean
  public Filter TracingFilter() {
    return new AWSXRayServletFilter(SegmentNamingStrategy.dynamic("MyApp",
 "*.example.com"));
  }
}
```

## Verfolgen von AWS SDK-Aufrufen mit dem X-Ray SDK for Java

Wenn Ihre Anwendung Aufrufe tätigt, um Daten AWS-Services zu speichern, in eine Warteschlange zu schreiben oder Benachrichtigungen zu senden, verfolgt das X-Ray SDK for Java die Aufrufe im Downstream in Untersegmenten. Verfolgte Ressourcen AWS-Services und Ressourcen, auf die Sie innerhalb dieser Services zugreifen (z. B. ein Amazon S3 S3-Bucket oder eine Amazon SQS SQS-Warteschlange), werden als Downstream-Knoten auf der Trace-Map in der X-Ray-Konsole angezeigt.

Das X-Ray-SDK SDK for Java instrumentiert automatisch alle AWS-SDK-Clients, wenn Sie die `aws-sdk` und die `aws-sdk-instrumentor` Submodule in Ihren Build aufnehmen. Wenn Sie das Instrumentor-Untermodul nicht einbeziehen, können Sie auswählen, welche Clients Sie instrumentieren möchten, und andere ausschließen.

Um einzelne Clients zu instrumentieren, entfernen Sie das `aws-sdk-instrumentor` Submodul aus Ihrem Build und fügen Sie mithilfe des Client-Builders des Services ein `XRayClient` AS `TracingHandler` auf Ihrem AWS SDK-Client hinzu.

Wenn Sie beispielsweise einen `AmazonDynamoDB`-Client instrumentieren möchten, übergeben Sie einen Ablaufverfolgungshandler an `AmazonDynamoDBClientBuilder`.

Example MyModel.java — DynamoDB-Client

```
import com.amazonaws.xray.AWSXRay;
```

```
import com.amazonaws.xray.handlers.TracingHandler;

...
public class MyModel {
  private AmazonDynamoDB client = AmazonDynamoDBClientBuilder.standard()
        .withRegion(Regions.fromName(System.getenv("AWS_REGION")))
        .withRequestHandlers(new TracingHandler(AWSXRay.getGlobalRecorder()))
        .build();
...
```

Für alle Dienste können Sie den Namen der aufgerufenen API in der X-Ray-Konsole sehen. Für eine Untergruppe von Diensten fügt das X-Ray SDK dem Segment Informationen hinzu, um die Service Map detaillierter zu gestalten.

Wenn Sie beispielsweise einen Aufruf mit einem instrumentierten DynamoDB-Client tätigen, fügt das SDK den Tabellennamen dem Segment für Aufrufe hinzu, die auf eine Tabelle abzielen. In der Konsole wird jede Tabelle als separater Knoten in der Service Map angezeigt, mit einem generischen DynamoDB-Knoten für Aufrufe, die nicht auf eine Tabelle abzielen.

Example Untersegment für einen Aufruf von DynamoDB zum Speichern eines Elements

```
{
  "id": "24756640c0d0978a",
  "start_time": 1.480305974194E9,
  "end_time": 1.4803059742E9,
  "name": "DynamoDB",
  "namespace": "aws",
  "http": {
    "response": {
      "content_length": 60,
      "status": 200
    }
  },
  "aws": {
    "table_name": "scorekeep-user",
    "operation": "UpdateItem",
    "request_id": "UBQNSO5AEM8T4FDA4RQDEB94OVTDRVV4K4HIRGVJF66Q9ASUAAJG",
  }
}
```

Wenn Sie auf benannte Ressourcen zugreifen, werden durch Aufrufe der folgenden Services weitere Knoten in der Service-Übersicht erstellt. Durch Aufrufe, die keinen bestimmten Ressourcen gelten, wird ein generischer Knoten für den Service erstellt.

- Amazon DynamoDB — Tabellenname
- Amazon Simple Storage Service — Bucket und Schlüsselname
- Amazon Simple Queue Service — Name der Warteschlange

Um Downstream-Aufrufe AWS-Services mit AWS SDK for Java 2.2 und höher zu instrumentieren, können Sie das `aws-xray-recorder-sdk-aws-sdk-v2-instrumentor` Modul aus Ihrer Build-Konfiguration weglassen. Fügen Sie stattdessen das `aws-xray-recorder-sdk-aws-sdk-v2 module` ein und instrumentieren Sie dann einzelne Clients, indem Sie sie mit einem `TracingInterceptor` konfigurieren.

Example AWS SDK for Java 2.2 und höher — Tracing Interceptor

```
import com.amazonaws.xray.interceptors.TracingInterceptor;
import software.amazon.awssdk.core.client.config.ClientOverrideConfiguration
import software.amazon.awssdk.services.dynamodb.DynamoDbClient;
//...
public class MyModel {
private DynamoDbClient client = DynamoDbClient.builder()
.region(Region.US_WEST_2)
.overrideConfiguration(ClientOverrideConfiguration.builder()
.addExecutionInterceptor(new TracingInterceptor())
.build()
)
.build();
//...
```

## Rückverfolgung von Aufrufen an Downstream-HTTP-Webservices mit dem X-Ray SDK for Java

Wenn Ihre Anwendung Microservices oder öffentliche HTTP-APIs aufruft, können Sie die Version von X-Ray SDK für Java verwenden, `HttpClient` um diese Aufrufe zu instrumentieren und die API als Downstream-Service zum Service Graph hinzuzufügen.

Das X-Ray-SDK SDK for Java umfasst `DefaultHttpClient` `HttpClientBuilder` Klassen, die anstelle der HttpComponents Apache-Äquivalente verwendet werden können, um ausgehende HTTP-Aufrufe zu instrumentieren.

- `com.amazonaws.xray.proxies.apache.http.DefaultHttpClient` - `org.apache.http.impl.client.DefaultHttpClient`
- `com.amazonaws.xray.proxies.apache.http.HttpClientBuilder` - `org.apache.http.impl.client.HttpClientBuilder`

Diese Bibliotheken befinden sich im `aws-xray-recorder-sdk-apache-http`-Untermodul.

Sie können Ihre vorhandenen Importanweisungen durch das X-Ray-Äquivalent ersetzen, um alle Clients zu instrumentieren, oder den vollqualifizierten Namen verwenden, wenn Sie einen Client für die Instrumentierung bestimmter Clients initialisieren.

Example HttpClientBuilder

```
import com.fasterxml.jackson.databind.ObjectMapper;
import org.apache.http.HttpEntity;
import org.apache.http.client.methods.CloseableHttpResponse;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.CloseableHttpClient;
import org.apache.http.util.EntityUtils;
import com.amazonaws.xray.proxies.apache.http.HttpClientBuilder;
...
  public String randomName() throws IOException {
    CloseableHttpClient httpclient = HttpClientBuilder.create().build();
    HttpGet httpGet = new HttpGet("http://names.example.com/api/");
    CloseableHttpResponse response = httpclient.execute(httpGet);
    try {
      HttpEntity entity = response.getEntity();
      InputStream inputStream = entity.getContent();
      ObjectMapper mapper = new ObjectMapper();
      Map<String, String> jsonMap = mapper.readValue(inputStream, Map.class);
      String name = jsonMap.get("name");
      EntityUtils.consume(entity);
      return name;
    } finally {
      response.close();
    }
  }
```

Wenn Sie einen Aufruf einer Downstream-Web-API instrumentieren, zeichnet das X-Ray SDK for Java ein Untersegment mit Informationen über die HTTP-Anfrage und -Antwort auf. X-Ray verwendet das Untersegment, um ein abgeleitetes Segment für die Remote-API zu generieren.

Example Untersegment für einen nachgelagerten HTTP-Aufruf

```
{
  "id": "004f72be19cddc2a",
  "start_time": 1484786387.131,
  "end_time": 1484786387.501,
  "name": "names.example.com",
  "namespace": "remote",
  "http": {
    "request": {
      "method": "GET",
      "url": "https://names.example.com/"
    },
    "response": {
      "content_length": -1,
      "status": 200
    }
  }
}
```

Example Abgeleitetes Segment für einen nachgelagerten HTTP-Anruf

```
{
  "id": "168416dc2ea97781",
  "name": "names.example.com",
  "trace_id": "1-62be1272-1b71c4274f39f122afa64eab",
  "start_time": 1484786387.131,
  "end_time": 1484786387.501,
  "parent_id": "004f72be19cddc2a",
  "http": {
    "request": {
      "method": "GET",
      "url": "https://names.example.com/"
    },
    "response": {
      "content_length": -1,
      "status": 200
    }
  },
```

```
  "inferred": true
}
```

## Verfolgen von SQL-Abfragen mit dem X-Ray SDK for Java

### SQL-Interzeptoren

Instrumentieren Sie SQL-Datenbankabfragen, indem Sie den X-Ray SDK for Java JDBC Interceptor zu Ihrer Datenquellenkonfiguration hinzufügen.

- PostgreSQL – `com.amazonaws.xray.sql.postgres.TracingInterceptor`
- MySQL – `com.amazonaws.xray.sql.mysql.TracingInterceptor`

Diese Interceptors befinden sich im `aws-xray-recorder-sql-postgres`-Untermodul bzw. `aws-xray-recorder-sql-mysql`. Sie implementieren `org.apache.tomcat.jdbc.pool.JdbcInterceptor` und sind mit Tomcat-Verbindungspools kompatibel.

> **Note**
>
> Aus Sicherheitsgründen zeichnen die SQL Interceptors die SQL-Abfrage selbst innerhalb von Teilsegmenten nicht auf.

Für Spring fügen Sie der Eigenschaftendatei einen Interceptor hinzu und erstellen Sie die Datenquelle mit dem `DataSourceBuilder` von Spring Boot.

Example **`src/main/java/resources/application.properties`** – PostgreSQL-JDBC-Interceptor

```
spring.datasource.continue-on-error=true
spring.jpa.show-sql=false
spring.jpa.hibernate.ddl-auto=create-drop
spring.datasource.jdbc-interceptors=com.amazonaws.xray.sql.postgres.TracingInterceptor
spring.jpa.database-platform=org.hibernate.dialect.PostgreSQL94Dialect
```

Example **`src/main/java/myapp/WebConfig.java`** – Datenquelle

```
import org.springframework.boot.autoconfigure.EnableAutoConfiguration;
```

```
import org.springframework.boot.autoconfigure.jdbc.DataSourceBuilder;
import org.springframework.boot.context.properties.ConfigurationProperties;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.data.jpa.repository.config.EnableJpaRepositories;

import javax.servlet.Filter;
import javax.sql.DataSource;
import java.net.URL;

@Configuration
@EnableAutoConfiguration
@EnableJpaRepositories("myapp")
public class RdsWebConfig {

  @Bean
  @ConfigurationProperties(prefix = "spring.datasource")
  public DataSource dataSource() {
      logger.info("Initializing PostgreSQL datasource");
      return DataSourceBuilder.create()
              .driverClassName("org.postgresql.Driver")
              .url("jdbc:postgresql://" + System.getenv("RDS_HOSTNAME") + ":" +
 System.getenv("RDS_PORT") + "/ebdb")
              .username(System.getenv("RDS_USERNAME"))
              .password(System.getenv("RDS_PASSWORD"))
              .build();
  }
...
}
```

Rufen Sie für Tomcat die JDBC-Datenquelle mit einem Verweis `setJdbcInterceptors` auf die Klasse X-Ray SDK for Java auf.

Example **`src/main/myapp/model.java`** – Datenquelle

```
import org.apache.tomcat.jdbc.pool.DataSource;
...
DataSource source = new DataSource();
source.setUrl(url);
source.setUsername(user);
source.setPassword(password);
source.setDriverClassName("com.mysql.jdbc.Driver");
source.setJdbcInterceptors("com.amazonaws.xray.sql.mysql.TracingInterceptor;");
```

Die Tomcat JDBC Data Source-Bibliothek ist im X-Ray SDK for Java enthalten, aber Sie können sie als bereitgestellte Abhängigkeit von dem Dokument deklarieren, in dem Sie sie verwenden.

Example **`pom.xml`** – JDBC-Datenquelle

```
<dependency>
  <groupId>org.apache.tomcat</groupId>
  <artifactId>tomcat-jdbc</artifactId>
  <version>8.0.36</version>
  <scope>provided</scope>
</dependency>
```

Nativer SQL Tracing Decorator

- Fügen Sie Ihre `aws-xray-recorder-sdk-sql`Abhängigkeiten hinzu.
- Dekorieren Sie Ihre Datenbank-Datenquelle, Verbindung oder Anweisung.

```
dataSource = TracingDataSource.decorate(dataSource)
connection = TracingConnection.decorate(connection)
statement = TracingStatement.decorateStatement(statement)
preparedStatement = TracingStatement.decoratePreparedStatement(preparedStatement,
 sql)
callableStatement = TracingStatement.decorateCallableStatement(callableStatement,
 sql)
```

## Generieren von benutzerdefinierten Untersegmenten mit dem X-Ray SDK for Java

Untersegmente erweitern das Segment eines Traces um Details über die Arbeit, die zur Bearbeitung einer Anfrage geleistet wurde. Jedes Mal, wenn Sie einen Anruf mit einem instrumentierten Client tätigen, zeichnet das X-Ray-SDK die in einem Untersegment generierten Informationen auf. Sie können zusätzliche Untersegmente erstellen, um andere Untersegmente zu gruppieren, die Leistung eines Codeabschnitts zu messen oder Anmerkungen und Metadaten aufzuzeichnen.

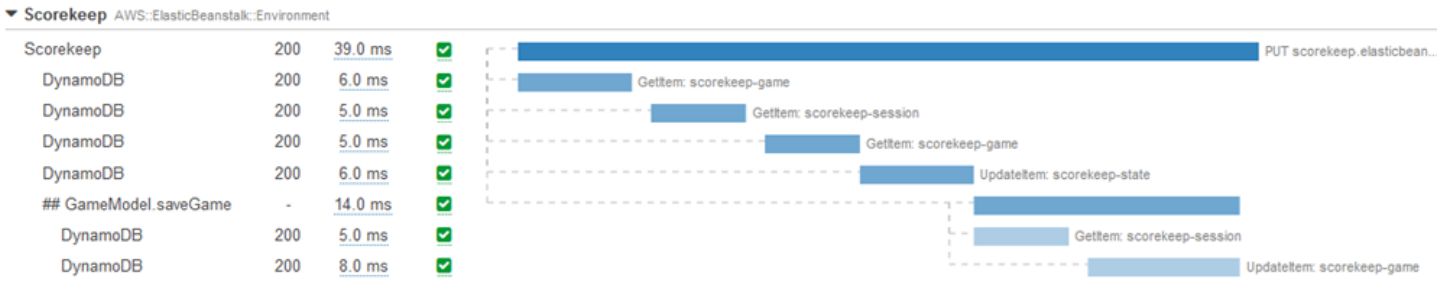
Um Untersegmente zu verwalten, verwenden Sie die Methoden `beginSubsegment` und `endSubsegment`.

Example GameModel.java — benutzerdefiniertes Untersegment

```
import com.amazonaws.xray.AWSXRay;
```

```
...
  public void saveGame(Game game) throws SessionNotFoundException {
    // wrap in subsegment
    Subsegment subsegment = AWSXRay.beginSubsegment("Save Game");
    try {
      // check session
      String sessionId = game.getSession();
      if (sessionModel.loadSession(sessionId) == null ) {
        throw new SessionNotFoundException(sessionId);
      }
      mapper.save(game);
    } catch (Exception e) {
      subsegment.addException(e);
      throw e;
    } finally {
      AWSXRay.endSubsegment();
    }
  }
```

In diesem Beispiel lädt der Code innerhalb des Untersegments die Spielsitzung aus DynamoDB mit einer Methode auf dem Sitzungsmodell und verwendet den DynamoDB-Mapper AWS SDK for Java des Spiels, um das Spiel zu speichern. Wenn Sie diesen Code in ein Untersegment einbinden, werden die Aufrufe zu DynamoDB-Unterelementen des `Save Game` Untersegments in der Trace-Ansicht in der Konsole.




Wenn der Code in Ihrem Untersegment geprüfte Ausnahmen auslöst, verpacken Sie ihn in einen `try`-Block und rufen Sie `AWSXRay.endSubsegment()` in einem `finally`-Block auf, um sicherzustellen, dass das Untersegment immer geschlossen ist. Wenn ein Untersegment nicht geschlossen ist, kann das übergeordnete Segment nicht abgeschlossen werden und wird nicht an X-Ray gesendet.

Für Code, der keine geprüften Ausnahmen auslöst, können Sie den Code `AWSXRay.CreateSubsegment` als Lambda-Funktion übergeben.

Example Untersegment-Lambda-Funktion

```
import com.amazonaws.xray.AWSXRay;

AWSXRay.createSubsegment("getMovies", (subsegment) -> {
    // function code
});
```

Wenn Sie ein Untersegment innerhalb eines Segments oder eines anderen Untersegments erstellen, generiert das X-Ray SDK for Java eine ID dafür und zeichnet die Start- und Endzeit auf.

Example Untersegment mit Metadaten

```
"subsegments": [{
  "id": "6f1605cd8a07cb70",
  "start_time": 1.480305974194E9,
  "end_time": 1.4803059742E9,
  "name": "Custom subsegment for UserModel.saveUser function",
  "metadata": {
    "debug": {
      "test": "Metadata string from UserModel.saveUser"
    }
  },
```

Bei asynchroner Programmierung und Multithread-Programmierung müssen Sie das Teilsegment manuell an die `endSubsegment()` Methode übergeben, um sicherzustellen, dass es korrekt geschlossen wird, da der X-Ray-Kontext während der asynchronen Ausführung geändert werden kann. Wenn ein asynchrones Teilsegment geschlossen wird, nachdem das übergeordnete Segment geschlossen wurde, streamt diese Methode automatisch das gesamte Segment an den X-Ray-Daemon.

Example Asynchrones Teilsegment

```
@GetMapping("/api")
public ResponseEntity<?> api() {
  CompletableFuture.runAsync(() -> {
      Subsegment subsegment = AWSXRay.beginSubsegment("Async Work");
      try {
          Thread.sleep(3000);
      } catch (InterruptedException e) {
          subsegment.addException(e);
```

```
            throw e;
        } finally {
            AWSXRay.endSubsegment(subsegment);
        }
    });
    return ResponseEntity.ok().build();
}
```

## Hinzufügen von Anmerkungen und Metadaten zu Segmenten mit dem X-Ray SDK for Java

Sie können Anmerkungen und Metadaten verwenden, um zusätzliche Informationen über Anfragen, die Umgebung oder Ihre Anwendung aufzuzeichnen. Sie können Anmerkungen und Metadaten zu den Segmenten hinzufügen, die das X-Ray SDK erstellt, oder zu benutzerdefinierten Untersegmenten, die Sie erstellen.

Anmerkungen sind Schlüssel-Wert-Paare mit Zeichenfolgen-, Zahlen- oder booleschen Werten. Anmerkungen sind für die Verwendung mit Filterausdrücken indexiert. Berücksichtigen Sie Anmerkungen, um Daten zur Gruppierung von Ablaufverfolgungen in der Konsole zu verwenden, oder wenn Sie die `GetTraceSummaries`-API aufrufen.

Metadaten sind Schlüssel-Wert-Paare, die Werte beliebigen Typs enthalten können, einschließlich Objekte und Listen, aber nicht für die Verwendung mit Filterausdrücken indexiert sind. Verwenden Sie Metadaten, um zusätzliche Daten aufzuzeichnen, die Sie im Trace speichern möchten, aber nicht für die Suche verwenden müssen.

Zusätzlich zu Anmerkungen und Metadaten können Sie auch Benutzer-ID-Zeichenfolgen in Segmenten aufzeichnen. Benutzer-IDs werden in einem eigenen Feld aufgezeichnet und zur Suche indiziert.

Sections



### Anmerkungen mit dem X-Ray SDK for Java aufnehmen

Verwenden Sie Anmerkungen, um Informationen zu Segmenten oder Untersegmenten, die zur Suche indiziert werden sollten, aufzuzeichnen.

Anmerkung zu Anforderungen

- Schlüssel — Der Schlüssel für eine X-Ray-Anmerkung kann bis zu 500 alphanumerische Zeichen enthalten. Sie können keine anderen Leerzeichen oder Symbole als den Unterstrich (_) verwenden.
- Werte — Der Wert für eine X-Ray-Anmerkung kann bis zu 1.000 Unicode-Zeichen enthalten.
- Die Anzahl der Anmerkungen — Sie können bis zu 50 Anmerkungen pro Spur verwenden.

So zeichnen Sie Anmerkungen auf

1. Eine Referenz des aktuellen Segments oder Untersegments finden Sie unter `AWSXRay`.

   ```
   import com.amazonaws.xray.AWSXRay;
   import com.amazonaws.xray.entities.Segment;
   ...
   Segment document = AWSXRay.getCurrentSegment();
   ```

   or

   ```
   import com.amazonaws.xray.AWSXRay;
   import com.amazonaws.xray.entities.Subsegment;
   ...
   Subsegment document = AWSXRay.getCurrentSubsegment();
   ```

2. Rufen Sie `putAnnotation` mit einem Aktivierungsschlüssel und einem booleschen Wert oder einem Zeichenfolgenwert auf.

   ```
   document.putAnnotation("mykey", "my value");
   ```

Das SDK zeichnet Anmerkungen als Schlüssel-Wert-Paare in einem `annotations`-Objekt im Segmentdokument auf. Wenn `putAnnotation` zweimal mit demselben Schlüssel aufgerufen wird, werden zuvor aufgezeichnete Werte im gleichen Segment oder Untersegment überschrieben.

Nutzen Sie das `annotations.`*`key`*-Schlüsselwort in einem Filterausdruck, um Ablaufverfolgungen durch Anmerkungen mit bestimmten Werten zu finden.

Example **`src/main/java/scorekeep/GameModel.java`**— Anmerkungen und Metadaten

```
import com.amazonaws.xray.AWSXRay;
```

```
import com.amazonaws.xray.entities.Segment;
import com.amazonaws.xray.entities.Subsegment;
...
  public void saveGame(Game game) throws SessionNotFoundException {
    // wrap in subsegment
    Subsegment subsegment = AWSXRay.beginSubsegment("## GameModel.saveGame");
    try {
      // check session
      String sessionId = game.getSession();
      if (sessionModel.loadSession(sessionId) == null ) {
        throw new SessionNotFoundException(sessionId);
      }
      Segment segment = AWSXRay.getCurrentSegment();
      subsegment.putMetadata("resources", "game", game);
      segment.putAnnotation("gameid", game.getId());
      mapper.save(game);
    } catch (Exception e) {
      subsegment.addException(e);
      throw e;
    } finally {
      AWSXRay.endSubsegment();
    }
  }
```

Metadaten mit dem X-Ray SDK for Java aufzeichnen

Verwenden Sie Metadaten, um Segment- oder Untersegmentinformationen aufzuzeichnen, die nicht zur Suche indiziert werden müssen. Metadatenwerte sind Zeichenfolgen, Zahlen, boolesche Werte oder andere Objekte, die in Form eines JSON-Objekts oder eines Arrays angeordnet sein können.

So zeichnen Sie Metadaten auf

1. Eine Referenz des aktuellen Segments oder Untersegments finden Sie unter `AWSXRay`.

   ```
   import com.amazonaws.xray.AWSXRay;
   import com.amazonaws.xray.entities.Segment;
   ...
   Segment document = AWSXRay.getCurrentSegment();
   ```

   or

   ```
   import com.amazonaws.xray.AWSXRay;
   ```

```
import com.amazonaws.xray.entities.Subsegment;
...
Subsegment document = AWSXRay.getCurrentSubsegment();
```

2. Rufen Sie `putMetadata` mit einem String-Namespace, einem Aktivierungsschlüssel sowie einem booleschen Wert, einer Zahl, einer Zeichenfolge oder einem Objektwert auf.

   ```
   document.putMetadata("my namespace", "my key", "my value");
   ```

   or

   Rufen Sie `putMetadata` nur mit einem Aktivierungsschlüssel und einem Wert auf.

   ```
   document.putMetadata("my key", "my value");
   ```

Wenn Sie keinen Namespace angeben, verwendet SDK `default`. Wenn `putMetadata` zweimal mit demselben Schlüssel aufgerufen wird, werden zuvor aufgezeichnete Werte im gleichen Segment oder Untersegment überschrieben.

Example **`src/main/java/scorekeep/GameModel.java`**— Anmerkungen und Metadaten

```
import com.amazonaws.xray.AWSXRay;
import com.amazonaws.xray.entities.Segment;
import com.amazonaws.xray.entities.Subsegment;
...
  public void saveGame(Game game) throws SessionNotFoundException {
    // wrap in subsegment
    Subsegment subsegment = AWSXRay.beginSubsegment("## GameModel.saveGame");
    try {
      // check session
      String sessionId = game.getSession();
      if (sessionModel.loadSession(sessionId) == null ) {
        throw new SessionNotFoundException(sessionId);
      }
      Segment segment = AWSXRay.getCurrentSegment();
      subsegment.putMetadata("resources", "game", game);
      segment.putAnnotation("gameid", game.getId());
      mapper.save(game);
    } catch (Exception e) {
      subsegment.addException(e);
      throw e;
```

```
    } finally {
      AWSXRay.endSubsegment();
    }
  }
```

Aufzeichnen von Benutzer-IDs mit dem X-Ray SDK for Java

Zeichnen Sie Benutzer-IDs in Anforderungssegmenten auf, um den Benutzer zu identifizieren, der die Anforderung gesendet hat.

So zeichnen Sie Benutzer-IDs auf

1. Eine Referenz des aktuellen Segments finden Sie unter `AWSXRay`.

   ```
   import com.amazonaws.xray.AWSXRay;
   import com.amazonaws.xray.entities.Segment;
   ...
   Segment document = AWSXRay.getCurrentSegment();
   ```

2. Rufen Sie `setUser` mit einer Zeichenfolgen-ID des Benutzers auf, der die Anforderung gesendet hat.

   ```
   document.setUser("U12345");
   ```

Sie können `setUser` in Ihrem Controller aufrufen, um die Benutzer-ID aufzuzeichnen, sobald die Anwendung mit der Bearbeitung einer Anfrage beginnt. Wenn Sie das Segment nur zur Einrichtung der Benutzer-ID verwenden, können Sie die Aufrufe in einer einzelnen Zeile anordnen.

Example src/main/java/scorekeep/ .java MoveController — Benutzer-ID

```
import com.amazonaws.xray.AWSXRay;
...
  @RequestMapping(value="/{userId}", method=RequestMethod.POST)
  public Move newMove(@PathVariable String sessionId, @PathVariable String
 gameId, @PathVariable String userId, @RequestBody String move) throws
 SessionNotFoundException, GameNotFoundException, StateNotFoundException,
 RulesException {
    AWSXRay.getCurrentSegment().setUser(userId);
    return moveFactory.newMove(sessionId, gameId, userId, move);
  }
```

Nutzen Sie das `user`-Schlüsselwort in einem Filterausdruck, um Ablaufverfolgungen einer Benutzer-ID zu finden.

## AWS X-Ray Metriken für das X-Ray SDK for Java

In diesem Thema werden der AWS X-Ray Namespace, die Metriken und die Dimensionen beschrieben. Sie können das X-Ray SDK for Java verwenden, um CloudWatch Amazon-Metriken ohne Stichproben aus Ihren gesammelten X-Ray-Segmenten zu veröffentlichen. Diese Metriken werden von der Start- und Endzeit des Segments sowie den Status-Flags für Fehler, Ausfall und Ablehnung abgeleitet. Mit diesen Trace-Metriken können Sie Wiederholungen und Abhängigkeitsprobleme in Teilsegmenten anzeigen.

CloudWatch ist ein Metrik-Repository. Eine Metrik ist das grundlegende Konzept von Datenpunkten CloudWatch und stellt eine zeitlich geordnete Menge von Datenpunkten dar. Sie (oder AWS-Services) veröffentlichen Metrikdatenpunkte in CloudWatch und rufen Statistiken über diese Datenpunkte als geordneten Satz von Zeitreihendaten ab.

Metriken werden eindeutig durch einen Namen, ein Namespace und eine oder mehrere Dimensionen definiert. Jeder Datenpunkt verfügt über einen Zeitstempel und optional über eine Maßeinheit. Wenn Sie Statistiken anfordern, wird der zurückgegebene Datenstrom durch den Namespace, den Metrik-Namen und die Dimension identifiziert.

Weitere Informationen zu CloudWatch finden Sie im CloudWatch Amazon-Benutzerhandbuch.

CloudWatch Röntgenmetriken

Der `ServiceMetrics/SDK`-Namespace enthält die folgenden Metriken.

| Metrik | Verfügbare Statistiken | Beschreibung | Einheiten |
| --- | --- | --- | --- |
| `Latency` | Durchschnitt, Minimum, Maximum, Anzahl | Die Differenz zwischen der Start- und Endzeit Durchschnitt, Minimum und Maximum beschreib en Betriebslatenz. „Anzahl“ beschreib | Millisekunden |

| Metrik | Verfügbare Statistiken | Beschreibung | Einheiten |
| --- | --- | --- | --- |
| | | t die Anzahl der Aufrufe. | |
| `ErrorRate` | Durchschnitt, Summe | Die Rate der Anforderungen, die mit dem Statuscod e „`4xx Client Error`“ fehlgesch lagen sind, was zu einem Fehler führt. | Prozent |
| `FaultRate` | Durchschnitt, Summe | Die Rate der Traces, die mit dem Statuscod e „`5xx Server Error`“ fehlgesch lagen sind, was zu einem Fehler führte. | Prozent |
| `ThrottleRate` | Durchschnitt, Summe | Die Rate der abgelehnten Traces, die einen `429`-Statuscode zurückgeb en. Dies ist eine Teilmenge der `ErrorRate` -Metrik. | Prozent |
| `OkRate` | Durchschnitt, Summe | Die Rate der verfolgte n Anforderungen, die zu einem `OK`-Statusco de führen. | Prozent |

CloudWatch Röntgenabmessungen

Verwenden Sie die Dimensionen in der folgenden Tabelle, um die für Ihre Java Anwendungen mit Röntgeninstrumenten zurückgegebenen Metriken zu verfeinern.

| Dimension | Beschreibung |
| --- | --- |
| `ServiceType` | Der Service-Typ, z. B. `AWS::EC2::Instance` oder `NONE`, falls nicht bekannt. |
| `ServiceName` | Der kanonische Name für den Service. |

CloudWatch X-Ray-Metriken aktivieren

Gehen Sie wie folgt vor, um Trace-Metriken in Ihrer instrumentierten Java Anwendung zu aktivieren.

So konfigurieren Sie Trace-Metriken

1. Fügen Sie das `aws-xray-recorder-sdk-metrics` Paket als Apache Maven Abhängigkeit hinzu. Weitere Informationen finden Sie unter X-Ray SDK for Java Java-Submodule.
2. Aktivieren Sie ein neues `MetricsSegmentListener()` als Teil des globalen Recorder-Builds.

   Example src/com/myapp/web/Startup.java

```
import com.amazonaws.xray.AWSXRay;
import com.amazonaws.xray.AWSXRayRecorderBuilder;
import com.amazonaws.xray.plugins.EC2Plugin;
import com.amazonaws.xray.plugins.ElasticBeanstalkPlugin;
import com.amazonaws.xray.strategy.sampling.LocalizedSamplingStrategy;

@Configuration
public class WebConfig {
...
  static {
    AWSXRayRecorderBuilder builder = AWSXRayRecorderBuilder
                                        .standard()
                                        .withPlugin(new EC2Plugin())
                                        .withPlugin(new ElasticBeanstalkPlugin())
                                        .withSegmentListener(new
 MetricsSegmentListener());

    URL ruleFile = WebConfig.class.getResource("/sampling-rules.json");
    builder.withSamplingStrategy(new LocalizedSamplingStrategy(ruleFile));

    AWSXRay.setGlobalRecorder(builder.build());
  }
```

```
}
```

3. Stellen Sie den CloudWatch Agenten bereit, um Metriken mithilfe von Amazon Elastic Compute Cloud (Amazon EC2), Amazon Elastic Container Service (Amazon ECS) oder Amazon Elastic Kubernetes Service (Amazon EKS) zu sammeln:

   - Informationen zur Konfiguration von Amazon EC2 finden Sie unter Installation des CloudWatch Agenten.
   - Informationen zur Konfiguration von Amazon ECS finden Sie unter Überwachen von Amazon ECS-Containern mithilfe von Container Insights.
   - Informationen zur Konfiguration von Amazon EKS finden Sie unter Installieren des CloudWatch Agenten mithilfe des Amazon CloudWatch Observability EKS-Add-ons.

4. Konfigurieren Sie das SDK für die Kommunikation mit dem CloudWatch Agenten. Standardmäßig kommuniziert das SDK mit dem CloudWatch Agenten über die Adresse`127.0.0.1`. Sie können alternative Adressen konfigurieren, indem Sie die Umgebungsvariable oder die Java-Eigenschaft auf `address:port` festlegen.

   Example Umgebungsvariable

   ```
   AWS_XRAY_METRICS_DAEMON_ADDRESS=address:port
   ```

   Example Java-Eigenschaft

   ```
   com.amazonaws.xray.metrics.daemonAddress=address:port
   ```

So überprüfen Sie die Konfiguration

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Öffnen Sie die Registerkarte Metriken, um den Zustrom Ihrer Metriken zu überwachen.
3. (Optional) Öffnen Sie in der CloudWatch Konsole auf der Registerkarte Protokolle die `ServiceMetricsSDK` Protokollgruppe. Suchen Sie nach einem Protokolldatenstrom, der den Host-Metriken entspricht, und bestätigen Sie die Protokollmeldungen.

## Übermitteln von Segmentkontext zwischen Threads in einer Multithread-Anwendung

Beim Erstellen eines neuen Threads in Ihrer Anwendung behält `AWSXRayRecorder` eine Referenz zur aktuellen Segment- oder Untersegment-Entity nicht bei. Wenn Sie im neuen Thread einen instrumentierten Client verwenden, versucht das SDK, in ein Segment zu schreiben, das nicht existiert, was zu einem führt. SegmentNotFoundException

Um zu vermeiden, dass während der Entwicklung Ausnahmen ausgelöst werden, können Sie den Rekorder so konfigurieren ContextMissingStrategy, dass er stattdessen einen Fehler protokollieren soll. Sie können die Strategie im Code mit einer SetContextMissingStrategyUmgebungsvariablen oder einer Systemeigenschaft konfigurieren oder entsprechende Optionen konfigurieren.

Eine Möglichkeit zur Behebung des Fehlers ist die Verwendung eines neuen Segments, indem Sie beginSegment aufrufen, wenn Sie den Thread starten, und endSegment, wenn Sie ihn schließen. Dies funktioniert, wenn Sie Code instrumentieren, der nicht als Reaktion auf eine HTTP-Anforderung ausgeführt wird, z. B. Code, der beim Starten Ihrer Anwendung ausgeführt wird.

Wenn Sie mehrere Threads zur Verarbeitung eingehender Anfragen verwenden, können Sie das aktuelle Segment oder Untersegment an den neuen Thread übergeben und für die globale Aufzeichnung bereitstellen. Auf diese Weise wird sichergestellt, dass die im neuen Thread aufgezeichneten Informationen mit demselben Segment verknüpft werden wie die übrigen zu dieser Anfrage aufgezeichneten Informationen. Sobald das Segment im neuen Thread verfügbar ist, können Sie mithilfe der Methode jedes Runnable ausführen, das Zugriff auf den `segment.run(() -> { ... })` Kontext dieses Segments hat.

Ein Beispiel finden Sie unter Verwenden instrumentierter Clients in Auftragnehmer-Threads.

X-Ray mit asynchroner Programmierung verwenden

Das X-Ray SDK for Java kann in asynchronen Java-Programmen mit SegmentContextExecutorsverwendet werden. Das SegmentContextExecutor implementiert das Executor-Interface, was bedeutet, dass es an alle asynchronen Operationen von a übergeben werden kann. CompletableFuture Dadurch wird sichergestellt, dass alle asynchronen Operationen mit dem richtigen Segment in ihrem Kontext ausgeführt werden.

Example Beispiel App.java: Übergabe an SegmentContextExecutor CompletableFuture

```
DynamoDbAsyncClient client = DynamoDbAsyncClient.create();
```

```
AWSXRay.beginSegment();

// ...

client.getItem(request).thenComposeAsync(response -> {
    // If we did not provide the segment context executor, this request would not be
 traced correctly.
    return client.getItem(request2);
}, SegmentContextExecutors.newSegmentContextExecutor());
```

## AOP mit Spring und dem X-Ray SDK for Java

In diesem Thema wird beschrieben, wie Sie das X-Ray SDK und das Spring Framework verwenden, um Ihre Anwendung zu instrumentieren, ohne ihre Kernlogik zu ändern. Das bedeutet, dass es jetzt eine nichtinvasive Methode zur Instrumentierung Ihrer Anwendungen gibt, die remote ausgeführt werden. AWS

So aktivieren Sie AOP in Spring

1. Konfigurieren von Spring
2. Fügen Sie Ihrer Anwendung einen Tracing-Filter hinzu
3. Kommentieren Ihres Codes oder implementieren einer Schnittstelle
4. Aktivieren von X-Ray in Ihrer Anwendung

### Konfigurieren von Spring

Sie können Maven oder Gradle verwenden, um Spring zu konfigurieren, um AOP für die Instrumentierung Ihrer Anwendung verwenden zu können.

Wenn Sie Ihre Anwendung mit Maven erstellen, fügen Sie die folgende Abhängigkeit in Ihrer `pom.xml`-Datei hinzu.

```
<dependency>
     <groupId>com.amazonaws</groupId>
     <artifactId>aws-xray-recorder-sdk-spring</artifactId>
     <version>2.11.0</version>
</dependency>
```

Für Gradle fügen Sie die folgende Abhängigkeit in Ihre `build.gradle`-Datei ein.

```
compile 'com.amazonaws:aws-xray-recorder-sdk-spring:2.11.0'
```

Spring Boot konfigurieren

Wenn Sie Spring Boot verwenden, fügen Sie zusätzlich zu der im vorherigen Abschnitt beschriebenen Spring-Abhängigkeit die folgende Abhängigkeit hinzu, sofern sie nicht bereits in Ihrem Klassenpfad enthalten ist.

Maven:

```
<dependency>
     <groupId>org.springframework.boot</groupId>
     <artifactId>spring-boot-starter-aop</artifactId>
     <version>2.5.2</version>
</dependency>
```

Gradle:

```
compile 'org.springframework.boot:spring-boot-starter-aop:2.5.2'
```

Hinzufügen eines Tracing-Filters zu Ihrer Anwendung

Fügen Sie Ihrer `WebConfig` Klasse eine `Filter` hinzu. Übermitteln Sie den Segmentnamen als Zeichenfolge an den `AWSXRayServletFilter`-Konstruktor. Weitere Informationen zur Verfolgung von Filtern und zur Instrumentierung eingehender Anfragen finden Sie unter. Nachverfolgung eingehender Anfragen mit dem X-Ray SDK for Java

Example src/main/java/myapp/ .java WebConfig — Frühling

```
package myapp;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.Bean;
import javax.servlet.Filter;
import com.amazonaws.xray.javax.servlet.AWSXRayServletFilter;

@Configuration
public class WebConfig {

  @Bean
  public Filter TracingFilter() {
    return new AWSXRayServletFilter("Scorekeep");
  }
```

```
}
```

Jakarta-Unterstützung

Spring 6 verwendet Jakarta anstelle von Javax für seine Enterprise Edition. Um diesen neuen Namespace zu unterstützen, hat X-Ray einen parallel Satz von Klassen erstellt, die in ihrem eigenen Jakarta-Namespace leben.

Ersetzen Sie für die Filterklassen durch. `javax` `jakarta` Wenn Sie eine Segmentbenennungsstrategie konfigurieren, fügen Sie `jakarta` vor der Benennungsstrategie den Klassennamen hinzu, wie im folgenden Beispiel:

```
package myapp;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.Bean;
import jakarta.servlet.Filter;
import com.amazonaws.xray.jakarta.servlet.AWSXRayServletFilter;
import com.amazonaws.xray.strategy.jakarta.SegmentNamingStrategy;

@Configuration
public class WebConfig {
    @Bean
    public Filter TracingFilter() {
        return new AWSXRayServletFilter(SegmentNamingStrategy.dynamic("Scorekeep"));
    }
}
```

Kommentieren Ihres Codes oder Implementieren einer Schnittstelle

Ihre Klassen müssen entweder mit der `@XRayEnabled` Anmerkung versehen sein oder die `XRayTraced` Schnittstelle implementieren. Damit wird das AOP-System angewiesen, die Funktionen der betroffenen Klasse für die X-Ray-Instrumentierung zu kapseln.

X-Ray in Ihrer Anwendung aktivieren

Um X-Ray Tracing in Ihrer Anwendung zu aktivieren, muss Ihr Code die abstrakte Klasse erweitern, `BaseAbstractXRayInterceptor` indem er die folgenden Methoden überschreibt.

- `generateMetadata`— Diese Funktion ermöglicht die Anpassung der Metadaten, die an den Trace der aktuellen Funktion angehängt sind. Standardmäßig wird der Klassenname der ausgeführten Funktion in den Metadaten aufgezeichnet. Sie können weitere Daten hinzufügen, wenn Sie zusätzliche Informationen benötigen.

- `xrayEnabledClasses`— Diese Funktion ist leer und sollte es auch bleiben. Sie dient als Host für ein Pointcut, das den Interceptor anweist, welche Methoden gekapselt werden sollen. Definieren Sie das Pointcut, indem Sie angeben, welche der Klassen mit `@XRayEnabled` kommentiert sind, um ein Tracing durchzuführen. Die folgende pointcut-Anweisung weist den Interceptor an, alle Controller-Beans einzukapseln, die mit dem Kommentar `@XRayEnabled` gekennzeichnet sind.

```
@Pointcut(“@within(com.amazonaws.xray.spring.aop.XRayEnabled) && bean(*Controller)”)
```

Wenn Ihr Projekt Spring Data JPA verwendet, sollten Sie eine Erweiterung von `AbstractXRayInterceptor` anstelle von in Betracht ziehen. `BaseAbstractXRayInterceptor`

Beispiel

Der folgende Code erweitert die abstrakte Klasse`BaseAbstractXRayInterceptor`.

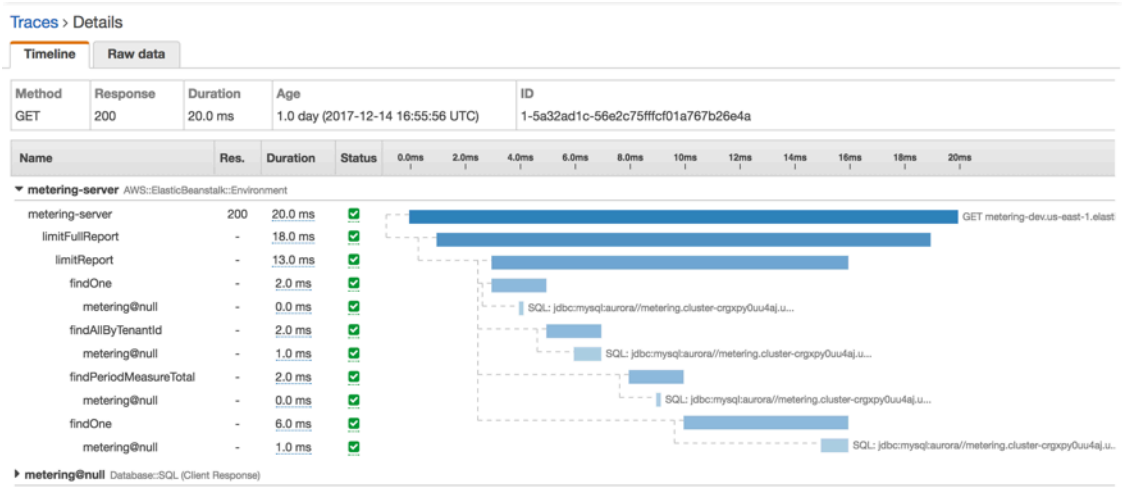
```
@Aspect
@Component
public class XRayInspector extends BaseAbstractXRayInterceptor {
    @Override
    protected Map<String, Map<String, Object>> generateMetadata(ProceedingJoinPoint
 proceedingJoinPoint, Subsegment subsegment) throws Exception {
        return super.generateMetadata(proceedingJoinPoint, subsegment);
    }

  @Override
  @Pointcut("@within(com.amazonaws.xray.spring.aop.XRayEnabled) && bean(*Controller)")

  public void xrayEnabledClasses() {}

}
```

Der folgenden Code ist eine Klasse, die von X-Ray instrumentiert wird.

```
@Service
@XRayEnabled
public class MyServiceImpl implements MyService {
    private final MyEntityRepository myEntityRepository;

    @Autowired
    public MyServiceImpl(MyEntityRepository myEntityRepository) {
```

```
        this.myEntityRepository = myEntityRepository;
    }

    @Transactional(readOnly = true)
    public List<MyEntity> getMyEntities(){
        try(Stream<MyEntity> entityStream = this.myEntityRepository.streamAll()){

            return entityStream.sorted().collect(Collectors.toList());
        }
    }
}
```

Wenn Sie Ihre Anwendung ordnungsgemäß konfiguriert haben, sollten Sie den vollständigen Aufruf-Stack der Anwendung sehen, vom Controller bis zu den Service-Aufrufe, wie im folgenden Screenshot der Konsole gezeigt.




# Instrumentieren Sie Ihre Bewerbung mit Node.js

Es gibt zwei Möglichkeiten, Ihre Anwendung Node.js so zu instrumentieren, dass sie Traces an X-Ray sendet:

- AWS Distro for OpenTelemetry JavaScript — Eine AWS Distribution, die eine Reihe von Open-Source-Bibliotheken zum Senden korrelierter Metriken und Traces über AWS Distro for Collector an mehrere AWS Überwachungslösungen CloudWatch AWS X-Ray, darunter Amazon und Amazon OpenSearch Service, bereitstellt. OpenTelemetry
- AWS X-Ray SDK für Node.js — Eine Reihe von Bibliotheken zum Generieren und Senden von Traces an X-Ray über den X-Ray-Daemon.

Weitere Informationen finden Sie unter Wahl zwischen den AWS SDKs Distro for OpenTelemetry und X-Ray.

## AWSDistro fürOpenTelemetry JavaScript

Mit demAWSDistro fürOpenTelemetry(ADOPTIEREN)JavaScript, können Sie Ihre Anwendungen einmal instrumentieren und korrelierte Metriken und Traces an mehrere sendenAWSÜberwachungslösungen, einschließlich AmazonCloudWatch,AWS X-Ray, und AmazonOpenSearchBedienung. X-Ray verwenden mitAWSDistribution fürOpenTelemetrybenötigt zwei Komponenten: eineOpenTelemetrySDKaktiviert für die Verwendung mit X-Ray und demAWSDistribution fürOpenTelemetrySammlerfür die Verwendung mit X-Ray aktiviert.

Informationen zu den ersten Schritten finden Sie unterAWSDistribution fürOpenTelemetry JavaScriptDokumentation.

> **Note**
>
> ADOPTIERENJavaScriptwird für alle serverseitigen Node.js Anwendungen unterstützt. ADOPTIERENJavaScriptist nicht in der Lage, Daten von Browser-Clients nach X-Ray zu exportieren.

Weitere Informationen zur Verwendung vonAWSDistribution fürOpenTelemetrymitAWS X-Rayund andereAWS-Services, sieheAWSDistribution fürOpenTelemetryoder dieAWSDistribution fürOpenTelemetryDokumentation.

Weitere Informationen zur Sprachunterstützung und Sprachverwendung finden Sie unterAWSBeobachtbarkeit aufGitHub.

## AWS X-Ray-SDK für Node.js

Das X-Ray-SDK für Node.js ist eine Bibliothek für Express-Webanwendungen und Node.js Lambda-Funktionen, die Klassen und Methoden zum Generieren und Senden von Trace-Daten an den X-Ray-Daemon bereitstellt. Zu den Trace-Daten gehören Informationen über eingehende HTTP-Anfragen, die von der Anwendung bedient werden, sowie über Aufrufe, die die Anwendung mithilfe des AWS SDK oder der HTTP-Clients an nachgeschaltete Dienste sendet.

> **Note**
>
> Das X-Ray SDK für Node.js ist ein Open-Source-Projekt, das für Node.js Versionen 14.x und höher unterstützt wird. Sie können das Projekt verfolgen und Probleme und Pull-Requests einreichen unter GitHub: github.com/aws/ aws-xray-sdk-node

Wenn Sie Express nutzen, beginnen Sie, indem Sie das SDK als Middleware auf Ihrem Anwendungsserver hinzufügen, um eingehende Anforderungen zu verfolgen. Der Middleware erstellt für jede verfolgte Anforderung ein Segment und vervollständigt das Segment, nachdem die Antwort gesendet wurde. Während das Segment geöffnet ist, können Sie die SDK-Client-Methoden nutzen, um dem Segment Informationen hinzuzufügen, Untersegmente zu erstellen und nachgelagerte Aufrufe rückzuverfolgen. Das SDK erfasst auch automatisch Ausnahmen, die Ihre Anwendung ausgibt, während das Segment geöffnet ist.

Bei Lambda-Funktionen, die von einer instrumentierten Anwendung oder einem Dienst aufgerufen werden, liest Lambda den Tracing-Header und verfolgt automatisch Sampling-Anfragen. Für andere Funktionen können Sie Lambda so konfigurieren, dass eingehende Anfragen abgefragt und verfolgt werden. In beiden Fällen erstellt Lambda das Segment und stellt es dem X-Ray SDK zur Verfügung.

> **Note**
>
> Auf Lambda ist das X-Ray SDK optional. Wenn Sie es nicht in Ihrer Funktion verwenden, enthält Ihre Service-Map immer noch einen Knoten für den Lambda-Service und einen für jede Lambda-Funktion. Durch Hinzufügen des SDK können Sie Ihren Funktionscode instrumentieren, um Untersegmente zu dem von Lambda aufgezeichneten Funktionssegment hinzuzufügen. Weitere Informationen finden Sie unter AWS Lambda und AWS X-Ray.

Verwenden Sie als Nächstes das X-Ray-SDK für Node.js, um Ihr AWS SDK für die Clients JavaScript in Node.js zu instrumentieren. Immer wenn Sie einen Downstream AWS-Service oder eine Ressource mit einem instrumentierten Client aufrufen, zeichnet das SDK Informationen über den Anruf in einem Untersegment auf. AWS-Services und die Ressourcen, auf die Sie innerhalb der Services zugreifen, werden in der Trace-Map als Downstream-Knoten angezeigt, sodass Sie Fehler und Drosselungsprobleme bei einzelnen Verbindungen leichter identifizieren können.

Das X-Ray-SDK für Node.js bietet auch Instrumentierung für Downstream-Aufrufe von HTTP-Web-APIs und SQL-Abfragen. Umhüllen Sie den HTTP-Client in der SDK-Erfassungsmethode, um

Informationen zu ausgehenden HTTP-Anforderungen aufzuzeichnen. Für SQL-Clients verwenden Sie die Capture-Methode für Ihre Datenbank.

Die Middleware wendet Samplingregeln auf eingehende Anforderungen an, um zu ermitteln, welche Anforderungen rückverfolgt werden. Sie können das X-Ray SDK für Node.js konfigurieren, um das Sampling-Verhalten anzupassen oder Informationen über die AWS Rechenressourcen aufzuzeichnen, auf denen Ihre Anwendung ausgeführt wird.

Zeichnen Sie zusätzliche Informationen zu Anforderungen und den Aufgaben, die Ihre Anwendung ausführt, in Anmerkungen und Metadaten auf. Anmerkungen sind einfache Schlüsselwertpaare, die für die Verwendung mit Filterausdrücken indiziert werden, damit Sie nach Ablaufverfolgen mit bestimmten Daten suchen können. Metadateneinträge sind weniger einschränkend und können ganze Objekte und Arrays aufzeichnen – alle Daten, die in eine JSON zusammengefasst werden können.

> **Anmerkungen und Metadaten**
>
> Anmerkungen und Metadaten sind beliebiger Text, den Sie Segmenten mit dem X-Ray SDK hinzufügen. Anmerkungen werden für die Verwendung mit Filterausdrücken indexiert. Metadaten werden nicht indexiert, können aber im Rohsegment mit der X-Ray-Konsole oder API angezeigt werden. Jeder, dem Sie Lesezugriff auf X-Ray gewähren, kann diese Daten einsehen.

Wenn Sie viele instrumentierten Clients in Ihrem Code haben, kann ein einzelnes Anforderungssegmente viele Untersegmente enthalten, eines für jeden Aufruf mit einem instrumentierten Client. Sie können Untersegmente organisieren und gruppieren, indem Sie Client-Aufrufe in benutzerdefinierten Untersegmenten zusammenfassen. Sie können ein benutzerdefiniertes Untersegment für eine ganze Funktion oder eine Code-Abschnitt erstellen und Metadaten und Anmerkungen im Untersegment festhalten, anstatt alles im übergeordneten Segment aufzuzeichnen.

Referenzdokumentation zu den Klassen und Methoden des SDK finden Sie in der API-Referenz zum AWS X-Ray SDK for Node.js.

## Voraussetzungen

Das X-Ray SDK für Node.js benötigt Node.js und die folgenden Bibliotheken:

- `atomic-batcher`— 1.0.2

- `cls-hooked`— 4.2.2
- `pkginfo`— 0,4,0
- `semver`— 5,3,0

Das SDK zieht diese Bibliotheken bei der Installation in NPM ein.

Um AWS SDK-Clients verfolgen zu können, benötigt das X-Ray-SDK für Node.js eine Mindestversion des AWS SDK für JavaScript in Node.js.

- `aws-sdk`— 2.7.15

## Abhängigkeitsmanagement

Das X-Ray-SDK für Node.js ist bei NPM erhältlich.

- Package — `aws-xray-sdk`

Installieren Sie das SDK für eine lokale Bereitstellung in Ihrem Projektverzeichnis mit npm.

```
~/nodejs-xray$ npm install aws-xray-sdk
aws-xray-sdk@3.3.3
  ### aws-xray-sdk-core@3.3.3
  # ### @aws-sdk/service-error-classification@3.15.0
  # ### @aws-sdk/types@3.15.0
  # ### @types/cls-hooked@4.3.3
  # # ### @types/node@15.3.0
  # ### atomic-batcher@1.0.2
  # ### cls-hooked@4.2.2
  # # ### async-hook-jl@1.7.6
  # # # ### stack-chain@1.3.7
  # # ### emitter-listener@1.1.2
  # #   ### shimmer@1.2.1
  # ### semver@5.7.1
  ### aws-xray-sdk-express@3.3.3
  ### aws-xray-sdk-mysql@3.3.3
  ### aws-xray-sdk-postgres@3.3.3
```

Verwenden Sie die `--save`-Option zum Speichern von SDK in Abhängigkeit von `package.json` in Ihrer Anwendung.

```
~/nodejs-xray$ npm install aws-xray-sdk --save
aws-xray-sdk@3.3.3
```

Wenn Ihre Anwendung Abhängigkeiten hat, deren Versionen mit den Abhängigkeiten des X-Ray-SDK in Konflikt stehen, werden beide Versionen installiert, um die Kompatibilität sicherzustellen. Weitere Informationen finden Sie in der offiziellen NPM-Dokumentation zur Auflösung von Abhängigkeiten.

## Node.js-Beispiele

Verwenden Sie das AWS X-Ray SDK für Node.js, um sich einen end-to-end Überblick über die Anfragen zu verschaffen, die Ihre Node.js -Anwendungen durchlaufen.

- Node.js Beispielanwendung aktiviert GitHub.

## Konfiguration des X-Ray-SDK für Node.js

Sie können das X-Ray-SDK für Node.js mit Plug-ins so konfigurieren, dass es Informationen über den Dienst enthält, auf dem Ihre Anwendung ausgeführt wird, das standardmäßige Sampling-Verhalten ändern oder Sampling-Regeln hinzufügen, die für Anfragen an bestimmte Pfade gelten.

Sections

- Service-Plugins
- Samplingregeln
- Protokollierung
- Adresse des X-Ray-Daemons
- Umgebungsvariablen

### Service-Plugins

Wird verwendet`plugins`, um Informationen über den Dienst aufzuzeichnen, der Ihre Anwendung hostet.

Plug-ins

- Amazon EC2 — `EC2Plugin` fügt die Instance-ID, die Availability Zone und die CloudWatch Logs-Gruppe hinzu.

- Elastic Beanstalk — `ElasticBeanstalkPlugin` fügt den Umgebungsnamen, die Versionsbezeichnung und die Bereitstellungs-ID hinzu.
- Amazon ECS — `ECSPlugin` fügt die Container-ID hinzu.

Um ein Plug-in zu verwenden, konfigurieren Sie das X-Ray SDK für den Client Node.js mithilfe der `config` Methode.

Example app.js – Plugins

```
var AWSXRay = require('aws-xray-sdk');
AWSXRay.config([AWSXRay.plugins.EC2Plugin,AWSXRay.plugins.ElasticBeanstalkPlugin]);
```

Das SDK verwendet auch Plugin-Einstellungen, um das `origin` Feld für das Segment festzulegen. Dies gibt den AWS Ressourcentyp an, auf dem Ihre Anwendung ausgeführt wird. Wenn Sie mehrere Plugins verwenden, verwendet das SDK die folgende Auflösungsreihenfolge, um den Ursprung zu bestimmen: ElasticBeanstalk > EKS > ECS > EC2.

Samplingregeln

Das SDK verwendet die Sampling-Regeln, die Sie in der X-Ray-Konsole definieren, um zu bestimmen, welche Anfragen aufgezeichnet werden sollen. Die Standardregel verfolgt die erste Anfrage jede Sekunde und fünf Prozent aller weiteren Anfragen aller Dienste, die Traces an X-Ray senden. Erstellen Sie zusätzliche Regeln in der X-Ray-Konsole, um die Menge der aufgezeichneten Daten für jede Ihrer Anwendungen anzupassen.

Das SDK wendet benutzerdefinierte Regeln in der Reihenfolge an, in der sie definiert sind. Wenn eine Anfrage mehreren benutzerdefinierten Regeln entspricht, wendet das SDK nur die erste Regel an.

> **Note**
>
> Wenn das SDK X-Ray nicht erreichen kann, um Sampling-Regeln abzurufen, kehrt es zu einer lokalen Standardregel zurück, die die erste zu Beginn jeder Sekunde empfangene Anfrage und fünf Prozent aller zusätzlichen Anfragen pro Host festlegt. Dies kann passieren, wenn der Host nicht berechtigt ist, Sampling-APIs aufzurufen, oder wenn er keine Verbindung zum X-Ray-Daemon herstellen kann, der als TCP-Proxy für API-Aufrufe durch das SDK fungiert.

Sie können das SDK auch so konfigurieren, dass Sampling-Regeln aus einem JSON-Dokument geladen werden. Das SDK kann lokale Regeln als Backup für Fälle verwenden, in denen X-Ray Sampling nicht verfügbar ist, oder ausschließlich lokale Regeln verwenden.

Example sampling-rules.json

```
{
  "version": 2,
  "rules": [
    {
      "description": "Player moves.",
      "host": "*",
      "http_method": "*",
      "url_path": "/api/move/*",
      "fixed_target": 0,
      "rate": 0.05
    }
  ],
  "default": {
    "fixed_target": 1,
    "rate": 0.1
  }
}
```

In diesem Beispiel werden eine benutzerdefinierte Regel und eine Standardregel definiert. Die benutzerdefinierte Regel wendet eine Stichprobenrate von fünf Prozent an, ohne dass eine Mindestanzahl von Anfragen für Pfade verfolgt werden muss. `/api/move/` Die Standardregel verfolgt die erste Anfrage jede Sekunde und 10 Prozent der weiteren Anfragen.

Der Nachteil der lokalen Definition von Regeln besteht darin, dass das feste Ziel von jeder Instanz des Rekorders unabhängig angewendet wird, anstatt vom X-Ray-Dienst verwaltet zu werden. Wenn Sie mehr Hosts bereitstellen, wird die feste Rate vervielfacht, wodurch es schwieriger wird, die Menge der aufgezeichneten Daten zu kontrollieren.

Wenn aktiviert AWS Lambda, können Sie die Samplerate nicht ändern. Wenn Ihre Funktion von einem instrumentierten Dienst aufgerufen wird, werden Aufrufe, die Anfragen generierten, die von diesem Dienst abgetastet wurden, von Lambda aufgezeichnet. Wenn aktives Tracing aktiviert ist und kein Tracing-Header vorhanden ist, trifft Lambda die Stichprobenentscheidung.

Um Backup-Regeln zu konfigurieren, weisen Sie das X-Ray SDK for Node.js an, Sampling-Regeln aus einer Datei mit zu lade`setSamplingRules`n.

Example app.js – Sampling-Regeln aus einer Datei

```
var AWSXRay = require('aws-xray-sdk');
AWSXRay.middleware.setSamplingRules('sampling-rules.json');
```

Sie können Ihre Regeln auch in Code definieren und als Objekt an `setSamplingRules` übergeben.

Example app.js – Sampling-Regeln aus einem Objekt

```
var AWSXRay = require('aws-xray-sdk');
var rules = {
  "rules": [ { "description": "Player moves.", "service_name": "*", "http_method": "*",
 "url_path": "/api/move/*", "fixed_target": 0, "rate": 0.05 } ],
  "default": { "fixed_target": 1, "rate": 0.1 },
  "version": 1
  }

AWSXRay.middleware.setSamplingRules(rules);
```

Um nur lokale Regeln zu verwenden, rufen Sie `disableCentralizedSampling` auf.

```
AWSXRay.middleware.disableCentralizedSampling()
```

Protokollierung

Zum Protokollieren der SDK-Ausgabe rufen Sie `AWSXRay.setLogger(logger)` auf, wobei `logger` ein Objekt ist, dass Standard-Protokollierungsmethoden (`warn`, `info`usw.) bereitstellt.

Standardmäßig protokolliert das SDK Fehlermeldungen auf der Konsole mithilfe der Standardmethoden für das Konsolenobjekt. Die Protokollebene des integrierten Loggers kann entweder mithilfe der `AWS_XRAY_LOG_LEVEL` Umgebungsvariablen `AWS_XRAY_DEBUG_MODE` oder festgelegt werden. Eine Liste der gültigen Werte auf Protokollebene finden Sie unter Umgebungsvariablen.

Wenn Sie ein anderes Format oder ein anderes Ziel für die Protokolle angeben möchten, können Sie dem SDK Ihre eigene Implementierung der Logger-Schnittstelle zur Verfügung stellen, wie unten gezeigt. Jedes Objekt, das diese Schnittstelle implementiert, kann verwendet werden. Das bedeutet, dass viele Logging-Bibliotheken, z. B. Winston, verwendet und direkt an das SDK übergeben werden könnten.

Example app.js – Protokollierung

```
var AWSXRay = require('aws-xray-sdk');

// Create your own logger, or instantiate one using a library.
var logger = {
  error: (message, meta) => { /* logging code */ },
  warn: (message, meta) => { /* logging code */ },
  info: (message, meta) => { /* logging code */ },
  debug: (message, meta) => { /* logging code */ }
}

AWSXRay.setLogger(logger);
AWSXRay.config([AWSXRay.plugins.EC2Plugin]);
```

Rufen Sie vor der Ausführung anderer Konfigurationsmethoden `setLogger` auf, um sicherzustellen, dass Sie die Ausgabe dieser Vorgänge erfassen.

Adresse des X-Ray-Daemons

Wenn der X-Ray-Daemon auf einem anderen Port oder Host lauscht als`127.0.0.1:2000`, können Sie das X-Ray-SDK für Node.js so konfigurieren, dass Trace-Daten an eine andere Adresse gesendet werden.

```
AWSXRay.setDaemonAddress('host:port');
```

Sie können den Host mit Namen oder IPv4-Adresse angeben.

Example app.js – Daemon-Adresse

```
var AWSXRay = require('aws-xray-sdk');
AWSXRay.setDaemonAddress('daemonhost:8082');
```

Wenn Sie den Daemon so konfiguriert haben, dass er auf verschiedenen Ports für TCP und UDP wartet, können Sie beides in den Einstellungen für die Daemon-Adresse angeben.

Example app.js – Daemon-Adresse auf separaten Ports

```
var AWSXRay = require('aws-xray-sdk');
AWSXRay.setDaemonAddress('tcp:daemonhost:8082 udp:daemonhost:8083');
```

Sie können die Daemon-Adresse auch festlegen, indem Sie die `AWS_XRAY_DAEMON_ADDRESS` Umgebungsvariable verwenden.

Umgebungsvariablen

Sie können Umgebungsvariablen verwenden, um das X-Ray SDK für Node.js zu konfigurieren. Das SDK unterstützt die folgenden Variablen.

- `AWS_XRAY_CONTEXT_MISSING`— Legt fest, `RUNTIME_ERROR` dass Ausnahmen ausgelöst werden, wenn Ihr instrumentierter Code versucht, Daten aufzuzeichnen, obwohl kein Segment geöffnet ist.

  Zulässige Werte

  - `RUNTIME_ERROR`— Löst eine Laufzeitausnahme aus.
  - `LOG_ERROR`— Einen Fehler protokollieren und fortfahren (Standard).
  - `IGNORE_ERROR`— Fehler ignorieren und fortfahren.

  Fehler im Zusammenhang mit fehlenden Segmenten oder Untersegmenten können auftreten, wenn Sie versuchen, einen instrumentierten Client in Startcode zu verwenden, der ausgeführt wird, wenn keine Anfrage geöffnet ist, oder in Code, der einen neuen Thread erzeugt.
- `AWS_XRAY_DAEMON_ADDRESS`— Legt den Host und den Port des X-Ray-Daemon-Listeners fest. Standardmäßig verwendet `127.0.0.1:2000` das SDK sowohl Trace-Daten (UDP) als auch Sampling-Daten (TCP). Verwenden Sie diese Variable, wenn Sie den Daemon so konfiguriert haben, dass er auf einem anderen Port lauscht oder wenn er auf einem anderen Host läuft.

  Format

  - Derselbe Port — *`address`*`:`*`port`*
  - Verschiedene Anschlüsse — `tcp:`*`address`*`:`*`port`* `udp:`*`address`*`:`*`port`*
- `AWS_XRAY_DEBUG_MODE`— Stellen Sie diese Option ein`TRUE`, um das SDK so zu konfigurieren, dass es Protokolle auf `debug` Ebene 2 ausgibt.
- `AWS_XRAY_LOG_LEVEL` — Legt eine Protokollebene für den Standard-Logger fest. Gültige Werte sind `debug`, `info`, `warn`, `error` und `silent`. Dieser Wert wird ignoriert, wenn AWS_XRAY_DEBUG_MODE auf `TRUE` gesetzt ist.
- `AWS_XRAY_TRACING_NAME`— Legen Sie einen Dienstnamen fest, den das SDK für Segmente verwendet. Überschreibt den für die Express-Middleware festgelegten Segmentnamen.

## Nachverfolgung eingehender Anfragen mit dem X-Ray SDK für Node.js

Sie können das X-Ray-SDK für Node.js verwenden, um eingehende HTTP-Anfragen zu verfolgen, die Ihre Express- und Restify-Anwendungen auf einer EC2-Instance in Amazon EC2 oder Amazon ECS bearbeiten. AWS Elastic Beanstalk

Das X-Ray SDK für Node.js bietet Middleware für Anwendungen, die die Express- und Restify-Frameworks verwenden. Wenn Sie die X-Ray-Middleware zu Ihrer Anwendung hinzufügen, erstellt das X-Ray-SDK für Node.js ein Segment für jede gesampelte Anfrage. Dieses Segment umfasst Dauer, Methode und Status der HTTP-Anforderung. Die zusätzliche Instrumentierung schafft Untersegmente zu diesem Segment.

> **Note**
>
> Für AWS Lambda Funktionen erstellt Lambda für jede abgetastete Anfrage ein Segment. Weitere Informationen finden Sie unter AWS Lambda und AWS X-Ray.

Jedes Segment hat einen Namen, der Ihre Anwendung in der Service Map identifiziert. Das Segment kann statisch benannt werden, oder Sie können das SDK so konfigurieren, dass es dynamisch auf der Grundlage des Host-Headers in der eingehenden Anfrage benannt wird. Mit der dynamischen Benennung können Sie Traces auf der Grundlage des Domainnamens in der Anfrage gruppieren und einen Standardnamen anwenden, wenn der Name nicht einem erwarteten Muster entspricht (z. B. wenn der Host-Header gefälscht ist).

> **Weitergeleitete Anfragen**
>
> Wenn ein Load Balancer oder ein anderer Vermittler eine Anfrage an Ihre Anwendung weiterleitet, nimmt X-Ray die Client-IP aus dem `X-Forwarded-For` Header in der Anfrage und nicht aus der Quell-IP im IP-Paket. Die Client-IP, die für eine weitergeleitete Anfrage aufgezeichnet wird, kann gefälscht sein und sollte daher nicht als vertrauenswürdig eingestuft werden.

Wenn eine Anfrage weitergeleitet wird, legt das SDK ein zusätzliches Feld im Segment fest, um dies anzuzeigen. Wenn das Segment das Feld enthält, das auf `x_forwarded_for` gesetzt ist`true`, wurde die Client-IP aus dem `X-Forwarded-For` Header in der HTTP-Anfrage übernommen.

Der Meldungshandler erzeugt für jede eingehende Anforderung ein Segment mit einem `http`-Block, der die folgenden Informationen enthält:

- HTTP-Methode — GET, POST, PUT, DELETE usw.
- Client-Adresse — Die IP-Adresse des Clients, der die Anfrage gesendet hat.
- Antwortcode — Der HTTP-Antwortcode für die abgeschlossene Anfrage.
- Timing — Die Startzeit (als die Anfrage empfangen wurde) und die Endzeit (als die Antwort gesendet wurde).
- Benutzeragent — Der `user-agent` aus der Anfrage.
- Länge des Inhalts — Die Länge `content-length` der Antwort.

Sections



## Nachverfolgung eingehender Anforderungen mit Express

Zum Verwenden der Express-Middleware initialisieren Sie den SDK-Client und nutzen Sie die Middleware, die von der `express.openSegment`-Funktion zurückgegeben wird, bevor Sie Ihre Routen festlegen.

Example app.js – Express

```
var app = express();

var AWSXRay = require('aws-xray-sdk');
app.use(AWSXRay.express.openSegment('MyApp'));

app.get('/', function (req, res) {
  res.render('index');
});

app.use(AWSXRay.express.closeSegment());
```

Nachdem Sie Ihre Routen definiert haben, verwenden Sie die Ausgabe von `express.closeSegment` wie abgebildet, um alle Fehler zu behandeln, die vom X-Ray SDK für Node.js zurückgegeben werden.

Nachverfolgung eingehender Anforderungen mit Restify

Um die Restify-Middleware zu verwenden, initialisieren Sie den SDK-Client und führen Sie `enable` aus. Übergeben Sie den Namen Ihres Restify-Servers und des Segments.

Example app.js – Restify

```
var AWSXRay = require('aws-xray-sdk');
var AWSXRayRestify = require('aws-xray-sdk-restify');

var restify = require('restify');
var server = restify.createServer();
AWSXRayRestify.enable(server, 'MyApp'));

server.get('/', function (req, res) {
  res.render('index');
});
```

Konfiguration einer Segmentbenennungsstrategie

AWS X-Ray verwendet einen Dienstnamen, um Ihre Anwendung zu identifizieren und sie von den anderen Anwendungen, Datenbanken, externen APIs und AWS Ressourcen zu unterscheiden, die Ihre Anwendung verwendet. Wenn das X-Ray SDK Segmente für eingehende Anfragen generiert, zeichnet es den Dienstnamen Ihrer Anwendung im Namensfeld des Segments auf.

Das X-Ray SDK kann Segmente nach dem Hostnamen im HTTP-Anforderungsheader benennen. Dieser Header kann jedoch gefälscht sein, was zu unerwarteten Knoten in Ihrer Service Map führen kann. Um zu verhindern, dass das SDK Segmente aufgrund von Anfragen mit gefälschten Host-Headern falsch benennt, müssen Sie einen Standardnamen für eingehende Anfragen angeben.

Wenn Ihre Anwendung Anfragen für mehrere Domänen bearbeitet, können Sie das SDK so konfigurieren, dass es eine dynamische Benennungsstrategie verwendet, um dies in Segmentnamen widerzuspiegeln. Eine dynamische Benennungsstrategie ermöglicht es dem SDK, den Hostnamen für Anfragen zu verwenden, die einem erwarteten Muster entsprechen, und den Standardnamen auf Anfragen anzuwenden, bei denen dies nicht der Fall ist.

Beispielsweise können Sie über eine einzige Anwendung verfügen, die Anfragen an drei Subdomänen — `www.example.com`api.example.com, und — bedient. `static.example.com` Sie können eine dynamische Benennungsstrategie mit dem Muster verwenden`*.example.com`, um Segmente für jede Subdomain mit einem anderen Namen zu identifizieren, was zu drei Dienstknoten auf der Service-Map führt. Wenn Ihre Anwendung Anfragen mit einem Hostnamen empfängt, der nicht dem Muster entspricht, wird auf der Service Map ein vierter Knoten mit einem von Ihnen angegebenen Fallback-Namen angezeigt.

Wenn Sie denselben Namen für alle Segmente verwenden möchten, geben Sie bei der Initialisierung der Middleware den Namen Ihrer Anwendung, wie in den vorherigen Abschnitten gezeigt, ein.

> **Note**
>
> Sie können den mit der `AWS_XRAY_TRACING_NAME`-Umgebungsvariablen in Code definierten standardmäßigen Dienstnamen überschreiben.

Eine dynamische Benennungsstrategie definiert ein Muster, dem Hostnamen entsprechen sollten, sowie einen Standardnamen, der verwendet wird, wenn der Hostname in der HTTP-Anforderung nicht mit diesem Muster übereinstimmt. Zur dynamischen Segmentbenennung verwenden Sie `AWSXRay.middleware.enableDynamicNaming`.

Example app.js – dynamische Segmentnamen

Wenn der Hostname in der Anforderung dem Muster `*.example.com` entspricht, verwenden Sie den Hostnamen. Verwenden Sie andernfalls `MyApp`.

```
var app = express();

var AWSXRay = require('aws-xray-sdk');
app.use(AWSXRay.express.openSegment('MyApp'));
AWSXRay.middleware.enableDynamicNaming('*.example.com');

app.get('/', function (req, res) {
  res.render('index');
});

app.use(AWSXRay.express.closeSegment());
```

## Nachverfolgen von AWS SDK-Aufrufen mit dem X-Ray SDK for Node.js

Wenn Ihre Anwendung Aufrufe an tätigt, AWS-Services um Daten zu speichern, in eine Warteschlange zu schreiben oder Benachrichtigungen zu senden, verfolgt das X-Ray SDK for Node.js die Aufrufe nachgelagert in Untersegmenten . Nachverfolgte AWS-Services, und Ressourcen, auf die Sie innerhalb dieser Services zugreifen (z. B. ein Amazon S3-Bucket oder eine Amazon SQS-Warteschlange), werden als nachgelagerte Knoten auf der Trace-Map in der X-Ray-Konsole angezeigt.

Instrumentieren Sie AWS SDK-Clients, die Sie über die AWS SDK for JavaScript V2 oder AWS SDK for JavaScript V3 erstellen. Jede AWS SDK-Version bietet verschiedene Methoden zur Instrumentierung von AWS SDK-Clients.

> **Note**
>
> Derzeit gibt das AWS X-Ray SDK for Node.js beim Instrumentieren von AWS SDK for JavaScript V3-Clients weniger Segmentinformationen zurück als beim Instrumentieren von V2-Clients. Beispielsweise geben Untersegmente, die Aufrufe an DynamoDB darstellen, den Tabellennamen nicht zurück. Wenn Sie diese Segmentinformationen in Ihren Ablaufverfolgungen benötigen, sollten Sie die Verwendung von AWS SDK for JavaScript V2 in Betracht ziehen.

### AWS SDK for JavaScript V2

Sie können alle AWS SDK-V2-Clients instrumentieren, indem Sie Ihre `aws-sdk` Anforderungsanweisung in einen Aufruf von packen`AWSXRay.captureAWS`.

Example app.js – AWS SDK-Instrumentierung

```
const AWS = AWSXRay.captureAWS(require('aws-sdk'));
```

Um einzelne Clients zu instrumentieren, schließen Sie Ihren AWS SDK-Client in einen Aufruf von ein`AWSXRay.captureAWSClient`. Instrumentieren Sie beispielsweise einen `AmazonDynamoDB`-Client wie folgt:

Example app.js – DynamoDB-Client-Instrumentierung

```
    const AWSXRay = require('aws-xray-sdk');
```

```
...
    const ddb = AWSXRay.captureAWSClient(new AWS.DynamoDB());
```

> **Warning**
>
> Verwenden Sie nicht `captureAWS` und `captureAWSClient` zusammen. Dies führt zu doppelten Untersegmenten.

Wenn Sie TypeScript mit ECMAScript-Modulen (ESM) verwenden möchten, um Ihren JavaScript Code zu laden, verwenden Sie das folgende Beispiel, um Bibliotheken zu importieren:

Example app.js – AWS SDK-Instrumentierung

```
import * as AWS from 'aws-sdk';
import * as AWSXRay from 'aws-xray-sdk';
```

Verwenden Sie den folgenden Code, um alle AWS Clients mit ESM zu instrumentieren:

Example app.js – AWS SDK-Instrumentierung

```
import * as AWS from 'aws-sdk';
import * as AWSXRay from 'aws-xray-sdk';
const XRAY_AWS = AWSXRay.captureAWS(AWS);
const ddb = new XRAY_AWS.DynamoDB();
```

Für alle Services können Sie den Namen der API mit dem Namen in der X-Ray-Konsole sehen. Für eine Teilmenge von -Services fügt das X-Ray-SDK dem Segment Informationen hinzu, um mehr Granularität in der Service-Übersicht zu gewährleisten.

Wenn Sie beispielsweise einen Aufruf mit einem instrumentierten DynamoDB-Client tätigen, fügt das SDK dem Segment den Tabellennamen für Aufrufe hinzu, die auf eine Tabelle abzielen. In der Konsole wird jede Tabelle als separater Knoten in der Service-Übersicht angezeigt, mit einem generischen DynamoDB-Knoten für Aufrufe, die keine Tabelle anvisieren.

Example Untersegment für einen Aufruf an DynamoDB zum Speichern eines Elements

```
{
```

```
  "id": "24756640c0d0978a",
  "start_time": 1.480305974194E9,
  "end_time": 1.4803059742E9,
  "name": "DynamoDB",
  "namespace": "aws",
  "http": {
    "response": {
      "content_length": 60,
      "status": 200
    }
  },
  "aws": {
    "table_name": "scorekeep-user",
    "operation": "UpdateItem",
    "request_id": "UBQNSO5AEM8T4FDA4RQDEB94OVTDRVV4K4HIRGVJF66Q9ASUAAJG",
  }
}
```

Wenn Sie auf benannte Ressourcen zugreifen, werden durch Aufrufe der folgenden Services weitere Knoten in der Service-Übersicht erstellt. Durch Aufrufe, die keinen bestimmten Ressourcen gelten, wird ein generischer Knoten für den Service erstellt.

- Amazon DynamoDB – Tabellenname
- Amazon Simple Storage Service – Bucket- und Schlüsselname
- Amazon Simple Queue Service – Name der Warteschlange

### AWS SDK for JavaScript V3

Die AWS SDK for JavaScript V3 ist modular, sodass Ihr Code nur die benötigten Module lädt. Aus diesem Grund ist es nicht möglich, alle AWS SDK-Clients zu instrumentieren, da V3 die `captureAWS` Methode nicht unterstützt.

Wenn Sie TypeScript mit ECMAScript Modules (ESM) verwenden möchten, um Ihren JavaScript Code zu laden, können Sie das folgende Beispiel verwenden, um Bibliotheken zu importieren:

```
import * as AWS from 'aws-sdk';
import * as AWSXRay from 'aws-xray-sdk';
```

Instrumentieren Sie jeden AWS SDK-Client mit der -`AWSXRay.captureAWSv3Client`Methode. Instrumentieren Sie beispielsweise einen `AmazonDynamoDB`-Client wie folgt:

Example app.js – DynamoDB-Client-Instrumentierung mit -SDK für Javascript V3

```
    const AWSXRay = require('aws-xray-sdk');
    const { DynamoDBClient } = require("@aws-sdk/client-dynamodb");
...
    const ddb = AWSXRay.captureAWSv3Client(new DynamoDBClient({ region:
 "region" }));
```

Bei Verwendung von AWS SDK for JavaScript V3 werden Metadaten wie Tabellenname, Bucket- und Schlüsselname oder Warteschlangenname derzeit nicht zurückgegeben, daher enthält die Ablaufverfolgungszuordnung keine diskreten Knoten für jede benannte Ressource wie bei der Instrumentierung von AWS SDK-Clients mit der AWS SDK for JavaScript V2.

Example Untersegment für einen Aufruf an DynamoDB zum Speichern eines Elements bei Verwendung von AWS SDK for JavaScript V3

```
{
  "id": "24756640c0d0978a",
  "start_time": 1.480305974194E9,
  "end_time": 1.4803059742E9,
  "name": "DynamoDB",
  "namespace": "aws",
  "http": {
    "response": {
      "content_length": 60,
      "status": 200
    }
  },
  "aws": {
    "operation": "UpdateItem",
    "request_id": "UBQNSO5AEM8T4FDA4RQDEB94OVTDRVV4K4HIRGVJF66Q9ASUAAJG",
  }
}
```

## Verfolgen von Aufrufen an Downstream-HTTP-Webservices mithilfe des X-Ray SDK für Node.js

Wenn Ihre Anwendung Microservices oder öffentliche HTTP-APIs aufruft, können Sie den X-Ray SDK for Node.js Client verwenden, um diese Aufrufe zu instrumentieren und die API als Downstream-Service zum Service Graph hinzuzufügen.

Übergeben Sie Ihren `http` oder `https` Client an die `captureHTTPs` Methode X-Ray SDK for Node.js, um ausgehende Anrufe zu verfolgen.

> **Note**
>
> Aufrufe, die HTTP-Anforderungsbibliotheken von Drittanbietern wie Axios oder Superagent verwenden, werden durch die `captureHTTPsGlobal()`-API unterstützt und werden weiterhin nachverfolgt, wenn sie das `http`-Modul verwenden.

Example app.js – HTTP-Client

```
var AWSXRay = require('aws-xray-sdk');
var http = AWSXRay.captureHTTPs(require('http'));
```

Zur Aktivierung der Nachverfolgung auf allen HTTP-Clients rufen Sie `captureHTTPsGlobal` auf, bevor Sie `http` laden.

Example app.js – HTTP-Client (Global)

```
var AWSXRay = require('aws-xray-sdk');
AWSXRay.captureHTTPsGlobal(require('http'));
var http = require('http');
```

Wenn Sie einen Aufruf einer Downstream-Web-API instrumentieren, zeichnet das X-Ray-SDK für Node.js ein Untersegment auf, das Informationen über die HTTP-Anfrage und -Antwort enthält. X-Ray verwendet das Untersegment, um ein abgeleitetes Segment für die Remote-API zu generieren.

Example Untersegment für einen nachgelagerten HTTP-Aufruf

```
{
  "id": "004f72be19cddc2a",
  "start_time": 1484786387.131,
  "end_time": 1484786387.501,
  "name": "names.example.com",
  "namespace": "remote",
  "http": {
    "request": {
      "method": "GET",
      "url": "https://names.example.com/"
```

```
    },
    "response": {
      "content_length": -1,
      "status": 200
    }
  }
}
```

Example Abgeleitetes Segment für einen nachgelagerten HTTP-Anruf

```
{
  "id": "168416dc2ea97781",
  "name": "names.example.com",
  "trace_id": "1-62be1272-1b71c4274f39f122afa64eab",
  "start_time": 1484786387.131,
  "end_time": 1484786387.501,
  "parent_id": "004f72be19cddc2a",
  "http": {
    "request": {
      "method": "GET",
      "url": "https://names.example.com/"
    },
    "response": {
      "content_length": -1,
      "status": 200
    }
  },
  "inferred": true
}
```

## Verfolgen von SQL-Abfragen mit dem X-Ray SDK für Node.js

Instrumentieren Sie SQL-Datenbankabfragen, indem Sie Ihren SQL-Client in die entsprechende Client-Methode des X-Ray SDK for Node.js einbinden.

- PostgreSQL – `AWSXRay.capturePostgres()`

```
var AWSXRay = require('aws-xray-sdk');
var pg = AWSXRay.capturePostgres(require('pg'));
var client = new pg.Client();
```

- MySQL – `AWSXRay.captureMySQL()`

```
var AWSXRay = require('aws-xray-sdk');
var mysql = AWSXRay.captureMySQL(require('mysql'));
...
var connection = mysql.createConnection(config);
```

Wenn Sie mit einem instrumentierten Client SQL-Abfragen vornehmen, zeichnet das X-Ray SDK for Node.js in einem Untersegment Informationen über die Verbindung und die Abfrage auf.

Zusätzliche Daten in SQL-Untersegmenten einbeziehen

Sie können Untersegmenten, die für SQL-Abfragen generiert wurden, zusätzliche Informationen hinzufügen, sofern diese einem SQL-Feld auf der Zulassungsliste zugeordnet sind. Um beispielsweise die bereinigte SQL-Abfragezeichenfolge in einem Untersegment aufzuzeichnen, können Sie sie direkt zum SQL-Objekt des Untersegments hinzufügen.

Example Weisen Sie SQL einem Untersegment zu

```
    const queryString = 'SELECT * FROM MyTable';
connection.query(queryString, ...);

// Retrieve the most recently created subsegment
const subs = AWSXRay.getSegment().subsegments;

if (subs & & subs.length > 0) {
  var sqlSub = subs[subs.length - 1];
  sqlSub.sql.sanitized_query = queryString;
}
```

Eine vollständige Liste der SQL-Felder auf der Zulassungsliste finden Sie im Abschnitt SQL-Abfragen in der. Dokumente für Röntgensegmente

## Generieren benutzerdefinierter Untersegmente mit dem X-Ray SDK für Node.js

Untersegmente erweitern das Segment eines Traces um Details über die Arbeit, die zur Bearbeitung einer Anfrage geleistet wurde. Jedes Mal, wenn Sie einen Anruf mit einem instrumentierten Client tätigen, zeichnet das X-Ray-SDK die in einem Untersegment generierten Informationen auf. Sie können zusätzliche Untersegmente erstellen, um andere Untersegmente zu gruppieren, die Leistung eines Codeabschnitts zu messen oder Anmerkungen und Metadaten aufzuzeichnen.

### Benutzerdefinierte Express-Untersegmente

Um ein benutzerdefiniertes Untersegment für eine Funktion zu erstellen, die Aufrufe des nachgelagerten Service vornimmt, verwenden Sie die `captureAsyncFunc`-Funktion.

Example app.js – benutzerdefinierte Untersegmente Express

```
var AWSXRay = require('aws-xray-sdk');

app.use(AWSXRay.express.openSegment('MyApp'));

app.get('/', function (req, res) {
  var host = 'api.example.com';

  AWSXRay.captureAsyncFunc('send', function(subsegment) {
    sendRequest(host, function() {
      console.log('rendering!');
      res.render('index');
      subsegment.close();
    });
  });
});

app.use(AWSXRay.express.closeSegment());

function sendRequest(host, cb) {
  var options = {
    host: host,
    path: '/',
  };

  var callback = function(response) {
    var str = '';

    response.on('data', function (chunk) {
      str += chunk;
    });

    response.on('end', function () {
      cb();
    });
  }

  http.request(options, callback).end();
```

```
};
```

In diesem Beispiel erstellt die Anwendung ein benutzerdefiniertes Untersegment namens `send` für Aufrufe der `sendRequest`-Funktion. `captureAsyncFunc` übergibt ein Untersegment, das Sie innerhalb der Callback-Funktion schließen müssen, wenn die asynchronen Aufrufe der Funktion abgeschlossen sind.

Für synchrone Funktionen können Sie die `captureFunc`-Funktion verwenden, mit der das Untersegment automatisch geschlossen wird, sobald der Funktionsblock ausgeführt wurde.

Wenn Sie ein Untersegment innerhalb eines Segments oder eines anderen Untersegments erstellen, generiert das X-Ray SDK for Node.js eine ID dafür und zeichnet die Start- und Endzeit auf.

Example Untersegment mit Metadaten

```
"subsegments": [{
  "id": "6f1605cd8a07cb70",
  "start_time": 1.480305974194E9,
  "end_time": 1.4803059742E9,
  "name": "Custom subsegment for UserModel.saveUser function",
  "metadata": {
    "debug": {
      "test": "Metadata string from UserModel.saveUser"
    }
  },
```

Benutzerdefinierte Lambda-Untersegmente

Das SDK ist so konfiguriert, dass es automatisch ein Platzhalter-Fassadensegment erstellt, wenn es erkennt, dass es in Lambda ausgeführt wird. Um ein grundlegendes Untersegment zu erstellen, das einen einzelnen `AWS::Lambda::Function` Knoten auf der X-Ray-Trace-Map erstellt, rufen Sie das Fassadensegment auf und verwenden es für neue Zwecke. Wenn Sie manuell ein neues Segment mit einer neuen ID erstellen (während Sie die Ablaufverfolgung-ID, die übergeordnete ID und die Sampling-Entscheidung teilen), können Sie ein neues Segment senden.

Example app.js – benutzerdefinierte manuelle Untersegmente

```
const segment = AWSXRay.getSegment(); //returns the facade segment
const subsegment = segment.addNewSubsegment('subseg');
...
subsegment.close();
```

```
//the segment is closed by the SDK automatically
```

## Hinzufügen von Anmerkungen und Metadaten zu Segmenten mit dem X-Ray SDK for Node.js

Sie können Anmerkungen und Metadaten verwenden, um zusätzliche Informationen über Anfragen, die Umgebung oder Ihre Anwendung aufzuzeichnen. Sie können Anmerkungen und Metadaten zu den Segmenten hinzufügen, die das X-Ray SDK erstellt, oder zu benutzerdefinierten Untersegmenten, die Sie erstellen.

Anmerkungen sind Schlüssel-Wert-Paare mit Zeichenfolgen-, Zahlen- oder booleschen Werten. Anmerkungen sind für die Verwendung mit Filterausdrücken indexiert. Berücksichtigen Sie Anmerkungen, um Daten zur Gruppierung von Ablaufverfolgungen in der Konsole zu verwenden, oder wenn Sie die `GetTraceSummaries`-API aufrufen.

Metadaten sind Schlüssel-Wert-Paare, die Werte beliebigen Typs enthalten können, einschließlich Objekte und Listen, aber nicht für die Verwendung mit Filterausdrücken indexiert sind. Verwenden Sie Metadaten, um zusätzliche Daten aufzuzeichnen, die Sie im Trace speichern möchten, aber nicht für die Suche verwenden müssen.

Zusätzlich zu Anmerkungen und Metadaten können Sie auch Benutzer-ID-Zeichenfolgen in Segmenten aufzeichnen. Benutzer-IDs werden in einem eigenen Feld aufgezeichnet und zur Suche indiziert.

Sections

- Anmerkungen mit dem X-Ray SDK für Node.js aufnehmen
- Aufzeichnen von Metadaten mit dem X-Ray SDK für Node.js
- Aufzeichnen von Benutzer-IDs mit dem X-Ray SDK für Node.js

### Anmerkungen mit dem X-Ray SDK für Node.js aufnehmen

Verwenden Sie Anmerkungen, um Informationen zu Segmenten oder Untersegmenten, die zur Suche indiziert werden sollten, aufzuzeichnen.

Anmerkung zu Anforderungen

- Schlüssel — Der Schlüssel für eine X-Ray-Anmerkung kann bis zu 500 alphanumerische Zeichen enthalten. Sie können keine anderen Leerzeichen oder Symbole als den Unterstrich (_) verwenden.

- Werte — Der Wert für eine X-Ray-Anmerkung kann bis zu 1.000 Unicode-Zeichen enthalten.
- Die Anzahl der Anmerkungen — Sie können bis zu 50 Anmerkungen pro Spur verwenden.

So zeichnen Sie Anmerkungen auf

1. Erhalten Sie eine Referenz zum aktuellen Segment oder Untersegment.

```
var AWSXRay = require('aws-xray-sdk');
...
var document = AWSXRay.getSegment();
```

2. Rufen Sie `addAnnotation` mit einem Aktivierungsschlüssel und einem booleschen Wert oder einem Zeichenfolgenwert auf.

```
document.addAnnotation("mykey", "my value");
```

Das SDK zeichnet Anmerkungen als Schlüssel-Wert-Paare in einem `annotations`-Objekt im Segmentdokument auf. Wenn `addAnnotation` zweimal mit demselben Schlüssel aufgerufen wird, werden zuvor aufgezeichnete Werte im gleichen Segment oder Untersegment überschrieben.

Nutzen Sie das `annotations.`*`key`*-Schlüsselwort in einem Filterausdruck, um Ablaufverfolgungen durch Anmerkungen mit bestimmten Werten zu finden.

Example app.js – Anmerkungen

```
var AWS = require('aws-sdk');
var AWSXRay = require('aws-xray-sdk');
var ddb = AWSXRay.captureAWSClient(new AWS.DynamoDB());
...
app.post('/signup', function(req, res) {
    var item = {
        'email': {'S': req.body.email},
        'name': {'S': req.body.name},
        'preview': {'S': req.body.previewAccess},
        'theme': {'S': req.body.theme}
    };

    var seg = AWSXRay.getSegment();
    seg.addAnnotation('theme', req.body.theme);
```

```
    ddb.putItem({
      'TableName': ddbTable,
      'Item': item,
      'Expected': { email: { Exists: false } }
  }, function(err, data) {
...
```

Aufzeichnen von Metadaten mit dem X-Ray SDK für Node.js

Verwenden Sie Metadaten, um Segment- oder Untersegmentinformationen aufzuzeichnen, die nicht zur Suche indiziert werden müssen. Metadatenwerte sind Zeichenfolgen, Zahlen, boolesche Werte oder andere Objekte, die in Form eines JSON-Objekts oder eines Arrays angeordnet sein können.

So zeichnen Sie Metadaten auf

1. Erhalten Sie eine Referenz zum aktuellen Segment oder Untersegment.

   ```
   var AWSXRay = require('aws-xray-sdk');
   ...
   var document = AWSXRay.getSegment();
   ```

2. Rufen Sie `addMetadata` mit einem Zeichenfolgenschlüssel, einem booleschen Wert, einer Zeichenfolge oder einem Objektwert und einem Zeichenfolgen-Namespace auf.

   ```
   document.addMetadata("my key", "my value", "my namespace");
   ```

   or

   Rufen Sie `addMetadata` nur mit einem Aktivierungsschlüssel und einem Wert auf.

   ```
   document.addMetadata("my key", "my value");
   ```

Wenn Sie keinen Namespace angeben, verwendet SDK `default`. Wenn `addMetadata` zweimal mit demselben Schlüssel aufgerufen wird, werden zuvor aufgezeichnete Werte im gleichen Segment oder Untersegment überschrieben.

Aufzeichnen von Benutzer-IDs mit dem X-Ray SDK für Node.js

Zeichnen Sie Benutzer-IDs in Anforderungssegmenten auf, um den Benutzer zu identifizieren, der die Anforderung gesendet hat. Dieser Vorgang ist nicht mit AWS Lambda Funktionen kompatibel,

da Segmente in Lambda-Umgebungen unveränderlich sind. Der `setUser`-Aufruf kann nur auf Segmente angewendet werden, nicht auf Untersegmente.

So zeichnen Sie Benutzer-IDs auf

1. Erhalten Sie eine Referenz zum aktuellen Segment oder Untersegment.

```
var AWSXRay = require('aws-xray-sdk');
...
var document = AWSXRay.getSegment();
```

2. Rufen Sie `setUser()` mit einer Zeichenfolgen-ID des Benutzers auf, der die Anforderung gesendet hat.

```
var user = 'john123';

AWSXRay.getSegment().setUser(user);
```

Sie können `setUser` aufrufen, um die Benutzer-ID aufzuzeichnen, sobald die Express-Anwendung mit der Verarbeitung einer Anfrage beginnt. Wenn Sie das Segment nur zur Einrichtung der Benutzer-ID verwenden, können Sie die Aufrufe in einer einzelnen Zeile anordnen.

Example app.js – Benutzer-ID

```
var AWS = require('aws-sdk');
var AWSXRay = require('aws-xray-sdk');
var uuidv4 = require('uuid/v4');
var ddb = AWSXRay.captureAWSClient(new AWS.DynamoDB());
...
    app.post('/signup', function(req, res) {
    var userId = uuidv4();
    var item = {
        'userId': {'S': userId},
        'email': {'S': req.body.email},
        'name': {'S': req.body.name}
    };

    var seg = AWSXRay.getSegment().setUser(userId);

    ddb.putItem({
      'TableName': ddbTable,
```

```
      'Item': item,
      'Expected': { email: { Exists: false } }
  }, function(err, data) {
...
```

Nutzen Sie das `user`-Schlüsselwort in einem Filterausdruck, um Ablaufverfolgungen einer Benutzer-ID zu finden.

# Instrumentieren Sie Ihre Bewerbung mit Python

Es gibt zwei Möglichkeiten, Ihre Python Anwendung so zu instrumentieren, dass sie Spuren an X-Ray sendet:

- AWS Distro for OpenTelemetry Python — Eine AWS Distribution, die eine Reihe von Open-Source-Bibliotheken zum Senden korrelierter Metriken und Traces über AWS Distro for Collector an mehrere AWS Überwachungslösungen CloudWatch AWS X-Ray, darunter Amazon und Amazon OpenSearch Service, bereitstellt. OpenTelemetry
- AWS X-Ray SDK für Python — Eine Reihe von Bibliotheken zum Generieren und Senden von Traces an X-Ray über den X-Ray-Daemon.

Weitere Informationen finden Sie unter Wahl zwischen den AWS SDKs Distro for OpenTelemetry und X-Ray.

## AWS Distro für OpenTelemetry Python

Mit AWS Distro for OpenTelemetry (ADOT) Python können Sie Ihre Anwendungen einmal instrumentieren und korrelierte Metriken und Traces an mehrere AWS Monitoring-Lösungen wie Amazon und Amazon CloudWatch Service AWS X-Ray senden. OpenSearch Die Verwendung von X-Ray mit ADOT erfordert zwei Komponenten: ein OpenTelemetry SDK, das für die Verwendung mit X-Ray aktiviert ist, und das AWS Distro for OpenTelemetry Collector, das für die Verwendung mit X-Ray aktiviert ist. ADOT unterstützt Python automatische Instrumentierung, sodass Ihre Anwendung Traces ohne Codeänderungen senden kann.

Informationen zu den ersten Schritten finden Sie in der Dokumentation der AWS Distribution. OpenTelemetry Python

Weitere Informationen zur Verwendung von AWS Distro for OpenTelemetry with AWS X-Ray und anderen AWS-Services finden Sie unter AWS Distro for OpenTelemetry oder The AWS Distro for Documentation. OpenTelemetry

Weitere Informationen zur Sprachunterstützung und -verwendung finden Sie unter AWS Observability on. GitHub

## AWS X-Ray SDK für Python

Das X-Ray-SDK für Python ist eine Bibliothek für Python Webanwendungen, die Klassen und Methoden zum Generieren und Senden von Trace-Daten an den X-Ray-Daemon bereitstellt. Zu den Trace-Daten gehören Informationen über eingehende HTTP-Anfragen, die von der Anwendung bedient werden, sowie über Aufrufe, die die Anwendung mithilfe des AWS SDK, HTTP-Clients oder eines SQL-Datenbank-Connectors an Downstream-Dienste sendet. Sie können Segmente auch manuell erstellen und Debug-Informationen Anmerkungen und Metadaten hinzufügen.

Sie können das SDK mit `pip` herunterladen.

```
$ pip install aws-xray-sdk
```

> **Note**
>
> Das X-Ray SDK für Python ist ein Open-Source-Projekt. Du kannst das Projekt verfolgen und Issues und Pull-Requests einreichen unter GitHub: github.com/aws/ aws-xray-sdk-python

Wenn Sie Django oder Flask nutzen, beginnen Sie, indem Sie die SDK-Middleware zu Ihrer Anwendung hinzufügen, um eingehende Anforderungen zu verfolgen. Der Middleware erstellt für jede verfolgte Anforderung ein Segment und vervollständigt das Segment, nachdem die Antwort gesendet wurde. Während das Segment geöffnet ist, können Sie die SDK-Client-Methoden nutzen, um dem Segment Informationen hinzuzufügen, Untersegmente zu erstellen und nachgelagerte Aufrufe rückzuverfolgen. Das SDK erfasst auch automatisch Ausnahmen, die Ihre Anwendung ausgibt, während das Segment geöffnet ist. Bei anderen Anwendungen können Sie Segmente manuell erstellen.

Bei Lambda-Funktionen, die von einer instrumentierten Anwendung oder einem Dienst aufgerufen werden, liest Lambda den Tracing-Header und verfolgt automatisch Sampling-Anfragen. Für andere Funktionen können Sie Lambda so konfigurieren, dass eingehende Anfragen abgefragt und verfolgt werden. In beiden Fällen erstellt Lambda das Segment und stellt es dem X-Ray SDK zur Verfügung.

> **Note**
>
> Auf Lambda ist das X-Ray SDK optional. Wenn Sie es nicht in Ihrer Funktion verwenden, enthält Ihre Service-Map immer noch einen Knoten für den Lambda-Service und einen für jede Lambda-Funktion. Durch Hinzufügen des SDK können Sie Ihren Funktionscode instrumentieren, um Untersegmente zu dem von Lambda aufgezeichneten Funktionssegment hinzuzufügen. Weitere Informationen finden Sie unter AWS Lambda und AWS X-Ray.

Eine in Lambda instrumentierte Python Beispielfunktion finden Worker Sie unter.

Verwenden Sie als Nächstes das X-Ray-SDK für Python, um Downstream-Aufrufe zu instrumentieren, indem Sie die Bibliotheken Ihrer Anwendung patchen. Das SDK unterstützt die folgenden Bibliotheken.

Unterstützte Bibliotheken

- `botocore`, `boto3` — AWS SDK for Python (Boto) Instrumenten-Clients.
- `pynamodb`— Instrumentieren Sie die Version des Amazon DynamoDB-Clients von PynamoDB.
- `aiobotocore`, `aioboto3` — Asyncio-integrierte Versionen des SDK für Python-Clients.
- `requests`, `aiohttp` — Instrumentieren Sie HTTP-Clients auf hoher Ebene.
- `httplib`, `http.client`— Instrumentieren Sie HTTP-Clients auf niedriger Ebene und die Bibliotheken auf höherer Ebene, die sie verwenden.
- `sqlite3`— Instrumentieren Sie SQLite-Clients.
- `mysql-connector-python`— Instrumentieren Sie MySQL-Clients.
- `pg8000`— Instrument Pure-Python PostgreSQL-Schnittstelle.
- `psycopg2`— Instrument PostgreSQL-Datenbankadapter.
- `pymongo`— Instrumentieren Sie MongoDB-Clients.
- `pymysql`— Instrument PyMy SQL-basierte Clients für MySQL und MariaDB.

Immer wenn Ihre Anwendung eine SQL-Datenbank oder andere HTTP-Dienste aufruft, zeichnet das SDK Informationen über den Aufruf in einem Untersegment auf. AWS AWS-Services und die Ressourcen, auf die Sie innerhalb der Dienste zugreifen, werden in der Trace-Map als Downstream-Knoten angezeigt, sodass Sie Fehler und Drosselungsprobleme bei einzelnen Verbindungen leichter identifizieren können.

Nachdem Sie das SDK verwendet haben, passen Sie sein Verhalten an, indem Sie den Rekorder und die Middleware konfigurieren. Sie können Plugins zum Festhalten von Daten über die Datenverarbeitungsressourcen, auf denen Ihre Anwendung ausgeführt wird, hinzufügen, das Samplingverhalten durch Samplingregeln anpassen und Protokollebenen einrichten, um mehr oder weniger Informationen von dem SDK in Ihren Anwendungsprotokollen zu sehen.

Zeichnen Sie zusätzliche Informationen zu Anforderungen und den Aufgaben, die Ihre Anwendung ausführt, in Anmerkungen und Metadaten auf. Anmerkungen sind einfache Schlüsselwertpaare, die für die Verwendung mit Filterausdrücken indiziert werden, damit Sie nach Ablaufverfolgen mit bestimmten Daten suchen können. Metadateneinträge sind weniger einschränkend und können ganze Objekte und Arrays aufzeichnen – alle Daten, die in eine JSON zusammengefasst werden können.

> **Anmerkungen und Metadaten**
>
> Anmerkungen und Metadaten sind beliebiger Text, den Sie Segmenten mit dem X-Ray SDK hinzufügen. Anmerkungen werden für die Verwendung mit Filterausdrücken indexiert. Metadaten werden nicht indexiert, können aber im Rohsegment mit der X-Ray-Konsole oder API angezeigt werden. Jeder, dem Sie Lesezugriff auf X-Ray gewähren, kann diese Daten einsehen.

Wenn Sie viele instrumentierten Clients in Ihrem Code haben, kann ein einzelnes Anforderungssegmente viele Untersegmente enthalten, eines für jeden Aufruf mit einem instrumentierten Client. Sie können Untersegmente organisieren und gruppieren, indem Sie Client-Aufrufe in benutzerdefinierten Untersegmenten zusammenfassen. Sie können ein benutzerdefiniertes Untersegment für die gesamte Funktion oder einen beliebigen Bereich des Codes erstellen. Sie können dann Metadaten und Anmerkungen im Untersegment aufzeichnen, anstatt alles in das übergeordnete Segment zu schreiben.

Eine Referenzdokumentation zu den Klassen und Methoden des SDK finden Sie in der PythonAPI-Referenz zum AWS X-Ray SDK.

## Voraussetzungen

Das X-Ray-SDK für Python unterstützt die folgenden Sprach- und Bibliotheksversionen.

- Python — 2.7, 3.4 und neuer
- Django — 1.10 und neuer

- Flask — 0.10 und neuer
- aiohttp — 2.3.0 und neuer
- AWS SDK for Python (Boto)— 1.4.0 und neuer
- botocore — 1.5.0 und neuer
- enum — 0.4.7 und neuer, für Versionen 3.4.0 und älter Python
- jsonpickle — 1.0.0 und neuer
- setuptools — 40.6.3 und neuer
- wrapt — 1.11.0 und neuer

## Abhängigkeitsmanagement

Das X-Ray-SDK für Python ist erhältlich unter`pip`.

- Package — `aws-xray-sdk`

Fügen Sie das SDK in Ihrer `requirements.txt`-Datei als abhängige Komponente hinzu.

Example requirements.txt

```
aws-xray-sdk==2.4.2
boto3==1.4.4
botocore==1.5.55
Django==1.11.3
```

Wenn Sie Elastic Beanstalk für die Bereitstellung Ihrer Anwendung verwenden, installiert Elastic Beanstalk alle Pakete automatisch. `requirements.txt`

## Konfiguration des X-Ray-SDK für Python

Das X-Ray-SDK für Python hat eine Klasse namens`xray_recorder`, die den globalen Rekorder bereitstellt. Sie können die globale Aufzeichnung so konfigurieren, dass die Middleware, die Segmente für eingehende HTTP-Aufrufe erstellt, angepasst wird.

Sections

- Service-Plugins
- Samplingregeln
- Protokollierung

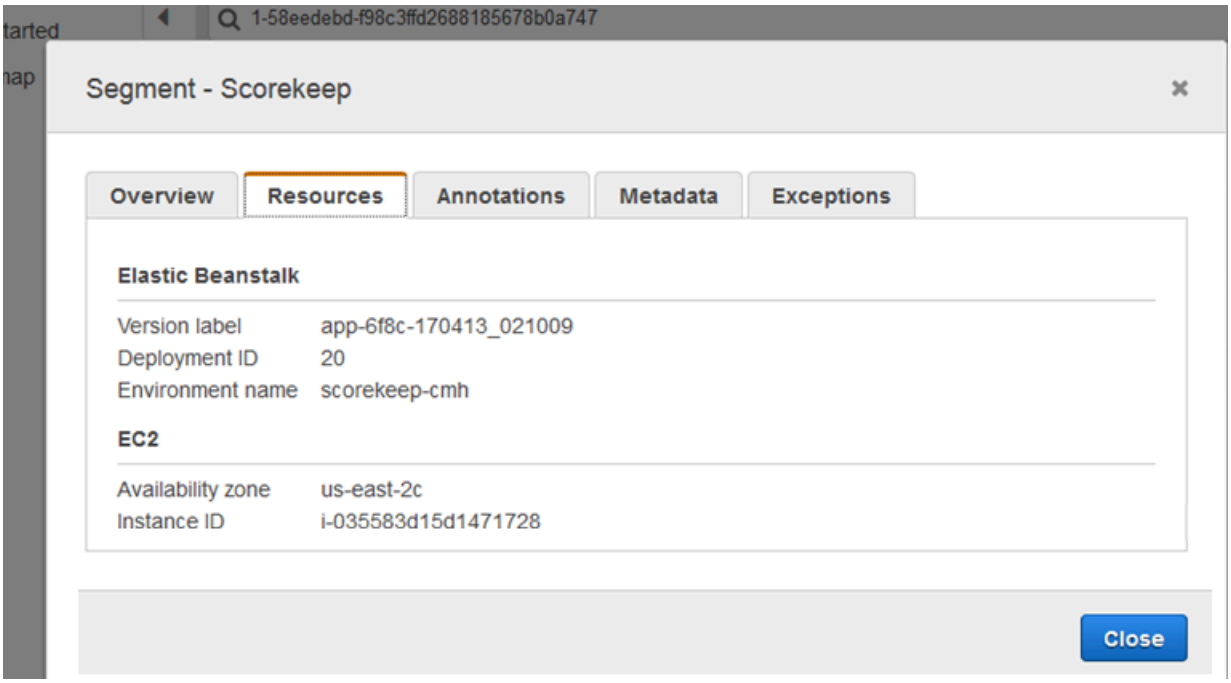
- Konfiguration des Recorders im Code
- Konfigurieren des Recorders mit Django
- Umgebungsvariablen

Service-Plugins

Wird verwendet`plugins`, um Informationen über den Dienst aufzuzeichnen, der Ihre Anwendung hostet.

Plug-ins

- Amazon EC2 — `EC2Plugin` fügt die Instance-ID, die Availability Zone und die CloudWatch Logs-Gruppe hinzu.
- Elastic Beanstalk — `ElasticBeanstalkPlugin` fügt den Umgebungsnamen, die Versionsbezeichnung und die Bereitstellungs-ID hinzu.
- Amazon ECS — `ECSPlugin` fügt die Container-ID hinzu.




Um ein Plugin zu verwenden, rufen Sie `configure` im `xray_recorder` auf.

```
from aws_xray_sdk.core import xray_recorder
from aws_xray_sdk.core import patch_all
```

```
xray_recorder.configure(service='My app')
plugins = ('ElasticBeanstalkPlugin', 'EC2Plugin')
xray_recorder.configure(plugins=plugins)
patch_all()
```

> **Note**
>
> Da `plugins` sie als Tupel übergeben werden, sollten Sie bei der Angabe eines einzelnen Plugins unbedingt ein abschließendes Zeichen `,` angeben. Beispiel: `plugins = ('EC2Plugin',)`

Sie können auch Umgebungsvariablen, die Vorrang über Werte im Code haben, zur Konfiguration des Recorders verwenden.

Konfigurieren Sie Plugins vor Patch-Bibliotheken, um nachgeschaltete Aufrufe aufzuzeichnen.

Das SDK verwendet auch Plugin-Einstellungen, um das `origin` Feld für das Segment festzulegen. Dies gibt den AWS Ressourcentyp an, auf dem Ihre Anwendung ausgeführt wird. Wenn Sie mehrere Plugins verwenden, verwendet das SDK die folgende Auflösungsreihenfolge, um den Ursprung zu bestimmen: ElasticBeanstalk > EKS > ECS > EC2.

Samplingregeln

Das SDK verwendet die Sampling-Regeln, die Sie in der X-Ray-Konsole definieren, um zu bestimmen, welche Anfragen aufgezeichnet werden sollen. Die Standardregel verfolgt die erste Anfrage jede Sekunde und fünf Prozent aller weiteren Anfragen aller Dienste, die Traces an X-Ray senden. Erstellen Sie zusätzliche Regeln in der X-Ray-Konsole, um die Menge der aufgezeichneten Daten für jede Ihrer Anwendungen anzupassen.

Das SDK wendet benutzerdefinierte Regeln in der Reihenfolge an, in der sie definiert sind. Wenn eine Anfrage mehreren benutzerdefinierten Regeln entspricht, wendet das SDK nur die erste Regel an.

> **Note**
>
> Wenn das SDK X-Ray nicht erreichen kann, um Sampling-Regeln abzurufen, kehrt es zu einer lokalen Standardregel zurück, die die erste zu Beginn jeder Sekunde empfangene Anfrage und fünf Prozent aller zusätzlichen Anfragen pro Host festlegt. Dies kann passieren, wenn der Host nicht berechtigt ist, Sampling-APIs aufzurufen, oder wenn er keine Verbindung

zum X-Ray-Daemon herstellen kann, der als TCP-Proxy für API-Aufrufe durch das SDK fungiert.

Sie können das SDK auch so konfigurieren, dass Sampling-Regeln aus einem JSON-Dokument geladen werden. Das SDK kann lokale Regeln als Backup für Fälle verwenden, in denen X-Ray Sampling nicht verfügbar ist, oder ausschließlich lokale Regeln verwenden.

Example sampling-rules.json

```
{
  "version": 2,
  "rules": [
    {
      "description": "Player moves.",
      "host": "*",
      "http_method": "*",
      "url_path": "/api/move/*",
      "fixed_target": 0,
      "rate": 0.05
    }
  ],
  "default": {
    "fixed_target": 1,
    "rate": 0.1
  }
}
```

In diesem Beispiel werden eine benutzerdefinierte Regel und eine Standardregel definiert. Die benutzerdefinierte Regel wendet eine Stichprobenrate von fünf Prozent an, ohne dass eine Mindestanzahl von Anfragen für Pfade verfolgt werden muss, unter `/api/move/` denen Pfade verfolgt werden müssen. Die Standardregel verfolgt die erste Anfrage jede Sekunde und 10 Prozent der weiteren Anfragen.

Der Nachteil der lokalen Definition von Regeln besteht darin, dass das feste Ziel von jeder Instanz des Rekorders unabhängig angewendet wird, anstatt vom X-Ray-Dienst verwaltet zu werden. Wenn Sie mehr Hosts bereitstellen, wird die feste Rate vervielfacht, wodurch es schwieriger wird, die Menge der aufgezeichneten Daten zu kontrollieren.

Wenn aktiviert AWS Lambda, können Sie die Samplerate nicht ändern. Wenn Ihre Funktion von einem instrumentierten Dienst aufgerufen wird, werden Aufrufe, die Anfragen generierten, die von

diesem Dienst abgetastet wurden, von Lambda aufgezeichnet. Wenn aktives Tracing aktiviert ist und kein Tracing-Header vorhanden ist, trifft Lambda die Stichprobenentscheidung.

Um Sicherungsregeln zu konfigurieren, rufen Sie `xray_recorder.configure` auf, wie im folgenden Beispiel dargestellt, wobei der Begriff *`Regeln`* entweder ein Wörterbuch mit Regeln oder den absoluten Pfad zu einer JSON-Datei mit Samplingregeln bezeichnet.

```
xray_recorder.configure(sampling_rules=rules)
```

Um nur lokale Regeln zu verwenden, konfigurieren Sie den Recorder mit einer `LocalSampler`.

```
from aws_xray_sdk.core.sampling.local.sampler import LocalSampler
xray_recorder.configure(sampler=LocalSampler())
```

Sie können auch die globale Aufzeichnung so konfigurieren, dass das Sampling deaktiviert und alle eingehenden Anfragen instrumentiert werden.

Example main.py — Sampling deaktivieren

```
xray_recorder.configure(sampling=False)
```

Protokollierung

Das SDK verwendet das in Python integrierte `logging` Modul mit einer `WARNING` Standard-Protokollierungsebene. Erhalten Sie eine Referenz zum Logger für die `aws_xray_sdk`-Klasse und rufen Sie darauf `setLevel` auf, um die verschiedenen Protokollebenen für die Bibliothek und den Rest Ihrer Anwendung zu konfigurieren.

Example app.py — Protokollierung

```
logging.basicConfig(level='WARNING')
logging.getLogger('aws_xray_sdk').setLevel(logging.ERROR)
```

Verwenden Sie Debug-Protokolle, um Probleme wie nicht geschlossene Untersegmente zu identifizieren, wenn Sie Untersegmente manuell generieren.

Konfiguration des Recorders im Code

Zusätzliche Einstellungen finden Sie in der `configure`-Methode auf `xray_recorder`.

- `context_missing`— Auf einstellen, um `LOG_ERROR` zu verhindern, dass Ausnahmen ausgelöst werden, wenn Ihr instrumentierter Code versucht, Daten aufzuzeichnen, obwohl kein Segment geöffnet ist.
- `daemon_address`— Legt den Host und den Port des X-Ray-Daemon-Listeners fest.
- `service`— Legen Sie einen Dienstnamen fest, den das SDK für Segmente verwendet.
- `plugins`— Notieren Sie Informationen über die AWS Ressourcen Ihrer Anwendung.
- `sampling`— Auf einstellen, `False` um die Probenahme zu deaktivieren.
- `sampling_rules`— Legen Sie den Pfad der JSON-Datei fest, die Ihre Sampling-Regeln enthält.

Example main.py — Deaktiviert Ausnahmen, die im Kontext fehlen

```
from aws_xray_sdk.core import xray_recorder

xray_recorder.configure(context_missing='LOG_ERROR')
```

Konfigurieren des Recorders mit Django

Wenn Sie das Django-Framework verwenden, können Sie die `settings.py`-Datei von Django zum Konfigurieren von Optionen in der globalen Aufzeichnung verwenden.

- `AUTO_INSTRUMENT`(Nur Django) — Zeichnet Untersegmente für integrierte Datenbank- und Vorlagen-Rendering-Operationen auf.
- `AWS_XRAY_CONTEXT_MISSING`— Auf einstellen, um `LOG_ERROR` zu vermeiden, dass Ausnahmen ausgelöst werden, wenn Ihr instrumentierter Code versucht, Daten aufzuzeichnen, wenn kein Segment geöffnet ist.
- `AWS_XRAY_DAEMON_ADDRESS`— Legt den Host und den Port des X-Ray-Daemon-Listeners fest.
- `AWS_XRAY_TRACING_NAME`— Legen Sie einen Dienstnamen fest, den das SDK für Segmente verwendet.
- `PLUGINS`— Notieren Sie Informationen über die AWS Ressourcen Ihrer Anwendung.
- `SAMPLING`— Auf einstellen, `False` um die Probenahme zu deaktivieren.
- `SAMPLING_RULES`— Legen Sie den Pfad der JSON-Datei fest, die Ihre Sampling-Regeln enthält.

Um die Konfiguration des Recorders in `settings.py` zu aktivieren, fügen Sie die Django-Middleware zur Liste der installierten Apps hinzu.

Example settings.py — Installierte Apps

```
INSTALLED_APPS = [
    ...
    'django.contrib.sessions',
    'aws_xray_sdk.ext.django',
]
```

Konfigurieren Sie die verfügbaren Einstellungen in einem dict-Objekt mit dem Namen `XRAY_RECORDER`.

Example settings.py — Installierte Apps

```
XRAY_RECORDER = {
    'AUTO_INSTRUMENT': True,
    'AWS_XRAY_CONTEXT_MISSING': 'LOG_ERROR',
    'AWS_XRAY_DAEMON_ADDRESS': '127.0.0.1:5000',
    'AWS_XRAY_TRACING_NAME': 'My application',
    'PLUGINS': ('ElasticBeanstalkPlugin', 'EC2Plugin', 'ECSPlugin'),
    'SAMPLING': False,
}
```

Umgebungsvariablen

Sie können Umgebungsvariablen verwenden, um das X-Ray SDK für Python zu konfigurieren. Das SDK unterstützt die folgenden Variablen:

- `AWS_XRAY_TRACING_NAME`— Legen Sie einen Dienstnamen fest, den das SDK für Segmente verwendet. Überschreibt den Servicenamen, den Sie programmgesteuert festgelegt haben.
- `AWS_XRAY_SDK_ENABLED`— Wenn auf gesetzt`false`, wird das SDK deaktiviert. Das SDK ist standardmäßig aktiviert, es sei denn, die Umgebungsvariable ist auf „false“ festgelegt.
  - Wenn diese Option deaktiviert ist, generiert die globale Aufzeichnung automatisch Dummy-Segmente und Untersegmente, die nicht an den Daemon gesendet werden, und das automatische Patchen wird deaktiviert. Middlewares werden als Wrapper über der globalen Aufzeichnung geschrieben. Auch alle Generierungen von Segmenten und Untersegmenten über die Middleware werden zu Dummy-Segmenten und Dummy-Untersegmenten.
  - Legen Sie den Wert `AWS_XRAY_SDK_ENABLED` über die Umgebungsvariable oder durch direkte Interaktion mit dem Objekt `global_sdk_config` aus der `aws_xray_sdk`-Bibliothek fest. Einstellungen der Umgebungsvariablen überschreiben diese Interaktionen.

- `AWS_XRAY_DAEMON_ADDRESS`— Legt den Host und den Port des X-Ray-Daemon-Listeners fest. Standardmäßig verwendet `127.0.0.1:2000` das SDK sowohl Trace-Daten (UDP) als auch Sampling-Daten (TCP). Verwenden Sie diese Variable, wenn Sie den Daemon so konfiguriert haben, dass er auf einem anderen Port lauscht oder wenn er auf einem anderen Host läuft.

  Format

  - Derselbe Port — *`address`*`:`*`port`*
  - Verschiedene Anschlüsse — `tcp:`*`address`*`:`*`port`* `udp:`*`address`*`:`*`port`*
- `AWS_XRAY_CONTEXT_MISSING`— Auf einstellen, `RUNTIME_ERROR` um Ausnahmen auszulösen, wenn Ihr instrumentierter Code versucht, Daten aufzuzeichnen, obwohl kein Segment geöffnet ist.

  Zulässige Werte

  - `RUNTIME_ERROR`— Löst eine Laufzeitausnahme aus.
  - `LOG_ERROR`— Einen Fehler protokollieren und fortfahren (Standard).
  - `IGNORE_ERROR`— Fehler ignorieren und fortfahren.

  Fehler im Zusammenhang mit fehlenden Segmenten oder Untersegmenten können auftreten, wenn Sie versuchen, einen instrumentierten Client in Startcode zu verwenden, der ausgeführt wird, wenn keine Anfrage geöffnet ist, oder in Code, der einen neuen Thread erzeugt.

Umgebungsvariablen überschreiben Werte im Code.

## Ablaufverfolgung eingehender Anfragen mit dem X-Ray-SDK für Python -Middleware

Wenn Sie die Middleware zu Ihrer Anwendung hinzufügen und einen Segmentnamen konfigurieren, erstellt das X-Ray SDK für Python ein Segment für jede gesampelte Anfrage. Dieses Segment umfasst Dauer, Methode und Status der HTTP-Anforderung. Die zusätzliche Instrumentierung schafft Untersegmente zu diesem Segment.

Das X-Ray-SDK für Python unterstützt die folgende Middleware zur Instrumentierung eingehender HTTP-Anforderungen:

- Django
- Flask
- Bottle

> **Note**
>
> FürAWS Lambda Funktionen erstellt Lambda für jede gesampelte Anfrage ein Segment. Weitere Informationen finden Sie unter AWS Lambda und AWS X-Ray.

Ein BeispielWorker für eine Python-Funktion, die in Lambda instrumentiert ist, finden Sie unter.

Für Skripts oder Python-Anwendungen auf anderen Frameworks können Sie Segmente manuell erstellen.

Jedes Segment hat einen Namen, der Ihre Anwendung in der Service Map identifiziert. Das Segment kann statisch benannt werden, oder Sie können das SDK so konfigurieren, dass es basierend auf dem Host-Header in der eingehenden Anfrage dynamisch benannt wird. Mit der dynamischen Benennung können Sie Traces basierend auf dem Domainnamen in der Anfrage gruppieren und einen Standardnamen anwenden, wenn der Name nicht mit einem erwarteten Muster übereinstimmt (z. B. wenn der Host-Header gefälscht ist).

> **Weitergeleitete Anfragen**
>
> Wenn ein Load Balancer oder ein anderer Vermittler eine Anfrage an Ihre Anwendung weiterleitet, entnimmt X-Ray die Client-IP aus dem`X-Forwarded-For` Header in der Anfrage und nicht von der Quell-IP im IP-Paket. Die Client-IP, die für eine weitergeleitete Anfrage aufgezeichnet wurde, kann gefälscht sein und sollte daher nicht vertrauenswürdig sein.

Wenn eine Anfrage weitergeleitet wird, legt das SDK ein zusätzliches Feld im Segment fest, um dies anzuzeigen. Wenn das Segment das Feld enthält, das auf`x_forwarded_for` gesetzt ist`true`, wurde die Client-IP dem`X-Forwarded-For` Header der HTTP-Anfrage entnommen.

Die Middleware erzeugt für jede eingehende Anfrage ein Segment mit einem `http`-Block mit folgenden Informationen:

- HTTP-Methode — GET, POST, PUT, DELETE usw.
- Client-Adresse — Die IP-Adresse des Clients, der die Anfrage gesendet hat.
- Antwortcode — Der HTTP-Antwortcode für die abgeschlossene Anfrage.
- Zeitpunkt — Die Startzeit (als die Anfrage empfangen wurde) und die Endzeit (als die Antwort gesendet wurde).

- Benutzeragent — Der`user-agent` aus der Anfrage.
- Inhaltslänge — Die`content-length` aus der Antwort.

Abschnitte



## Hinzufügen der Middleware zu Ihrer Anwendung (Django)

Fügen Sie die Middleware zur `MIDDLEWARE`-Liste in Ihrer `settings.py`-Datei hinzu. Die X-Ray-Middleware sollte die erste Zeile in Ihrer `settings.py`-Datei darstellen, um sicherzugehen, dass Anfragen, die in anderen Middlewares fehlschlagen, aufgezeichnet werden.

Example settings.py - X-Ray-SDK für Python-Middleware

```
MIDDLEWARE = [
    'aws_xray_sdk.ext.django.middleware.XRayMiddleware',
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware'
]
```

Fügen Sie die X-Ray SDK Django-App zur`INSTALLED_APPS` Liste in Ihrer`settings.py` Datei hinzu. Dadurch kann der X-Ray-Recorder während des Starts Ihrer App konfiguriert werden.

Example settings.py - X-Ray-SDK für die Python Django-App

```
INSTALLED_APPS = [
    'aws_xray_sdk.ext.django',
    'django.contrib.admin',
    'django.contrib.auth',
```

```
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
]
```

Konfigurieren Sie einen Segmentnamen in Ihrer `settings.py`-Datei.

Example settings.py — Segmentname

```
XRAY_RECORDER = {
    'AWS_XRAY_TRACING_NAME': 'My application',
    'PLUGINS': ('EC2Plugin',),
}
```

Dadurch wird der X-Ray-Recorder angewiesen, Anfragen, die von Ihrer Django-Anwendung bedient werden, mit der standardmäßigen Samplingrate zu verfolgen. Sie können den Recorder in Ihrer Django-Einstellungsdatei konfigurieren, um benutzerdefinierte Samplingregeln anzuwenden oder andere Einstellungen zu ändern.

> **Note**
>
> Da `plugins` sie als Tupel übergeben werden, stellen Sie sicher, dass Sie `,` bei der Angabe eines einzelnen Plugins einen Nachbau angeben. Beispiel: `plugins = ('EC2Plugin',)`

Hinzufügen der Middleware zu Ihrer Anwendung (Flask)

Um Ihre Flask-Anwendung zu instrumentieren, konfigurieren Sie zunächst einen Segmentnamen im `xray_recorder`. Verwenden Sie dann die `XRayMiddleware`-Funktion zum Patchen Ihrer Flask-Anwendung im Code.

Example app.py

```
from aws_xray_sdk.core import xray_recorder
from aws_xray_sdk.ext.flask.middleware import XRayMiddleware

app = Flask(__name__)

xray_recorder.configure(service='My application')
XRayMiddleware(app, xray_recorder)
```

Dadurch wird der X-Ray-Recorder angewiesen, Anfragen, die von Ihrer Flask-Anwendung bearbeitet wurden, mit der standardmäßigen Samplingrate zu verfolgen. Sie können den Recorder im Code konfigurieren, um benutzerdefinierte Samplingregeln anzuwenden oder andere Einstellungen zu ändern.

Hinzufügen der Middleware zu Ihrer Anwendung (Bottle)

Um Ihre Bottle-Anwendung zu instrumentieren, konfigurieren Sie zunächst einen Segmentnamen im `xray_recorder`. Verwenden Sie dann die `XRayMiddleware`-Funktion zum Patchen Ihrer Bottle-Anwendung im Code.

Example app.py

```
from aws_xray_sdk.core import xray_recorder
from aws_xray_sdk.ext.bottle.middleware import XRayMiddleware

app = Bottle()

xray_recorder.configure(service='fallback_name', dynamic_naming='My application')
app.install(XRayMiddleware(xray_recorder))
```

Dadurch wird der X-Ray-Recorder angewiesen, Anfragen, die von Ihrer Bottle-Anwendung bearbeitet wurden, mit der standardmäßigen Samplingrate zu verfolgen. Sie können den Recorder im Code konfigurieren, um benutzerdefinierte Samplingregeln anzuwenden oder andere Einstellungen zu ändern.

Manuelle Instrumentierung von Python-Code

Wenn Sie weder Django noch Flask verwenden, können Sie Segmente manuell erstellen. Sie können für jede eingehende Anfrage ein Segment erstellen oder Segmente um gepatchte HTTP- oderAWS SDK-Clients erstellen, um dem Rekorder den Kontext zum Hinzufügen von Untersegmenten bereitzustellen.

Example main.py — Manuelle Instrumentierung

```
from aws_xray_sdk.core import xray_recorder

# Start a segment
segment = xray_recorder.begin_segment('segment_name')
# Start a subsegment
subsegment = xray_recorder.begin_subsegment('subsegment_name')
```

```
# Add metadata and annotations
segment.put_metadata('key', dict, 'namespace')
subsegment.put_annotation('key', 'value')

# Close the subsegment and segment
xray_recorder.end_subsegment()
xray_recorder.end_segment()
```

Konfiguration einer Segmentbenennungsstrategie

AWS X-Rayverwendet einen Dienstnamen, um Ihre Anwendung zu identifizieren und sie von den anderen Anwendungen, Datenbanken, externen APIs undAWS Ressourcen zu unterscheiden, die Ihre Anwendung verwendet. Wenn das X-Ray SDK Segmente für eingehende Anfragen generiert, zeichnet es den Dienstnamen Ihrer Anwendung im Namensfeld des Segments auf.

Das X-Ray SDK kann Segmente nach dem Hostnamen im HTTP-Request-Header benennen. Dieser Header kann jedoch gefälscht sein, was zu unerwarteten Knoten in Ihrer Service-Map führen kann. Um zu verhindern, dass das SDK Segmente aufgrund von Anfragen mit gefälschten Host-Headern falsch benennt, müssen Sie einen Standardnamen für eingehende Anfragen angeben.

Wenn Ihre Anwendung Anfragen für mehrere Domänen verarbeitet, können Sie das SDK so konfigurieren, dass es eine dynamische Benennungsstrategie verwendet, die dies in Segmentnamen widerspiegelt. Eine dynamische Benennungsstrategie ermöglicht es dem SDK, den Hostnamen für Anfragen zu verwenden, die einem erwarteten Muster entsprechen, und den Standardnamen auf Anfragen anzuwenden, die dies nicht tun.

Beispielsweise könnten Sie eine einzelne Anwendung haben, die Anfragen an drei Subdomänen verarbeitet —`www.example.com`api.example.com, und`static.example.com`. Sie können eine dynamische Benennungsstrategie mit dem Muster verwenden,`*.example.com` um Segmente für jede Subdomain mit einem anderen Namen zu identifizieren, was zu drei Service-Knoten auf der Service-Map führt. Wenn Ihre Anwendung Anfragen mit einem Hostnamen empfängt, der nicht mit dem Muster übereinstimmt, sehen Sie in der Service-Map einen vierten Knoten mit einem von Ihnen angegebenen Ausweichnamen.

Wenn Sie denselben Namen für alle Anfragesegmente verwenden möchten, geben Sie bei der Konfiguration des Recorders den Namen Ihrer Anwendung, wie in den vorherigen Abschnitten gezeigt, ein.

Eine dynamische Benennungsstrategie definiert ein Muster, dem Hostnamen entsprechen sollten, sowie einen Standardnamen, der verwendet wird, wenn der Hostname in der HTTP-Anforderung

nicht mit diesem Muster übereinstimmt. Zur dynamischen Segmentbenennung fügen Sie die DYNAMIC_NAMING-Einstellung zu Ihrer settings.py-Datei hinzu.

Example settings.py — Dynamische Benennung

```
XRAY_RECORDER = {
    'AUTO_INSTRUMENT': True,
    'AWS_XRAY_TRACING_NAME': 'My application',
    'DYNAMIC_NAMING': '*.example.com',
    'PLUGINS': ('ElasticBeanstalkPlugin', 'EC2Plugin')
}
```

Sie können "*" im Muster verwenden, um eine Übereinstimmung mit einer beliebigen Zeichenfolge zu erzielen, oder "?" für die Übereinstimmung mit einem einzelnen Zeichen. Für Flask konfigurieren Sie den Recorder im Code.

Example main.py — Segmentname

```
from aws_xray_sdk.core import xray_recorder
xray_recorder.configure(service='My application')
xray_recorder.configure(dynamic_naming='*.example.com')
```

> **Note**
>
> Sie können den mit der AWS_XRAY_TRACING_NAME-Umgebungsvariablen in Code definierten standardmäßigen Dienstnamen überschreiben.

## Patchen von Bibliotheken zum Instrumentieren von nachgelagerten Aufrufen

Um Downstream-Aufrufe zu instrumentieren, verwenden Sie das X-Ray-SDK für Python, um die Bibliotheken zu patchen, die Ihre Anwendung verwendet. Das X-Ray-SDK für Python kann die folgenden Bibliotheken patchen.

Unterstützte Bibliotheken

- botocore,boto3— InstrumentAWS SDK for Python (Boto)Kunden.
- pynamodb— Instrumentieren Sie die Version des Amazon DynamoDB-Clients von PynamoDB.
- aiobotocore,aioboto3— Instrumentsynchron-integrierte Versionen des SDK für Python-Clients.

- `requests`,`aiohttp`— Instrumentieren Sie HTTP-Clients auf hoher Ebene.
- `httplib`,`http.client`— Instrumentieren Sie Low-Level-HTTP-Clients und die Bibliotheken auf höherer Ebene, die sie verwenden.
- `sqlite3`— Instrumentieren Sie SQLite-Clients.
- `mysql-connector-python`— Instrumentieren Sie MySQL-Clients.
- `pg8000`— Instrument Pure-Python-PostgreSQL-Schnittstelle.
- `psycopg2`— Instrument: PostgreSQL-Datenbankadapter.
- `pymongo`— Instrumentieren Sie MongoDB-Clients.
- `pymysql`— InstrumentPyMySQL-basierte Clients für MySQL und MariaDB.

Wenn Sie eine gepatchte Bibliothek verwenden, erstellt das X-Ray-SDK für Python ein Untersegment für den Aufruf und zeichnet Informationen aus der Anfrage und Antwort auf. Für das SDK muss ein Segment zur Verfügung stehen, damit es ein Untersegment aus der SDK-Middleware oder aus AWS Lambda erstellen kann.

> **Note**
>
> Wenn Sie SQLAlchemy ORM verwenden, können Sie Ihre SQL-Abfragen instrumentieren, indem Sie die SDK-Version der Sitzungs- und Abfrageklassen von SQLAlchemy importieren. Anweisungen hierzu finden Sie unter Verwenden von SQLAlchemy ORM.

Um alle verfügbaren Bibliotheken zu patchen, verwenden Sie die `patch_all`-Funktion in `aws_xray_sdk.core`. Einige Bibliotheken, wie z. B. `httplib` und `urllib`, müssen möglicherweise doppelte Patches durch Aufruf von `patch_all(double_patch=True)` aktivieren.

Example main.py — Patchen Sie alle unterstützten Bibliotheken

```
import boto3
import botocore
import requests
import sqlite3

from aws_xray_sdk.core import xray_recorder
from aws_xray_sdk.core import patch_all

patch_all()
```

Rufen Sie zum Patchen einer einzelnen Bibliothek `patch` mit einem Tupel des Bibliotheksnamens auf. Um dies zu erreichen, müssen Sie eine einzelne Elementliste bereitstellen.

Example main.py — Patchen Sie bestimmte Bibliotheken

```
import boto3
import botocore
import requests
import mysql-connector-python

from aws_xray_sdk.core import xray_recorder
from aws_xray_sdk.core import patch

libraries = (['botocore'])
patch(libraries)
```

> **Note**
>
> In einigen Fällen stimmt der Schlüssel, mit dem Sie eine Bibliothek patchen, nicht mit dem Namen der Bibliothek überein. Einige Schlüssel dienen als Aliase für eine oder mehrere Bibliotheken.
>
> Aliase für Bibliotheken
>
> - `httplib`—`httplib` und `http.client`
> - `mysql` – `mysql-connector-python`

Ablaufverfolgungskontext für asynchrone Vorgänge

Für `asyncio`integrierte Bibliotheken oder zu Untersegmente für asynchrone Funktionen erstellen, müssen Sie auch das X-Ray-SDK für Python mit einem asynchronen Kontext konfigurieren. Importieren Sie das `AsyncContext`Klasse und übergibt eine Instanz davon an den X-Ray-Recorder.

> **Note**
>
> Bibliotheken für die Unterstützung von Web-Frameworks, wie beispielsweise AIOHTTP, werden nicht über das Modul `aws_xray_sdk.core.patcher` bearbeitet. Sie werden nicht im `patcher`-Katalog unterstützter Bibliotheken angezeigt.

Example main.py — Patch aioboto3

```
import asyncio
import aioboto3
import requests

from aws_xray_sdk.core.async_context import AsyncContext
from aws_xray_sdk.core import xray_recorder
xray_recorder.configure(service='my_service', context=AsyncContext())
from aws_xray_sdk.core import patch

libraries = (['aioboto3'])
patch(libraries)
```

## Nachverfolgen von AWS SDK-Aufrufen mit dem X-Ray-SDK für Python

Wenn Ihre Anwendung Aufrufe an tätigt, AWS-Services um Daten zu speichern, in eine Warteschlange zu schreiben oder Benachrichtigungen zu senden, verfolgt das X-Ray SDK for Python die Aufrufe nachgelagert in Untersegmenten . Nachverfolgte AWS-Services und Ressourcen, auf die Sie innerhalb dieser Services zugreifen (z. B. ein Amazon S3-Bucket oder eine Amazon SQS-Warteschlange), werden auf der Ablaufverfolgungskarte in der X-Ray-Konsole als nachgelagerte Knoten angezeigt.

Das X-Ray SDK for Python instrumentiert automatisch alle AWS SDK-Clients, wenn Sie die `botocore` Bibliothek patchen. Einzelne Clients können nicht instrumentiert werden.

Für alle Services können Sie den Namen der API mit dem Namen in der X-Ray-Konsole sehen. Für eine Teilmenge von -Services fügt das X-Ray-SDK dem Segment Informationen hinzu, um mehr Granularität in der Service-Übersicht zu gewährleisten.

Wenn Sie beispielsweise einen Aufruf mit einem instrumentierten DynamoDB-Client tätigen, fügt das SDK dem Segment den Tabellennamen für Aufrufe hinzu, die auf eine Tabelle abzielen. In der Konsole wird jede Tabelle als separater Knoten in der Service-Übersicht angezeigt, mit einem generischen DynamoDB-Knoten für Aufrufe, die keine Tabelle anvisieren.

Example Untersegment für einen Aufruf an DynamoDB zum Speichern eines Elements

```
{
  "id": "24756640c0d0978a",
  "start_time": 1.480305974194E9,
```

```
  "end_time": 1.4803059742E9,
  "name": "DynamoDB",
  "namespace": "aws",
  "http": {
    "response": {
      "content_length": 60,
      "status": 200
    }
  },
  "aws": {
    "table_name": "scorekeep-user",
    "operation": "UpdateItem",
    "request_id": "UBQNSO5AEM8T4FDA4RQDEB94OVTDRVV4K4HIRGVJF66Q9ASUAAJG",
  }
}
```

Wenn Sie auf benannte Ressourcen zugreifen, werden durch Aufrufe der folgenden Services weitere Knoten in der Service-Übersicht erstellt. Durch Aufrufe, die keinen bestimmten Ressourcen gelten, wird ein generischer Knoten für den Service erstellt.

- Amazon DynamoDB – Tabellenname
- Amazon Simple Storage Service – Bucket- und Schlüsselname
- Amazon Simple Queue Service – Name der Warteschlange

## Nachverfolgen von Aufrufen nachgelagerter HTTP-HTTP-Nachgelagerter HTTP-HTTP-Web-Services mit

Wenn Ihre Anwendung Microservices oder öffentliche HTTP-APIs aufruft, können Sie diese Aufrufe mit dem X-Ray SDK für Aufrufe hinzufügen und die API der Service-Grafik als nachgelagerten Service hinzufügen.

Um HTTP-Clients zu instrumentieren, patchen Sie die Bibliothek, mit der Sie ausgehende Aufrufe durchführen. Wenn Sie `requests` oder den integrierten HTTP-Client von Python verwenden, brauchen Sie nichts weiter zu tun. Bei `aiohttp` müssen Sie zudem den Recorder mit einem asynchronen Kontext konfigurieren.

Wenn Sie die `aiohttp` 3-Client-API verwenden, müssen Sie auch die `ClientSession` mit einer Instance der Nachverfolgungskonfiguration konfigurieren, die vom SDK bereitgestellt wird.

Example **`aiohttp`** 3 Client-API

```
from aws_xray_sdk.ext.aiohttp.client import aws_xray_trace_config

async def foo():
    trace_config = aws_xray_trace_config()
    async with ClientSession(loop=loop, trace_configs=[trace_config]) as session:
        async with session.get(url) as resp
            await resp.read()
```

Wenn Sie einen Aufruf an eine nachgelagerte HTTP--API instrumentieren, erfasst das X-Ray SDK für Python ein Untersegment mit Informationen über die HTTP-Anforderung und Antwort. X-Ray verwendet das Untersegment, um ein abgeleitetes Segment für die entfernte API zu generieren.

Example Untersegment für einen nachgelagerten HTTP-Aufruf

```
{
  "id": "004f72be19cddc2a",
  "start_time": 1484786387.131,
  "end_time": 1484786387.501,
  "name": "names.example.com",
  "namespace": "remote",
  "http": {
    "request": {
      "method": "GET",
      "url": "https://names.example.com/"
    },
    "response": {
      "content_length": -1,
      "status": 200
    }
  }
}
```

Example Abgeleitetes Segment für einen nachgelagerten HTTP-Anruf

```
{
  "id": "168416dc2ea97781",
  "name": "names.example.com",
  "trace_id": "1-62be1272-1b71c4274f39f122afa64eab",
  "start_time": 1484786387.131,
  "end_time": 1484786387.501,
```

```
  "parent_id": "004f72be19cddc2a",
  "http": {
    "request": {
      "method": "GET",
      "url": "https://names.example.com/"
    },
    "response": {
      "content_length": -1,
      "status": 200
    }
  },
  "inferred": true
}
```

## Generieren benutzerdefinierter Untersegmente mit dem X-Ray-SDK für Python

Teilsegmente erweitern ein TraceAbschnittmit Details über die geleistete Arbeit, um eine Anfrage zu stellen. Jedes Mal, wenn Sie einen Aufruf mit einem instrumentierten Client erstellen, erfasst das X-Ray-SDK die in einem Untersegment generierten Informationen. Sie können zusätzliche Teilsegmente erstellen, um andere Teilsegmente zu gruppieren, die Leistung eines Codeabschnitts zu messen oder Anmerkungen und Metadaten aufzuzeichnen.

Um Untersegmente zu verwalten, verwenden Sie die Methoden `begin_subsegment` und `end_subsegment`.

Example main.py — benutzerdefiniertes Untersegment

```
from aws_xray_sdk.core import xray_recorder

subsegment = xray_recorder.begin_subsegment('annotations')
subsegment.put_annotation('id', 12345)
xray_recorder.end_subsegment()
```

Um ein Untersegment für eine synchrone Funktion zu erstellen, verwenden Sie den `@xray_recorder.capture`-Decorator. Sie können einen Namen für das Untersegment an die Erfassungsfunktion übergeben oder diesen weglassen und den Funktionsnamen verwenden.

Example main.py — Funktionsuntersegment

```
from aws_xray_sdk.core import xray_recorder
```

```
@xray_recorder.capture('## create_user')
def create_user():
...
```

Verwenden Sie bei einer asynchronen Funktion den `@xray_recorder.capture_async`-Decorator und übergeben Sie einen asynchronen Kontext an den Recorder.

Example main.py — Untersegment asynchrones Funktionsuntersegment

```
from aws_xray_sdk.core.async_context import AsyncContext
from aws_xray_sdk.core import xray_recorder
xray_recorder.configure(service='my_service', context=AsyncContext())

@xray_recorder.capture_async('## create_user')
async def create_user():
    ...

async def main():
    await myfunc()
```

Beim Erstellen eines Untersegments innerhalb eines Segments oder eines anderen Untersegments generiert das X-Ray-SDK für Python eine ID dafür und erfasst die Start- und Endzeit.

Example Untersegment mit Metadaten

```
"subsegments": [{
  "id": "6f1605cd8a07cb70",
  "start_time": 1.480305974194E9,
  "end_time": 1.4803059742E9,
  "name": "Custom subsegment for UserModel.saveUser function",
  "metadata": {
    "debug": {
      "test": "Metadata string from UserModel.saveUser"
    }
  },
```

## Hinzufügen von Anmerkungen und Metadaten zu Segmenten mit dem X-Ray SDK für Python

Sie können Anmerkungen und Metadaten verwenden, um zusätzliche Informationen über Anfragen, die Umgebung oder Ihre Anwendung aufzuzeichnen. Sie können Anmerkungen und

Metadaten zu den Segmenten hinzufügen, die das X-Ray SDK erstellt, oder zu benutzerdefinierten Untersegmenten, die Sie erstellen.

Anmerkungen sind Schlüssel-Wert-Paare mit Zeichenfolgen-, Zahlen- oder booleschen Werten. Anmerkungen sind für die Verwendung mit Filterausdrücken indexiert. Berücksichtigen Sie Anmerkungen, um Daten zur Gruppierung von Ablaufverfolgungen in der Konsole zu verwenden, oder wenn Sie die `GetTraceSummaries`-API aufrufen.

Metadaten sind Schlüssel-Wert-Paare, die Werte beliebigen Typs enthalten können, einschließlich Objekte und Listen, aber nicht für die Verwendung mit Filterausdrücken indexiert sind. Verwenden Sie Metadaten, um zusätzliche Daten aufzuzeichnen, die Sie im Trace speichern möchten, aber nicht für die Suche verwenden müssen.

Zusätzlich zu Anmerkungen und Metadaten können Sie auch Benutzer-ID-Zeichenfolgen in Segmenten aufzeichnen. Benutzer-IDs werden in einem eigenen Feld aufgezeichnet und zur Suche indiziert.

Sections

- Anmerkungen mit dem X-Ray SDK für Python aufnehmen
- Metadaten mit dem X-Ray SDK für Python aufzeichnen
- Aufzeichnen von Benutzer-IDs mit dem X-Ray SDK für Python

## Anmerkungen mit dem X-Ray SDK für Python aufnehmen

Verwenden Sie Anmerkungen, um Informationen zu Segmenten oder Untersegmenten, die zur Suche indiziert werden sollten, aufzuzeichnen.

### Anmerkung zu Anforderungen

- Schlüssel — Der Schlüssel für eine X-Ray-Anmerkung kann bis zu 500 alphanumerische Zeichen enthalten. Sie können keine anderen Leerzeichen oder Symbole als den Unterstrich (_) verwenden.
- Werte — Der Wert für eine X-Ray-Anmerkung kann bis zu 1.000 Unicode-Zeichen enthalten.
- Die Anzahl der Anmerkungen — Sie können bis zu 50 Anmerkungen pro Spur verwenden.

### So zeichnen Sie Anmerkungen auf

1. Eine Referenz des aktuellen Segments oder Untersegments finden Sie unter `xray_recorder`.

```
from aws_xray_sdk.core import xray_recorder
...
document = xray_recorder.current_segment()
```

or

```
from aws_xray_sdk.core import xray_recorder
...
document = xray_recorder.current_subsegment()
```

2. Rufen Sie `put_annotation` mit einem Aktivierungsschlüssel und einem booleschen Wert oder einem Zeichenfolgenwert auf.

```
document.put_annotation("mykey", "my value");
```

Alternativ können Sie auch die `put_annotation`-Methode im `xray_recorder` verwenden. Mit dieser Methode werden Anmerkungen zum aktuellen Untersegment oder zum Segment aufgezeichnet, wenn kein Untersegment geöffnet ist.

```
xray_recorder.put_annotation("mykey", "my value");
```

Das SDK zeichnet Anmerkungen als Schlüssel-Wert-Paare in einem `annotations`-Objekt im Segmentdokument auf. Wenn `put_annotation` zweimal mit demselben Schlüssel aufgerufen wird, werden zuvor aufgezeichnete Werte im gleichen Segment oder Untersegment überschrieben.

Nutzen Sie das `annotations.`*`key`*-Schlüsselwort in einem Filterausdruck, um Ablaufverfolgungen durch Anmerkungen mit bestimmten Werten zu finden.

Metadaten mit dem X-Ray SDK für Python aufzeichnen

Verwenden Sie Metadaten, um Segment- oder Untersegmentinformationen aufzuzeichnen, die nicht zur Suche indiziert werden müssen. Metadatenwerte sind Zeichenfolgen, Zahlen, boolesche Werte oder andere Objekte, die in Form eines JSON-Objekts oder eines Arrays angeordnet sein können.

So zeichnen Sie Metadaten auf

1. Eine Referenz des aktuellen Segments oder Untersegments finden Sie unter `xray_recorder`.

```
from aws_xray_sdk.core import xray_recorder
...
document = xray_recorder.current_segment()
```

or

```
from aws_xray_sdk.core import xray_recorder
...
document = xray_recorder.current_subsegment()
```

2. Rufen Sie `put_metadata` mit einem Zeichenfolgenschlüssel, einem booleschen Wert, einer Zahl, einer Zeichenfolge oder einem Objektwert und einem Zeichenfolgen-Namespace auf.

```
document.put_metadata("my key", "my value", "my namespace");
```

or

Rufen Sie `put_metadata` nur mit einem Aktivierungsschlüssel und einem Wert auf.

```
document.put_metadata("my key", "my value");
```

Alternativ können Sie auch die `put_metadata`-Methode im `xray_recorder` verwenden. Mit dieser Methode werden Metadaten zum aktuellen Untersegment oder zum Segment aufgezeichnet, wenn kein Untersegment geöffnet ist.

```
xray_recorder.put_metadata("my key", "my value");
```

Wenn Sie keinen Namespace angeben, verwendet SDK `default`. Wenn `put_metadata` zweimal mit demselben Schlüssel aufgerufen wird, werden zuvor aufgezeichnete Werte im gleichen Segment oder Untersegment überschrieben.

Aufzeichnen von Benutzer-IDs mit dem X-Ray SDK für Python

Zeichnen Sie Benutzer-IDs in Anforderungssegmenten auf, um den Benutzer zu identifizieren, der die Anforderung gesendet hat.

So zeichnen Sie Benutzer-IDs auf

1. Eine Referenz des aktuellen Segments finden Sie unter `xray_recorder`.

   ```
   from aws_xray_sdk.core import xray_recorder
   ...
   document = xray_recorder.current_segment()
   ```

2. Rufen Sie `setUser` mit einer Zeichenfolgen-ID des Benutzers auf, der die Anforderung gesendet hat.

   ```
   document.set_user("U12345");
   ```

Sie können `set_user` in Ihrem Controller aufrufen, um die Benutzer-ID aufzuzeichnen, sobald die Anwendung mit der Bearbeitung einer Anfrage beginnt.

Nutzen Sie das `user`-Schlüsselwort in einem Filterausdruck, um Ablaufverfolgungen einer Benutzer-ID zu finden.

## Instrumentieren von in serverlosen Umgebungen bereitgestellten Web-Frameworks

Das AWS X-Ray SDK for Python unterstützt die Instrumentierung von Web-Frameworks, die in Serverless-Anwendungen bereitgestellt werden. Die Cloud ist von Natur aus serverlos und dies bietet Ihnen die Möglichkeit, immer mehr Ihrer betrieblichen Verantwortung auf AWS zu verlagern. Der unschätzbare Nutzen für Sie sind mehr Agilität und Innovation.

Eine serverlose Architektur ist ein Softwareanwendungsmodell, mit dem Sie Anwendungen und Services entwickeln und ausführen können, ohne über Server nachdenken zu müssen. Für Sie verringern sich die Aufgaben der Infrastrukturverwaltung wie die Bereitstellung von Servern oder Clustern, Patching, Betriebssystemwartung und Kapazitätsbereitstellung. Sie können serverlose Lösungen für praktisch jeden Anwendungstyp oder Backend-Service erstellen und alles, was zum Ausführen und Skalieren Ihrer Anwendung mit hoher Verfügbarkeit erforderlich ist, wird für Sie durchgeführt.

Dieses Tutorial zeigt Ihnen, wie Sie automatisch AWS X-Ray in einem Web-Framework wie Flask oder Django instrumentieren, das in einer Serverless-Umgebung bereitgestellt wird. Mit der X-Ray-Instrumentierung der Anwendung können Sie alle nachgelagerten Aufrufe anzeigen, die von Amazon API Gateway über Ihre AWS Lambda Funktion und die ausgehenden Aufrufe Ihrer Anwendung getätigt werden.

Das X-Ray SDK for Python unterstützt die folgenden Python-Anwendungs-Frameworks:

- Flask-Version 0.8 oder höher
- Django-Version 1.0 oder höher

In diesem Tutorial wird eine Beispielanwendung für Serverless entwickelt, die auf Lambda bereitgestellt und von API Gateway aufgerufen wird. In diesem Tutorial wird Zappa verwendet, um die Anwendung automatisch in Lambda bereitzustellen und den API Gateway-Endpunkt zu konfigurieren.

Voraussetzungen

- Zappa
- Python – Version 2.7 oder 3.6.
- AWS CLI – Stellen Sie sicher, dass Ihr mit dem Konto und konfiguriert AWS CLI ist, AWS-Region in dem Sie Ihre Anwendung bereitstellen.
- Pip
- Virtualenv

Schritt 1: Erstellen einer Umgebung

In diesem Schritt erstellen Sie eine virtuelle Umgebung mit `virtualenv` zum Hosten einer Anwendung.

1. AWS CLI Erstellen Sie mithilfe der ein Verzeichnis für die Anwendung. Wechseln Sie anschließend zum neuen Verzeichnis.

   ```
   mkdir serverless_application
   cd serverless_application
   ```

2. Erstellen Sie dann eine virtuelle Umgebung in Ihrem neuen Verzeichnis. Aktivieren Sie sie mit dem folgenden Befehl.

   ```
   # Create our virtual environment
   virtualenv serverless_env

   # Activate it
   source serverless_env/bin/activate
   ```

3. Installieren Sie X-Ray, Flask, Zappa und die Requests-Bibliothek in Ihrer Umgebung.

```
# Install X-Ray, Flask, Zappa, and Requests into your environment
pip install aws-xray-sdk flask zappa requests
```

4. Fügen Sie Anwendungscode zum Verzeichnis `serverless_application` hinzu. In diesem Beispiel bauen wir auf dem Flask-Beispiel Hallo Welt auf.

   Erstellen Sie in dem Verzeichnis `serverless_application` eine Datei namens `my_app.py`. Fügen Sie anschließend mit einem Texteditor die folgenden Befehle hinzu. Diese Anwendung instrumentiert die Anfragebibliothek, führt Patches für die Middleware der Flask-Anwendung aus und öffnet den Endpunkt `'/'`.

```
# Import the X-Ray modules
from aws_xray_sdk.ext.flask.middleware import XRayMiddleware
from aws_xray_sdk.core import patcher, xray_recorder
from flask import Flask
import requests

# Patch the requests module to enable automatic instrumentation
patcher.patch(('requests',))

app = Flask(__name__)

# Configure the X-Ray recorder to generate segments with our service name
xray_recorder.configure(service='My First Serverless App')

# Instrument the Flask application
XRayMiddleware(app, xray_recorder)

@app.route('/')
def hello_world():
    resp = requests.get("https://aws.amazon.com")
    return 'Hello, World: %s' % resp.url
```

### Schritt 2: Erstellen und Bereitstellen einer Zappa-Umgebung

In diesem Schritt verwenden Sie Zappa, um automatisch einen API Gateway-Endpunkt zu konfigurieren und dann in Lambda bereitzustellen.

1. Initialisieren Sie Zappa über das Verzeichnis `serverless_application`. In diesem Beispiel verwenden wir die Standardeinstellungen. Wenn Sie jedoch über individuelle Voreinstellungen verfügen, zeigt Zappa Anleitungen für die Konfiguration an.

```
zappa init
```

```
What do you want to call this environment (default 'dev'): dev
...
What do you want to call your bucket? (default 'zappa-*******'): zappa-*******
...
...
It looks like this is a Flask application.
What's the modular path to your app's function?
This will likely be something like 'your_module.app'.
We discovered: my_app.app
Where is your app's function? (default 'my_app.app'): my_app.app
...
Would you like to deploy this application globally? (default 'n') [y/n/
(p)rimary]: n
```

2. Aktivieren Sie X-Ray. Öffnen Sie die Datei `zappa_settings.json` und stellen Sie sicher, dass sie dem Beispiel ähnelt.

```
{
    "dev": {
        "app_function": "my_app.app",
        "aws_region": "us-west-2",
        "profile_name": "default",
        "project_name": "serverless-exam",
        "runtime": "python2.7",
        "s3_bucket": "zappa-*********"
    }
}
```

3. Fügen Sie `"xray_tracing": true` als Eintrag zur Konfigurationsdatei hinzu.

```
{
    "dev": {
        "app_function": "my_app.app",
        "aws_region": "us-west-2",
        "profile_name": "default",
        "project_name": "serverless-exam",
```

```
        "runtime": "python2.7",
        "s3_bucket": "zappa-*********",
        "xray_tracing": true
    }
}
```

4. Stellen Sie die Anwendung bereit. Dadurch wird der API Gateway-Endpunkt automatisch konfiguriert und Ihr Code wird in Lambda hochgeladen.

```
zappa deploy
```

```
...
Deploying API Gateway..
Deployment complete!: https://**********.execute-api.us-west-2.amazonaws.com/dev
```

Schritt 3: Aktivieren der X-Ray-Ablaufverfolgung für API Gateway

In diesem Schritt interagieren Sie mit der API Gateway-Konsole, um die X-Ray-Nachverfolgung zu aktivieren.

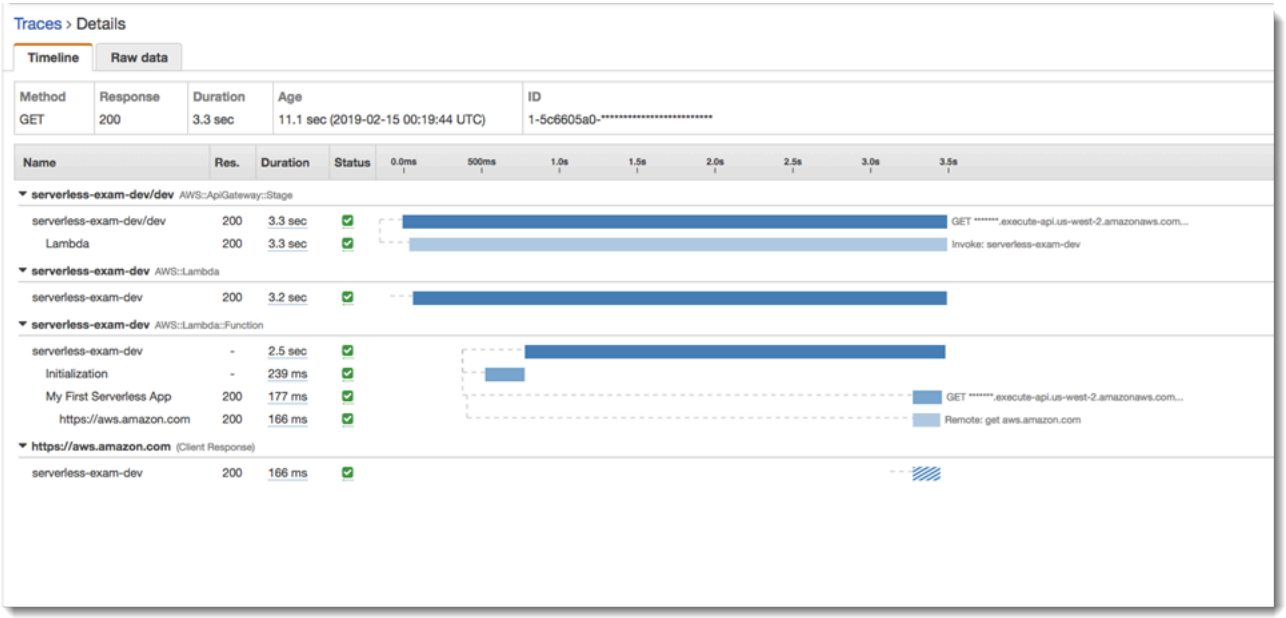
1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die API Gateway-Konsole unter https://console.aws.amazon.com/apigateway/.
2. Suchen Sie Ihre neu generierte API. Sie sollte in etwa so aussehen: `serverless-exam-dev`.
3. Wählen Sie Stages.
4. Wählen Sie den Namen der Bereitstellungsstufe. Der Standardwert ist `dev`.
5. Wählen Sie auf der Registerkarte Logs/Tracing (Protokolle/Nachverfolgung) die Option Enable X-Ray Tracing (X-Ray-Tracing aktivieren) aus.
6. Wählen Sie Save Changes.
7. Rufen Sie den Endpunkt in Ihrem Browser auf. Wenn Sie die Beispielanwendung `Hello World` verwendet haben, sollte Folgendes angezeigt werden.

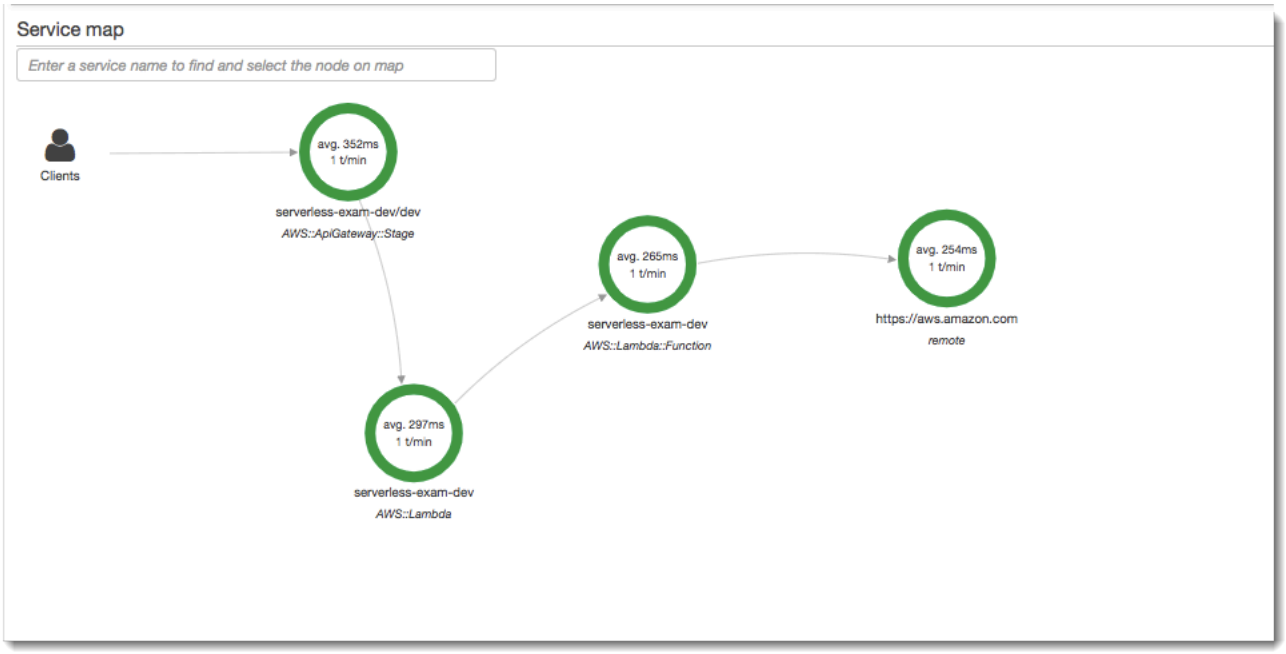
```
"Hello, World: https://aws.amazon.com/"
```

### Schritt 4: Anzeigen der erstellten Nachverfolgung

In diesem Schritt interagieren Sie mit der X-Ray-Konsole, um die von der Beispielanwendung erstellte Ablaufverfolgung anzuzeigen. Einen detaillierteren Leitfaden für die Analyse von Nachverfolgungen finden Sie unter Anzeigen der Service-Übersicht.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die X-Ray-Konsole unter https://console.aws.amazon.com/xray/home.
2. Zeigen Sie von API Gateway generierte Segmente, die Lambda-Funktion und den Lambda-Container an.
3. Zeigen Sie im Lambda-Funktionssegment ein Untersegment mit dem Namen an`My First Serverless App`. Dahinter folgt ein zweites Untersegment mit dem Namen `https://aws.amazon.com`.
4. Während der Initialisierung generiert Lambda möglicherweise auch ein drittes Untersegment mit dem Namen `initialization`.






Schritt 5: Bereinigen

Sie sollten Ressourcen, die Sie nicht mehr verwenden, stets beenden, um unerwartete Kosten zu vermeiden. Wie in diesem Tutorial gezeigt, lässt sich die serverlose erneute Bereitstellung mit Tools wie Zappa optimieren.

Um Ihre Anwendung aus Lambda, API Gateway und Amazon S3 zu entfernen, führen Sie den folgenden Befehl in Ihrem Projektverzeichnis mithilfe der aus AWS CLI.

```
zappa undeploy dev
```

Nächste Schritte

Fügen Sie Ihrer Anwendung weitere Funktionen hinzu, indem Sie AWS Clients hinzufügen und sie mit X-Ray instrumentieren. Weitere Informationen zu Serverless-Computing-Optionen finden Sie unter Serverless auf AWS.

# Instrumentieren Sie Ihre Bewerbung mit .NET

Es gibt zwei Möglichkeiten, Ihre .NET Anwendung so zu instrumentieren, dass sie Spuren an X-Ray sendet:

- AWS Distro for OpenTelemetry .NET — Eine AWS Distribution, die eine Reihe von Open-Source-Bibliotheken zum Senden korrelierter Metriken und Traces über AWS Distro for Collector an

mehrere AWS Überwachungslösungen wie Amazon CloudWatch und Amazon OpenSearch Service bereitstellt. AWS X-Ray OpenTelemetry

- AWS X-Ray SDK für .NET — Eine Reihe von Bibliotheken zum Generieren und Senden von Traces an X-Ray über den X-Ray-Daemon.

Weitere Informationen finden Sie unter Wahl zwischen den AWS SDKs Distro for OpenTelemetry und X-Ray.

## AWS Distribution für OpenTelemetry .NET

Mit der AWS Distro for OpenTelemetry .NET können Sie Ihre Anwendungen einmal instrumentieren und korrelierte Metriken und Traces an mehrere AWS Überwachungslösungen wie Amazon CloudWatch und Amazon AWS X-Ray OpenSearch Service senden. Die Verwendung von X-Ray mit AWS Distro for OpenTelemetry erfordert zwei Komponenten: ein OpenTelemetry SDK, das für die Verwendung mit X-Ray aktiviert ist, und das AWS Distro for OpenTelemetry Collector, das für die Verwendung mit X-Ray aktiviert ist.

Informationen zu den ersten Schritten finden Sie in der Dokumentation zur AWS Distribution. OpenTelemetry .NET

Weitere Informationen zur Verwendung von AWS Distro for OpenTelemetry with AWS X-Ray und anderen AWS-Services finden Sie unter AWS Distro for OpenTelemetry oder The AWS Distro for Documentation. OpenTelemetry

Weitere Informationen zur Sprachunterstützung und -verwendung finden Sie unter AWS Observability on. GitHub

## AWS X-Ray SDK für .NET

Das X-Ray SDK for .NET ist eine Bibliothek zur Instrumentierung von C#-.NET-Webanwendungen, .NET Core-Webanwendungen und .NET Core-Funktionen in AWS Lambda. Es bietet Klassen und Methoden zum Generieren und Senden von Ablaufverfolgungsdaten an den X-Ray-Daemon . Dazu gehören Informationen über eingehende Anforderungen, die von der Anwendung bedient werden, und Aufrufe, die die Anwendung an nachgelagerte AWS-Services, HTTP-Web-APIs und SQL-Datenbanken durchführt.

> **Note**
>
> Das X-Ray SDK for .NET ist ein Open-Source-Projekt. Sie können dem Projekt folgen und Probleme und Pull-Anfragen an senden GitHub: github.com/aws/aws-xray-sdk-dotnet

Für Webanwendungen fügen Sie zunächst einen Message Handler zu Ihrer Web-Konfiguration hinzu, um eingehende Anforderungen zu verfolgen. Der Message Handler erstellt für jede rückverfolgte Anforderung ein Segment und vervollständigt das Segment, nachdem die Antwort gesendet wurde. Während das Segment geöffnet ist, können Sie die SDK-Client-Methoden nutzen, um dem Segment Informationen hinzuzufügen, Untersegmente zu erstellen und nachgelagerte Aufrufe rückzuverfolgen. Das SDK erfasst auch automatisch Ausnahmen, die Ihre Anwendung ausgibt, während das Segment geöffnet ist.

Für Lambda-Funktionen, die von einer instrumentierten Anwendung oder einem instrumentierten Service aufgerufen werden, liest Lambda den Nachverfolgungs-Header und verfolgt Stichprobenanforderungen automatisch. Für andere Funktionen können Sie Lambda so konfigurieren, dass eingehende Anforderungen erfasst und verfolgt werden. In beiden Fällen erstellt Lambda das Segment und stellt es dem X-Ray-SDK zur Verfügung.

> **Note**
>
> Auf Lambda ist das X-Ray-SDK optional. Wenn Sie es nicht in Ihrer Funktion verwenden, enthält Ihre Service-Übersicht immer noch einen Knoten für den Lambda-Service und einen für jede Lambda-Funktion. Durch Hinzufügen des SDK können Sie Ihren Funktionscode instrumentieren, um dem von Lambda aufgezeichneten Funktionssegment Untersegmente hinzuzufügen. Weitere Informationen finden Sie unter AWS Lambda und AWS X-Ray.

Verwenden Sie als Nächstes das X-Ray SDK for .NET, um Ihre AWS SDK for .NET Clients zu instrumentieren. Jedes Mal, wenn Sie einen Aufruf an eine Downstream- AWS-Service oder -Ressource mit einem instrumentierten Client tätigen, zeichnet das SDK Informationen über den Aufruf in einem Untersegment auf. AWS -Services und die Ressourcen, auf die Sie innerhalb der Services zugreifen, werden als Downstream-Knoten auf der Ablaufverfolgungskarte angezeigt, um Ihnen bei der Identifizierung von Fehlern und Drosselungsproblemen bei einzelnen Verbindungen zu helfen.

Das X-Ray SDK for .NET bietet auch Instrumentierung für nachgelagerte Aufrufe von HTTP-Web-APIs und SQL-Datenbanken. Die `GetResponseTraced`-Erweiterungsmethode für

`System.Net.HttpWebRequest` verfolgt ausgehende HTTP-Aufrufe. Sie können die Version des X-Ray SDK for .NET verwenden`SqlCommand`, um SQL-Abfragen zu instrumentieren.

Nachdem Sie das SDK verwendet haben, passen Sie sein Verhalten an, indem Sie den Recorder und den Message Handler konfigurieren. Sie können Plugins zum Festhalten von Daten über die Datenverarbeitungsressourcen, auf denen Ihre Anwendung ausgeführt wird, hinzufügen, das Samplingverhalten durch Samplingregeln anpassen und Protokollebenen einrichten, um mehr oder weniger Informationen von dem SDK in Ihren Anwendungsprotokollen zu sehen.

Zeichnen Sie zusätzliche Informationen zu Anforderungen und den Aufgaben, die Ihre Anwendung ausführt, in Anmerkungen und Metadaten auf. Anmerkungen sind einfache Schlüsselwertpaare, die für die Verwendung mit Filterausdrücken indiziert werden, damit Sie nach Ablaufverfolgen mit bestimmten Daten suchen können. Metadateneinträge sind weniger einschränkend und können ganze Objekte und Arrays aufzeichnen – alle Daten, die in eine JSON zusammengefasst werden können.

> **Anmerkungen und Metadaten**
>
> Anmerkungen und Metadaten sind beliebiger Text, den Sie Segmenten mit dem X-Ray-SDK hinzufügen. Anmerkungen werden für die Verwendung mit Filterausdrücken indiziert. Metadaten werden nicht indiziert, können aber im Rohsegment mit der X-Ray-Konsole oder API angezeigt werden. Jeder, dem Sie Lesezugriff auf X-Ray gewähren, kann diese Daten anzeigen.

Wenn Sie viele instrumentierten Clients in Ihrem Code haben, kann ein einzelnes Anforderungssegmente viele Untersegmente enthalten, eines für jeden Aufruf mit einem instrumentierten Client. Sie können Untersegmente organisieren und gruppieren, indem Sie Client-Aufrufe in benutzerdefinierten Untersegmenten zusammenfassen. Sie können ein benutzerdefiniertes Untersegment für eine ganze Funktion oder eine Code-Abschnitt erstellen und Metadaten und Anmerkungen im Untersegment festhalten, anstatt alles im übergeordneten Segment aufzuzeichnen.

Referenzdokumentation zu den SDK-Klassen und -Methoden finden Sie unter:

- AWS X-Ray API-Referenz zum -SDK für .NET
- AWS X-Ray SDK für .NET Core – API-Referenz

Das Paket unterstützt sowohl .NET als auch .NET Core. Die verwendeten Klassen sind jedoch unterschiedlich. Beispiele in diesem Kapitel verweisen auf die .NET API-Referenz, es sei denn, die Klasse ist für .NET Core spezifisch.

## Voraussetzungen

Das X-Ray SDK for .NET erfordert das .NET Framework 4.5 oder höher und AWS SDK for .NET.

Für .NET Core-Anwendungen und -Funktionen erfordert das SDK .NET Core 2.0 oder höher.

## Hinzufügen des X-Ray SDK for .NET zu Ihrer Anwendung

Verwenden Sie NuGet , um das X-Ray SDK for .NET zu Ihrer Anwendung hinzuzufügen.

So installieren Sie das X-Ray SDK for .NET mit dem NuGet Paketmanager in Visual Studio

1. Wählen Sie Tools , NuGet Package Manager , NuGet Pakete für Lösung verwalten aus.
2. Suchen Sie nach AWSXRayRecorder.
3. Wählen sie das Paket und dann Installieren.

## Abhängigkeitsmanagement

Das X-Ray SDK for .NET ist über Nuget verfügbar. Installieren Sie das SDK mit dem Paketmanager:

```
Install-Package AWSXRayRecorder -Version 2.10.1
```

Das nuget`AWSXRayRecorder v2.10.1`-Paket hat die folgenden Abhängigkeiten:

NET Framework 4.5

```
AWSXRayRecorder (2.10.1)
|
|-- AWSXRayRecorder.Core (>= 2.10.1)
|   |-- AWSSDK.Core (>= 3.3.25.1)
|
|-- AWSXRayRecorder.Handlers.AspNet (>= 2.7.3)
|   |-- AWSXRayRecorder.Core (>= 2.10.1)
|
|-- AWSXRayRecorder.Handlers.AwsSdk (>= 2.8.3)
```

```
|   |-- AWSXRayRecorder.Core (>= 2.10.1)
|
|-- AWSXRayRecorder.Handlers.EntityFramework (>= 1.1.1)
|   |-- AWSXRayRecorder.Core (>= 2.10.1)
|   |-- EntityFramework (>= 6.2.0)
|
|-- AWSXRayRecorder.Handlers.SqlServer (>= 2.7.3)
|   |-- AWSXRayRecorder.Core (>= 2.10.1)
|
|-- AWSXRayRecorder.Handlers.System.Net (>= 2.7.3)
    |-- AWSXRayRecorder.Core (>= 2.10.1)
```

### NET Framework 2.0

```
AWSXRayRecorder (2.10.1)
|
|-- AWSXRayRecorder.Core (>= 2.10.1)
|   |-- AWSSDK.Core (>= 3.3.25.1)
|   |-- Microsoft.AspNetCore.Http (>= 2.0.0)
|   |-- Microsoft.Extensions.Configuration (>= 2.0.0)
|   |-- System.Net.Http (>= 4.3.4)
|
|-- AWSXRayRecorder.Handlers.AspNetCore (>= 2.7.3)
|   |-- AWSXRayRecorder.Core (>= 2.10.1)
|   |-- Microsoft.AspNetCore.Http.Extensions (>= 2.0.0)
|   |-- Microsoft.AspNetCore.Mvc.Abstractions (>= 2.0.0)
|
|-- AWSXRayRecorder.Handlers.AwsSdk (>= 2.8.3)
|   |-- AWSXRayRecorder.Core (>= 2.10.1)
|
|-- AWSXRayRecorder.Handlers.EntityFramework (>= 1.1.1)
|   |-- AWSXRayRecorder.Core (>= 2.10.1)
|   |-- Microsoft.EntityFrameworkCore.Relational (>= 3.1.0)
|
|-- AWSXRayRecorder.Handlers.SqlServer (>= 2.7.3)
|   |-- AWSXRayRecorder.Core (>= 2.10.1)
|   |-- System.Data.SqlClient (>= 4.4.0)
|
|-- AWSXRayRecorder.Handlers.System.Net (>= 2.7.3)
    |-- AWSXRayRecorder.Core (>= 2.10.1)
```

Weitere Informationen zur Abhängigkeitsverwaltung finden Sie in der Microsoft-Dokumentation zu Nuget-Abhängigkeit und Nuget-Abhängigkeitsauflösung.

## Konfiguration des X-Ray-SDK SDK for .NET

Sie können das X-Ray-SDK SDK for .NET mit Plug-ins so konfigurieren, dass es Informationen über den Dienst enthält, auf dem Ihre Anwendung ausgeführt wird, das standardmäßige Sampling-Verhalten ändern oder Sampling-Regeln hinzufügen, die für Anfragen an bestimmte Pfade gelten.

Für .NET-Webanwendungen fügen Sie dem Abschnitt `appSettings` Ihrer `Web.config`-Datei Schlüssel hinzu.

Example Web.config

```
<configuration>
  <appSettings>
    <add key="AWSXRayPlugins" value="EC2Plugin"/>
    <add key="SamplingRuleManifest" value="sampling-rules.json"/>
  </appSettings>
</configuration>
```

Erstellen Sie für .NET Core eine Datei mit dem Namen `appsettings.json` mit einem Top-Level-Schlüssel namens `XRay`.

Example .NET appsettings.json

```
{
  "XRay": {
    "AWSXRayPlugins": "EC2Plugin",
    "SamplingRuleManifest": "sampling-rules.json"
  }
}
```

Erstellen Sie dann in Ihrem Anwendungscode ein Konfigurationsobjekt und verwenden Sie es, um den X-Ray-Recorder zu initialisieren. Führen Sie dies aus, bevor Sie den Recorder initialisieren.

Example .NET Core Program.cs — Rekorder-Konfiguration

```
using Amazon.XRay.Recorder.Core;
...
AWSXRayRecorder.InitializeInstance(configuration);
```

Wenn Sie eine .NET Core-Webanwendung instrumentieren, können Sie bei der Konfiguration des Message-Handlers auch das Konfigurationsobjekt an die `UseXRay` Methode übergeben. Verwenden Sie für Lambda-Funktionen die oben gezeigte `InitializeInstance` Methode.

Weitere Informationen zur .NET Core-Konfigurations-API finden Sie unter Konfigurieren einer ASP.NET-Core-App auf docs.microsoft.com.

Sections

- Plug-ins
- Samplingregeln
- Protokollierung (.NET)
- Protokollierung (.NET Core)
- Umgebungsvariablen

## Plug-ins

Verwenden Sie Plugins zum Hinzufügen von Daten über den Service, der Ihre Anwendung hostet.

Plug-ins

- Amazon EC2 — `EC2Plugin` fügt die Instance-ID, die Availability Zone und die CloudWatch Logs-Gruppe hinzu.
- Elastic Beanstalk — `ElasticBeanstalkPlugin` fügt den Umgebungsnamen, die Versionsbezeichnung und die Bereitstellungs-ID hinzu.
- Amazon ECS — `ECSPlugin` fügt die Container-ID hinzu.

Um ein Plugin zu verwenden, konfigurieren Sie das X-Ray-SDK SDK for .NET .NET-Client, indem Sie die `AWSXRayPlugins` Einstellung hinzufügen. Wenn mehrere Plugins auf Ihre Anwendung zutreffen, geben Sie alle Plugins in der gleichen Einstellung an (getrennt durch Kommata).

Example Web.config – Plugins

```
<configuration>
  <appSettings>
    <add key="AWSXRayPlugins" value="EC2Plugin,ElasticBeanstalkPlugin"/>
  </appSettings>
</configuration>
```

Example .NET Core appsettings.json — Plug-ins

```
{
  "XRay": {
    "AWSXRayPlugins": "EC2Plugin,ElasticBeanstalkPlugin"
  }
}
```

Samplingregeln

Das SDK verwendet die Sampling-Regeln, die Sie in der X-Ray-Konsole definieren, um zu bestimmen, welche Anfragen aufgezeichnet werden sollen. Die Standardregel verfolgt die erste Anfrage jede Sekunde und fünf Prozent aller weiteren Anfragen aller Dienste, die Traces an X-Ray senden. Erstellen Sie zusätzliche Regeln in der X-Ray-Konsole, um die Menge der aufgezeichneten Daten für jede Ihrer Anwendungen anzupassen.

Das SDK wendet benutzerdefinierte Regeln in der Reihenfolge an, in der sie definiert sind. Wenn eine Anfrage mehreren benutzerdefinierten Regeln entspricht, wendet das SDK nur die erste Regel an.

> **Note**
>
> Wenn das SDK X-Ray nicht erreichen kann, um Sampling-Regeln abzurufen, kehrt es zu einer lokalen Standardregel zurück, die die erste zu Beginn jeder Sekunde empfangene Anfrage und fünf Prozent aller zusätzlichen Anfragen pro Host festlegt. Dies kann passieren, wenn der Host nicht berechtigt ist, Sampling-APIs aufzurufen, oder wenn er keine Verbindung zum X-Ray-Daemon herstellen kann, der als TCP-Proxy für API-Aufrufe durch das SDK fungiert.

Sie können das SDK auch so konfigurieren, dass Sampling-Regeln aus einem JSON-Dokument geladen werden. Das SDK kann lokale Regeln als Backup für Fälle verwenden, in denen X-Ray Sampling nicht verfügbar ist, oder ausschließlich lokale Regeln verwenden.

Example sampling-rules.json

```
{
  "version": 2,
  "rules": [
    {
      "description": "Player moves.",
      "host": "*",
```

```
      "http_method": "*",
      "url_path": "/api/move/*",
      "fixed_target": 0,
      "rate": 0.05
    }
  ],
  "default": {
    "fixed_target": 1,
    "rate": 0.1
  }
}
```

Dieses Beispiel definiert eine benutzerdefinierte Regel und eine Standardregel. Die benutzerdefinierte Regel wendet eine Stichprobenrate von fünf Prozent an, ohne dass eine Mindestanzahl von Anfragen für Pfade verfolgt werden muss. `/api/move/` Die Standardregel verfolgt die erste Anfrage jede Sekunde und 10 Prozent der weiteren Anfragen.

Der Nachteil der lokalen Definition von Regeln besteht darin, dass das feste Ziel von jeder Instanz des Rekorders unabhängig angewendet wird, anstatt vom X-Ray-Dienst verwaltet zu werden. Wenn Sie mehr Hosts bereitstellen, wird die feste Rate vervielfacht, wodurch es schwieriger wird, die Menge der aufgezeichneten Daten zu kontrollieren.

Wenn aktiviert AWS Lambda, können Sie die Samplerate nicht ändern. Wenn Ihre Funktion von einem instrumentierten Dienst aufgerufen wird, werden Aufrufe, die Anfragen generierten, die von diesem Dienst abgetastet wurden, von Lambda aufgezeichnet. Wenn aktives Tracing aktiviert ist und kein Tracing-Header vorhanden ist, trifft Lambda die Stichprobenentscheidung.

Um Backup-Regeln zu konfigurieren, weisen Sie das X-Ray SDK for .NET an, Sampling-Regeln aus einer Datei mit der `SamplingRuleManifest` Einstellung zu laden.

Example .NET Web.config – Samplingregeln

```
<configuration>
  <appSettings>
    <add key="SamplingRuleManifest" value="sampling-rules.json"/>
  </appSettings>
</configuration>
```

Example .NET Core appsettings.json — Sampling-Regeln

```
{
  "XRay": {
```

```
    "SamplingRuleManifest": "sampling-rules.json"
  }
}
```

Um nur lokale Regeln zu verwenden, erstellen Sie den Recorder mit einer `LocalizedSamplingStrategy`. Wenn Sie Sicherungsregeln konfiguriert haben, entfernen Sie die Konfiguration.

Example .NET global.asax — Lokale Sampling-Regeln

```
var recorder = new AWSXRayRecorderBuilder().WithSamplingStrategy(new
 LocalizedSamplingStrategy("samplingrules.json")).Build();
AWSXRayRecorder.InitializeInstance(recorder: recorder);
```

Example .NET Core Program.cs — Lokale Sampling-Regeln

```
var recorder = new AWSXRayRecorderBuilder().WithSamplingStrategy(new
 LocalizedSamplingStrategy("sampling-rules.json")).Build();
AWSXRayRecorder.InitializeInstance(configuration,recorder);
```

Protokollierung (.NET)

Das X-Ray-SDK SDK for .NET verwendet denselben Protokollierungsmechanismus wie das AWS SDK for .NET. Wenn Sie Ihre Anwendung bereits für die Protokollierung der AWS SDK for .NET Ausgabe konfiguriert haben, gilt dieselbe Konfiguration für die Ausgabe aus dem X-Ray SDK for .NET.

Zum Konfigurieren von Protokollierung fügen Sie einen Konfigurationsabschnitt mit dem Namen `aws` Ihrer `App.config`-Datei oder `Web.config`-Datei hinzu.

Example Web.config – Protokollierung

```
...
<configuration>
  <configSections>
    <section name="aws" type="Amazon.AWSSection, AWSSDK.Core"/>
  </configSections>
  <aws>
    <logging logTo="Log4Net"/>
  </aws>
</configuration>
```

Weitere Informationen finden Sie unter Konfigurieren Ihrer AWS SDK for .NET -Anwendung im AWS SDK for .NET -Entwicklerhandbuch.

Protokollierung (.NET Core)

Das X-Ray-SDK SDK for .NET verwendet dieselben Protokollierungsoptionen wie das AWS SDK for .NET. Um die Protokollierung für .NET Core-Anwendungen zu konfigurieren, übergeben Sie die Protokollierungsoption an die `AWSXRayRecorder.RegisterLogger` Methode.

Um beispielsweise log4net zu verwenden, erstellen Sie eine Konfigurationsdatei, die den Logger, das Ausgabeformat und den Speicherort der Datei definiert.

Example .NET Core log4net.config

```
<?xml version="1.0" encoding="utf-8" ?>
<log4net>
  <appender name="FileAppender" type="log4net.Appender.FileAppender,log4net">
    <file value="c:\logs\sdk-log.txt" />
    <layout type="log4net.Layout.PatternLayout">
      <conversionPattern value="%date [%thread] %level %logger - %message%newline" />
    </layout>
  </appender>
  <logger name="Amazon">
    <level value="DEBUG" />
    <appender-ref ref="FileAppender" />
  </logger>
</log4net>
```

Erstellen Sie dann den Logger und übernehmen Sie die Konfiguration in Ihren Programmcode.

Example .NET Core Program.cs — Protokollierung

```
using log4net;
using Amazon.XRay.Recorder.Core;

class Program
{
  private static ILog log;
  static Program()
  {
    var logRepository = LogManager.GetRepository(Assembly.GetEntryAssembly());
    XmlConfigurator.Configure(logRepository, new FileInfo("log4net.config"));
    log = LogManager.GetLogger(typeof(Program));
```

```
    AWSXRayRecorder.RegisterLogger(LoggingOptions.Log4Net);
  }
  static void Main(string[] args)
  {
  ...
  }
}
```

Weitere Informationen zur Konfiguration von log4net finden Sie unter Konfiguration auf logging.apache.org.

Umgebungsvariablen

Sie können Umgebungsvariablen verwenden, um das X-Ray SDK for .NET zu konfigurieren. Das SDK unterstützt die folgenden Variablen.

- `AWS_XRAY_TRACING_NAME`— Legen Sie einen Dienstnamen fest, den das SDK für Segmente verwendet. Überschreibt den für die Segmentbenennungsstrategie des Servlet-Filters festgelegten Dienstnamen.
- `AWS_XRAY_DAEMON_ADDRESS`— Legt den Host und den Port des X-Ray-Daemon-Listeners fest. Standardmäßig verwendet `127.0.0.1:2000` das SDK sowohl Trace-Daten (UDP) als auch Sampling-Daten (TCP). Verwenden Sie diese Variable, wenn Sie den Daemon so konfiguriert haben, dass er auf einem anderen Port lauscht oder wenn er auf einem anderen Host läuft.

  Format

  - Derselbe Port — *`address`*`:`*`port`*
  - Verschiedene Anschlüsse — `tcp:`*`address`*`:`*`port`* `udp:`*`address`*`:`*`port`*
- `AWS_XRAY_CONTEXT_MISSING`— Auf einstellen, `RUNTIME_ERROR` um Ausnahmen auszulösen, wenn Ihr instrumentierter Code versucht, Daten aufzuzeichnen, obwohl kein Segment geöffnet ist.

  Zulässige Werte

  - `RUNTIME_ERROR`— Löst eine Laufzeitausnahme aus.
  - `LOG_ERROR`— Fehler protokollieren und fortfahren (Standard).
  - `IGNORE_ERROR`— Fehler ignorieren und fortfahren.

  Fehler im Zusammenhang mit fehlenden Segmenten oder Untersegmenten können auftreten, wenn Sie versuchen, einen instrumentierten Client in Startcode zu verwenden, der ausgeführt wird, wenn keine Anfrage geöffnet ist, oder in Code, der einen neuen Thread erzeugt.

## Instrumentieren von eingehenden HTTP-Anforderungen mit dem X-Ray SDK for .NET

Sie können mit dem X-Ray SDK eingehende HTTP-Anforderungen verfolgen, die Ihre Anwendung auf einer EC2-Instance in Amazon EC2 bedient, verfolgen.AWS Elastic Beanstalkoder Amazon ECS.

Verwenden Sie einen Meldungs-Handler, um eingehende HTTP-Anforderungen zu instrumentieren. Wenn Sie den X-Ray Meldungshandler Ihrer Anwendung hinzufügen, erstellt das X-Ray SDK for .NET ein Segment für jede stichprobenartige Anforderung. Dieses Segment umfasst Dauer, Methode und Status der HTTP-Anforderung. Die zusätzliche Instrumentierung schafft Untersegmente zu diesem Segment.

> **Note**
>
> FürAWS LambdaFunktionen erstellt Lambda ein Segment für jede stichprobenartige Anforderung. Weitere Informationen finden Sie unter AWS Lambda und AWS X-Ray.

Jedes Segment hat einen Namen, der Ihre Anwendung in der Service-Map identifiziert. Das Segment kann statisch benannt werden, oder Sie können das SDK so konfigurieren, dass es basierend auf dem Host-Header in der eingehenden Anforderung dynamisch benennt. Mit der dynamischen Benennung können Sie Traces basierend auf dem Domänennamen in der Anforderung gruppieren und einen Standardnamen anwenden, wenn der Name nicht mit einem erwarteten Muster übereinstimmt (z. B. wenn der Host-Header gefälscht ist).

> **Weitergeleitete Anfragen**
>
> Wenn ein Load Balancer oder ein anderer Vermittler eine Anfrage an Ihre Anwendung weiterleitet, übernimmt X-Ray die Client-IP von der`X-Forwarded-For`Header in der Anforderung statt von der Quell-IP im IP-Paket. Die Client-IP, die für eine weitergeleitete Anfrage aufgezeichnet wird, kann gefälscht werden und sollte daher nicht vertrauenswürdig sein.

Der Meldungshandler erzeugt für jede eingehende Anforderung ein Segment mit einem `http`-Block, der die folgenden Informationen enthält:

- HTTP method (HTTP-Methode)- HOLEN, POSTEN, PUTEN, LÖSCHEN usw.
- Client-Adresse— Die IP-Adresse des Clients, der die Anforderung gesendet hat.

- Antwortcode— Der HTTP-Antwortcode für die abgeschlossene Anforderung.
- Timing— Die Startzeit (zu der die Anforderung empfangen wurde) und die Endzeit (zu der die Antwort gesendet wurde).
- Benutzer-Agent— Die`user-agent`aus der Anfrage.
- Länge des Inhalts— Die`content-length`aus der Antwort.

Abschnitte



## Instrumentierung eingehender Anforderungen (.NET)

Um die von Ihrer Anwendung verarbeiteten Anforderungen zu instrumentieren, rufen Sie in der Methode `RegisterXRay` Ihrer `Init`-Datei die Methode `global.asax` auf.

Example global.asax – Meldungshandler

```
using System.Web.Http;
using Amazon.XRay.Recorder.Handlers.AspNet;

namespace SampleEBWebApplication
{
  public class MvcApplication : System.Web.HttpApplication
  {
    public override void Init()
    {
      base.Init();
      AWSXRayASPNET.RegisterXRay(this, "MyApp");
    }
  }
}
```

## Instrumentierung eingehender Anforderungen (.NET Core)

Um die von Ihrer Anwendung bearbeiteten Anforderungen zu instrumentieren, rufen Sie`UseXRay`-Methode vor jeder anderen Middleware im`Configure`Methode Ihrer Startup-Klasse als idealerweise

sollte X-Ray Middleware die erste Middleware sein, die die Anforderung und die letzte Middleware verarbeitet, um die Antwort in der Pipeline zu verarbeiten.

> **Note**
>
> Wenn Sie für .NET Core 2.0 einen`UseExceptionHandler`Methode in der Anwendung, stellen Sie sicher, dass Sie aufrufen`UseXRay`nach`UseExceptionHandler`Methode, um sicherzustellen, dass Ausnahmen aufgezeichnet werden.

Example Startup.cs

<caption>.NET Core 2.1 and above</caption>

```
using Microsoft.AspNetCore.Builder;

public void Configure(IApplicationBuilder app, IHostingEnvironment env)
  {
    app.UseXRay("MyApp");
    // additional middleware
    ...
  }
```

<caption>.NET Core 2.0</caption>

```
using Microsoft.AspNetCore.Builder;

public void Configure(IApplicationBuilder app, IHostingEnvironment env)
  {
    app.UseExceptionHandler("/Error");
    app.UseXRay("MyApp");
    // additional middleware
    ...
  }
```

Die `UseXRay`-Methode kann außerdem ein Konfigurationsobjekt als zweites Argument entgegennehmen.

```
app.UseXRay("MyApp", configuration);
```

### Konfiguration einer Segmentbenennungsstrategie

AWS X-Rayverwendet eineService-Nameum Ihre Anwendung zu identifizieren und von den anderen Anwendungen, Datenbanken, externen APIs zu unterscheiden undAWSRessourcen, die Ihre Anwendung verwendet. Wenn das X-Ray SDK Segmente für eingehende Anfragen generiert, zeichnet es den Dienstnamen Ihrer Anwendung im Segment aufNamen-Feldaus.

Das X-Ray SDK kann Segmente nach dem Hostnamen im HTTP-Anforderungsheader benennen. Dieser Header kann jedoch gefälscht werden, was zu unerwarteten Knoten in Ihrer Service-Map führen kann. Um zu verhindern, dass das SDK Segmente aufgrund von Anforderungen mit gefälschten Host-Headern falsch benennt, müssen Sie einen Standardnamen für eingehende Anfragen angeben.

Wenn Ihre Anwendung Anfragen für mehrere Domänen bereitstellt, können Sie das SDK so konfigurieren, dass es eine dynamische Benennungsstrategie verwendet, die dies in Segmentnamen widerspiegelt. Eine dynamische Benennungsstrategie ermöglicht es dem SDK, den Hostnamen für Anfragen zu verwenden, die einem erwarteten Muster entsprechen, und den Standardnamen auf Anfragen anzuwenden, die dies nicht tun.

Beispiel: Sie haben eine einzige Anwendung, die Anforderungen an drei Sub-Domains sendet —`www.example.com`,`api.example.com`, und`static.example.com`aus. Sie können eine dynamische Benennungsstrategie mit dem Muster verwenden`*.example.com`um Segmente für jede Subdomain mit einem anderen Namen zu identifizieren, was zu drei Serviceknoten auf der Service-Map führt. Wenn Ihre Anwendung Anfragen mit einem Hostnamen erhält, der nicht mit dem Muster übereinstimmt, wird auf der Service-Map ein vierter Knoten mit einem von Ihnen angegebenen Fallback-Namen angezeigt.

Wenn Sie denselben Namen für alle Segmente verwenden möchten, geben Sie bei der Initialisierung des Message-Handlers den Namen Ihrer Anwendung, wie im vorherigen Abschnitt gezeigt, ein. Dies hat den gleichen Effekt wie das Anlegen einer `FixedSegmentNamingStrategy` und das Übergeben an die `RegisterXRay`-Methode.

```
AWSXRayASPNET.RegisterXRay(this, new FixedSegmentNamingStrategy("MyApp"));
```

> **Note**
>
> Sie können den mit der `AWS_XRAY_TRACING_NAME`-Umgebungsvariablen in Code definierten standardmäßigen Dienstnamen überschreiben.

Eine dynamische Benennungsstrategie definiert ein Muster, dem Hostnamen entsprechen sollten, sowie einen Standardnamen, der verwendet wird, wenn der Hostname in der HTTP-Anforderung nicht mit diesem Muster übereinstimmt. Zur dynamischen Segmentbenennung erstellen Sie eine `DynamicSegmentNamingStrategy` und übergeben diese an die `RegisterXRay`-Methode.

```
AWSXRayASPNET.RegisterXRay(this, new DynamicSegmentNamingStrategy("MyApp",
 "*.example.com"));
```

## Nachverfolgen von AWS SDK-Aufrufen mit dem X-Ray SDK for .NET

Wenn Ihre Anwendung Aufrufe an tätigt, AWS-Services um Daten zu speichern, in eine Warteschlange zu schreiben oder Benachrichtigungen zu senden, verfolgt das X-Ray SDK for .NET die Aufrufe nachgelagert in Teilsegmenten . Nachverfolgte AWS-Services und Ressourcen, auf die Sie innerhalb dieser Services zugreifen (z. B. ein Amazon S3-Bucket oder eine Amazon SQS-Warteschlange), werden auf der Ablaufverfolgungskarte in der X-Ray-Konsole als nachgelagerte Knoten angezeigt.

Sie können alle Ihre AWS SDK for .NET Clients instrumentieren, indem Sie aufrufen, `RegisterXRayForAllServices` bevor Sie sie erstellen.

Example SampleController.cs – DynamoDB-Client-Instrumentierung

```
using Amazon;
using Amazon.Util;
using Amazon.DynamoDBv2;
using Amazon.DynamoDBv2.DocumentModel;
using Amazon.XRay.Recorder.Core;
using Amazon.XRay.Recorder.Handlers.AwsSdk;

namespace SampleEBWebApplication.Controllers
{
  public class SampleController : ApiController
  {
    AWSSDKHandler.RegisterXRayForAllServices();
    private static readonly Lazy<AmazonDynamoDBClient> LazyDdbClient = new
 Lazy<AmazonDynamoDBClient>(() =>
    {
      var client = new AmazonDynamoDBClient(EC2InstanceMetadata.Region ??
 RegionEndpoint.USEast1);
      return client;
    });
```

Um Clients nur für einige Services zu instrumentieren, rufen Sie `RegisterXRay` statt `RegisterXRayForAllServices` auf. Ersetzen Sie den markierten Text durch den Namen der Client-Schnittstelle des Services.

```
AWSSDKHandler.RegisterXRay<IAmazonDynamoDB>()
```

Für alle Services können Sie den Namen der API mit dem Namen in der X-Ray-Konsole sehen. Für eine Teilmenge von Services fügt das X-Ray-SDK dem Segment Informationen hinzu, um eine höhere Granularität in der Service-Übersicht zu gewährleisten.

Wenn Sie beispielsweise einen Aufruf mit einem instrumentierten DynamoDB-Client tätigen, fügt das SDK dem Segment den Tabellennamen für Aufrufe hinzu, die auf eine Tabelle abzielen. In der Konsole wird jede Tabelle als separater Knoten in der Service-Übersicht angezeigt, mit einem generischen DynamoDB-Knoten für Aufrufe, die keine Tabelle anvisieren.

Example Untersegment für einen Aufruf an DynamoDB zum Speichern eines Elements

```
{
  "id": "24756640c0d0978a",
  "start_time": 1.480305974194E9,
  "end_time": 1.4803059742E9,
  "name": "DynamoDB",
  "namespace": "aws",
  "http": {
    "response": {
      "content_length": 60,
      "status": 200
    }
  },
  "aws": {
    "table_name": "scorekeep-user",
    "operation": "UpdateItem",
    "request_id": "UBQNSO5AEM8T4FDA4RQDEB94OVTDRVV4K4HIRGVJF66Q9ASUAAJG",
  }
}
```

Wenn Sie auf benannte Ressourcen zugreifen, werden durch Aufrufe der folgenden Services weitere Knoten in der Service-Übersicht erstellt. Durch Aufrufe, die keinen bestimmten Ressourcen gelten, wird ein generischer Knoten für den Service erstellt.

- Amazon DynamoDB – Tabellenname

- Amazon Simple Storage Service – Bucket- und Schlüsselname
- Amazon Simple Queue Service – Name der Warteschlange

## Nachverfolgen von Aufrufen nachgelagerter HTTP-Web-Services mit dem X-Ray SDK for .NET

Wenn Ihre Anwendung Microservices oder öffentliche HTTP-APIs aufruft, können Sie das X-Ray SDK für .NET`GetResponseTraced`Erweiterungsversionen für`System.Net.HttpWebRequest`diese Aufrufe instrumentieren und die API der Service-Grafik als nachgelagerten Service hinzufügen.

Example HttpWebRequest

```
using System.Net;
using Amazon.XRay.Recorder.Core;
using Amazon.XRay.Recorder.Handlers.System.Net;

private void MakeHttpRequest()
{
  HttpWebRequest request = (HttpWebRequest)WebRequest.Create("http://names.example.com/
api");
  request.GetResponseTraced();
}
```

Für asynchrone Aufrufe verwenden Sie `GetAsyncResponseTraced`.

```
request.GetAsyncResponseTraced();
```

Zeichnen Sie bei der Verwendung von [system.net.http.httpclient](#) Aufrufe mit dem delegierenden `HttpClientXRayTracingHandler`-Handler auf.

Example HttpClient

```
using System.Net.Http;
using Amazon.XRay.Recorder.Core;
using Amazon.XRay.Recorder.Handlers.System.Net;

private void MakeHttpRequest()
{
  var httpClient = new HttpClient(new HttpClientXRayTracingHandler(new
 HttpClientHandler()));
```

```
  httpClient.GetAsync(URL);
}
```

Wenn Sie einen Aufruf einer nachgelagerten Web-API instrumentieren, erfasst das X-Ray SDK for .NET ein Untersegment mit Informationen über die HTTP-Anforderung und Antwort. X-Ray verwendet das Untersegment, um ein abgeleitetes Segment für die API zu generieren.

Example Untersegment für einen nachgelagerten HTTP-Aufruf

```
{
  "id": "004f72be19cddc2a",
  "start_time": 1484786387.131,
  "end_time": 1484786387.501,
  "name": "names.example.com",
  "namespace": "remote",
  "http": {
    "request": {
      "method": "GET",
      "url": "https://names.example.com/"
    },
    "response": {
      "content_length": -1,
      "status": 200
    }
  }
}
```

Example Abgeleitetes Segment für einen nachgelagerten HTTP-Anruf

```
{
  "id": "168416dc2ea97781",
  "name": "names.example.com",
  "trace_id": "1-62be1272-1b71c4274f39f122afa64eab",
  "start_time": 1484786387.131,
  "end_time": 1484786387.501,
  "parent_id": "004f72be19cddc2a",
  "http": {
    "request": {
      "method": "GET",
      "url": "https://names.example.com/"
    },
    "response": {
      "content_length": -1,
```

```
      "status": 200
    }
  },
  "inferred": true
}
```

## Verfolgung von SQL-Abfragen mit dem X-Ray SDK for .NET

Das X-Ray SDK for .NET bietet eine Wrapper-Klasse für`System.Data.SqlClient.SqlCommand`, benannt`TraceableSqlCommand`, die Sie anstelle von verwenden können`SqlCommand`aus. Sie können einen SQL-Befehl mit der Klasse `TraceableSqlCommand` initialisieren.

Ablaufverfolgung von SQL-Abfragen mit synchronen und asynchronen Methoden

Die folgenden Beispiele zeigen, wie Sie SQL Server-Abfragen mit `TraceableSqlCommand` automatisch synchron und asynchron verfolgen.

Example **`Controller.cs`** – SQL-Client-Instrumentierung (synchron)

```
using Amazon;
using Amazon.Util;
using Amazon.XRay.Recorder.Core;
using Amazon.XRay.Recorder.Handlers.SqlServer;

private void QuerySql(int id)
{
  var connectionString = ConfigurationManager.AppSettings["RDS_CONNECTION_STRING"];
  using (var sqlConnection = new SqlConnection(connectionString))
  using (var sqlCommand = new TraceableSqlCommand("SELECT " + id, sqlConnection))
  {
    sqlCommand.Connection.Open();
    sqlCommand.ExecuteNonQuery();
  }
}
```

Sie können die Abfrage mithilfe der `ExecuteReaderAsync`-Methode asynchron ausführen.

Example **`Controller.cs`** – SQL-Client-Instrumentierung (asynchron)

```
using Amazon;
using Amazon.Util;
using Amazon.XRay.Recorder.Core;
using Amazon.XRay.Recorder.Handlers.SqlServer;
```

```
private void QuerySql(int id)
{
  var connectionString = ConfigurationManager.AppSettings["RDS_CONNECTION_STRING"];
  using (var sqlConnection = new SqlConnection(connectionString))
  using (var sqlCommand = new TraceableSqlCommand("SELECT " + id, sqlConnection))
  {
    await sqlCommand.ExecuteReaderAsync();
  }
}
```

Erfassen von SQL-Abfragen an SQL Server

Sie können die Erfassung von `SqlCommand.CommandText` als Teil des Untersegments aktivieren, das von Ihrer SQL-Abfrage erstellt wurde. `SqlCommand.CommandText` wird als Feld `sanitized_query` im JSON-Untersegment angezeigt. Standardmäßig ist diese Funktion aus Sicherheitsgründen deaktiviert.

> **Note**
>
> Aktivieren Sie die Sammlungsfunktion nicht, wenn Sie sensible Informationen als Klartext in Ihre SQL-Abfragen einfügen.

Sie können die Sammlung von SQL-Abfragen auf zwei Arten aktivieren:

- Legen Sie die Eigenschaft `CollectSqlQueries` in der globalen Konfiguration für Ihre Anwendung auf `true` fest.
- Legen Sie den Parameter `collectSqlQueries` in der Instance `TraceableSqlCommand` auf `true` fest, um Aufrufe innerhalb der Instance zu erfassen.

Aktivieren der globalen CollectSqlQueries-Eigenschaft

Die folgenden Beispiele zeigen, wie Sie die Eigenschaft `CollectSqlQueries` für .NET und .NET Core aktivieren.

.NET

Um die Eigenschaft `CollectSqlQueries` in der globalen Konfiguration Ihrer Anwendung in .NET auf `true` festzulegen, ändern Sie die `appsettings` Ihrer `App.config`- oder `Web.config`-Datei wie dargestellt.

Example **`App.config`**Or**`Web.config`**— Globales Aktivieren der SQL-Abfragesammlung

```
<configuration>
<appSettings>
    <add key="CollectSqlQueries" value="true">
</appSettings>
</configuration>
```

.NET Core

So stellen Sie das`CollectSqlQueries`immobilien zu`true`ändern Sie in der globalen Konfiguration Ihrer Anwendung in .NET Core`appsettings.json`-Datei unter der X-Ray-Taste, wie gezeigt.

Example **`appsettings.json`**— Globales Aktivieren der SQL-Abfragesammlung

```
{
  "XRay": {
    "CollectSqlQueries":"true"
  }
}
```

Aktivieren des collectSqlQueries-Parameters

Sie können den Parameter `collectSqlQueries` in der Instance `TraceableSqlCommand` auf `true` setzen, um den SQL-Abfragetext für SQL Server-Abfragen zu erfassen, die mit dieser Instance durchgeführt wurden. Wenn Sie den Parameter auf `false` setzen, wird die Funktion `CollectSqlQuery` für die Instance `TraceableSqlCommand` deaktiviert.

> **Note**
>
> Der Wert von `collectSqlQueries` in der Instance `TraceableSqlCommand` überschreibt den in der globalen Konfiguration der Eigenschaft `CollectSqlQueries` festgelegten Wert.

Example Beispiel**`Controller.cs`**— Aktivieren der SQL-Abfragesplung für die Instance

```
using Amazon;
```

```
using Amazon.Util;
using Amazon.XRay.Recorder.Core;
using Amazon.XRay.Recorder.Handlers.SqlServer;

private void QuerySql(int id)
{
  var connectionString = ConfigurationManager.AppSettings["RDS_CONNECTION_STRING"];
  using (var sqlConnection = new SqlConnection(connectionString))
  using (var command = new TraceableSqlCommand("SELECT " + id, sqlConnection,
 collectSqlQueries: true))
  {
    command.ExecuteNonQuery();
  }
}
```

## Erstellen zusätzlicher Untersegmente

Teilsegmente erweitern ein TraceAbschnittmit Details über die geleistete Arbeit, um eine Anfrage zu stellen. Jedes Mal, wenn Sie einen Aufruf mit einem instrumentierten Client erstellen, erfasst das X-Ray SDK die in einem Untersegment generierten Informationen. Sie können zusätzliche Teilsegmente erstellen, um andere Teilsegmente zu gruppieren, die Leistung eines Codeabschnitts zu messen oder Anmerkungen und Metadaten aufzuzeichnen.

Um Untersegmente zu verwalten, verwenden Sie die Methoden `BeginSubsegment` und `EndSubsegment`. Führen Sie die gewünschte Bearbeitung im Untersegment in einem `try`-Block aus und verwenden Sie `AddException`, um Ausnahmen zu verfolgen. Rufen Sie `EndSubsegment` in einem `finally`-Block auf, um sicherzustellen, dass das Untersegment geschlossen ist.

Example Controller.cs — benutzerdefiniertes Untersegment

```
AWSXRayRecorder.Instance.BeginSubsegment("custom method");
try
{
  DoWork();
}
catch (Exception e)
{
  AWSXRayRecorder.Instance.AddException(e);
}
finally
{
  AWSXRayRecorder.Instance.EndSubsegment();
```

```
}
```

Beim Erstellen eines Untersegments innerhalb eines Segments oder eines anderen Untersegments generiert das X-Ray SDK for .NET eine ID dafür und erfasst die Start- und Endzeit.

Example Untersegment mit Metadaten

```
"subsegments": [{
  "id": "6f1605cd8a07cb70",
  "start_time": 1.480305974194E9,
  "end_time": 1.4803059742E9,
  "name": "Custom subsegment for UserModel.saveUser function",
  "metadata": {
    "debug": {
      "test": "Metadata string from UserModel.saveUser"
    }
  },
```

## Hinzufügen von Anmerkungen und Metadaten zu Segmenten mit dem X-Ray SDK for .NET

Sie können Anmerkungen und Metadaten verwenden, um zusätzliche Informationen über Anfragen, die Umgebung oder Ihre Anwendung aufzuzeichnen. Sie können Anmerkungen und Metadaten zu den Segmenten hinzufügen, die das X-Ray SDK erstellt, oder zu benutzerdefinierten Untersegmenten, die Sie erstellen.

Anmerkungen sind Schlüssel-Wert-Paare mit Zeichenfolgen-, Zahlen- oder booleschen Werten. Anmerkungen sind für die Verwendung mit Filterausdrücken indexiert. Berücksichtigen Sie Anmerkungen, um Daten zur Gruppierung von Ablaufverfolgungen in der Konsole zu verwenden, oder wenn Sie die `GetTraceSummaries`-API aufrufen.

Metadaten sind Schlüssel-Wert-Paare, die Werte beliebigen Typs enthalten können, einschließlich Objekte und Listen, aber nicht für die Verwendung mit Filterausdrücken indexiert sind. Verwenden Sie Metadaten, um zusätzliche Daten aufzuzeichnen, die Sie im Trace speichern möchten, aber nicht für die Suche verwenden müssen.

Sections

- Anmerkungen mit dem X-Ray SDK for .NET aufnehmen
- Metadaten mit dem X-Ray SDK for .NET aufzeichnen

### Anmerkungen mit dem X-Ray SDK for .NET aufnehmen

Verwenden Sie Anmerkungen, um Informationen zu Segmenten oder Untersegmenten, die zur Suche indiziert werden sollten, aufzuzeichnen.

Folgendes ist für alle Anmerkungen in X-Ray erforderlich:

Anmerkung zu Anforderungen

- Schlüssel — Der Schlüssel für eine X-Ray-Anmerkung kann bis zu 500 alphanumerische Zeichen enthalten. Sie können keine anderen Leerzeichen oder Symbole als den Unterstrich (_) verwenden.
- Werte — Der Wert für eine X-Ray-Anmerkung kann bis zu 1.000 Unicode-Zeichen enthalten.
- Die Anzahl der Anmerkungen — Sie können bis zu 50 Anmerkungen pro Spur verwenden.

Um Anmerkungen außerhalb einer Funktion aufzuzeichnen AWS Lambda

1. Rufen Sie eine Instance von `AWSXRayRecorder` ab.

   ```
   using Amazon.XRay.Recorder.Core;
   ...
   AWSXRayRecorder recorder = AWSXRayRecorder.Instance;
   ```

2. Rufen Sie `addAnnotation` mit einem Zeichenfolgenschlüssel und einem booleschen, Int32-, Int64-, Double- oder Zeichenfolgenwert auf.

   ```
   recorder.AddAnnotation("mykey", "my value");
   ```

Um Anmerkungen innerhalb einer Funktion aufzuzeichnen AWS Lambda

Sowohl Segmente als auch Untersegmente innerhalb einer Lambda-Funktion werden von der Lambda-Laufzeitumgebung verwaltet. Wenn Sie einem Segment oder Untersegment innerhalb einer Lambda-Funktion eine Anmerkung hinzufügen möchten, müssen Sie wie folgt vorgehen:

1. Erstellen Sie das Segment oder Untersegment in der Lambda-Funktion.
2. Fügen Sie die Anmerkung dem Segment oder Untersegment hinzu.
3. Beenden Sie das Segment oder Untersegment.

Das folgende Codebeispiel zeigt Ihnen, wie Sie einem Untersegment innerhalb einer Lambda-Funktion eine Anmerkung hinzufügen:

```
#Create the subsegment
AWSXRayRecorder.Instance.BeginSubsegment("custom method");
#Add an annotation
AWSXRayRecorder.Instance.AddAnnotation("My", "Annotation");
try
{
  YourProcess(); #Your function
}
catch (Exception e)
{
  AWSXRayRecorder.Instance.AddException(e);
}
finally #End the subsegment
{
  AWSXRayRecorder.Instance.EndSubsegment();
}
```

Das X-Ray SDK zeichnet Anmerkungen als Schlüssel-Wert-Paare in einem `annotations` Objekt im Segmentdokument auf. Durch `addAnnotation` zweimaliges Aufrufen der Operation mit demselben Schlüssel wird ein zuvor aufgezeichneter Wert für dasselbe Segment oder Untersegment überschrieben.

Um Spuren zu finden, die Anmerkungen mit bestimmten Werten enthalten, verwenden Sie das `annotations.key` Schlüsselwort in einem Filterausdruck. Weitere Informationen finden Sie unter Verwenden Sie Filterausdrücke.

Metadaten mit dem X-Ray SDK for .NET aufzeichnen

Verwenden Sie Metadaten, um Informationen zu Segmenten oder Untersegmenten aufzuzeichnen, die Sie für die Verwendung in einer Suche nicht indizieren müssen. Bei Metadatenwerten kann es sich um Zeichenketten, Zahlen, Boolesche Werte oder jedes andere Objekt handeln, das in ein JSON-Objekt oder -Array serialisiert werden kann.

So zeichnen Sie Metadaten auf

1. Rufen Sie eine Instanz von ab`AWSXRayRecorder`, wie im folgenden Codebeispiel gezeigt:

   ```
   using Amazon.XRay.Recorder.Core;
   ...
   ```

```
AWSXRayRecorder recorder = AWSXRayRecorder.Instance;
```

2. Rufen Sie `AddMetadata` mit einem Zeichenfolgen-Namespace, einem Zeichenkettenschlüssel und einem Objektwert auf, wie im folgenden Codebeispiel gezeigt:

   ```
   recorder.AddMetadata("my namespace", "my key", "my value");
   ```

   Sie können den `AddMetadata` Vorgang auch nur mit einem Schlüssel- und Wertepaar aufrufen, wie im folgenden Codebeispiel gezeigt:

   ```
   recorder.AddMetadata("my key", "my value");
   ```

Wenn Sie keinen Wert für den Namespace angeben, verwendet `default` das X-Ray SDK.
Wenn Sie den `AddMetadata` Vorgang zweimal mit demselben Schlüssel aufrufen, wird ein zuvor aufgezeichneter Wert für dasselbe Segment oder Untersegment überschrieben.

# Instrumentieren Sie Ihre Anwendung mit Ruby

Es gibt zwei Möglichkeiten, Ihre Ruby-Anwendung so zu instrumentieren, dass sie Traces an X-Ray sendet:

- AWS Distro for OpenTelemetry Ruby — Eine AWS Distribution, die eine Reihe von Open-Source-Bibliotheken zum Senden korrelierter Metriken und Traces über AWS Distro for Collector an mehrere AWS Überwachungslösungen CloudWatch AWS X-Ray, darunter Amazon und Amazon OpenSearch Service, bereitstellt. OpenTelemetry
- AWS X-Ray SDK for Ruby — Eine Reihe von Bibliotheken zum Generieren und Senden von Traces an X-Ray über den X-Ray-Daemon.

Weitere Informationen finden Sie unter Wahl zwischen den AWS SDKs Distro for OpenTelemetry und X-Ray.

## AWSDistro fürOpenTelemetryRuby

Mit demAWSDistro fürOpenTelemetry(ADOT) Ruby, Sie können Ihre Anwendungen einmal instrumentieren und dann korrelierte Metriken und Traces an mehrere sendenAWSMonitoring-Lösungen, einschließlich AmazonCloudWatch,AWS X-Ray, und AmazonOpenSearchBedienung. Die Verwendung von X-Ray mit ADOT erfordert zwei Komponenten:OpenTelemetrySDKaktiviert für die

Verwendung mit X-Ray, undAWSDistribution fürOpenTelemetrySammlerfür die Verwendung mit X-Ray aktiviert.

Informationen zu den ersten Schritten finden Sie imAWSDistribution fürOpenTelemetryRuby-Dokumentation.

Für weitere Informationen zur Verwendung desAWSDistribution fürOpenTelemetrymitAWS X-Rayund andereAWS-Services, sieheAWSDistribution fürOpenTelemetryoder dieAWSDistribution fürOpenTelemetryDokumentation.

Weitere Informationen zur Sprachunterstützung und Sprachverwendung finden Sie unterAWSBeobachtbarkeit aufGitHub.

## AWS X-Ray SDK für Ruby

Das X-Ray SDK ist eine Bibliothek für Ruby-Webanwendungen, die Klassen und Methoden zum Generieren und Senden von Ablaufverfolgungsdaten an den X-Ray-Daemon bereitstellt. Trace-Daten enthalten Informationen über eingehende HTTP-Anfragen, die von der Anwendung bedient werden, und Aufrufe, die die Anwendung an nachgelagerte Services über das AWS SDK, HTTP-Clients oder einen aktiven Datensatzclient durchführt. Sie können Segmente auch manuell erstellen und Debug-Informationen Anmerkungen und Metadaten hinzufügen.

Sie können das SDK herunterladen, indem Sie es Ihrer Gemfile-Datei hinzufügen und `bundle install` ausführen.

Example Gemfile

```
gem 'aws-sdk'
```

Wenn Sie Rails verwenden, fügen Sie zunächst die X-Ray-SDK-Middleware hinzu, um eingehende Anfragen nachzuverfolgen. Ein Anforderungsfilter erstellt ein Segment. Während das Segment geöffnet ist, können Sie die SDK-Client-Methoden nutzen, um dem Segment Informationen hinzuzufügen, Untersegmente zu erstellen und nachgelagerte Aufrufe rückzuverfolgen. Das SDK erfasst auch automatisch Ausnahmen, die Ihre Anwendung ausgibt, während das Segment geöffnet ist. Bei anderen Anwendungen als Rails können Sie Segmente manuell erstellen.

Verwenden Sie als Nächstes das X-Ray-SDK, um Ihre AWS SDK for Ruby-, HTTP- und SQL-Clients zu instrumentieren, indem Sie den Recorder so konfigurieren, dass er die zugehörigen Bibliotheken patcht. Jedes Mal, wenn Sie einen Aufruf an eine Downstream- AWS-Service oder -Ressource mit einem instrumentierten Client tätigen, zeichnet das SDK Informationen über den Aufruf in einem

Untersegment auf. AWS-Services und die Ressourcen, auf die Sie innerhalb der Services zugreifen, werden als Downstream-Knoten auf der Ablaufverfolgungskarte angezeigt, um Ihnen zu helfen, Fehler und Drosselungsprobleme bei einzelnen Verbindungen zu identifizieren.

Sobald Sie die ersten Schritte mit dem SDK gemacht haben, können Sie das Verhalten durch die Konfiguration des Recorders individualisieren. Sie können Plugins zum Festhalten von Daten über die Datenverarbeitungsressourcen, auf denen Ihre Anwendung ausgeführt wird, hinzufügen, das Samplingverhalten durch Samplingregeln anpassen und einen Logger bereitstellen, um mehr oder weniger Informationen von dem SDK in Ihren Anwendungsprotokollen zu sehen.

Zeichnen Sie zusätzliche Informationen zu Anforderungen und den Aufgaben, die Ihre Anwendung ausführt, in Anmerkungen und Metadaten auf. Anmerkungen sind einfache Schlüsselwertpaare, die für die Verwendung mit Filterausdrücken indiziert werden, damit Sie nach Ablaufverfolgen mit bestimmten Daten suchen können. Metadateneinträge sind weniger einschränkend und können ganze Objekte und Arrays aufzeichnen – alle Daten, die in eine JSON zusammengefasst werden können.

> **Anmerkungen und Metadaten**
>
> Anmerkungen und Metadaten sind beliebiger Text, den Sie Segmenten mit dem X-Ray-SDK hinzufügen. Anmerkungen werden für die Verwendung mit Filterausdrücken indiziert. Metadaten werden nicht indiziert, können aber im Rohsegment mit der X-Ray-Konsole oder API angezeigt werden. Jeder, dem Sie Lesezugriff auf X-Ray gewähren, kann diese Daten anzeigen.

Wenn Sie viele instrumentierten Clients in Ihrem Code haben, kann ein einzelnes Anforderungssegmente viele Untersegmente enthalten, eines für jeden Aufruf mit einem instrumentierten Client. Sie können Untersegmente organisieren und gruppieren, indem Sie Client-Aufrufe in benutzerdefinierten Untersegmenten zusammenfassen. Sie können ein benutzerdefiniertes Untersegment für eine ganze Funktion oder eine Code-Abschnitt erstellen und Metadaten und Anmerkungen im Untersegment festhalten, anstatt alles im übergeordneten Segment aufzuzeichnen.

Referenzdokumentation zu den Klassen und Methoden des SDK finden Sie in der AWS X-Ray API-Referenz zum SDK für Ruby.

## Voraussetzungen

Das X-Ray-SDK erfordert Ruby 2.3 oder höher und ist mit den folgenden Bibliotheken kompatibel:

- AWS SDK for Ruby Version 3.0 oder höher
- Rails Version 5.1 oder höher

## Konfiguration des X-Ray-SDK SDK for Ruby

Das X-Ray-SDK SDK for Ruby hat eine Klasse namens`XRay.recorder`, die den globalen Rekorder bereitstellt. Sie können die globale Aufzeichnung so konfigurieren, dass die Middleware, die Segmente für eingehende HTTP-Aufrufe erstellt, angepasst wird.

Sections

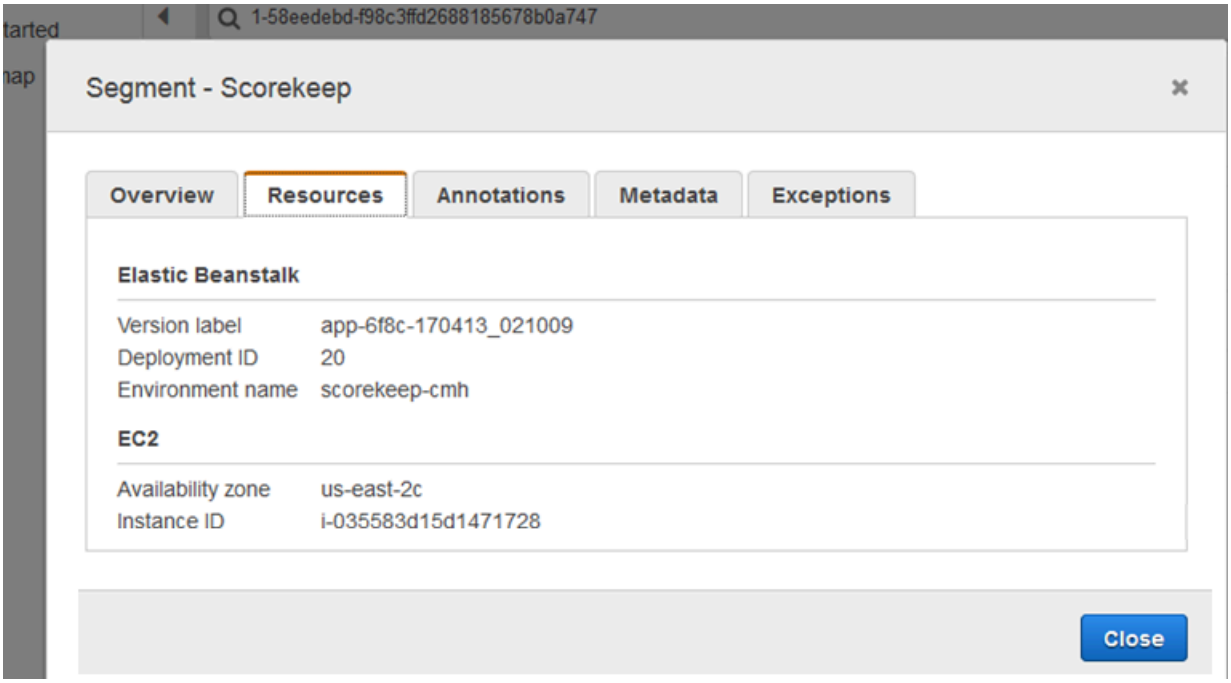
- Service-Plugins
- Samplingregeln
- Protokollierung
- Konfiguration des Recorders im Code
- Konfigurieren des Recorders mit Rails
- Umgebungsvariablen

Service-Plugins

Wird verwendet`plugins`, um Informationen über den Dienst aufzuzeichnen, der Ihre Anwendung hostet.

Plug-ins

- Amazon EC2 — `ec2` fügt die Instance-ID und die Availability Zone hinzu.
- Elastic Beanstalk — `elastic_beanstalk` fügt den Umgebungsnamen, die Versionsbezeichnung und die Bereitstellungs-ID hinzu.
- Amazon ECS — `ecs` fügt die Container-ID hinzu.




Um Plugins verwenden zu können, geben Sie sie in dem Konfigurationsobjekt an, das sie dem Recorder übergeben.

Example main.rb — Plugin-Konfiguration

```
my_plugins = %I[ec2 elastic_beanstalk]

config = {
  plugins: my_plugins,
  name: 'my app',
}

XRay.recorder.configure(config)
```

Sie können auch Umgebungsvariablen, die Vorrang über Werte im Code haben, zur Konfiguration des Recorders verwenden.

Das SDK verwendet auch Plugin-Einstellungen, um das `origin` Feld im Segment festzulegen. Dies gibt den AWS Ressourcentyp an, auf dem Ihre Anwendung ausgeführt wird. Wenn Sie mehrere Plugins verwenden, verwendet das SDK die folgende Auflösungsreihenfolge, um den Ursprung zu bestimmen: ElasticBeanstalk > EKS > ECS > EC2.

### Samplingregeln

Das SDK verwendet die Sampling-Regeln, die Sie in der X-Ray-Konsole definieren, um zu bestimmen, welche Anfragen aufgezeichnet werden sollen. Die Standardregel verfolgt die erste Anfrage jede Sekunde und fünf Prozent aller weiteren Anfragen aller Dienste, die Traces an X-Ray senden. Erstellen Sie zusätzliche Regeln in der X-Ray-Konsole, um die Menge der aufgezeichneten Daten für jede Ihrer Anwendungen anzupassen.

Das SDK wendet benutzerdefinierte Regeln in der Reihenfolge an, in der sie definiert sind. Wenn eine Anfrage mehreren benutzerdefinierten Regeln entspricht, wendet das SDK nur die erste Regel an.

> **Note**
>
> Wenn das SDK X-Ray nicht erreichen kann, um Sampling-Regeln abzurufen, kehrt es zu einer lokalen Standardregel zurück, die die erste zu Beginn jeder Sekunde empfangene Anfrage und fünf Prozent aller zusätzlichen Anfragen pro Host festlegt. Dies kann passieren, wenn der Host nicht berechtigt ist, Sampling-APIs aufzurufen, oder wenn er keine Verbindung zum X-Ray-Daemon herstellen kann, der als TCP-Proxy für API-Aufrufe durch das SDK fungiert.

Sie können das SDK auch so konfigurieren, dass Sampling-Regeln aus einem JSON-Dokument geladen werden. Das SDK kann lokale Regeln als Backup für Fälle verwenden, in denen X-Ray Sampling nicht verfügbar ist, oder ausschließlich lokale Regeln verwenden.

Example sampling-rules.json

```
{
  "version": 2,
  "rules": [
    {
      "description": "Player moves.",
      "host": "*",
      "http_method": "*",
      "url_path": "/api/move/*",
      "fixed_target": 0,
      "rate": 0.05
    }
  ],
  "default": {
    "fixed_target": 1,
```

```
    "rate": 0.1
  }
}
```

In diesem Beispiel werden eine benutzerdefinierte Regel und eine Standardregel definiert. Die benutzerdefinierte Regel wendet eine Stichprobenrate von fünf Prozent an, ohne dass eine Mindestanzahl von Anfragen für Pfade verfolgt werden muss, unter `/api/move/` denen Pfade verfolgt werden müssen. Die Standardregel verfolgt die erste Anfrage jede Sekunde und 10 Prozent der weiteren Anfragen.

Der Nachteil der lokalen Definition von Regeln besteht darin, dass das feste Ziel von jeder Instanz des Rekorders unabhängig angewendet wird, anstatt vom X-Ray-Dienst verwaltet zu werden. Wenn Sie mehr Hosts bereitstellen, wird die feste Rate vervielfacht, wodurch es schwieriger wird, die Menge der aufgezeichneten Daten zu kontrollieren.

Um Sicherungsregeln zu konfigurieren, definieren Sie einen Hash für das Dokument im Konfigurationsobjekt, das Sie dem Recorder übergeben.

Example main.rb — Konfiguration der Backup-Regeln

```
require 'aws-xray-sdk'
my_sampling_rules =  {
  version: 1,
  default: {
    fixed_target: 1,
    rate: 0.1
  }
}
config = {
  sampling_rules: my_sampling_rules,
  name: 'my app',
}
XRay.recorder.configure(config)
```

Speichern Sie die Samplingregeln unabhängig voneinander, definieren Sie den Hash in einer separaten Datei und fordern Sie die Datei an, damit er in Ihre Anwendung übernommen wird.

Example config/sampling-rules.rb

```
my_sampling_rules =  {
  version: 1,
```

```
  default: {
    fixed_target: 1,
    rate: 0.1
  }
}
```

Example main.rb — Sampling-Regel aus einer Datei

```
require 'aws-xray-sdk'
require 'config/sampling-rules.rb'

config = {
  sampling_rules: my_sampling_rules,
  name: 'my app',
}
XRay.recorder.configure(config)
```

Um nur lokale Regeln zu verwenden, fordern Sie die Samplingregeln an und konfigurieren den `LocalSampler`.

Example main.rb — Sampling mit lokalen Regeln

```
require 'aws-xray-sdk'
require 'aws-xray-sdk/sampling/local/sampler'

config = {
  sampler: LocalSampler.new,
  name: 'my app',
}
XRay.recorder.configure(config)
```

Sie können auch die globale Aufzeichnung so konfigurieren, dass das Sampling deaktiviert und alle eingehenden Anfragen instrumentiert werden.

Example main.rb — Deaktiviert das Sampling

```
require 'aws-xray-sdk'
config = {
  sampling: false,
  name: 'my app',
}
XRay.recorder.configure(config)
```

Protokollierung

Standardmäßig gibt der Recorder Ereignisse auf Informationsebene in `$stdout` aus. Sie können die Protokollierung anpassen, indem Sie einen Logger in dem Konfigurationsobjekt definieren, das Sie dem Recorder übergeben.

Example main.rb — Protokollierung

```
require 'aws-xray-sdk'
config = {
  logger: my_logger,
  name: 'my app',
}
XRay.recorder.configure(config)
```

Verwenden Sie Debug-Protokolle, um Probleme wie nicht geschlossene Untersegmente zu identifizieren, wenn Sie Untersegmente manuell generieren.

Konfiguration des Recorders im Code

Zusätzliche Einstellungen finden Sie in der `configure`-Methode auf `XRay.recorder`.

- `context_missing`— Auf einstellen, um `LOG_ERROR` zu verhindern, dass Ausnahmen ausgelöst werden, wenn Ihr instrumentierter Code versucht, Daten aufzuzeichnen, wenn kein Segment geöffnet ist.
- `daemon_address`— Legt den Host und den Port des X-Ray-Daemon-Listeners fest.
- `name`— Legen Sie einen Dienstnamen fest, den das SDK für Segmente verwendet.
- `naming_pattern`— Legen Sie ein Domainnamenmuster fest, um dynamische Benennung zu verwenden.
- `plugins`— Erfassen Sie Informationen über die AWS Ressourcen Ihrer Anwendung mit Plugins.
- `sampling`— Auf einstellen, `false` um das Sampling zu deaktivieren.
- `sampling_rules`— Legen Sie den Hash fest, der Ihre Sampling-Regeln enthält.

Example main.rb — Deaktiviert im Kontext fehlende Ausnahmen

```
require 'aws-xray-sdk'
config = {
  context_missing: 'LOG_ERROR'
```

```
}

XRay.recorder.configure(config)
```

Konfigurieren des Recorders mit Rails

Wenn Sie das Rails-Framework verwenden, können Sie die Optionen für den globalen Recorder in einer Ruby-Datei unter `app_root/initializers` konfigurieren. Das X-Ray SDK unterstützt einen zusätzlichen Konfigurationsschlüssel für die Verwendung mit Rails.

- `active_record`— Auf einstellen, `true` um Untersegmente für Active Record-Datenbanktransaktionen aufzuzeichnen.

Konfigurieren Sie die verfügbaren Einstellungen in einem Konfigurationsobjekt mit dem Namen `Rails.application.config.xray`.

Example config/initializers/aws_xray.rb

```
Rails.application.config.xray = {
  name: 'my app',
  patch: %I[net_http aws_sdk],
  active_record: true
}
```

Umgebungsvariablen

Sie können Umgebungsvariablen verwenden, um das X-Ray SDK for Ruby zu konfigurieren. Das SDK unterstützt die folgenden Variablen:

- `AWS_XRAY_TRACING_NAME`— Legen Sie einen Dienstnamen fest, den das SDK für Segmente verwendet. Überschreibt den für die Segmentbenennungsstrategie des Servlet-Filters festgelegten Dienstnamen.
- `AWS_XRAY_DAEMON_ADDRESS`— Legt den Host und den Port des X-Ray-Daemon-Listeners fest. Standardmäßig sendet das SDK Trace-Daten an. `127.0.0.1:2000` Verwenden Sie diese Variable, wenn Sie den Daemon so konfiguriert haben, dass er auf einem anderen Port lauscht, oder wenn er auf einem anderen Host läuft.
- `AWS_XRAY_CONTEXT_MISSING`— Legt fest, `RUNTIME_ERROR` dass Ausnahmen ausgelöst werden, wenn Ihr instrumentierter Code versucht, Daten aufzuzeichnen, obwohl kein Segment geöffnet ist.

Zulässige Werte

- `RUNTIME_ERROR`— Löst eine Laufzeitausnahme aus.
- `LOG_ERROR`— Einen Fehler protokollieren und fortfahren (Standard).
- `IGNORE_ERROR`— Fehler ignorieren und fortfahren.

Fehler im Zusammenhang mit fehlenden Segmenten oder Untersegmenten können auftreten, wenn Sie versuchen, einen instrumentierten Client in Startcode zu verwenden, der ausgeführt wird, wenn keine Anfrage geöffnet ist, oder in Code, der einen neuen Thread erzeugt.

Umgebungsvariablen überschreiben Werte im Code.

## Nachverfolgen eingehender Anfragen mit der Middleware des X-Ray-SDK für Ruby

Sie können das X-Ray-SDK verwenden, um eingehende HTTP-Anforderungen zu verfolgen, die Ihre Anwendung auf einer EC2-Instance in Amazon EC2 AWS Elastic Beanstalk oder Amazon ECS bedient.

Wenn Sie Rails einsetzen, verwenden Sie die Rails-Middleware, um eingehenden HTTP-Anforderungen zu instrumentieren. Wenn Sie die Middleware zu Ihrer Anwendung hinzufügen und einen Segmentnamen konfigurieren, erstellt das X-Ray SDK for Ruby ein Segment für jede Stichprobenanforderung. Alle durch eine zusätzliche Instrumentierung erstellten Segmente werden zu Untersegmenten des Segments auf Anfrageebene, das Informationen über die HTTP-Anforderung und Antwort bereitstellt. Diese Informationen umfassen Dauer, Methode und Status der Anfrage.

Jedes Segment hat einen Namen, der Ihre Anwendung in der Service-Übersicht identifiziert. Das Segment kann statisch benannt werden, oder Sie können das SDK so konfigurieren, dass es es dynamisch auf der Grundlage des Host-Headers in der eingehenden Anforderung benannt wird. Mit der dynamischen Benennung können Sie Ablaufverfolgungen basierend auf dem Domänennamen in der Anforderung gruppieren und einen Standardnamen anwenden, wenn der Name nicht mit einem erwarteten Muster übereinstimmt (z. B. wenn der Host-Header gefälscht ist).

> **Weitergeleitete Anfragen**
>
> Wenn ein Load Balancer oder ein anderer Zwischendienst eine Anfrage an Ihre Anwendung weiterleitet, nimmt X-Ray die Client-IP aus dem `-X-Forwarded-For`Header in der Anfrage

und nicht aus der Quell-IP im IP-Paket. Die Client-IP, die für eine weitergeleitete Anforderung aufgezeichnet wird, kann gefälscht werden, daher sollte sie nicht vertrauenswürdig sein.

Wenn eine Anforderung weitergeleitet wird, legt das SDK ein zusätzliches Feld im Segment fest, um dies anzuzeigen. Wenn das Segment das auf `x_forwarded_for` eingestellte Feld enthält`true`, wurde die Client-IP aus dem -`X-Forwarded-For`Header in der HTTP-Anforderung übernommen.

Die Middleware erzeugt für jede eingehende Anfrage ein Segment mit einem `http`-Block mit folgenden Informationen:

- HTTP-Methode – GET, POST, PUT, DELETE usw.
- Client-Adresse – Die IP-Adresse des Clients, der die Anforderung gesendet hat.
- Antwortcode – Der HTTP-Antwortcode für die abgeschlossene Anforderung.
- Timing – Die Startzeit (als die Anforderung empfangen wurde) und die Endzeit (als die Antwort gesendet wurde).
- Benutzeragent – Die `user-agent` aus der Anforderung.
- Inhaltslänge – Die `content-length` aus der Antwort.

Verwenden der Rails-Middleware

Um die Middleware zu verwenden, aktualisieren Sie Ihre Gemfile-Datei, sodass sie das geforderte railtie enthält.

Example Gemfile – Rails

```
gem 'aws-xray-sdk', require: ['aws-xray-sdk/facets/rails/railtie']
```

Um die Middleware zu verwenden, müssen Sie den Recorder auch mit einem Namen konfigurieren, der die Anwendung in der Ablaufverfolgungszuordnung darstellt.

Example config/initializers/aws_xray.rb

```
Rails.application.config.xray = {
  name: 'my app'
}
```

### Code manuell instrumentieren

Wenn Sie Rails nicht verwenden, erstellen Sie Segmente manuell. Sie können für jede eingehende Anfrage ein Segment erstellen oder Segmente um gepatchte HTTP- oder AWS SDK-Clients erstellen, um dem Recorder Kontext zum Hinzufügen von Untersegmenten bereitzustellen.

```
# Start a segment
segment = XRay.recorder.begin_segment 'my_service'
# Start a subsegment
subsegment = XRay.recorder.begin_subsegment 'outbound_call', namespace: 'remote'

# Add metadata or annotation here if necessary
my_annotations = {
  k1: 'v1',
  k2: 1024
}
segment.annotations.update my_annotations

# Add metadata to default namespace
subsegment.metadata[:k1] = 'v1'

# Set user for the segment (subsegment is not supported)
segment.user = 'my_name'

# End segment/subsegment
XRay.recorder.end_subsegment
XRay.recorder.end_segment
```

### Konfiguration einer Segmentbenennungsstrategie

AWS X-Ray verwendet einen Servicenamen, um Ihre Anwendung zu identifizieren und sie von den anderen Anwendungen, Datenbanken, externen APIs und AWS Ressourcen zu unterscheiden, die Ihre Anwendung verwendet. Wenn das X-Ray-SDK Segmente für eingehende Anfragen generiert, zeichnet es den Servicenamen Ihrer Anwendung im Namensfeld des Segments auf.

Das X-Ray-SDK kann Segmente nach dem Hostnamen im HTTP-Anfrage-Header benennen. Dieser Header kann jedoch gefälscht werden, was zu unerwarteten Knoten in Ihrer Service-Übersicht führen kann. Um zu verhindern, dass das SDK Segmente aufgrund von Anfragen mit gefälschten Host-Headern falsch benennt, müssen Sie einen Standardnamen für eingehende Anfragen angeben.

Wenn Ihre Anwendung Anforderungen für mehrere Domains bedient, können Sie das SDK so konfigurieren, dass es eine dynamische Benennungsstrategie verwendet, um dies in Segmentnamen

widerzuspiegeln. Eine dynamische Benennungsstrategie ermöglicht es dem SDK, den Hostnamen für Anforderungen zu verwenden, die einem erwarteten Muster entsprechen, und den Standardnamen auf Anforderungen anzuwenden, die dies nicht tun.

Sie können beispielsweise eine einzelne Anwendung haben, die Anforderungen an drei Subdomänen verarbeitet – `www.example.comapi.example.com`, und `static.example.com`. Sie können eine dynamische Benennungsstrategie mit dem Muster verwenden, `*.example.com` um Segmente für jede Subdomäne mit einem anderen Namen zu identifizieren, was zu drei Serviceknoten auf der Service-Übersicht führt. Wenn Ihre Anwendung Anforderungen mit einem Hostnamen empfängt, der nicht dem Muster entspricht, sehen Sie auf der Service-Übersicht einen vierten Knoten mit einem von Ihnen angegebenen Fallback-Namen.

Wenn Sie denselben Namen für alle Anfragesegmente verwenden möchten, geben Sie bei der Konfiguration des Recorders den Namen Ihrer Anwendung, wie in den vorherigen Abschnitten gezeigt, ein.

Eine dynamische Benennungsstrategie definiert ein Muster, dem Hostnamen entsprechen sollten, sowie einen Standardnamen, der verwendet wird, wenn der Hostname in der HTTP-Anforderung nicht mit diesem Muster übereinstimmt. Um Segmente dynamisch zu benennen, geben Sie ein Namensmuster im Config-Hash an.

Example main.rb – Dynamische Benennung

```
config = {
  naming_pattern: '*mydomain*',
  name: 'my app',
}

XRay.recorder.configure(config)
```

Sie können "*" im Muster verwenden, um eine Übereinstimmung mit einer beliebigen Zeichenfolge zu erzielen, oder "?" für die Übereinstimmung mit einem einzelnen Zeichen.

> **Note**
>
> Sie können den mit der `AWS_XRAY_TRACING_NAME`-Umgebungsvariablen in Code definierten standardmäßigen Dienstnamen überschreiben.

## Patchen von Bibliotheken zum Instrumentieren von nachgelagerten Aufrufen

Um Downstream-Aufrufe zu instrumentieren, verwenden Sie das X-Ray-SDK für Ruby, um die Bibliotheken zu patchen, die Ihre Anwendung verwendet. Das X-Ray-SDK für Ruby kann die folgenden Bibliotheken patchen.

Unterstützte Bibliotheken

- `net/http`— Instrumentieren Sie HTTP-Clients.
- `aws-sdk`— InstrumentAWS SDK for RubyKunden.

Wenn Sie eine gepatchte Bibliothek verwenden, erstellt das X-Ray-SDK für Ruby ein Untersegment für den Anruf und zeichnet Informationen aus der Anfrage und Antwort auf. Für das SDK muss ein Segment zur Verfügung stehen, damit es ein Untersegment aus der SDK-Middleware oder einem Aufruf von `XRay.recorder.begin_segment` erstellen kann.

Um Bibliotheken zu patchen, geben Sie sie im Konfigurationsobjekt an, das Sie an den X-Ray-Recorder übergeben.

Example main.rb — Patch-Bibliotheken

```
require 'aws-xray-sdk'

config = {
  name: 'my app',
  patch: %I[net_http aws_sdk]
}

XRay.recorder.configure(config)
```

## Nachverfolgen von AWS SDK-Aufrufen mit dem X-Ray SDK for Ruby

Wenn Ihre Anwendung Aufrufe an tätigt, AWS-Services um Daten zu speichern, in eine Warteschlange zu schreiben oder Benachrichtigungen zu senden, verfolgt das X-Ray SDK for Ruby die Aufrufe nachgelagert in Untersegmenten . Nachverfolgte AWS-Services und Ressourcen, auf die Sie innerhalb dieser Services zugreifen (z. B. ein Amazon S3-Bucket oder eine Amazon SQS-Warteschlange), werden auf der Ablaufverfolgungskarte in der X-Ray-Konsole als nachgelagerte Knoten angezeigt.

Das X-Ray SDK for Ruby instrumentiert automatisch alle AWS SDK-Clients, wenn Sie die `aws-sdk` Bibliothek patchen. Einzelne Clients können nicht instrumentiert werden.

Für alle Services können Sie den Namen der API mit dem Namen in der X-Ray-Konsole sehen. Für eine Teilmenge von -Services fügt das X-Ray-SDK dem Segment Informationen hinzu, um mehr Granularität in der Service-Übersicht zu gewährleisten.

Wenn Sie beispielsweise einen Aufruf mit einem instrumentierten DynamoDB-Client tätigen, fügt das SDK dem Segment den Tabellennamen für Aufrufe hinzu, die auf eine Tabelle abzielen. In der Konsole wird jede Tabelle als separater Knoten in der Service-Übersicht angezeigt, mit einem generischen DynamoDB-Knoten für Aufrufe, die keine Tabelle anvisieren.

Example Untersegment für einen Aufruf an DynamoDB zum Speichern eines Elements

```
{
  "id": "24756640c0d0978a",
  "start_time": 1.480305974194E9,
  "end_time": 1.4803059742E9,
  "name": "DynamoDB",
  "namespace": "aws",
  "http": {
    "response": {
      "content_length": 60,
      "status": 200
    }
  },
  "aws": {
    "table_name": "scorekeep-user",
    "operation": "UpdateItem",
    "request_id": "UBQNSO5AEM8T4FDA4RQDEB94OVTDRVV4K4HIRGVJF66Q9ASUAAJG",
  }
}
```

Wenn Sie auf benannte Ressourcen zugreifen, werden durch Aufrufe der folgenden Services weitere Knoten in der Service-Übersicht erstellt. Durch Aufrufe, die keinen bestimmten Ressourcen gelten, wird ein generischer Knoten für den Service erstellt.

- Amazon DynamoDB – Tabellenname
- Amazon Simple Storage Service – Bucket- und Schlüsselname
- Amazon Simple Queue Service – Name der Warteschlange

## Generieren Sie mit dem X-Ray-SDK benutzerdefinierte Untersegmente

Teilsegmente erweitern ein TraceAbschnittmit Details über die geleistete Arbeit, um eine Anfrage zu stellen. Beim Aufruf mit einem instrumentierten Client erfasst das X-Ray-SDK die in einem Untersegment generierten Informationen. Sie können zusätzliche Teilsegmente erstellen, um andere Teilsegmente zu gruppieren, die Leistung eines Codeabschnitts zu messen oder Anmerkungen und Metadaten aufzuzeichnen.

Um Untersegmente zu verwalten, verwenden Sie die Methoden `begin_subsegment` und `end_subsegment`.

```
subsegment = XRay.recorder.begin_subsegment name: 'annotations', namespace: 'remote'
my_annotations = { id: 12345 }
subsegment.annotations.update my_annotations
XRay.recorder.end_subsegment
```

Um ein Subsegment für eine Funktion zu erstellen, kapseln Sie es in einen Aufruf von `XRay.recorder.capture` ein.

```
XRay.recorder.capture('name_for_subsegment') do |subsegment|
  resp = myfunc() # myfunc is your function
  subsegment.annotations.update k1: 'v1'
  resp
end
```

Beim Erstellen eines Untersegments innerhalb eines Segments oder eines anderen Untersegments generiert das X-Ray-SDK eine ID dafür und erfasst die Start- und Endzeit.

Example Untersegment mit Metadaten

```
"subsegments": [{
  "id": "6f1605cd8a07cb70",
  "start_time": 1.480305974194E9,
  "end_time": 1.4803059742E9,
  "name": "Custom subsegment for UserModel.saveUser function",
  "metadata": {
    "debug": {
      "test": "Metadata string from UserModel.saveUser"
    }
  },
```

# Hinzufügen von Anmerkungen und Metadaten zu Segmenten mit dem X-Ray SDK for Ruby

Sie können Anmerkungen und Metadaten verwenden, um zusätzliche Informationen über Anfragen, die Umgebung oder Ihre Anwendung aufzuzeichnen. Sie können Anmerkungen und Metadaten zu den Segmenten hinzufügen, die das X-Ray SDK erstellt, oder zu benutzerdefinierten Untersegmenten, die Sie erstellen.

Anmerkungen sind Schlüssel-Wert-Paare mit Zeichenfolgen-, Zahlen- oder booleschen Werten. Anmerkungen sind für die Verwendung mit Filterausdrücken indexiert. Berücksichtigen Sie Anmerkungen, um Daten zur Gruppierung von Ablaufverfolgungen in der Konsole zu verwenden, oder wenn Sie die `GetTraceSummaries`-API aufrufen.

Metadaten sind Schlüssel-Wert-Paare, die Werte beliebigen Typs enthalten können, einschließlich Objekte und Listen, aber nicht für die Verwendung mit Filterausdrücken indexiert sind. Verwenden Sie Metadaten, um zusätzliche Daten aufzuzeichnen, die Sie im Trace speichern möchten, aber nicht für die Suche verwenden müssen.

Zusätzlich zu Anmerkungen und Metadaten können Sie auch Benutzer-ID-Zeichenfolgen in Segmenten aufzeichnen. Benutzer-IDs werden in einem eigenen Feld aufgezeichnet und zur Suche indiziert.

Sections

- Anmerkungen mit dem X-Ray SDK for Ruby aufnehmen
- Metadaten mit dem X-Ray SDK for Ruby aufnehmen
- Benutzer-IDs mit dem X-Ray SDK for Ruby aufzeichnen

## Anmerkungen mit dem X-Ray SDK for Ruby aufnehmen

Verwenden Sie Anmerkungen, um Informationen zu Segmenten oder Untersegmenten, die zur Suche indiziert werden sollten, aufzuzeichnen.

Anmerkung zu Anforderungen

- Schlüssel — Der Schlüssel für eine X-Ray-Anmerkung kann bis zu 500 alphanumerische Zeichen enthalten. Sie können keine anderen Leerzeichen oder Symbole als den Unterstrich (_) verwenden.
- Werte — Der Wert für eine X-Ray-Anmerkung kann bis zu 1.000 Unicode-Zeichen enthalten.
- Die Anzahl der Anmerkungen — Sie können bis zu 50 Anmerkungen pro Spur verwenden.

So zeichnen Sie Anmerkungen auf

1. Eine Referenz des aktuellen Segments oder Untersegments finden Sie unter `xray_recorder`.

   ```
   require 'aws-xray-sdk'
   ...
   document = XRay.recorder.current_segment
   ```

   or

   ```
   require 'aws-xray-sdk'
   ...
   document = XRay.recorder.current_subsegment
   ```

2. Rufen Sie `update` mit einem Hash-Wert auf.

   ```
   my_annotations = { id: 12345 }
   document.annotations.update my_annotations
   ```

Das SDK zeichnet Anmerkungen als Schlüssel-Wert-Paare in einem `annotations`-Objekt im Segmentdokument auf. Wenn `add_annotations` zweimal mit demselben Schlüssel aufgerufen wird, werden zuvor aufgezeichnete Werte im gleichen Segment oder Untersegment überschrieben.

Nutzen Sie das `annotations.`*`key`*-Schlüsselwort in einem Filterausdruck, um Ablaufverfolgungen durch Anmerkungen mit bestimmten Werten zu finden.

Metadaten mit dem X-Ray SDK for Ruby aufnehmen

Verwenden Sie Metadaten, um Segment- oder Untersegmentinformationen aufzuzeichnen, die nicht zur Suche indiziert werden müssen. Metadatenwerte sind Zeichenfolgen, Zahlen, boolesche Werte oder andere Objekte, die in Form eines JSON-Objekts oder eines Arrays angeordnet sein können.

So zeichnen Sie Metadaten auf

1. Eine Referenz des aktuellen Segments oder Untersegments finden Sie unter `xray_recorder`.

   ```
   require 'aws-xray-sdk'
   ...
   document = XRay.recorder.current_segment
   ```

or

```
require 'aws-xray-sdk'
...
document = XRay.recorder.current_subsegment
```

2. Rufen Sie `metadata` mit einem Zeichenfolgenschlüssel, einem booleschen Wert, einer Zahl, einer Zeichenfolge oder einem Objektwert und einem Zeichenfolgen-Namespace auf.

```
my_metadata = {
  my_namespace: {
    key: 'value'
  }
}
subsegment.metadata my_metadata
```

Wenn `metadata` zweimal mit demselben Schlüssel aufgerufen wird, werden zuvor aufgezeichnete Werte im gleichen Segment oder Untersegment überschrieben.

Benutzer-IDs mit dem X-Ray SDK for Ruby aufzeichnen

Zeichnen Sie Benutzer-IDs in Anforderungssegmenten auf, um den Benutzer zu identifizieren, der die Anforderung gesendet hat.

So zeichnen Sie Benutzer-IDs auf

1. Eine Referenz des aktuellen Segments finden Sie unter `xray_recorder`.

```
require 'aws-xray-sdk'
...
document = XRay.recorder.current_segment
```

2. Setzen Sie das Benutzerfeld auf dem Segment auf eine Zeichenfolgen-ID des Benutzers, der die Anforderung gesendet hat.

```
segment.user = 'U12345'
```

Sie können den Benutzern in Ihren Controllern festlegen, um die Benutzer-ID aufzuzeichnen, sobald die Anwendung mit der Bearbeitung einer Anfrage beginnt.

Nutzen Sie das `user`-Schlüsselwort in einem Filterausdruck, um Ablaufverfolgungen einer Benutzer-ID zu finden.

# Integrieren Sie AWS X-Ray mit anderen AWS-Services

Viele AWS-Services bieten unterschiedliche Stufen der X-Ray-Integration, darunter Sampling und Hinzufügen von Headern zu eingehenden Anfragen, Ausführung des X-Ray-Daemons und automatisches Senden von Trace-Daten an X-Ray. Die Integration mit X-Ray kann Folgendes beinhalten:

- Aktive Instrumentierung — Eingehende Anfragen für Proben und Instrumente
- Passive Instrumentierung — Anfragen zu Instrumenten, die von einem anderen Dienst gesampelt wurden
- Anforderungsverfolgung — Fügt allen eingehenden Anfragen einen Tracing-Header hinzu und leitet ihn anschließend weiter
- Tooling — Führt den X-Ray-Daemon aus, um Segmente vom X-Ray SDK zu empfangen

> **Note**
>
> Die X-Ray-SDKs enthalten Plugins für eine zusätzliche Integration mit AWS-Services. Sie können beispielsweise das Elastic Beanstalk-Plug-In X-Ray SDK for Java verwenden, um Informationen über die Elastic Beanstalk Beanstalk-Umgebung hinzuzufügen, in der Ihre Anwendung ausgeführt wird, einschließlich des Umgebungsnamens und der ID.

Hier sind einige Beispiele dafür AWS-Services , die in X-Ray integriert sind:

- AWS Distro for OpenTelemetry (ADOT) — Mit ADOT können Ingenieure ihre Anwendungen einmal instrumentieren und korrelierte Metriken und Traces an mehrere AWS Überwachungslösungen senden, darunter Amazon CloudWatch AWS X-Ray, Amazon Service und Amazon Managed OpenSearch Service for Prometheus.
- AWS Lambda— Aktive und passive Instrumentierung eingehender Anfragen zu allen Laufzeiten. AWS Lambda fügt Ihrer Trace-Map zwei Knoten hinzu, einen für den AWS Lambda Service und einen für die Funktion. Wenn Sie die Instrumentierung aktivieren, wird AWS Lambda auch der X-Ray-Daemon auf Java- und Node.js Runtimes zur Verwendung mit dem X-Ray SDK ausgeführt.
- Amazon API Gateway — Aktive und passive Instrumentierung. API Gateway verwendet Stichprobenregeln, um zu bestimmen, welche Anfragen aufgezeichnet werden sollen, und fügt Ihrer Service-Map einen Knoten für die Gateway-Phase hinzu.

- AWS Elastic Beanstalk— Werkzeuge. Elastic Beanstalk enthält den X-Ray-Daemon auf den folgenden Plattformen:
  - Java SE — 2.3.0 und spätere Konfigurationen
  - Tomcat — 2.4.0 und spätere Konfigurationen
  - Node.js — 3.2.0 und spätere Konfigurationen
  - Windows Server — Alle Konfigurationen außer Windows Server Core, die nach dem 9. Dezember 2016 veröffentlicht wurden

  Sie können die Elastic Beanstalk-Konsole verwenden, um Elastic Beanstalk anzuweisen, den Daemon auf diesen Plattformen auszuführen, oder Sie können die `XRayEnabled` Option im Namespace verwenden. `aws:elasticbeanstalk:xray`
- Elastic Load Balancing — Ablaufverfolgung von Anfragen auf Application Load Balancers. Der Application Load Balancer fügt die Trace-ID dem Anforderungsheader hinzu, bevor er ihn an eine Zielgruppe sendet.
- Amazon EventBridge — Passive Instrumentierung. Wenn ein Dienst, der Ereignisse veröffentlicht, mit dem X-Ray SDK instrumentiert EventBridge ist, erhalten die Ereignisziele den Tracing-Header und können weiterhin die ursprüngliche Trace-ID weitergeben.
- Amazon Simple Notification Service — Passive Instrumentierung. Wenn ein Amazon SNS SNS-Publisher seinen Kunden mit dem X-Ray SDK verfolgt, können Abonnenten den Tracing-Header abrufen und den ursprünglichen Trace vom Herausgeber mit derselben Trace-ID weiterleiten.
- Amazon Simple Queue Service — Passive Instrumentierung. Wenn ein Service Anfragen mithilfe des X-Ray-SDK verfolgt, kann Amazon SQS den Tracing-Header senden und den ursprünglichen Trace mit einer konsistenten Trace-ID weiterhin vom Absender an den Verbraucher weitergeben.

Wählen Sie eines der folgenden Themen aus, um sich mit allen integrierten Funktionen vertraut zu machen. AWS-Services

Themen

- AWSDistro fürOpenTelemetryundAWS X-Ray
- Unterstützung für aktives Amazon API Gateway-Tracing für AWS X-Ray
- Amazon EC2 und AWS App Mesh
- AWSApp Runner und X-Ray
- AWS AppSync und AWS X-Ray
- Protokollieren von X-Ray-API-Aufrufen mit AWS CloudTrail

- CloudWatch -Integration mit X-Ray
- Verfolgung von Konfigurationsänderungen der X-Ray-Verschlüsselung mitAWS Config
- Amazon Elastic Compute Cloud undAWS X-Ray
- AWS Elastic Beanstalk und AWS X-Ray
- Elastic Load Balancing und AWS X-Ray
- Amazon EventBridge und AWS X-Ray
- AWS Lambda und AWS X-Ray
- Amazon SNS und AWS X-Ray
- AWS Step Functions und AWS X-Ray
- Amazon SQS und AWS X-Ray
- Amazon S3 und AWS X-Ray

# AWSDistro fürOpenTelemetryundAWS X-Ray

Verwenden Sie dieAWSDistribution fürOpenTelemetry(ADOT) zum Sammeln und Senden von Metriken und Traces anAWS X-Rayund andere Überwachungslösungen wie AmazonCloudWatch, AmazonOpenSearchService und Amazon Managed Service für Prometheus.

## AWS Distro für OpenTelemetry

DerAWSDistribution fürOpenTelemetry(ADOT) ist einAWSVertrieb auf der Grundlage der Cloud Native Computing Foundation (CNCF)OpenTelemetryprojekt. OpenTelemetrybietet einen einzigen Satz von Open-Source-APIs, -Bibliotheken und -Agenten zur Erfassung verteilter Traces und Metriken. Dieses Toolkit ist eine Distribution von UpstreamOpenTelemetryKomponenten wie SDKs, Agenten für automatische Instrumentierung und Collectors, die getestet, optimiert, gesichert und unterstützt werden vonAWS.

Mit ADOT können Ingenieure ihre Anwendungen einmal instrumentieren und korrelierte Metriken und Traces an mehrere sendenAWSÜberwachungslösungen wie AmazonCloudWatch,AWS X-Ray, AmazonasOpenSearchService und Amazon Managed Service für Prometheus.

ADOT ist in eine wachsende Anzahl von integriertAWS-Servicesum das Senden von Traces und Metriken an Monitoring-Lösungen wie X-Ray zu vereinfachen. Einige Beispiele für Dienste, die in ADOT integriert sind, umfassen:

- AWS Lambda—AWSverwaltete Lambda-Schichten für ADOT bietenplug-and-playBenutzererlebnis durch automatische Instrumentierung einer Lambda-Funktion, PaketierungOpenTelemetryzusammen mit einemout-of-the-boxKonfiguration fürAWS Lambdaund X-Ray in einer einfach einzurichtenden Ebene. Benutzer können es aktivieren und deaktivierenOpenTelemetryfür ihre Lambda-Funktion, ohne den Code zu ändern. Weitere Informationen finden Sie unterAWSDistribution fürOpenTelemetryLambda
- Amazon Elastic Container Service (ECS)— Erfassen Sie Metriken und Traces aus Amazon ECS-Anwendungen mithilfe derAWSDistribution fürOpenTelemetryCollector, zum Senden an X-Ray und andere Überwachungslösungen. Weitere Informationen finden Sie unterSammeln von Anwendungs-Trace-Datenim Amazon ECS-Entwicklerhandbuch.
- AWSApp Runner— App Runner unterstützt das Senden von Traces an X-Ray mithilfe derAWSDistro fürOpenTelemetry(ADOPTIEREN). Verwenden Sie ADOT SDKs, um Trace-Daten für Ihre containerisierten Anwendungen zu sammeln, und verwenden Sie X-Ray, um Ihre instrumentierte Anwendung zu analysieren und Einblicke in sie zu gewinnen. Weitere Informationen finden Sie unterAWSApp Runner und X-Ray.

Für weitere Informationen überAWSDistribution fürOpenTelemetry, einschließlich Integration mit zusätzlichenAWS-Services, sieheAWSDistribution fürOpenTelemetryDokumentation.

Für weitere Informationen zur Instrumentierung Ihrer Anwendung mitAWSDistribution fürOpenTelemetryund X-Ray, sieheInstrumentieren Sie Ihre Anwendung mit demAWSDistribution fürOpenTelemetry.

# Unterstützung für aktives Amazon API Gateway-Tracing für AWS X-Ray

Sie können X-Ray verwenden, um Benutzeranfragen zu verfolgen und zu analysieren, während sie über Ihre Amazon API Gateway-APIs zu den zugrunde liegenden Services gelangen. API Gateway unterstützt X-Ray-Ablaufverfolgung für alle API Gateway-Endpunkttypen: regional, Edge-optimiert und privat. Sie können X-Ray mit Amazon API Gateway in allen verwenden, in AWS-Regionen denen X-Ray verfügbar ist. Weitere Informationen finden Sie unter Verfolgen der API Gateway-API-Ausführung mit AWS X-Ray im Amazon API Gateway-Entwicklerhandbuch.

> **Note**
>
> X-Ray unterstützt nur die Nachverfolgung für REST-APIs über API Gateway.

Amazon API Gateway bietet Unterstützung für aktives Tracing für AWS X-Ray. Aktivieren Sie die aktive Nachverfolgung für Ihre API-Stufen, um eingehende Anfragen zu erfassen und Ablaufverfolgungen an X-Ray zu senden.

So aktivieren Sie die aktive Ablaufverfolgung auf einer API-Stufe

1. Öffnen Sie die API Gateway-Konsole unter https://console.aws.amazon.com/apigateway/.
2. Wählen Sie eine API aus.
3. Wählen Sie eine Stufe aus.
4. Wählen Sie auf der Registerkarte Protokolle/Ablaufverfolgung die Option X-Ray-Ablaufverfolgung aktivieren und dann Änderungen speichern aus.
5. Wählen Sie im Navigationsbereich links Resources (Ressourcen) aus.
6. Um die API mit den neuen Einstellungen erneut bereitzustellen, wählen Sie das Dropdown-Menü Aktionen und dann API bereitstellen aus.

API Gateway verwendet Samplingregeln, die Sie in der X-Ray-Konsole definieren, um zu bestimmen, welche Anforderungen aufgezeichnet werden sollen. Sie können Regeln erstellen, die nur für APIs oder gelten, die nur für Anforderungen gelten, die bestimmte Header enthalten. API Gateway zeichnet Header in Attributen im Segment zusammen mit Details zur Stufe und Anforderung auf. Weitere Informationen finden Sie unter Konfigurieren Sie Stichprobenregeln.

> **Note**
>
> Bei der Nachverfolgung von REST-APIs mit API Gateway-HTTP-Integration wird der Servicename jedes Segments auf den Anforderungs-URL-Pfad von API Gateway zu Ihrem HTTP-Integrationsendpunkt festgelegt, was zu einem Serviceknoten auf der X-Ray-Ablaufverfolgungszuordnung für jeden eindeutigen URL-Pfad führt. Eine große Anzahl von URL-Pfaden kann dazu führen, dass die Ablaufverfolgungszuordnung das Limit von 10.000 Knoten überschreitet, was zu einem Fehler führt.
> Um die Anzahl der von API Gateway erstellten Serviceknoten zu minimieren, sollten Sie Parameter innerhalb der URL-Abfragezeichenfolge oder im Anforderungstext über POST übergeben. Beide Ansätze stellen sicher, dass Parameter nicht Teil des URL-Pfads sind, was zu weniger unterschiedlichen URL-Pfaden und Serviceknoten führen kann.

Für alle eingehenden Anforderungen fügt API Gateway eingehenden HTTP-Anforderungen, die noch keinen haben, einen Ablaufverfolgungs-Header hinzu.

```
X-Amzn-Trace-Id: Root=1-5759e988-bd862e3fe1be46a994272793
```

X-Ray-Trace-ID-Format

Ein X-Ray `trace_id` besteht aus drei Zahlen, die durch Bindestriche getrennt sind. Beispiel: `1-58406520-a006649127e371903a2de979` Dies umfasst:

- Die Versionsnummer, 1.
- Die Uhrzeit der ursprünglichen Anforderung in Unix-Epochenzeit mit 8 Hexadezimalziffern.

  Beispielsweise beträgt 10:00 Uhr am 1. Dezember 2016 PST in der Epochenzeit `1480615200` Sekunden oder `58406520` Hexadezimalziffern.
- Eine global eindeutige 96-Bit-Kennung für die Ablaufverfolgung in 24 Hexadezimalziffern.

Wenn die aktive Ablaufverfolgung deaktiviert ist, zeichnet die Stufe dennoch ein Segment auf, wenn die Anforderung von einem Service stammt, der die Anforderung geprüft und eine Nachverfolgung gestartet hat. Beispielsweise kann eine instrumentierte Webanwendung eine API Gateway-API mit einem HTTP-Client aufrufen. Wenn Sie einen HTTP-Client mit dem X-Ray-SDK instrumentieren, fügt er der ausgehenden Anforderung einen Ablaufverfolgungs-Header hinzu, der die Sampling-Entscheidung enthält. API Gateway liest den Ablaufverfolgungs-Header und erstellt ein Segment für Stichprobenanforderungen.

Wenn Sie API Gateway verwenden, um ein Java-SDK für Ihre API zu generieren, können Sie den SDK-Client instrumentieren, indem Sie einen Anfrage-Handler mit dem Client-Builder hinzufügen, genauso wie Sie einen AWS SDK-Client manuell instrumentieren würden. Detaillierte Anweisungen finden Sie unter Verfolgen von AWS SDK-Aufrufen mit dem X-Ray SDK for Java.

## Amazon EC2 und AWS App Mesh

AWS X-Ray lässt sich in integrierenAWS App Mesh, um Envoy-Proxys für Microservices zu verwalten. App Mesh bietet eine Version von Envoy, die Sie so konfigurieren können, dass Ablaufverfolgungsdaten an den X-Ray-Daemon gesendet werden, der in einem Container derselben Aufgabe oder desselben Pods ausgeführt wird. X-Ray unterstützt die Nachverfolgung mit den folgenden App Mesh-kompatiblen Services:

- Amazon Elastic Container Service (Amazon ECS)
- Amazon Elastic Kubernetes Service (Amazon EKS)
- Amazon Elastic Compute Cloud (Amazon EC2)

Verwenden Sie die folgenden Anweisungen, um zu erfahren, wie Sie das X-Ray-Tracing durch App Mesh aktivieren.

Um den Envoy-Proxy für das Senden von Daten an X-Ray zu konfigurieren, legen Sie die `ENABLE_ENVOY_XRAY_TRACING` Umgebungsvariable in seiner Containerdefinition fest.

> **Note**
>
> Die App-Mesh-Version von Envoy sendet derzeit keine Ablaufverfolgungen basierend auf konfigurierten Samplingregeln. Stattdessen wird eine feste Abtastrate von 5 % für Envoy Version 1.16.3 oder höher oder eine Abtastrate von 50 % für Envoy-Versionen vor 1.16.3 verwendet.

Example Envoy-Containerdefinition für Amazon ECS

```
{
      "name": "envoy",
      "image": "public.ecr.aws/appmesh/aws-appmesh-envoy:envoy-version",
      "essential": true,
      "environment": [
        {
          "name": "APPMESH_VIRTUAL_NODE_NAME",
          "value": "mesh/myMesh/virtualNode/myNode"
        },
        {
          "name": "ENABLE_ENVOY_XRAY_TRACING",
          "value": "1"
        }
      ],
      "healthCheck": {
        "command": [
          "CMD-SHELL",
          "curl -s http://localhost:9901/server_info | cut -d' ' -f3 | grep -q live"
        ],
        "startPeriod": 10,
        "interval": 5,
        "timeout": 2,
        "retries": 3
      }
```

> **Note**
>
> Weitere Informationen zu verfügbaren Adressen in der Envoy-Region finden Sie unter Envoy image im AWS App Mesh -Benutzerhandbuch.

Weitere Informationen zum Ausführen des X-Ray-Daemons in einem Container finden Sie unter Ausführen von X-Ray-Daemon auf Amazon ECs. Stellen Sie für eine Beispielanwendung, die ein Servicegitter, Microservice, Envoy-Proxy und X-Ray-Daemon enthält, das `colorapp` Beispiel im App Mesh Examples GitHub-Repository bereit.

Weitere Informationen

- Erste Schritte mit AWS App Mesh
- Erste Schritte mit AWS App Mesh und Amazon ECS

## AWSApp Runner und X-Ray

AWSApp Runner ist einAWS-Servicedas bietet eine schnelle, einfache und kostengünstige Möglichkeit, Quellcode oder ein Container-Image direkt in eine skalierbare und sichere Webanwendung in derAWS Cloud. Sie müssen sich nicht in neue Technologien einarbeiten, entscheiden, welchen Compute-Service Sie nutzen möchten, oder wissen, wie Sie AWS-Ressourcen bereitstellen und konfigurieren. SiehstWas istAWSApp Runnerfür weitere Informationen.

AWSApp Runner sendet Traces an X-Ray durch die Integration mitAWSDistribution fürOpenTelemetry(ADOPTIEREN). Verwenden Sie ADOT SDKs, um Trace-Daten für Ihre containerisierten Anwendungen zu sammeln, und verwenden Sie X-Ray, um Ihre instrumentierte Anwendung zu analysieren und Einblicke in sie zu gewinnen. Weitere Informationen finden Sie unterTracing für Ihre App Runner-Anwendung mit X-Ray.

## AWS AppSync und AWS X-Ray

Sie können Anforderungen für aktivieren und verfolgenAWSAppSync. Weitere Informationen finden Sie unterAblaufverfolgung mitAWSX-Rayfür Anweisungen.

Wenn die X-Ray-Nachverfolgung für eine aktiviert istAWSAppSync API, einAWSIdentity and Access ManagementService-verknüpfte -RolleWird in Ihrem Konto automatisch mit den entsprechenden

Berechtigungen erstellt. Dies ermöglichtAWSAppSync, um Ablaufverfolgungen auf sichere Weise an X-Ray zu senden.

# Protokollieren von X-Ray-API-Aufrufen mit AWS CloudTrail

AWS X-Ray ist in integriert, einem ServiceAWS CloudTrail, der die Aktionen eines Benutzers, einer Rolle oder eines aufzeichnet AWS-Service. CloudTrail erfasst alle API-Aufrufe für X-Ray als Ereignisse. Zu den erfassten Aufrufen gehören Aufrufe von der X-Ray-Konsole und Codeaufrufe der X-Ray-API-Operationen. Anhand der von CloudTrailgesammelten Informationen können Sie die an X-Ray gestellte Anfrage, die IP-Adresse, von der die Anfrage gestellt wurde, den Zeitpunkt der Anfrage und zusätzliche Details bestimmen.

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Ob die Anfrage mit Anmeldeinformationen des Root-Benutzers oder des Benutzers gestellt wurde.
- Ob die Anforderung im Namen eines IAM-Identity-Center-Benutzers gestellt wurde.
- Gibt an, ob die Anforderung mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen Verbundbenutzer gesendet wurde.
- Ob die Anforderung aus einem anderen AWS-Service gesendet wurde.

CloudTrail ist in Ihrem aktiv AWS-Konto , wenn Sie das Konto erstellen und automatisch Zugriff auf den CloudTrail Ereignisverlauf haben. Der CloudTrail Ereignisverlauf bietet eine anzeigbare, durchsuchbare, herunterladbare und unveränderliche Aufzeichnung der aufgezeichneten Verwaltungsereignisse der letzten 90 Tage in einem AWS-Region. Weitere Informationen finden Sie unter Arbeiten mit dem CloudTrail Ereignisverlauf im AWS CloudTrail -Benutzerhandbuch. Für die Anzeige des Ereignisverlaufs fallen keine CloudTrail Gebühren an.

Erstellen Sie für eine fortlaufende Aufzeichnung der Ereignisse in den AWS-Konto letzten 90 Tagen einen Trail oder einen CloudTrail Lake-Ereignisdatenspeicher.

CloudTrail Trails

Ein Trail ermöglicht CloudTrail die Bereitstellung von Protokolldateien an einen Amazon S3-Bucket. Alle Trails, die mit der erstellt wurden, AWS Management Console sind multiregional. Sie können einen Trail für eine oder mehrere Regionen erstellen, indem Sie die verwenden AWS CLI. Das Erstellen eines multiregionalen Trails wird empfohlen, da Sie Aktivitäten in allen AWS-Regionen in Ihrem Konto erfassen. Wenn Sie einen Trail für eine einzelne Region erstellen,

können Sie nur die Ereignisse anzeigen, die im des Trails protokolliert wurden AWS-Region. Weitere Informationen zu Trails finden Sie unter Erstellen eines Trails für Ihr AWS-Konto und Erstellen eines Trails für eine Organisation im AWS CloudTrail -Benutzerhandbuch.

Sie können eine Kopie Ihrer laufenden Verwaltungsereignisse in Ihrem Amazon S3-Bucket kostenlos von bereitstellen, CloudTrail indem Sie einen Trail erstellen. Es fallen jedoch Amazon S3-Speichergebühren an. Weitere Informationen zu CloudTrail Preisen finden Sie unter -AWS CloudTrail Preise. Informationen zu Amazon-S3-Preisen finden Sie unter Amazon S3-Preise.

CloudTrail Lake-Ereignisdatenspeicher

Mit CloudTrail Lake können Sie SQL-basierte Abfragen für Ihre Ereignisse ausführen. CloudTrail Lake konvertiert vorhandene Ereignisse im zeilenbasierten JSON-Format in das Apache-ORC-Format. ORC ist ein spaltenförmiges Speicherformat, das für den schnellen Abruf von Daten optimiert ist. Die Ereignisse werden in Ereignisdatenspeichern zusammengefasst, bei denen es sich um unveränderliche Sammlungen von Ereignissen handelt, die auf Kriterien basieren, die Sie mit Hilfe von erweiterten Ereignisselektoren auswählen. Die Selektoren, die Sie auf einen Ereignisdatenspeicher anwenden, steuern, welche Ereignisse bestehen bleiben und für Sie zur Abfrage verfügbar sind. Weitere Informationen zu CloudTrail Lake finden Sie unter Arbeiten mit AWS CloudTrail Lake im AWS CloudTrail -Benutzerhandbuch.

CloudTrail Für Lake-Ereignisdatenspeicher und Abfragen fallen Kosten an. Beim Erstellen eines Ereignisdatenspeichers wählen Sie die Preisoption aus, die für den Ereignisdatenspeicher genutzt werden soll. Die Preisoption bestimmt die Kosten für die Erfassung und Speicherung von Ereignissen sowie die standardmäßige und maximale Aufbewahrungsdauer für den Ereignisdatenspeicher. Weitere Informationen zu CloudTrail Preisen finden Sie unter -AWS CloudTrail Preise.

Themen



## X-Ray-Verwaltungsereignisse in CloudTrail

AWS X-Ray lässt sich in integrieren, AWS CloudTrail um API-Aktionen eines Benutzers, einer Rolle oder eines AWS-Service in X-Ray aufzuzeichnen. Sie können verwenden CloudTrail , um X-Ray-API-

Anforderungen in Echtzeit zu überwachen und Protokolle in Amazon S3, Amazon CloudWatch Logs und Amazon CloudWatch Events zu speichern. X-Ray unterstützt die Protokollierung der folgenden Aktionen als Ereignisse in CloudTrail Protokolldateien:

Unterstützte API-Aktionen

- PutEncryptionConfig
- GetEncryptionConfig
- CreateGroup
- UpdateGroup
- DeleteGroup
- GetGroup
- GetGroups
- GetInsight
- GetInsightEvents
- GetInsightImpactGraph
- GetInsightSummaries
- GetSamplingStatisticSummaries

## X-Ray-Datenereignisse in CloudTrail

Datenereignisse liefern Informationen über die Ressourcenoperationen, die für oder in einer Ressource ausgeführt werden (z. B. PutTraceSegments, die Segmentdokumente in X-Ray hochlädt).

Sie werden auch als Vorgänge auf Datenebene bezeichnet. Datenereignisse sind oft Aktivitäten mit hohem Volume. Standardmäßig protokolliert CloudTrail keine Datenereignisse. Der CloudTrail Ereignisverlauf zeichnet keine Datenereignisse auf.

Für Datenereignisse werden zusätzliche Gebühren fällig. Weitere Informationen zu CloudTrail Preisen finden Sie unter AWS CloudTrail – Preise.

Sie können Datenereignisse für die X-Ray-Ressourcentypen mithilfe der CloudTrail Konsole AWS CLI oder API CloudTrail -Operationen protokollieren. Weitere Informationen zum Protokollieren von Datenereignissen finden Sie unter Protokollieren von Datenereignissen mit der AWS Management

Console und Protokollieren von Datenereignissen mit der AWS Command Line Interface im AWS CloudTrail -Benutzerhandbuch.

In der folgenden Tabelle sind die X-Ray-Ressourcentypen aufgeführt, für die Sie Datenereignisse protokollieren können. In der Spalte Datenereignistyp (Konsole) wird der Wert angezeigt, der aus der Liste Datenereignistyp in der CloudTrail Konsole ausgewählt werden kann. Die Spalte resources.type value zeigt den `resources.type` Wert an, den Sie bei der Konfiguration erweiterter Ereignisselektoren mit der AWS CLI oder CloudTrail APIs angeben würden. Die Spalte Daten-APIs, die in protokolliert CloudTrail wurden, zeigt die API-Aufrufe an, die CloudTrail für den Ressourcentyp protokolliert wurden.

| Typ des Datenereignisses (Konsole) | resources.type-Wert | Daten-APIs, die bei protokoll iert wurden CloudTrail |
| --- | --- | --- |
| X-Ray-Ablaufverfolgung | `AWS::XRay::Trace` | • PutTraceSegments<br>• GetTraceSummaries<br>• GetTraceGraph<br>• GetServiceGraph<br>• BatchGetTraces<br>• GetTimeSeriesServi ceStatistics<br>• PutTelemetryRecords<br>• GetSamplingTargets |

Sie können erweiterte Ereignisselektoren konfigurieren, um nach den `readOnly` Feldern `eventName` und zu filtern und nur die Ereignisse zu protokollieren, die für Sie wichtig sind. Sie können Ereignisse jedoch nicht auswählen, indem Sie die `resources.ARN` Feldauswahl hinzufügen, da X-Ray-Ablaufverfolgungen keine ARNs haben. Weitere Informationen zu diesen Feldern finden Sie unter AdvancedFieldSelector in der APIAWS CloudTrail -Referenz zu . Im Folgenden finden Sie ein Beispiel für die Ausführung des `put-event-selectors` AWS CLI Befehls zum Protokollieren von Datenereignissen in einem CloudTrail Trail. Sie müssen den Befehl in ausführen oder die Region angeben, in der der Trail erstellt wurde. Andernfalls gibt die Operation eine `InvalidHomeRegionException` Ausnahme zurück.

```
aws cloudtrail put-event-selectors --trail-name myTrail --advanced-event-selectors \
```

```
'{
   "AdvancedEventSelectors": [
      {
         "FieldSelectors": [
            { "Field": "eventCategory", "Equals": ["Data"] },
            { "Field": "resources.type", "Equals": ["AWS::XRay::Trace"] },
            { "Field": "eventName", "Equals":
 ["PutTraceSegments","GetSamplingTargets"] }
         ],
         "Name": "Log X-Ray PutTraceSegments and GetSamplingTargets data events"
      }
   ]
}'
```

# X-Ray-Ereignisbeispiele

## Beispiel für Verwaltungsereignisse, **`GetEncryptionConfig`**

Im Folgenden finden Sie ein Beispiel für den X-Ray-GetEncryptionConfigProtokolleintrag in CloudTrail.

Example

```
{
    "eventVersion"=>"1.05",
    "userIdentity"=>{
        "type"=>"AssumedRole",
        "principalId"=>"AROAJVHBZWD3DN6CI2MHM:MyName",
        "arn"=>"arn:aws:sts::123456789012:assumed-role/MyRole/MyName",
        "accountId"=>"123456789012",
        "accessKeyId"=>"AKIAIOSFODNN7EXAMPLE",
        "sessionContext"=>{
            "attributes"=>{
                "mfaAuthenticated"=>"false",
                "creationDate"=>"2023-7-01T00:24:36Z"
            },
            "sessionIssuer"=>{
                "type"=>"Role",
                "principalId"=>"AROAJVHBZWD3DN6CI2MHM",
                "arn"=>"arn:aws:iam::123456789012:role/MyRole",
                "accountId"=>"123456789012",
                "userName"=>"MyRole"
            }
```

```
        }
    },
    "eventTime"=>"2023-7-01T00:24:36Z",
    "eventSource"=>"xray.amazonaws.com",
    "eventName"=>"GetEncryptionConfig",
    "awsRegion"=>"us-east-2",
    "sourceIPAddress"=>"33.255.33.255",
    "userAgent"=>"aws-sdk-ruby2/2.11.19 ruby/2.3.1 x86_64-linux",
    "requestParameters"=>nil,
    "responseElements"=>nil,
    "requestID"=>"3fda699a-32e7-4c20-37af-edc2be5acbdb",
    "eventID"=>"039c3d45-6baa-11e3-2f3e-e5a036343c9f",
    "eventType"=>"AwsApiCall",
    "recipientAccountId"=>"123456789012"
}
```

## Beispiel für ein Datenereignis, **`PutTraceSegments`**

Im Folgenden finden Sie ein Beispiel für den X-Ray-PutTraceSegmentsDatenereignisprotokolleintrag in CloudTrail.

Example

```
{
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAWYXPW54Y4NEXAMPLE:i-0dzz2ac111c83zz0z",
    "arn": "arn:aws:sts::012345678910:assumed-role/my-service-role/
i-0dzz2ac111c83zz0z",
    "accountId": "012345678910",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAWYXPW54Y4NEXAMPLE",
        "arn": "arn:aws:iam::012345678910:role/service-role/my-service-role",
        "accountId": "012345678910",
        "userName": "my-service-role"
      },
      "attributes": {
        "creationDate": "2024-01-22T17:34:11Z",
        "mfaAuthenticated": "false"
```

```
      },
      "ec2RoleDelivery": "2.0"
    }
  },
  "eventTime": "2024-01-22T18:22:05Z",
  "eventSource": "xray.amazonaws.com",
  "eventName": "PutTraceSegments",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "198.51.100.0",
  "userAgent": "aws-sdk-ruby3/3.190.0 md/internal ua/2.0 api/xray#1.0.0 os/linux md/
x86_64 lang/ruby#2.7.8 md/2.7.8 cfg/retry-mode#legacy",
  "requestParameters": {
    "traceSegmentDocuments": [
      "trace_id:1-00zzz24z-EXAMPLE4f4e41754c77d0000",
      "trace_id:1-00zzz24z-EXAMPLE4f4e41754c77d0000",
      "trace_id:1-00zzz24z-EXAMPLE4f4e41754c77d0001",
      "trace_id:1-00zzz24z-EXAMPLE4f4e41754c77d0002"
    ]
  },
  "responseElements": {
    "unprocessedTraceSegments": []
  },
  "requestID": "5zzzzz64-acbd-46ff-z544-451a3ebcb2f8",
  "eventID": "4zz51z7z-77f9-44zz-9bd7-6c8327740f2e",
  "readOnly": false,
  "resources": [
    {
      "type": "AWS::XRay::Trace"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": false,
  "recipientAccountId": "012345678910",
  "eventCategory": "Data",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ZZZZZ-RSA-AAA128-GCM-SHA256",
    "clientProvidedHostHeader": "example.us-west-2.xray.cloudwatch.aws.dev"
  }
}
```

# CloudWatch -Integration mit X-Ray

AWS X-Ray lässt sich in CloudWatch Application Signals , CloudWatch RUM und CloudWatch Synthetics integrieren, um die Überwachung des Zustands Ihrer Anwendungen zu erleichtern. Aktivieren Sie Ihre Anwendung für Application Signals, um den Betriebszustand Ihrer Services, Clientseiten, Synthetics-Canarys und Serviceabhängigkeiten zu überwachen und zu beheben.

Durch die Korrelation von CloudWatch Metriken, Protokollen und X-Ray-Ablaufverfolgungen bietet die X-Ray-Ablaufverfolgungskarte einen end-to-end Überblick über Ihre Services, mit denen Sie Leistungsengpässe schnell erkennen und betroffene Benutzer identifizieren können.

Mit CloudWatch RUM können Sie eine echte Benutzerüberwachung durchführen, um clientseitige Daten über die Leistung Ihrer Webanwendung aus tatsächlichen Benutzersitzungen nahezu in Echtzeit zu erfassen und anzuzeigen. Mit AWS X-Ray und CloudWatch RUM können Sie den Anforderungspfad von Endbenutzern Ihrer Anwendung über nachgelagerte AWS verwaltete Services analysieren und debuggen. Auf diese Weise können Sie Latenztrends und Fehler identifizieren, die sich auf Ihre Endbenutzer auswirken.

Themen



## CloudWatch RUM und AWS X-Ray

Mit Amazon CloudWatch RUM können Sie eine echte Benutzerüberwachung durchführen, um clientseitige Daten über die Leistung Ihrer Webanwendung aus tatsächlichen Benutzersitzungen nahezu in Echtzeit zu erfassen und anzuzeigen. Mit AWS X-Ray und CloudWatch RUM können Sie den Anforderungspfad von Endbenutzern Ihrer Anwendung über nachgelagerte AWS verwaltete Services analysieren und debuggen. Auf diese Weise können Sie Latenztrends und Fehler identifizieren, die sich auf Ihre Endbenutzer auswirken.

Nachdem Sie die X-Ray-Nachverfolgung von Benutzersitzungen aktiviert haben, fügt CloudWatch RUM einen X-Ray-Ablaufverfolgungs-Header zu zulässigen HTTP-Anforderungen hinzu und zeichnet ein X-Ray-Segment für zulässige HTTP-Anforderungen auf. Sie können dann Ablaufverfolgungen und Segmente aus diesen Benutzersitzungen in den X-Ray- und - CloudWatch Konsolen sehen, einschließlich der X-Ray-Ablaufverfolgungszuordnung.

> **Note**
>
> CloudWatch RUM lässt sich nicht in X-Ray-Samplingregeln integrieren. Wählen Sie stattdessen einen Stichprobenprozentsatz aus, wenn Sie Ihre Anwendung für die Verwendung von CloudWatch RUM einrichten. Von CloudWatch RUM gesendete Ablaufverfolgungen können zusätzliche Kosten verursachen. Weitere Informationen finden Sie unter AWS X-Ray Preise.

Standardmäßig sind clientseitige Ablaufverfolgungen, die von CloudWatch RUM gesendet werden, nicht mit serverseitigen Ablaufverfolgungen verbunden. Um clientseitige Ablaufverfolgungen mit serverseitigen Ablaufverfolgungen zu verbinden, konfigurieren Sie den CloudWatch RUM-Webclient so, dass diesen HTTP-Anforderungen ein X-Ray-Ablaufverfolgungs-Header hinzugefügt wird.

> **Warning**
>
> Die Konfiguration des CloudWatch RUM-Webclients zum Hinzufügen eines X-Ray-Ablaufverfolgungs-Headers zu HTTP-Anforderungen kann dazu führen, dass Cross-Origin Resource Sharing (CORS) fehlschlägt. Um dies zu vermeiden, fügen Sie den `X-Amzn-Trace-Id` HTTP-Header der Liste der zulässigen Header in der CORS-Konfiguration Ihres Downstream-Services hinzu. Wenn Sie API Gateway als Downstream verwenden, finden Sie weitere Informationen unter Aktivieren von CORS für eine REST-API-Ressource. Wir empfehlen dringend, Ihre Anwendung zu testen, bevor Sie einen clientseitigen X-Ray-Ablaufverfolgungs-Header in einer Produktionsumgebung hinzufügen. Weitere Informationen finden Sie in der CloudWatch Dokumentation zum RUM-Webclient.

Weitere Informationen zur realen Benutzerüberwachung in finden Sie CloudWatchunter Verwenden von CloudWatch RUM. Informationen zum Einrichten Ihrer Anwendung für die Verwendung von CloudWatch RUM, einschließlich der Nachverfolgung von Benutzersitzungen mit X-Ray, finden Sie unter Einrichten einer Anwendung für die Verwendung von CloudWatch RUM.

## Debuggen von CloudWatch synthetischen Kanarienvögeln mit X-Ray

CloudWatch Synthetics ist ein vollständig verwalteter Service, mit dem Sie Ihre Endpunkte und APIs mithilfe von Scripted Canaries überwachen können, die 24 Stunden am Tag und einmal pro Minute ausgeführt werden.

Sie können Canary-Skripte anpassen, um nach Änderungen zu suchen in:

- Verfügbarkeit
- Latency
- Transaktionen
- Unterbrochene oder tote Links
- tep-by-step Erledigung von S-Aufgaben
- Fehler beim Laden der Seite
- Ladelatenzen für Komponenten der Benutzeroberfläche
- Komplexe Assistentenabläufe
- Checkout-Abläufe in Ihrer Anwendung

Canarys folgen denselben Routen, agieren und verhalten sich wie Ihre Kunden und überprüfen die Kundenerfahrung kontinuierlich.

Weitere Informationen zum Einrichten von Synthetics-Tests finden Sie unter Erstellen und Verwalten von Canarys mit Synthetics.


Die folgenden Beispiele zeigen häufige Anwendungsfälle für Debugging-Probleme, die von Ihren Synthetics-Canarys ausgelöst werden. Jedes Beispiel zeigt eine wichtige Strategie für das Debuggen mit der Trace-Map oder der X-Ray Analytics-Konsole.

Weitere Informationen zum Lesen und Interagieren mit der Trace-Map finden Sie unter Service-Map anzeigen.

Weitere Informationen zum Lesen und Interagieren mit der X-Ray Analytics-Konsole finden Sie unter Interaktion mit der AWS X-Ray Analytics-Konsole.

Themen

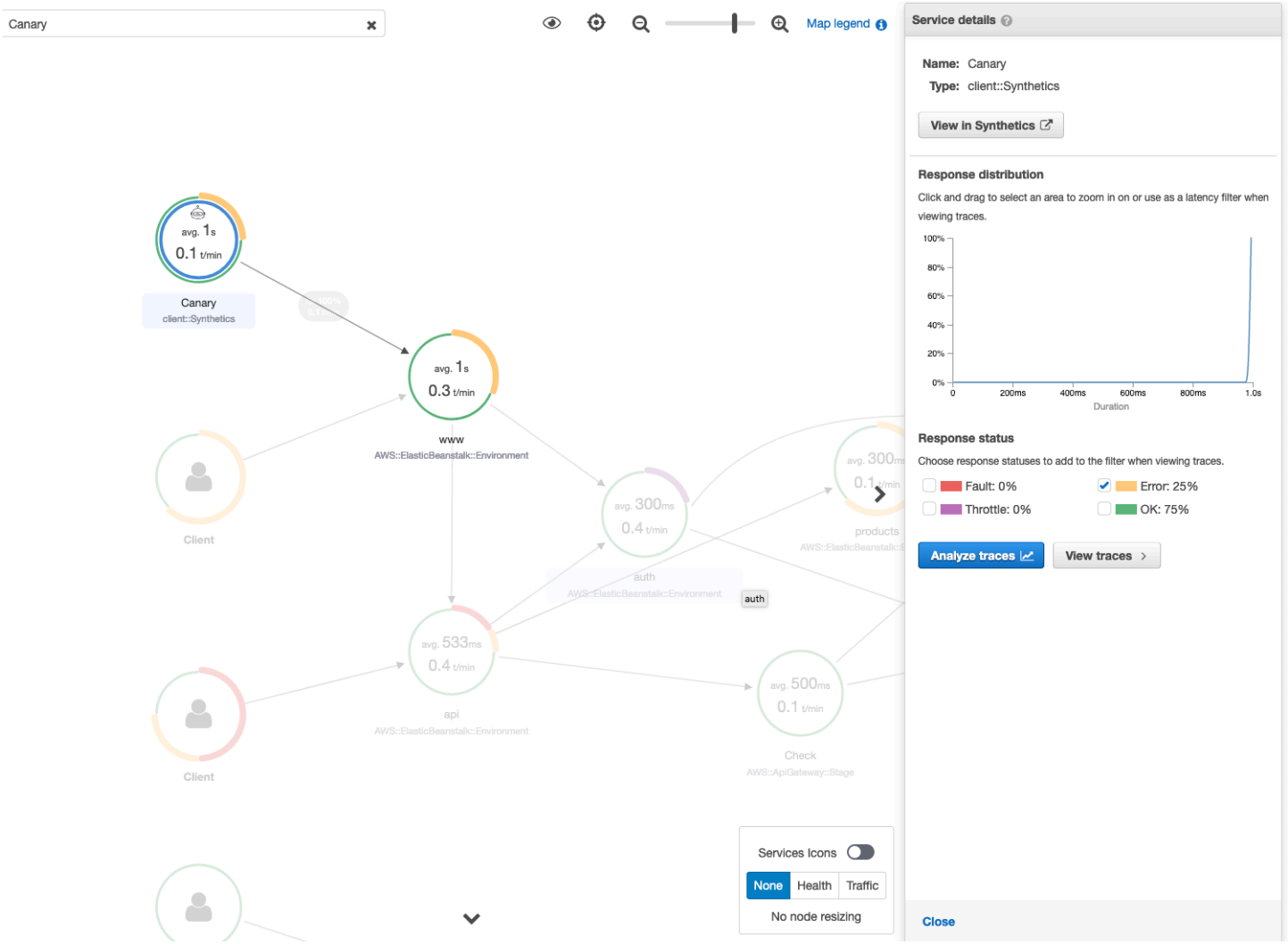
- In der Trace-Map werden Kanarienvögel angezeigt, bei denen häufiger Fehler gemeldet werden
- Verwenden Sie Trace-Detailkarten für einzelne Traces, um sich jede Anfrage im Detail anzusehen

- Die Ursache fortlaufender Fehler in Upstream- und Downstream-Services ermitteln
- Performance-Engpässe und Trends identifizieren
- Latenz und Fehler- oder Ausfallraten vor und nach Änderungen vergleichen
- Die erforderliche Canary-Abdeckung für alle APIs und URLs ermitteln
- Gruppen für die Konzentration auf Synthetics-Tests verwenden

## In der Trace-Map werden Kanarienvögel angezeigt, bei denen häufiger Fehler gemeldet werden

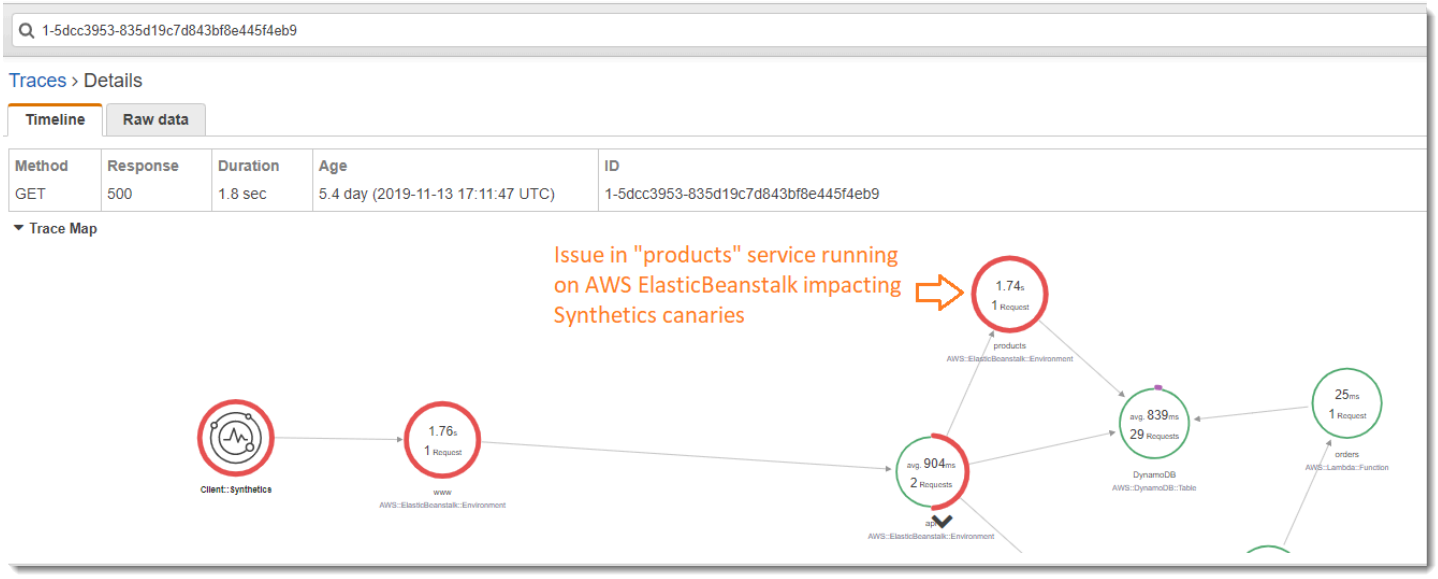
Um zu sehen, welche Kanaren in Ihrer X-Ray-Trace-Map häufiger Fehler, Störungen, Drosselungsraten oder langsame Reaktionszeiten aufweisen, können Sie die Canary-Clientknoten von Synthetics mithilfe des Filters hervorheben. `Client::Synthetic` Weitere Informationen finden Sie unter Verwenden Sie Filterausdrücke. Wenn Sie einen Knoten auswählen, wird die Verteilung der Antwortzeit für die gesamte Anfrage angezeigt. Wenn Sie eine Kante zwischen zwei Knoten auswählen, werden Details zu den Anfragen angezeigt, die diese Verbindung verarbeitet haben. Sie können in Ihrer Trace-Map auch abgeleitete „entfernte“ Knoten für verwandte Downstream-Dienste anzeigen.

Wenn Sie den Synthetics-Knoten auswählen, befindet sich im Seitenbereich die Schaltfläche In Synthetics anzeigen, über die Sie zur Synthetics-Konsole weitergeleitet werden, wo Sie die Canary-Details überprüfen können.




## Verwenden Sie Trace-Detailkarten für einzelne Traces, um sich jede Anfrage im Detail anzusehen

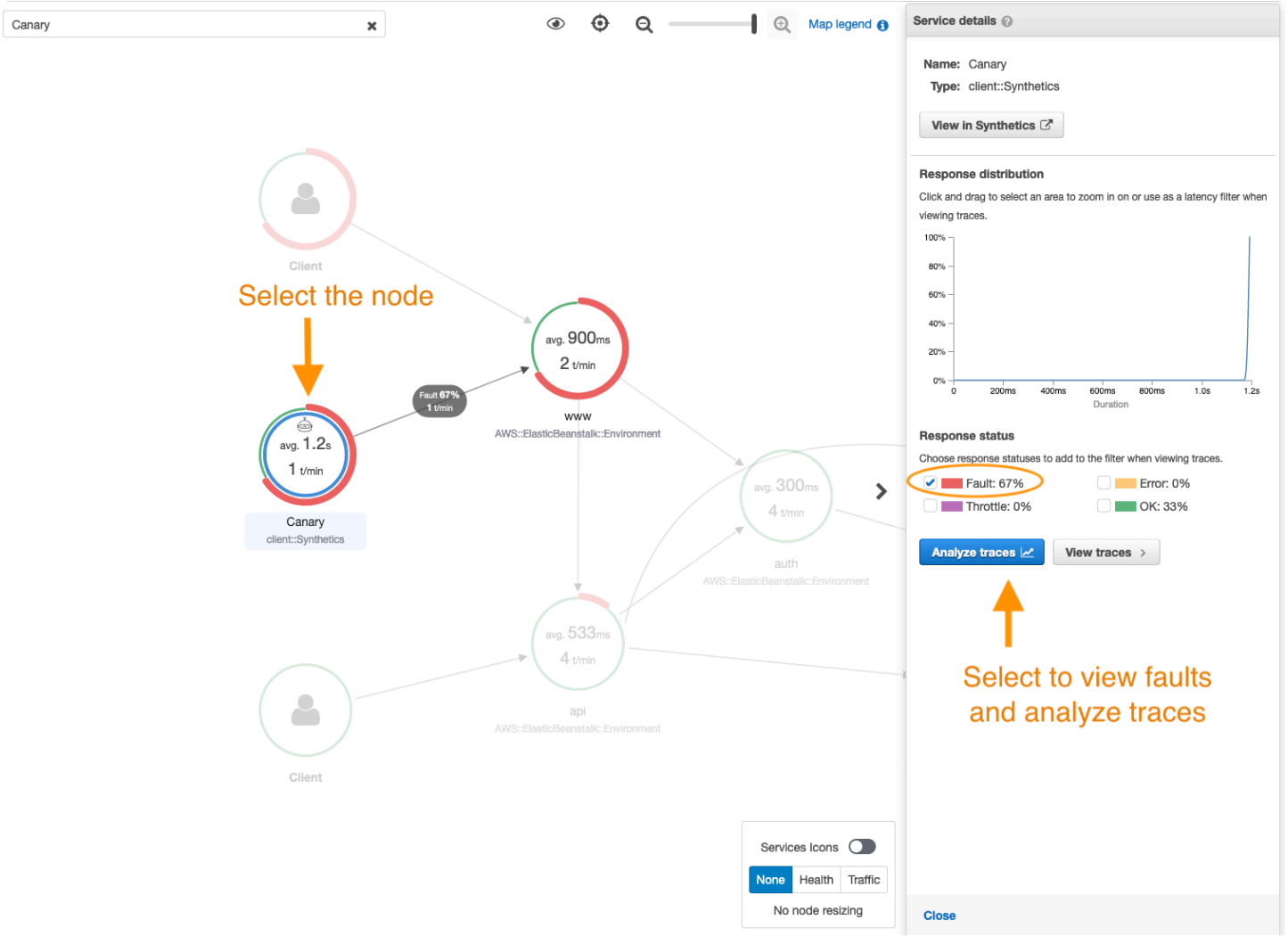
Um zu ermitteln, welcher Service zu der höchsten Latenz führt oder einen Fehler verursacht, rufen Sie die Trace-Details-Map auf, indem Sie die Trace in der Trace-Map auswählen. Die einzelnen Trace-Detailkarten zeigen den end-to-end Pfad einer einzelnen Anfrage an. Mit dieser Option können Sie die aufgerufenen Services verstehen und die Upstream- und Dowstream-Services visualisieren.

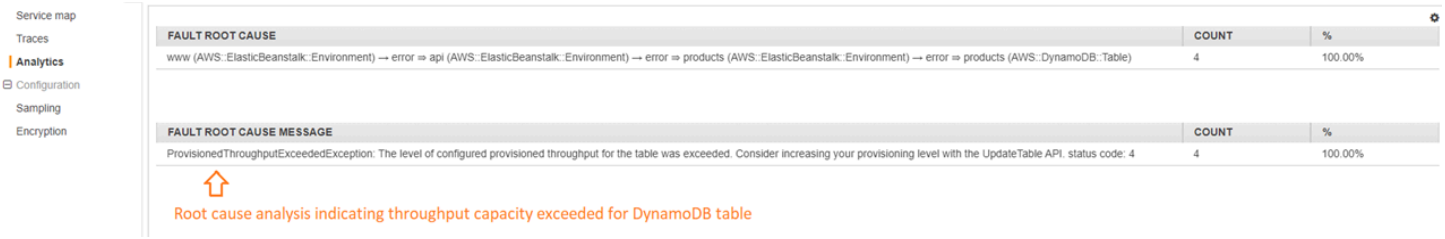


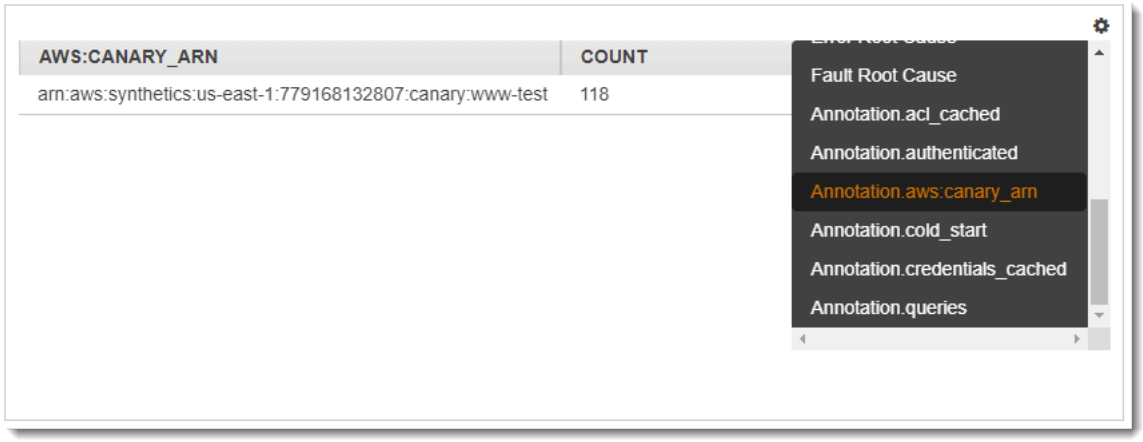

## Die Ursache fortlaufender Fehler in Upstream- und Downstream-Services ermitteln

Sobald Sie einen CloudWatch Alarm für Fehler in einem Synthetics Canary erhalten, verwenden Sie die statistische Modellierung der Trace-Daten in X-Ray, um die wahrscheinliche Ursache des Problems in der X-Ray Analytics-Konsole zu ermitteln. In der Analytics-Konsole werden in der Tabelle mit den Ursachen der Antwortzeiten aufgezeichnete Entitätenpfade angezeigt. X-Ray bestimmt, welcher Pfad in Ihrer Spur die wahrscheinlichste Ursache für die Reaktionszeit ist. Das Format steht für eine Hierarchie von vorgefundenen Entitäten. Dies führt zu einer Reaktionszeit-Ursache.

Das folgende Beispiel zeigt, dass der Synthetics-Test für API „XXX“, der auf API Gateway ausgeführt wird, aufgrund einer Durchsatzkapazitätsausnahme aus der Amazon DynamoDB-Tabelle fehlschlägt.

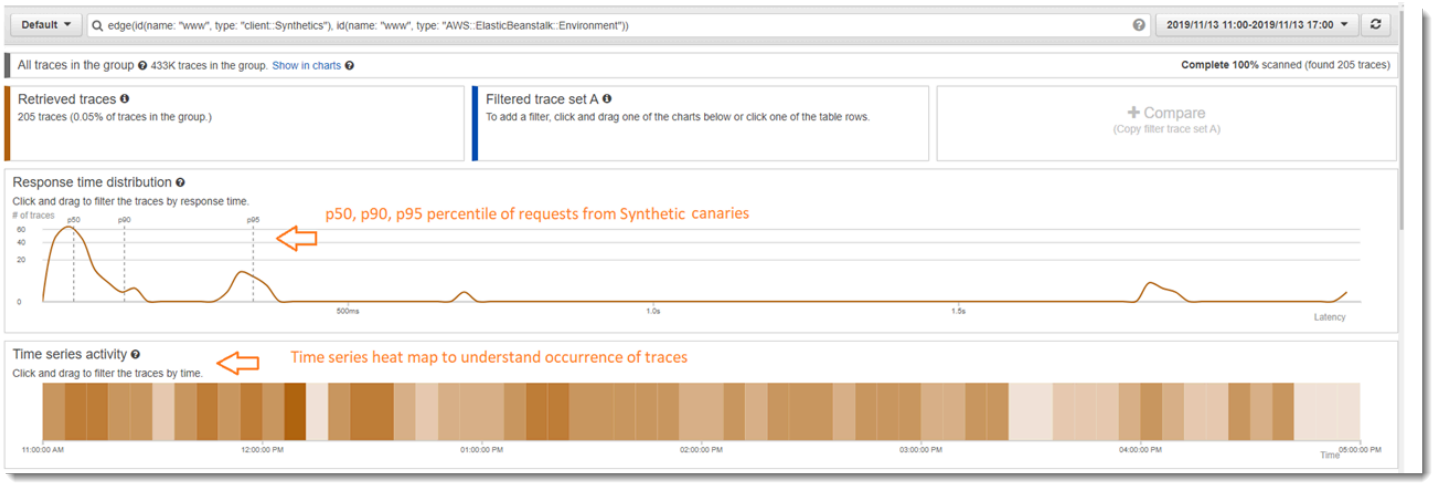

Canary
Map legend
Service details
Name: Canary
Type: client::Synthetics
View in Synthetics
Response distribution
Click and drag to select an area to zoom in on or use as a latency filter when viewing traces.
100%
80%
60%
40%
20%
0%
0
200ms
400ms
600ms
800ms
1.0s
1.2s
Duration
Response status
Choose response statuses to add to the filter when viewing traces.
Fault: 67%
Error: 0%
Throttle: 0%
OK: 33%
Analyze traces
View traces
Select to view faults and analyze traces
Client
Select the node
avg. 900ms
2 t/min
Fault 67%
1 t/min
www
AWS::ElasticBeanstalk::Environment
avg. 1.2s
1 t/min
Canary
client::Synthetics
avg. 300ms
4 t/min
auth
AWS::ElasticBeanstalk::Environment
avg. 533ms
4 t/min
api
AWS::ElasticBeanstalk::Environment
Client
Services Icons
None
Health
Traffic
No node resizing
Close


Service map
Traces
Analytics
Configuration
Sampling
Encryption
FAULT ROOT CAUSE
COUNT
%
www (AWS::ElasticBeanstalk::Environment) ⟶ error ⇒ api (AWS::ElasticBeanstalk::Environment) ⟶ error ⇒ products (AWS::ElasticBeanstalk::Environment) ⟶ error ⇒ products (AWS::DynamoDB::Table)
4
100.00%
FAULT ROOT CAUSE MESSAGE
COUNT
%
ProvisionedThroughputExceededException: The level of configured provisioned throughput for the table was exceeded. Consider increasing your provisioning level with the UpdateTable API. status code: 4
4
100.00%
Root cause analysis indicating throughput capacity exceeded for DynamoDB table



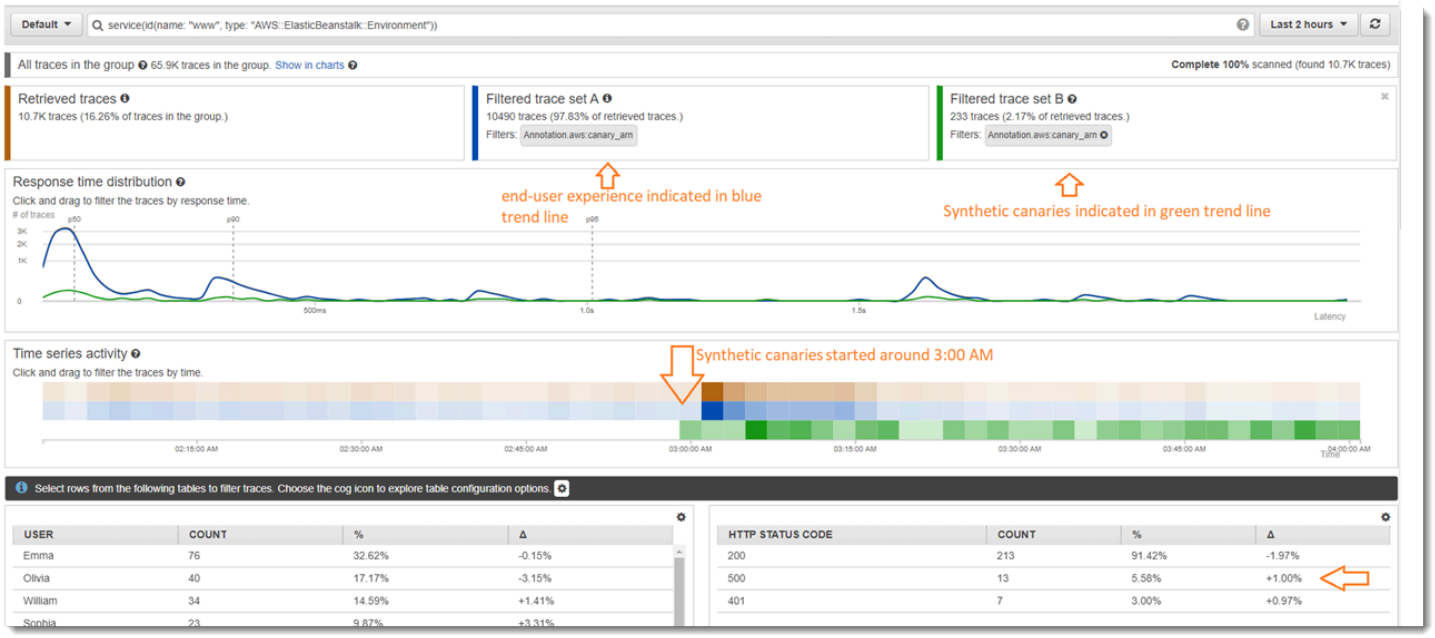

## Performance-Engpässe und Trends identifizieren

Sie können Trends in der Leistung Ihres Endpunkts im Zeitverlauf anzeigen, indem Sie kontinuierlichen Datenverkehr von Ihren Synthetics-Kanaren verwenden, um eine Karte mit Trace-Details über einen bestimmten Zeitraum zu füllen.



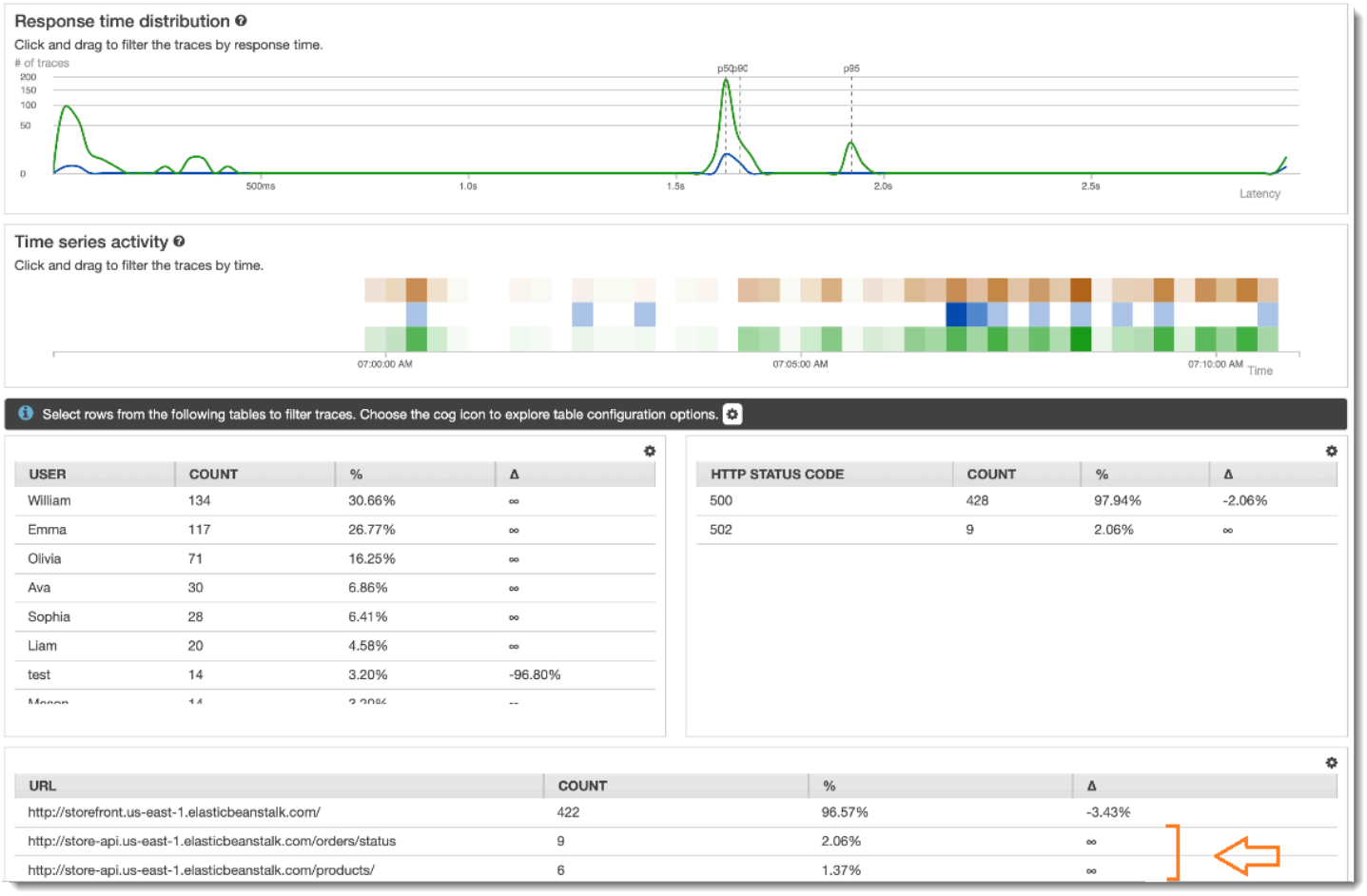

## Latenz und Fehler- oder Ausfallraten vor und nach Änderungen vergleichen

Ermitteln Sie den Zeitpunkt, zu dem eine Änderung eingetreten ist, um diese Änderung mit einer Zunahme von Problemen zu korrelieren, die auf Ihren Kanaren aufgetreten sind. Verwenden Sie die X-Ray Analytics-Konsole, um die Zeitbereiche „Vorher“ und „Nachher“ als unterschiedliche Trace-Sets zu definieren und so eine visuelle Differenzierung in der Verteilung der Antwortzeiten zu erzielen.




## Die erforderliche Canary-Abdeckung für alle APIs und URLs ermitteln

Verwenden Sie X-Ray Analytics, um die Erfahrung von Canarys mit den Benutzern zu vergleichen. Die folgende Benutzeroberfläche zeigt eine blaue Trendlinie für Canarys und eine grüne Linie für Benutzer an. Sie können auch feststellen, dass zwei von drei URLs über keine Canary-Tests verfügen.

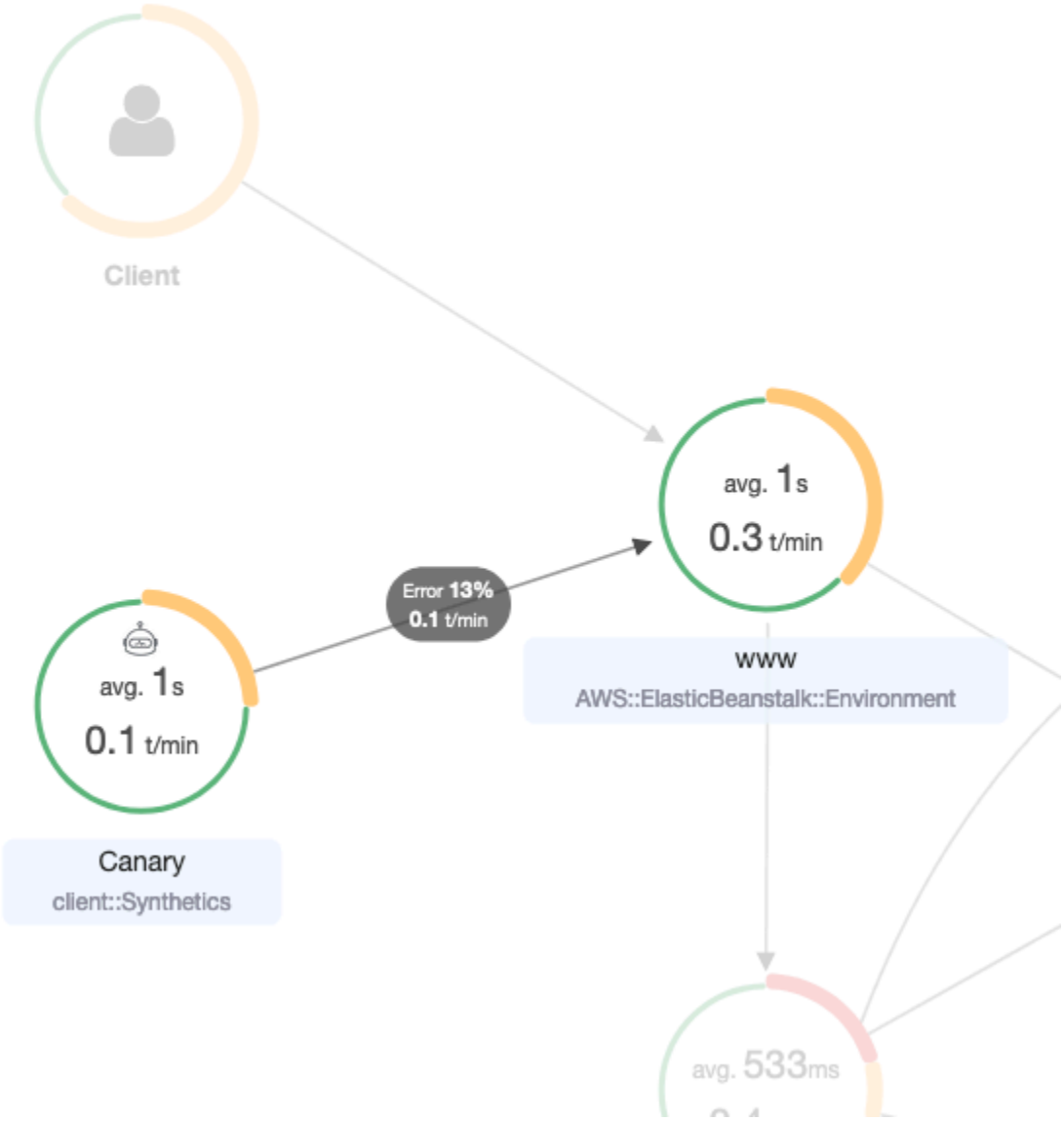



## Gruppen für die Konzentration auf Synthetics-Tests verwenden

Sie können mithilfe eines Filterausdrucks eine X-Ray-Gruppe erstellen, um sich auf eine bestimmte Gruppe von Workflows zu konzentrieren, z. B. einen Synthetics für die Anwendung „www“, auf der AWS Elastic Beanstalk ausgeführt wird. Verwenden Sie die komplexen Schlüsselwörter `service()` und`edge()`, um nach Services und Edges zu filtern. Weitere Informationen finden Sie im Abschnitt Komplexe Schlüsselwörter unterVerwenden Sie Filterausdrücke.

Example Gruppenfilterausdruck

```
"edge(id(name: "www", type: "client::Synthetics"), id(name: "www", type:
 "AWS::ElasticBeanstalk::Environment"))"
```




# Verfolgung von Konfigurationsänderungen der X-Ray-Verschlüsselung mitAWS Config

AWS X-RayintegriertAWS Config, um Konfigurationsänderungen an Ihren X-Ray-Verschlüsselungsressourcen aufzuzeichnen. Sie könnenAWS Config damit X-Ray-Verschlüsselungsressourcen inventarisieren, den X-Ray-Konfigurationsverlauf überprüfen und Benachrichtigungen auf der Grundlage von Ressourcenänderungen senden.

AWS Configunterstützt die Protokollierung der folgenden Ressourcenänderungen für die X-Ray-Verschlüsselung als Ereignisse:

- Konfigurationsänderungen — Änderung oder Hinzufügen eines Verschlüsselungsschlüssels oder Rückkehr zur Standardeinstellung für die X-Ray-Verschlüsselung.

Verwenden Sie die folgenden Anweisungen, um zu erfahren, wie Sie eine grundlegende Verbindung zwischen X-Ray und herstellenAWS Config.

## Auslösen von Lambda-Funktionen

Sie benötigen den ARN einer benutzerdefinierten AWS Lambda-Funktion, um eine benutzerdefinierte AWS Config-Regel generieren zu können. Befolgen Sie diese Anweisungen zum Erstellen einer einfachen Funktion mit Node.js, die basierend auf dem Zustand der `XrayEncryptionConfig`-Ressource als Wert an AWS Config zurückgibt, ob sie konform oder nicht konform ist.

Um eine Lambda-Funktion mit einem AWS::XrayEncryptionConfig Change-Trigger zu erstellen

1. Öffnen Sie die Lambda-Konsole. Wählen Sie Create function (Funktion erstellen).
2. Wählen Sie Blueprints aus, und filtern Sie dann die Blueprints-Bibliothek nach dem config-rule-change-triggeredBlueprint. Klicken Sie auf den Link im Namen der Vorlage oder wählen Sie Configure (Konfigurieren), um fortzufahren.
3. Definieren Sie die folgenden Felder zum Konfigurieren der Vorlage:
   - Geben Sie im Feld Name einen Namen ein.
   - Für die Option Role (Rolle) wählen Sie Create new role from template(s) (Neue Rolle aus Vorlage[n] erstellen) aus.
   - Bei Role Name geben Sie einen Namen ein.
   - Wählen Sie für Policy Templates (Richtlinienvorlagen) die Option AWS ConfigRules permissions (-Regelberechtigungen) aus.
4. Klicken Sie auf Create function (Funktion erstellen), um Ihre Funktion zu erstellen und in der AWS Lambda-Konsole anzuzeigen.
5. Bearbeiten Sie Ihren Funktionscode, indem Sie `AWS::EC2::Instance` durch `AWS::XrayEncryptionConfig` ersetzen. Sie können auch das Beschreibungsfeld aktualisieren, damit die Änderung wirksam wird.

   Standardcode

```
    if (configurationItem.resourceType !== 'AWS::EC2::Instance') {
        return 'NOT_APPLICABLE';
```

```
    } else if (ruleParameters.desiredInstanceType ===
configurationItem.configuration.instanceType) {
        return 'COMPLIANT';
    }
        return 'NON_COMPLIANT';
```

Aktualisierter Code

```
    if (configurationItem.resourceType !== 'AWS::XRay::EncryptionConfig') {
        return 'NOT_APPLICABLE';
    } else if (ruleParameters.desiredInstanceType ===
configurationItem.configuration.instanceType) {
        return 'COMPLIANT';
    }
        return 'NON_COMPLIANT';
```

6. Fügen Sie Ihrer Ausführungsrolle in IAM Folgendes hinzu, um auf X-Ray zuzugreifen. Diese Berechtigungen erlauben schreibgeschützten Zugriff auf Ihre X-Ray-Ressourcen. Wenn der Zugriff auf die entsprechenden Ressourcen nicht bereitgestellt wird, wird bei der Auswertung der mit der Regel verknüpftenAWS Config Lambda-Funktion eine Meldung angezeigt, dass der Geltungsbereich abgelaufen ist.

```
    {
        "Sid": "Stmt1529350291539",
        "Action": [
            "xray:GetEncryptionConfig"
        ],
        "Effect": "Allow",
        "Resource": "*"
     }
```

## Erstellen einer benutzerdefinierten AWS Config-Regel für X-Ray

Wenn die Lambda-Funktion erstellt wird, notieren Sie sich den ARN der Funktion und gehen Sie zurAWS Config Konsole, um Ihre benutzerdefinierte Regel zu erstellen.

Zur Erstellung einerAWS Config Regel für X-Ray-Ablaufverfolgung

1. Öffnen Sie die Seite Rules (Regeln) der AWS Config-Konsole.

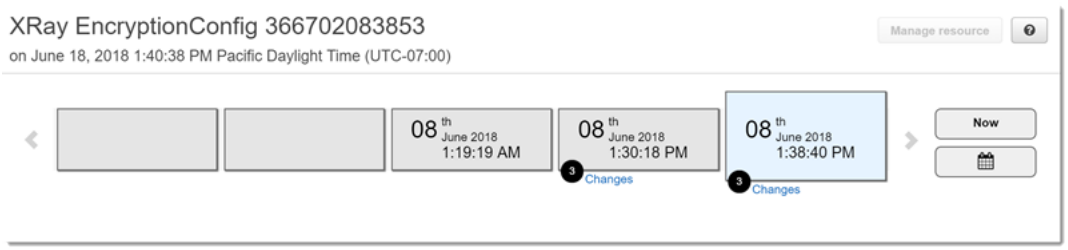
2. Klicken Sie auf Add Rule (Regel hinzufügen) und danach auf Add custom rule (Benutzerdefinierte Regel hinzufügen).
3. Geben Sie in AWS LambdaFunction ARN den ARN ein, der der Lambda-Funktion zugeordnet ist, die Sie verwenden möchten.
4. Wählen Sie, welche Art von Auslöser festgelegt werden soll:
   - Konfigurationsänderungen —AWS Config löst die Bewertung aus, wenn sich eine Ressource, die dem Geltungsbereich der Regel entspricht, in der Konfiguration ändert. Die Auswertung wird durchgeführt, nachdem AWS Config eine Änderungsbenachrichtigung wegen eines Konfigurationselements sendet.
   - Periodisch —AWS Config führt die Evaluierungen für die Regel in einer von Ihnen gewählten Häufigkeit durch (z. B. alle 24 Stunden).
5. Wählen Sie EncryptionConfigim X-Ray-Bereich als Ressourcentyp die Option aus.
6. Wählen Sie Save (Speichern) aus.

Die AWS Config-Konsole beginnt sofort mit der Auswertung der Compliance der Regel. Die Auswertung kann mehrere Minuten in Anspruch nehmen.

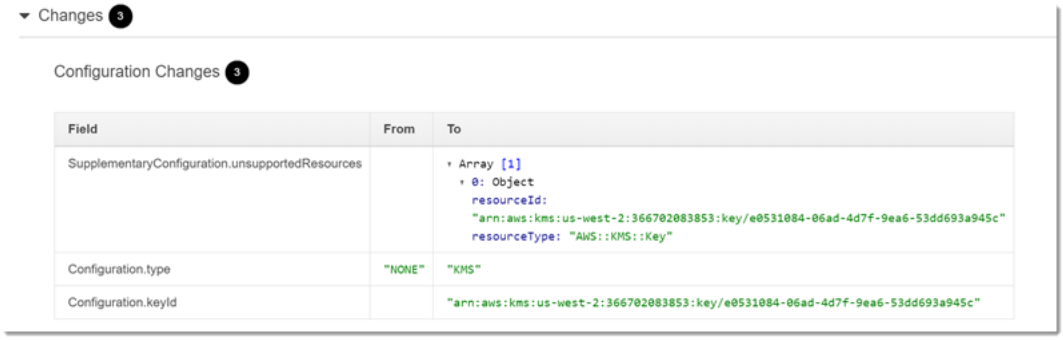
Wenn diese Regel konform ist, kann AWS Config mit dem Kompilieren eines Prüfungsverlaufs beginnen. AWS Config erfasst Ressourcenänderungen in Form einer Timeline. Für jede Änderung in der Timeline der Ereignisse erstellt AWS Config eine Tabelle im Format "Von/Zu", um aufzuzeigen, was sich in der JSON-Darstellung des Verschlüsselungsschlüssels geändert hat. Die beiden damit verbundenen Feldänderungen EncryptionConfig sind`Configuration.type` und`Configuration.keyID`.

## Beispielergebnisse

Im Folgenden sehen Sie ein Beispiel einer AWS Config-Timeline mit Änderungen an bestimmten Tagen und Uhrzeiten.



Nachfolgend finden Sie ein Beispiel eines AWS Config-Änderungseintrags. Das Format "Von/Zu" veranschaulicht, was sich geändert hat. Dieses Beispiel zeigt, dass die Standardeinstellungen für die X-Ray-Verschlüsselung in einen definierten Verschlüsselungsschlüssel geändert wurden.




## Amazon-SNS-Benachrichtigungen

Zur Benachrichtigung über Konfigurationsänderungen stellen, stellen von Amazon-SNS-BenachrichtigungenAWS Config auf die Zustellung Amazon SNS Amazon-SNS-Benachrichtigungen. Weitere Informationen finden Sie unter Überwachen von AWS Config-Ressourcenänderungen per E-Mail.

# Amazon Elastic Compute Cloud undAWS X-Ray

Sie können den X-Ray -Daemon in einer Amazon EC2 EC2-Instance mit einem Benutzerdatenskript installieren und ausführen. Detaillierte Anweisungen finden Sie unter Ausführen des X-Ray-Daemons auf Amazon EC2.

Verwenden Sie ein Instance-Profil zum Hochladen von Ablaufverfolgungsdaten auf X-Ray. Weitere Informationen finden Sie unter Dem Daemon die Erlaubnis geben, Daten an X-Ray zu senden.

# AWS Elastic Beanstalk und AWS X-Ray

AWS Elastic BeanstalkZu den Plattformen gehört der X-Ray-Daemon. Sie haben folgende MöglichkeitenAusführen des Daemonsindem Sie eine Option in der Elastic Beanstalk-Konsole oder mit einer Konfigurationsdatei festlegen.

Auf der Java SE-Plattform können Sie eine Buildfile-Datei zur Erstellung Ihrer Anwendung mit einer Maven- oder Gradle-Instance verwenden. Das X-Ray-SDK SDK for Java undAWS SDK for JavaSie sind von Maven verfügbar, sodass Sie nur Ihren Anwendungscode bereitstellen und die Anwendung in einer einzelnen Instanz erstellen können, damit Sie nicht alle Abhängigkeiten bündeln und hochladen müssen.

Sie können mithilfe von -Umgebungseigenschaften von Elastic Beanstalk das X-Ray SDK konfigurieren. Das Verfahren, mit dem Elastic Beanstalk Umgebungseigenschaften an Ihre Anwendung übergibt, variiert je nach Plattform. Verwenden Sie die Umgebungsvariablen des X-Ray SDK oder Systemeigenschaften, je nach Ihrer Plattform.

- Die Node.js Plattform— Verwendung vonUmgebungsvariablen
- Java SE-Plattform— Verwendung vonUmgebungsvariablen
- Tomcat-Plattform— Verwendung vonSystemeigenschaften

Weitere Informationen finden Sie unter Konfigurieren der AWS X-Ray-Fehlersuche im Entwicklerhandbuch für AWS Elastic Beanstalk.

# Elastic Load Balancing und AWS X-Ray

Elastic Load Balancing Application Load Balancer fügen eingehenden HTTP-Anforderungen in einem Header mit dem Namen eine Ablaufverfolgungs-ID hinzu`X-Amzn-Trace-Id`.

```
X-Amzn-Trace-Id: Root=1-5759e988-bd862e3fe1be46a994272793
```

X-Ray-Trace-ID-Format

Ein X-Ray `trace_id` besteht aus drei Zahlen, die durch Bindestriche getrennt sind. Beispiel: `1-58406520-a006649127e371903a2de979` Dies umfasst:

- Die Versionsnummer, 1.
- Die Uhrzeit der ursprünglichen Anforderung in Unix-Epochenzeit mit 8 Hexadezimalziffern.

  Beispielsweise beträgt 10:00 Uhr am 1. Dezember 2016 PST in der Epochenzeit `1480615200` Sekunden oder `58406520` Hexadezimalziffern.
- Eine global eindeutige 96-Bit-Kennung für die Ablaufverfolgung in 24 Hexadezimalziffern.

Load Balancer senden keine Daten an X-Ray und werden nicht als Knoten auf Ihrer Service-Übersicht angezeigt.

Weitere Informationen finden Sie unter Anforderungsnachverfolgung für Ihren Application Load Balancer im Elastic Load Balancing-Entwicklerhandbuch.

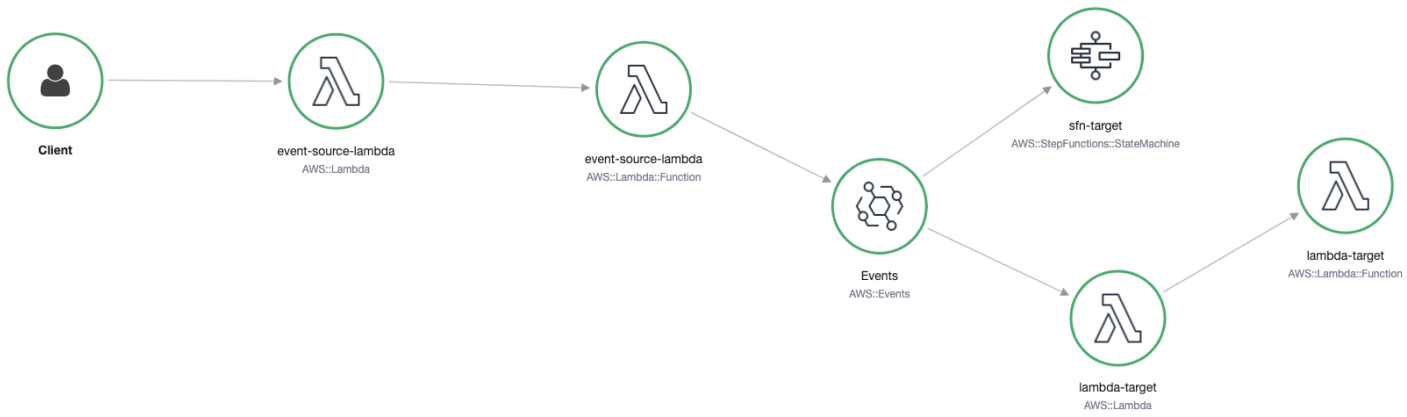
# Amazon EventBridge und AWS X-Ray

AWS X-Ray lässt sich in Amazon integrieren EventBridge , um Ereignisse nachzuverfolgen, die durchlaufen wurden EventBridge. Wenn ein Dienst, der mit dem X-Ray SDK instrumentiert ist, Ereignisse sendet EventBridge, wird der Trace-Kontext an nachgeschaltete Ereignisziele innerhalb des Tracing-Headers weitergegeben. Das X-Ray SDK nimmt den Tracing-Header automatisch auf und wendet ihn auf alle nachfolgenden Instrumente an. Diese Kontinuität ermöglicht es Benutzern, alle nachgelagerten Dienste zu verfolgen, zu analysieren und zu debuggen, und bietet einen vollständigeren Überblick über ihr System.

Weitere Informationen finden Sie unter EventBridge X-Ray Integration im EventBridge Benutzerhandbuch.

## Quelle und Ziele auf der X-Ray-Servicekarte anzeigen

Die X-Ray-Trace-Map zeigt einen EventBridge Ereignisknoten an, der Quell- und Zieldienste verbindet. Weitere Informationen finden Sie unter Verwenden Sie die X-Ray-Trace-Map. Im Folgenden finden Sie ein Beispiel für eine Trace-Map:




## Verbreiten Sie den Trace-Kontext an die Ereignisziele

Das X-Ray SDK ermöglicht es der EventBridge Ereignisquelle, den Trace-Kontext an nachgeschaltete Ereignisziele weiterzugeben. Die folgenden sprachspezifischen Beispiele demonstrieren den Aufruf EventBridge von einer Lambda-Funktion aus, für die die aktive Ablaufverfolgung aktiviert ist:

Java

Fügen Sie die erforderlichen Abhängigkeiten für X-Ray hinzu:

- AWS X-Ray SDK für Java
- AWS X-Ray Rekorder-SDK SDK for Java

```
package example;

import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.lambda.runtime.RequestHandler;
import com.amazonaws.services.lambda.runtime.events.SQSEvent;
import com.amazonaws.xray.AWSXRay;
import com.amazonaws.services.eventbridge.AmazonEventBridge;
import com.amazonaws.services.eventbridge.AmazonEventBridgeClientBuilder;
import com.amazonaws.services.eventbridge.model.PutEventsRequest;
import com.amazonaws.services.eventbridge.model.PutEventsRequestEntry;
import com.amazonaws.services.eventbridge.model.PutEventsResult;
import com.amazonaws.services.eventbridge.model.PutEventsResultEntry;
import com.amazonaws.xray.handlers.TracingHandler;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import java.lang.StringBuilder;
import java.util.Map;
import java.util.List;
import java.util.Date;
import java.util.Collections;

/*
   Add the necessary dependencies for XRay:
   https://mvnrepository.com/artifact/com.amazonaws/aws-java-sdk-xray
   https://mvnrepository.com/artifact/com.amazonaws/aws-xray-recorder-sdk-aws-sdk
*/
public class Handler implements RequestHandler<SQSEvent, String>{
  private static final Logger logger = LoggerFactory.getLogger(Handler.class);

  /*
    build EventBridge client
  */
  private static final AmazonEventBridge eventsClient =
 AmazonEventBridgeClientBuilder
         .standard()
         // instrument the EventBridge client with the XRay Tracing Handler.
         // the AWSXRay globalRecorder will retrieve the tracing-context
```

```
          // from the lambda function and inject it into the HTTP header.
          // be sure to enable 'active tracing' on the lambda function.
          .withRequestHandlers(new TracingHandler(AWSXRay.getGlobalRecorder()))
          .build();

  @Override
  public String handleRequest(SQSEvent event, Context context)
  {
    PutEventsRequestEntry putEventsRequestEntry0 = new PutEventsRequestEntry();
    putEventsRequestEntry0.setTime(new Date());
    putEventsRequestEntry0.setSource("my-lambda-function");
    putEventsRequestEntry0.setDetailType("my-lambda-event");
    putEventsRequestEntry0.setDetail("{\"lambda-source\":\"sqs\"}");
    PutEventsRequest putEventsRequest = new PutEventsRequest();
    putEventsRequest.setEntries(Collections.singletonList(putEventsRequestEntry0));
    // send the event(s) to EventBridge
    PutEventsResult putEventsResult = eventsClient.putEvents(putEventsRequest);
    try {
      logger.info("Put Events Result: {}", putEventsResult);
    } catch(Exception e) {
      e.getStackTrace();
    }
    return "success";
  }
}
```

### Python

Fügen Sie Ihrer Datei requirements.txt die folgende Abhängigkeit hinzu:

```
aws-xray-sdk==2.4.3
```

```
import boto3
from aws_xray_sdk.core import xray_recorder
from aws_xray_sdk.core import patch_all

# apply the XRay handler to all clients.
patch_all()

client = boto3.client('events')

def lambda_handler(event, context):
    response = client.put_events(
```

```
        Entries=[
            {
                'Source': 'foo',
                'DetailType': 'foo',
                'Detail': '{\"foo\": \"foo\"}'
            },
        ]
    )
    return response
```

Go

```
package main

import (
  "context"
  "github.com/aws/aws-lambda-go/lambda"
  "github.com/aws/aws-lambda-go/events"
  "github.com/aws/aws-sdk-go/aws/session"
  "github.com/aws/aws-xray-sdk-go/xray"
  "github.com/aws/aws-sdk-go/service/eventbridge"
  "fmt"
)

var client = eventbridge.New(session.New())


func main() {
 //Wrap the eventbridge client in the AWS XRay tracer
  xray.AWS(client.Client)
  lambda.Start(handleRequest)
}

func handleRequest(ctx context.Context, event events.SQSEvent) (string, error) {
  _, err := callEventBridge(ctx)
  if err != nil {
    return "ERROR", err
  }
  return "success", nil
}


func callEventBridge(ctx context.Context) (string, error) {
```

```
    entries := make([]*eventbridge.PutEventsRequestEntry, 1)
    detail := "{ \"foo\": \"foo\"}"
    detailType := "foo"
    source := "foo"
    entries[0] = &eventbridge.PutEventsRequestEntry{
        Detail: &detail,
        DetailType: &detailType,
        Source: &source,
    }

  input := &eventbridge.PutEventsInput{
     Entries: entries,
  }

  // Example sending a request using the PutEventsRequest method.
  resp, err := client.PutEventsWithContext(ctx, input)

  success := "yes"
  if err == nil { // resp is now filled
      success = "no"
      fmt.Println(resp)
  }
  return success, err
}
```

### Node.js

```
const AWSXRay = require('aws-xray-sdk')
//Wrap the aws-sdk client in the AWS XRay tracer
const AWS = AWSXRay.captureAWS(require('aws-sdk'))
const eventBridge = new AWS.EventBridge()

exports.handler = async (event) => {

  let myDetail = { "name": "Alice" }

  const myEvent = {
    Entries: [{
      Detail: JSON.stringify({ myDetail }),
      DetailType: 'myDetailType',
      Source: 'myApplication',
      Time: new Date
    }]
```

```
  }

  // Send to EventBridge
  const result = await eventBridge.putEvents(myEvent).promise()

  // Log the result
  console.log('Result: ', JSON.stringify(result, null, 2))

}
```

C#

Fügen Sie die folgenden X-Ray-Pakete zu Ihren C#-Abhängigkeiten hinzu:

```
<PackageReference Include="AWSXRayRecorder.Core" Version="2.6.2" />
<PackageReference Include="AWSXRayRecorder.Handlers.AwsSdk" Version="2.7.2" />
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Amazon;
using Amazon.Util;
using Amazon.Lambda;
using Amazon.Lambda.Model;
using Amazon.Lambda.Core;
using Amazon.EventBridge;
using Amazon.EventBridge.Model;
using Amazon.Lambda.SQSEvents;
using Amazon.XRay.Recorder.Core;
using Amazon.XRay.Recorder.Handlers.AwsSdk;
using Newtonsoft.Json;
using Newtonsoft.Json.Serialization;

[assembly:
 LambdaSerializer(typeof(Amazon.Lambda.Serialization.Json.JsonSerializer))]

namespace blankCsharp
{
  public class Function
  {
    private static AmazonEventBridgeClient eventClient;
```

```
    static Function() {
      initialize();
    }

    static async void initialize() {
      //Wrap the AWS SDK clients in the AWS XRay tracer
      AWSSDKHandler.RegisterXRayForAllServices();
      eventClient = new AmazonEventBridgeClient();
    }

    public async Task<PutEventsResponse> FunctionHandler(SQSEvent invocationEvent,
 ILambdaContext context)
    {
      PutEventsResponse response;
      try
      {
        response = await callEventBridge();
      }
      catch (AmazonLambdaException ex)
      {
        throw ex;
      }

      return response;
    }

    public static async Task<PutEventsResponse> callEventBridge()
    {
      var request = new PutEventsRequest();
      var entry = new PutEventsRequestEntry();
      entry.DetailType = "foo";
      entry.Source = "foo";
      entry.Detail = "{\"instance_id\":\"A\"}";
      List<PutEventsRequestEntry> entries = new List<PutEventsRequestEntry>();
      entries.Add(entry);
      request.Entries = entries;
      var response = await eventClient.PutEventsAsync(request);
      return response;
    }
  }
}
```

# AWS Lambda und AWS X-Ray

Sie können verwenden AWS X-Ray , um Ihre AWS Lambda Funktionen zu verfolgen. Lambda führt den X-Ray-Daemon aus und zeichnet ein Segment mit Details zum Aufrufen und Ausführen der Funktion auf. Zur weiteren Instrumentierung können Sie das X-Ray-SDK mit Ihrer Funktion bündeln, um ausgehende Aufrufe aufzuzeichnen und Anmerkungen und Metadaten hinzuzufügen.

Wenn Ihre Lambda-Funktion von einem anderen instrumentierten Service aufgerufen wird, verfolgt Lambda Anforderungen, die bereits ohne zusätzliche Konfiguration erfasst wurden. Der Upstream-Service kann eine instrumentierte Webanwendung oder eine andere Lambda-Funktion sein. Ihr Service kann die Funktion direkt mit einem instrumentierten AWS SDK-Client oder durch Aufrufen einer API Gateway-API mit einem instrumentierten HTTP-Client aufrufen.

AWS X-Ray unterstützt die Nachverfolgung von ereignisgesteuerten Anwendungen mit AWS Lambda und Amazon SQS . Verwenden Sie die - CloudWatch Konsole, um eine verbundene Ansicht jeder Anfrage anzuzeigen, während sie mit Amazon SQS in die Warteschlange gestellt und von einer nachgelagerten Lambda-Funktion verarbeitet wird. Ablaufverfolgungen von Produzenten von Upstream-Nachrichten werden automatisch mit Ablaufverfolgungen von Downstream-Lambda-Konsumentenknoten verknüpft, wodurch eine end-to-end Ansicht der Anwendung erstellt wird. Weitere Informationen finden Sie unter Nachverfolgung von ereignisgesteuerten Anwendungen.

> **Note**
>
> Wenn Sie Ablaufverfolgungen für eine nachgelagerte Lambda-Funktion aktiviert haben, müssen Sie auch Ablaufverfolgungen für die Lambda-Stammfunktion aktiviert haben, die die nachgelagerte Funktion aufruft, damit die nachgelagerte Funktion Ablaufverfolgungen generiert.

Wenn Ihre Lambda-Funktion nach einem Zeitplan ausgeführt wird oder von einem Service aufgerufen wird, der nicht instrumentiert ist, können Sie Lambda so konfigurieren, dass Aufrufe mit aktiver Ablaufverfolgung erfasst und aufgezeichnet werden.

So konfigurieren Sie die X-Ray-Integration für eine - AWS Lambda Funktion

1. Öffnen Sie die AWS Lambda -Konsole.
2. Wählen Sie in der linken Navigationsleiste Funktionen aus.
3. Wählen Sie Ihre Funktion.

4. Scrollen Sie auf der Registerkarte Konfiguration nach unten zur Karte Zusätzliche Überwachungstools. Sie finden diese Karte auch, indem Sie im linken Navigationsbereich die Option Überwachungstools und Betriebswerkzeuge auswählen.
5. Wählen Sie Bearbeiten aus.
6. Aktivieren Sie unter AWS X-Ray die Option Aktive Ablaufverfolgung.

Auf Laufzeiten mit einem entsprechenden X-Ray-SDK führt Lambda auch den X-Ray-Daemon aus.

X-Ray-SDKs auf Lambda

- X-Ray SDK for Go – Go 1.7 und neuere Laufzeiten
- X-Ray SDK for Java – Java-8-Laufzeit
- X-Ray SDK for Node.js – Node.js 4.3 und neuere Laufzeiten
- X-Ray SDK for Python – Python 2.7, Python 3.6 und neuere Laufzeiten
- X-Ray SDK for .NET – .NET Core 2.0 und neuere Laufzeiten

Um das X-Ray-SDK auf Lambda zu verwenden, bündeln Sie es jedes Mal mit Ihrem Funktionscode, wenn Sie eine neue Version erstellen. Sie können Ihre Lambda-Funktionen mit denselben Methoden instrumentieren, die Sie verwenden, um Anwendungen zu instrumentieren, die auf anderen -Services ausgeführt werden. Der Hauptunterschied besteht darin, dass Sie nicht das SDK dazu verwenden, eingehende Anforderungen zu instrumentieren, Samplingentscheidungen zu treffen und Segmente zu erstellen.

Der andere Unterschied zwischen der Instrumentierung von Lambda-Funktionen und Webanwendungen besteht darin, dass das Segment, das Lambda erstellt und an X-Ray sendet, nicht durch Ihren Funktionscode geändert werden kann. Sie können Untersegmente erstellen und Anmerkungen und Metadaten aufzeichnen. Sie können dem übergeordneten Segment jedoch keine Anmerkungen und Metadaten hinzufügen.

Weitere Informationen finden Sie unter Verwenden von AWS X-Ray im AWS Lambda - Entwicklerhandbuch.

# Amazon SNS und AWS X-Ray

Sie können AWS X-Ray mit Amazon Simple Notification Service (Amazon SNS) verwenden, um Anfragen zu verfolgen und zu analysieren, während sie durch Ihre SNS-Themen zu Ihren von SNS

unterstützten Abonnementservices gelangen. Verwenden Sie die X-Ray-Nachverfolgung mit Amazon SNS, um Latenzen in Ihren Nachrichten und deren Backend-Services zu analysieren, z. B. wie lange eine Anfrage in einem Thema verbringt und wie lange es dauert, die Nachricht an jedes Abonnement des Themas zuzustellen. Amazon SNS unterstützt X-Ray-Tracing für Standard- und FIFO-Themen.

Wenn Sie in einem Amazon SNS-Thema von einem Service veröffentlichen, der bereits mit X-Ray instrumentiert ist, übergibt Amazon SNS den Ablaufverfolgungskontext vom Publisher an Abonnenten. Darüber hinaus können Sie die aktive Nachverfolgung aktivieren, um Segmentdaten zu Ihren Amazon SNS-Abonnements für Nachrichten, die von einem instrumentierten SNS-Client veröffentlicht werden, an X-Ray zu senden. Aktivieren Sie die aktive Nachverfolgung für ein Amazon SNS-Thema mithilfe der Amazon SNS-Konsole oder mithilfe der Amazon SNS-API oder -CLI. Weitere Informationen zur Instrumentierung Ihrer SNS-Clients finden Sie unter Instrumentierung Ihrer Anwendung.

## Konfigurieren der aktiven Amazon SNS-Nachverfolgung

Sie können die Amazon SNS-Konsole oder die AWS CLI oder das SDK verwenden, um die aktive Amazon SNS-Nachverfolgung zu konfigurieren.

Wenn Sie die Amazon SNS-Konsole verwenden, versucht Amazon SNS, die erforderlichen Berechtigungen für SNS zum Aufrufen von X-Ray zu erstellen. Der Versuch kann abgelehnt werden, wenn Sie nicht über ausreichende Berechtigungen zum Ändern von X-Ray-Ressourcenrichtlinien verfügen. Weitere Informationen zu diesen Berechtigungen finden Sie unter Identitäts- und Zugriffsverwaltung in Amazon SNS und Beispielfälle für die Amazon SNS-Zugriffskontrolle im Entwicklerhandbuch für Amazon Simple Notification Service. Weitere Informationen zum Aktivieren der aktiven Ablaufverfolgung mithilfe der Amazon SNS-Konsole finden Sie unter Aktivieren der aktiven Ablaufverfolgung für ein Amazon SNS-Thema im Amazon-Simple-Notification-Service-Entwicklerhandbuch.

Wenn Sie die AWS CLI oder das SDK verwenden, um die aktive Ablaufverfolgung zu aktivieren, müssen Sie die Berechtigungen mithilfe ressourcenbasierter Richtlinien manuell konfigurieren. Verwenden Sie `PutResourcePolicy`, um X-Ray mit der erforderlichen ressourcenbasierten Richtlinie zu konfigurieren, damit Amazon SNS Ablaufverfolgungen an X-Ray senden kann.

Example Beispiel für eine ressourcenbasierte X-Ray-Richtlinie für aktives Tracing in Amazon SNS

Dieses Beispielrichtliniendokument gibt die Berechtigungen an, die Amazon SNS zum Senden von Ablaufverfolgungsdaten an X-Ray benötigt:

```
{
```

```
    Version: "2012-10-17",
    Statement: [
      {
        Sid: "SNSAccess",
        Effect: Allow,
        Principal: {
          Service: "sns.amazonaws.com",
        },
        Action: [
          "xray:PutTraceSegments",
          "xray:GetSamplingRules",
          "xray:GetSamplingTargets"
        ],
        Resource: "*",
        Condition: {
          StringEquals: {
            "aws:SourceAccount": "account-id"
          },
          StringLike: {
            "aws:SourceArn": "arn:partition:sns:region:account-id:topic-name"
          }
        }
      }
    ]
  }
```

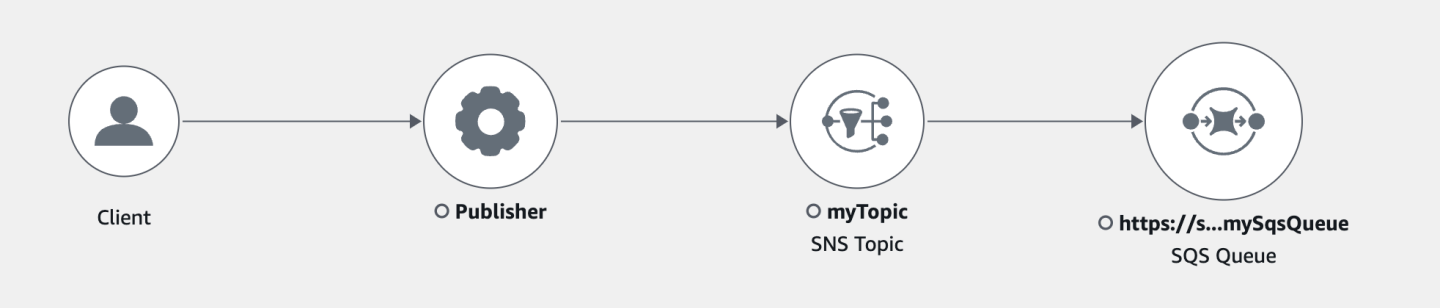
Verwenden Sie die CLI, um eine ressourcenbasierte Richtlinie zu erstellen, die Amazon SNS Berechtigungen zum Senden von Ablaufverfolgungsdaten an X-Ray erteilt:

```
aws xray put-resource-policy --policy-name MyResourcePolicy --policy-document
 '{ "Version": "2012-10-17", "Statement": [ { "Sid": "SNSAccess", "Effect": "Allow",
 "Principal": { "Service": "sns.amazonaws.com" }, "Action": [ "xray:PutTraceSegments",
 "xray:GetSamplingRules", "xray:GetSamplingTargets" ], "Resource": "*",
 "Condition": { "StringEquals": { "aws:SourceAccount": "account-id" }, "StringLike":
 { "aws:SourceArn": "arn:partition:sns:region:account-id:topic-name" } } } ] }'
```

Um diese Beispiele zu verwenden, ersetzen Sie *partition*, *regionaccount-id*, und *topic-name* durch Ihre spezifische AWS Partition, Region, Konto-ID und den Namen des Amazon SNS-Themas. Um allen Amazon SNS-Themen die Berechtigung zum Senden von Ablaufverfolgungsdaten an X-Ray zu erteilen, ersetzen Sie den Themennamen durch *.

# Anzeigen von Amazon SNS-Publisher- und Subscriber-Traces in der X-Ray-Konsole

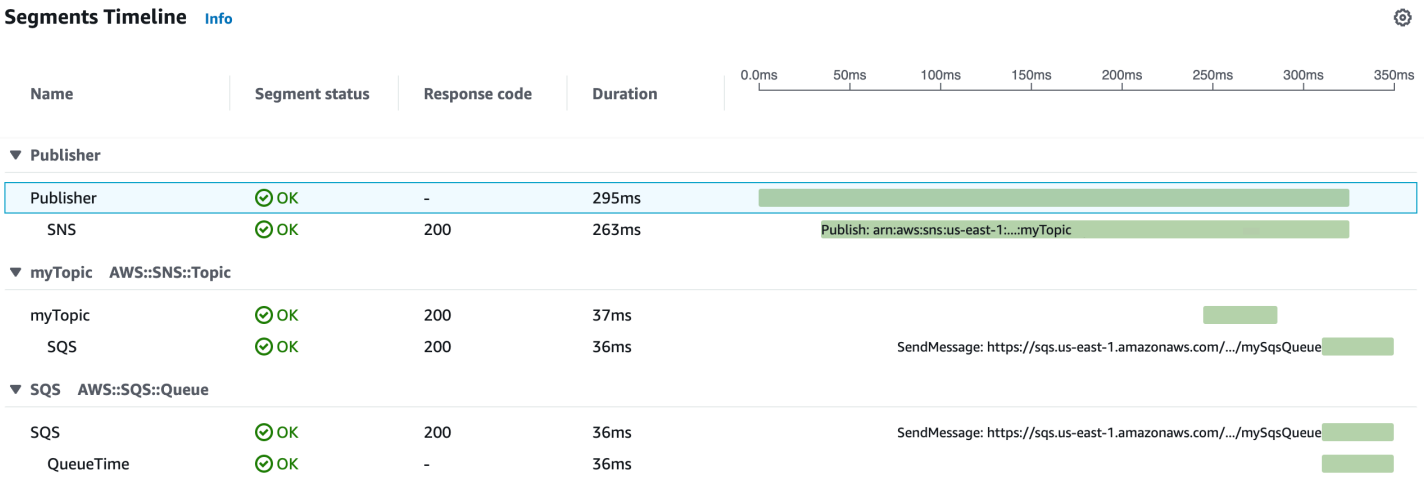
Verwenden Sie die X-Ray-Konsole, um eine Ablaufverfolgungskarte und Ablaufverfolgungsdetails anzuzeigen, die eine verbundene Ansicht von Amazon SNS-Herausgebern und -Abonnenten anzeigen. Wenn die aktive Ablaufverfolgung von Amazon SNS für ein Thema aktiviert ist, zeigt die X-Ray-Ablaufverfolgungs- und Ablaufverfolgungsdetail-Übersicht verbundene Knoten für Amazon SNS-Herausgeber, das Amazon SNS-Thema und nachgeschaltete Abonnenten an:




Nachdem Sie eine Ablaufverfolgung ausgewählt haben, die sich über einen Amazon SNS-Herausgeber und -Abonnenten erstreckt, zeigt die X-Ray-Ablaufverfolgungsdetailseite eine Zuordnung von Ablaufverfolgungsdetails und eine Segmentzeitleiste an.

Example Beispiel-Zeitleiste mit Amazon SNS-Herausgeber und Abonnent

Dieses Beispiel zeigt einen Zeitplan, der einen Amazon SNS-Herausgeber enthält, der eine Nachricht an ein Amazon SNS-Thema sendet, das von einem Amazon SQSAbonnenten verarbeitet wird.



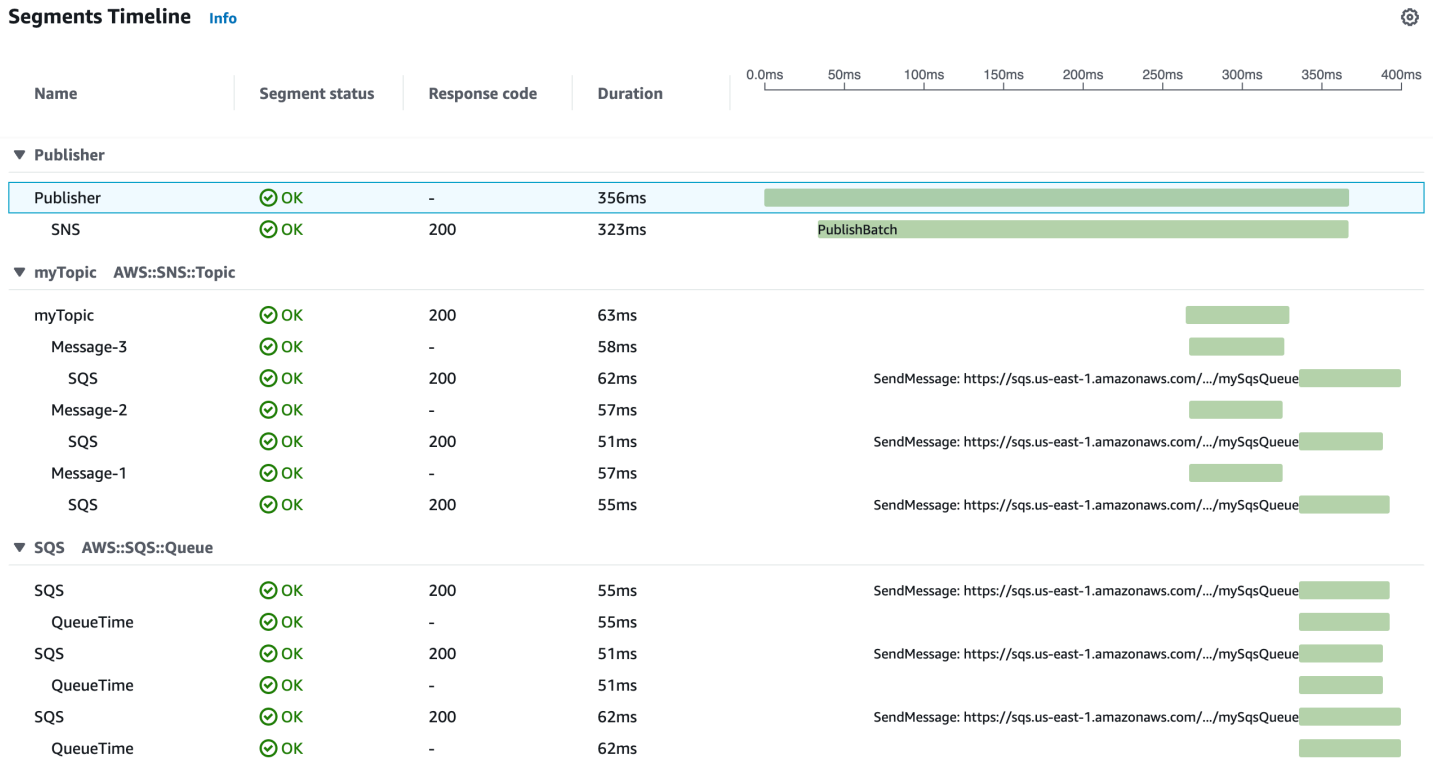

Die obige Beispielzeitleiste enthält Details zum Amazon SNS-Nachrichtenablauf:

- Das SNS-Segment stellt die Roundtrip-Dauer des `Publish` API-Aufrufs vom Client dar.

- Das myTopic-Segment stellt die Latenz der Amazon SNS-Antwort auf die Veröffentlichungsanforderung dar.
- Das SQS-Teilsegment stellt die Round-Trip-Zeit dar, die Amazon SNS benötigt, um die Nachricht in einer Amazon SQS-Warteschlange zu veröffentlichen.
- Die Zeit zwischen dem myTopic-Segment und dem SQS-Teilsegment stellt die Zeit dar, die die Nachricht im Amazon SNS-System verbringt.

Example Beispiel-Zeitachse mit gestapelten Amazon SNS-Nachrichten

Wenn mehrere Amazon SNS-Nachrichten in einem einzigen Trace gestapelt werden, zeigt die Segment-Zeitleiste Segmente an, die jede verarbeitete Nachricht darstellen.




# AWS Step Functions und AWS X-Ray

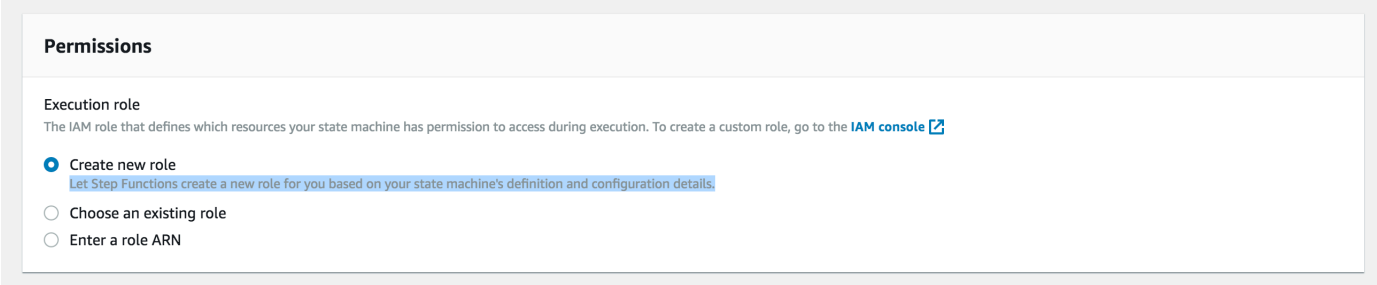
AWS X-Rayist in integriert.AWS Step Functionsum Anfragen für Step Functions zu verfolgen und zu analysieren. Sie können die Komponenten IhresZustandsautomat, identifizieren Sie Leistungsengpässe und beheben Sie Anforderungen, die zu einem Fehler geführt haben. Weitere Informationen finden Sie unterAWS X-Rayund Step FunctionsimAWS Step FunctionsEntwicklerhandbuch.

So aktivieren Sie die Röntgenprotokollierung beim Erstellen eines neuen Zustandsmaschiners

1. Öffnen Sie die Step Functions Functions-Konsolehttps://console.aws.amazon.com/states/aus.
2. Klicken Sie aufErstellen eines Zustandsautomatenaus.
3. Auf derDefinieren eines Zustandsautomaten-Seite, wählen Sie entwederAutor mit CodeausschnittenoderBeginne mit einer Vorlageaus. Wenn Sie ein Beispielprojekt ausführen möchten, können Siekann nichtAktivieren der X-Ray-Ablaufverfolgung währendaus. Stattdessen,Aktivieren der X-Ray-AblaufverfolgungErstellen IhrerZustandsautomataus.
4. Wählen Sie Next (Weiter) aus.
5. Auf derGeben Sie Details an-Seite konfigurieren Sie Ihren Zustandsautomaten.
6. Klicken Sie aufX-Ray-Ablaufverfolgungaus.

So aktivieren Sie die X-Ray-Ablaufverfolgung in einem vorhandenen Zustandsautomaten

1. Wählen Sie in der Step Functions Functions-Konsole den Zustandscomputer aus, für den Sie die Ablaufverfolgung aktivieren möchten.
2. Wählen Sie Edit.
3. Klicken Sie aufX-Ray-Ablaufverfolgungaus.
4. (Optional) Generieren Sie automatisch eine neue Rolle für Ihren Statuscomputer, um X-Ray-Berechtigungen einzuschließen, indem SieNeue Rolle erstellenaus dem Fenster Berechtigungen.




5. Wählen Sie Save (Speichern) aus.

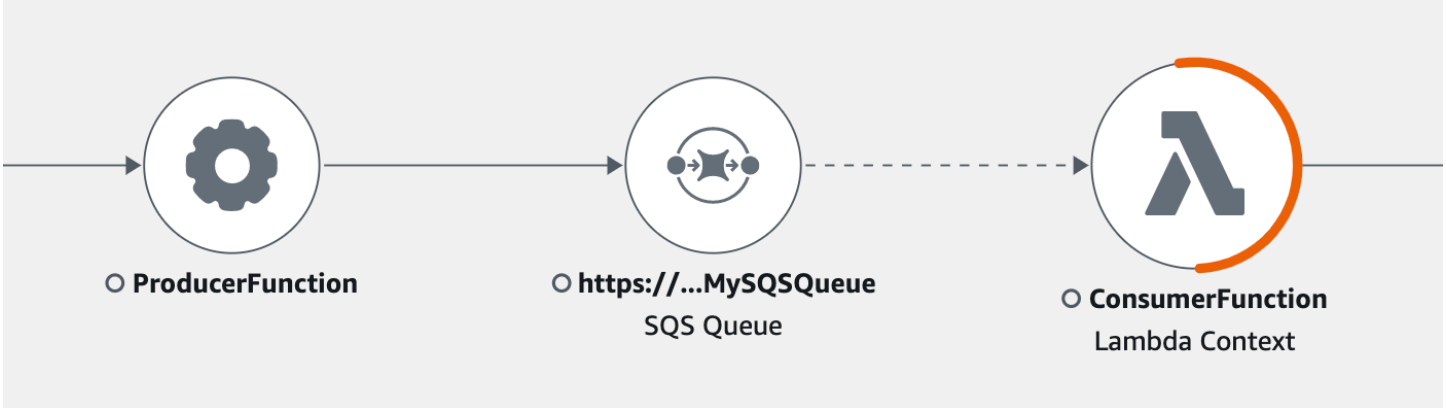
> **Note**
>
> Wenn Sie einen neuen Zustandsautomaten erstellen,es's wird automatisch verfolgt, wenn die Anforderung abgetastet wird und die Verfolgung in einem Upstream-Service wie Amazon API Gateway aktiviert ist oderAWS Lambdaaus. Für jeden vorhandenen Zustandscomputer, der nicht über die Konsole konfiguriert ist, z. B. über eineAWS

CloudFormation-Vorlage,überprüfen Sie, ob Sie über eine IAM-Richtlinie verfügen, die ausreichende Berechtigungen zum Aktivieren von X-Ray-Traces gewährt.

# Amazon SQS und AWS X-Ray

AWS X-Ray lässt sich in Amazon Simple Queue Service (Amazon SQS) integrieren, um Nachrichten zu verfolgen, die über eine Amazon SQS-Warteschlange übergeben werden. Wenn ein Service Anfragen mithilfe des X-Ray-SDK verfolgt, kann Amazon SQS den Nachverfolgungs-Header senden und die ursprüngliche Nachverfolgung mit einer konsistenten Nachverfolgungs-ID vom Sender an den Verbraucher weitergeben. Die Nachverfolgungskontinuität ermöglicht Benutzern das Nachverfolgen, Analysieren und Debuggen in allen nachgeschalteten Services.

AWS X-Ray unterstützt die Nachverfolgung von ereignisgesteuerten Anwendungen mit Amazon SQS und AWS Lambda. Verwenden Sie die - CloudWatch Konsole, um eine verbundene Ansicht jeder Anfrage anzuzeigen, während sie mit Amazon SQS in die Warteschlange gestellt und von einer nachgelagerten Lambda-Funktion verarbeitet wird. Ablaufverfolgungen von Produzenten von Upstream-Nachrichten werden automatisch mit Ablaufverfolgungen von Downstream-Lambda-Konsumentenknoten verknüpft, wodurch eine end-to-end Ansicht der Anwendung erstellt wird. Weitere Informationen finden Sie unter Nachverfolgung von ereignisgesteuerten Anwendungen.




Amazon SQS unterstützt die folgende Tracing-Header-Instrumentierung:

- Standard-HTTP-Header – Das X-Ray-SDK füllt den Trace-Header automatisch als HTTP-Header aus, wenn Sie Amazon SQS über das AWS SDK aufrufen. Der Standard-Nachverfolgungs-Header wird von `X-Amzn-Trace-Id` übernommen und entspricht allen Nachrichten, die in einer `SendMessage`- oder `SendMessageBatch`-Anfrage enthalten sind. Weitere Informationen zum Standard-HTTP-Header finden Sie unter Ablaufverfolgungs-Header.

- **`AWSTraceHeader`** Systemattribut – Das `AWSTraceHeader` ist ein Nachrichtensystemattribut, das von Amazon SQS für die Übertragung des X-Ray-Ablaufverfolgungs-Headers mit Nachrichten in der Warteschlange reserviert wurde. `AWSTraceHeader` ist für die Verwendung verfügbar, auch wenn die automatische Instrumentierung über das X-Ray-SDK nicht erfolgt, z. B. beim Erstellen eines Ablaufverfolgungs-SDK für eine neue Sprache. Wenn beide Header-Instrumentierungen festgelegt sind, überschreibt das Nachrichtensystemattribut den HTTP-Nachverfolgungs-Header.

Bei der Ausführung auf Amazon EC2 unterstützt Amazon SQS die Verarbeitung einer Nachricht nach der anderen. Dies gilt für die Ausführung auf einem On-Premises-Host und für die Verwendung von Container-Services wie , AWS Fargate Amazon ECS oder AWS App Mesh.

Der Trace-Header ist sowohl von Amazon SQS-Nachrichtengröße als auch von den Kontingenten für Nachrichtenattribute ausgeschlossen. Die Aktivierung der X-Ray-Ablaufverfolgung wird Ihre Amazon SQS-Kontingente nicht überschreiten. Weitere Informationen zu AWS Kontingenten finden Sie unter Amazon SQS-Kontingente.

## Senden des HTTP-Nachverfolgungs-Headers

Senderkomponenten in Amazon SQS können den Trace-Header automatisch über den -`SendMessageBatch` oder -`SendMessage`Aufruf senden. Wenn AWS SDK-Clients instrumentiert sind, können sie automatisch über alle Sprachen verfolgt werden, die vom X-Ray-SDK unterstützt werden. Nachverfolgte AWS-Services und Ressourcen, auf die Sie innerhalb dieser Services zugreifen (z. B. ein Amazon S3-Bucket oder eine Amazon SQS-Warteschlange), werden auf der Ablaufverfolgungskarte in der X-Ray-Konsole als nachgelagerte Knoten angezeigt.

Informationen zum Nachverfolgen von AWS SDK-Aufrufen mit Ihrer bevorzugten Sprache finden Sie in den folgenden Themen in den unterstützten SDKs:

- Go – Nachverfolgen von AWS SDK-Aufrufen mit dem X-Ray SDK for Go
- Java – Verfolgen von AWS SDK-Aufrufen mit dem X-Ray SDK for Java
- Node.js – Nachverfolgen von AWS SDK-Aufrufen mit dem X-Ray SDK for Node.js
- Python – Nachverfolgen von AWS SDK-Aufrufen mit dem X-Ray-SDK für Python
- Ruby – Nachverfolgen von AWS SDK-Aufrufen mit dem X-Ray SDK for Ruby
- .NET – Nachverfolgen von AWS SDK-Aufrufen mit dem X-Ray SDK for .NET

# Abrufen des Nachverfolgungs-Headers und Wiederherstellen des Nachverfolgungskontexts

Wenn Sie einen nachgelagerten Lambda-Verbraucher verwenden, erfolgt die Verbreitung des Ablaufverfolgungskontexts automatisch. Um die Kontextverbreitung mit anderen Amazon SQS-Konsumenten fortzusetzen, müssen Sie die Übergabe manuell an die Empfängerkomponente instrumentieren.

Es gibt drei Hauptschritte für die Wiederherstellung des Nachverfolgungskontexts:

- Empfangen Sie die Nachricht aus der Warteschlange für das `AWSTraceHeader`-Attribut, indem Sie die `ReceiveMessage`-API aufrufen.
- Rufen Sie den Nachverfolgungs-Header aus dem Attribut ab.
- Stellen Sie die Nachverfolgungs-ID aus dem Header wieder her. Optional können Sie dem Segment weitere Metriken hinzufügen.

Im Folgenden finden Sie ein Beispiel für eine Implementierung, die mit dem X-Ray SDK for Java geschrieben wurde.

Example Abrufen des Nachverfolgungs-Headers und Wiederherstellen des Nachverfolgungskontexts

```
// Receive the message from the queue, specifying the "AWSTraceHeader"
ReceiveMessageRequest receiveMessageRequest = new ReceiveMessageRequest()
        .withQueueUrl(QUEUE_URL)
        .withAttributeNames("AWSTraceHeader");
List<Message> messages = sqs.receiveMessage(receiveMessageRequest).getMessages();

if (!messages.isEmpty()) {
    Message message = messages.get(0);

    // Retrieve the trace header from the AWSTraceHeader message system attribute
    String traceHeaderStr = message.getAttributes().get("AWSTraceHeader");
    if (traceHeaderStr != null) {
        TraceHeader traceHeader = TraceHeader.fromString(traceHeaderStr);

        // Recover the trace context from the trace header
        Segment segment = AWSXRay.getCurrentSegment();
        segment.setTraceId(traceHeader.getRootTraceId());
        segment.setParentId(traceHeader.getParentId());
```

```
 segment.setSampled(traceHeader.getSampled().equals(TraceHeader.SampleDecision.SAMPLED));
    }
}
```

# Amazon S3 und AWS X-Ray

AWS X-Ray lässt sich in Amazon S3 integrieren, um Upstream-Anforderungen zur Aktualisierung der S3-Buckets Ihrer Anwendung nachzuverfolgen. Wenn ein Service Anfragen mithilfe des X-Ray-SDK verfolgt, kann Amazon S3 die Nachverfolgungs-Header an nachgelagerte Ereignisabonnenten wie AWS Lambda, Amazon SQS und Amazon SNS senden. X-Ray aktiviert Ablaufverfolgungsnachrichten für Amazon S3-Ereignisbenachrichtigungen.

Sie können die X-Ray-Ablaufverfolgungszuordnung verwenden, um die Verbindungen zwischen Amazon S3 und anderen -Services anzuzeigen, die Ihre Anwendung verwendet. Sie können mithilfe der Konsole auch Metriken, wie durchschnittliche Latenz- und Ausfallraten, anzuzeigen. Weitere Informationen zur X-Ray-Konsole finden Sie unter Erkunden Sie die X-Ray-Konsole.

Amazon S3 unterstützt die standardmäßige HTTP-Header-Instrumentierung. Das X-Ray-SDK füllt den Trace-Header automatisch als HTTP-Header aus, wenn Sie Amazon S3 über das AWS SDK aufrufen. Der Standard-Trace-Header wird von übertragen`X-Amzn-Trace-Id`. Weitere Informationen zur Nachverfolgung von Headern finden Sie unter Ablaufverfolgungs-Header auf der Konzeptseite. Amazon S3-Ablaufverfolgungskontext-Verbreitung unterstützt die folgenden Abonnenten: Lambda, SQS und SNS. Da SQS und SNS selbst keine Segmentdaten ausgeben, werden sie nicht in Ihrer Ablaufverfolgungs- oder Ablaufverfolgungszuordnung angezeigt, wenn sie von S3 ausgelöst werden, obwohl sie den Ablaufverfolgungs-Header an nachgelagerte Services weitergeben.

## Konfigurieren von Amazon S3-Ereignisbenachrichtigungen

Mit der Amazon S3-Benachrichtigungsfunktion erhalten Sie Benachrichtigungen, wenn bestimmte Ereignisse in Ihrem Bucket auftreten. Diese Benachrichtigungen können dann an die folgenden Ziele in Ihrer Anwendung weitergegeben werden:

- Amazon Simple Notification Service (Amazon SNS)
- Amazon Simple Queue Service (Amazon SQS)
- AWS Lambda

Eine Liste der unterstützten Ereignisse finden Sie unter Unterstützte Ereignistypen im Amazon S3-Entwicklerhandbuch.

## Amazon SNS und Amazon SQS

Um Benachrichtigungen in einem SNS-Thema oder einer SQS-Warteschlange zu veröffentlichen, müssen Sie zunächst Amazon S3-Berechtigungen erteilen. Um diese Berechtigungen zu erteilen, fügen Sie eine AWS Identity and Access Management (IAM)-Richtlinie an das SNS-Zielthema oder die SQS-Warteschlange an. Weitere Informationen zu den erforderlichen IAM-Richtlinien finden Sie unter Erteilen von Berechtigungen zum Veröffentlichen von Nachrichten in einem SNS-Thema oder einer SQS-Warteschlange.

Informationen zur Integration von SNS und SQS mit X-Ray finden Sie unter Amazon SNS und AWS X-Ray und Amazon SQS und AWS X-Ray.

## AWS Lambda

Wenn Sie die Amazon S3-Konsole verwenden, um Ereignisbenachrichtigungen für einen S3-Bucket für eine Lambda-Funktion zu konfigurieren, richtet die Konsole die erforderlichen Berechtigungen für die Lambda-Funktion ein, sodass Amazon S3 über Berechtigungen zum Aufrufen der Funktion aus dem Bucket verfügt. Weitere Informationen finden Sie unter Wie aktiviere und konfiguriere ich Ereignisbenachrichtigungen für einen S3-Bucket? im Konsolenbenutzerhandbuch für Amazon Simple Storage Service.

Sie können Amazon S3 auch Berechtigungen von erteilen AWS Lambda , um Ihre Lambda-Funktion aufzurufen. Weitere Informationen finden Sie unter Tutorial: Verwenden von AWS Lambda mit Amazon S3 im AWS Lambda-Entwicklerhandbuch.

Weitere Informationen zur Integration von Lambda in X-Ray finden Sie unter Instrumentieren von Java-Code in AWS Lambda .

# Ressourcen in X-Ray verwalten

Diese Anleitung zeigt Ihnen, wie Sie Ressourcen in X-Ray mithilfe einer Vorlage verwalten, indem Sie Ressourcen konfigurieren und Ressourcen mithilfe eines Schlüssels und eines optionalen Wertepaars taggen.

Mithilfe einer AWS CloudFormation Vorlage können Sie Ihre Ressourcen und Infrastruktur automatisch einrichten und verwalten. Verwenden Sie diese Vorlage im JSON YAML Oder-Format, um eine X-Ray-Gruppe, eine Stichprobenregel oder eine Ressourcenrichtlinie in einer einzigen Datei zu erstellen. Sie können eine AWS CloudFormation Vorlage beispielsweise für Folgendes verwenden:

- Verwenden Sie eine einzige Vorlage, um Ressourcen in mehreren Bereitstellungen einheitlich zu konfigurieren und manuelle Konfigurationsfehler zu vermeiden.
- Verwenden Sie eine Vorlagendatei, um Ressourcen über AWS Konten und Entwicklungsumgebungen hinweg zu verwalten und Vorlagendateien zwischen Teams auszutauschen.
- Steuern und verfolgen Sie Änderungen an Ihrer Vorlage mithilfe der Versionskontrolle und machen Sie Änderungen bei Bedarf rückgängig.

Sie können Ihrer Ressource auch Tags zuweisen, sodass Sie Ressourcen anhand von Stichwörtern suchen und filtern und Tag-basierte Berechtigungen durchsetzen können. Sie können Tags beispielsweise für Folgendes verwenden:

- Kennzeichnen Sie ein bestimmtes Team, eine Abteilung oder eine Anwendung, um nachzuverfolgen, welche Ressourcen sie verwenden.
- Kennzeichnen Sie Ressourcen, die eine besondere Behandlung erfordern, wie z. B. sensible Dokumente.
- Verwalten Sie markierte Ressourcen automatisch mithilfe von Skripten, um die Nutzung zu Spitzenzeiten zu unterbrechen.

Die folgenden Abschnitte enthalten zusätzliche Informationen zur Verwaltung von Ressourcen mit AWS CloudFormation Vorlagen und markierten Ressourcen.

Themen

- Erstellen von X-Ray-Ressourcen mit AWS CloudFormation

- Kennzeichnen von Regeln und Gruppen für die Röntgenprobenahme

# Erstellen von X-Ray-Ressourcen mit AWS CloudFormation

AWS X-Ray ist in einen Service integriert AWS CloudFormation, der Ihnen hilft, Ihre AWS Ressourcen zu modellieren und einzurichten, sodass Sie weniger Zeit mit der Erstellung und Verwaltung Ihrer Ressourcen und Infrastruktur verbringen müssen. Sie erstellen eine Vorlage, die alle AWS Ressourcen beschreibt, die Sie verwenden möchten, und die diese Ressourcen für Sie AWS CloudFormation bereitstellt und konfiguriert.

Wenn Sie es verwenden AWS CloudFormation, können Sie Ihre Vorlage wiederverwenden, um Ihre X-Ray-Ressourcen konsistent und wiederholt einzurichten. Beschreiben Sie Ihre Ressourcen einmal und stellen Sie dann dieselben Ressourcen immer wieder in mehreren AWS-Konten Regionen bereit.

## X-Ray und AWS CloudFormation Vorlagen

Verwenden Sie eine AWS CloudFormation Vorlage, um Ressourcen für X-Ray und verwandte Dienste bereitzustellen und zu konfigurieren. Vorlagen sind Textdateien, die in JSON oder YAML formatiert sind. Diese Vorlagen beschreiben die Ressourcen, die Sie in Ihren AWS CloudFormation Stacks bereitstellen möchten. Wenn Sie mit JSON oder YAML nicht vertraut sind, können Sie AWS CloudFormation Designer verwenden, um Ihnen die ersten Schritte mit Vorlagen zu erleichtern. AWS CloudFormation Weitere Informationen finden Sie unter Was ist AWS CloudFormation -Designer? im AWS CloudFormation -Benutzerhandbuch.

X-Ray unterstützt das Erstellen von AWS::XRay::GroupAWS::XRay::SamplingRule, und AWS::XRay::ResourcePolicyRessourcen in AWS CloudFormation. Weitere Informationen, einschließlich Beispielen für JSON- und YAML-Vorlagen, finden Sie in der Referenz zum X-Ray-Ressourcentyp im AWS CloudFormation Benutzerhandbuch.

## Erfahren Sie mehr über AWS CloudFormation

Weitere Informationen AWS CloudFormation dazu finden Sie in den folgenden Ressourcen:

- AWS CloudFormation
- AWS CloudFormation Benutzerhandbuch
- AWS CloudFormation API Reference
- AWS CloudFormation -Benutzerhandbuch für die Befehlszeilenschnittstelle

# Kennzeichnen von Regeln und Gruppen für die Röntgenprobenahme

Stichwörter sind Wörter oder Ausdrücke, mit denen Sie Ihre AWS Ressourcen identifizieren und organisieren können. Sie können jeder Ressource mehrere Tags hinzufügen. Jedes Tag enthält einen Schlüssel und einen optionalen Wert, den Sie definieren. Ein Tag-Schlüssel könnte beispielsweise sein**`domain`**, und der Tag-Wert könnte sein**`example.com`**. Sie können Ihre Ressourcen anhand der von Ihnen hinzugefügten Tags suchen und filtern. Weitere Informationen zur Verwendung von Tags finden Sie unter AWS Ressourcen taggen in der AWS allgemeinen Referenz.

Sie können Tags verwenden, um tagbasierte Berechtigungen für Distributionen durchzusetzen. CloudFront Weitere Informationen finden Sie unter Steuern des Zugriffs auf AWS Ressourcen mithilfe von Ressourcen-Tags.

> **Note**
>
> Tag Editor und AWS Resource Groups unterstützen derzeit keine X-Ray-Ressourcen. Sie können Tags mithilfe der AWS X-Ray Konsole oder der API hinzufügen und verwalten.

Sie können Tags auf Ressourcen anwenden, indem Sie die X-Ray-Konsole, API AWS CLI, SDKs und AWS Tools for Windows PowerShell verwenden. Weitere Informationen finden Sie in der folgenden - Dokumentation:

- X-Ray-API — Die folgenden Operationen finden Sie in der AWS X-Ray API-Referenz:
  - ListTagsForResource
  - CreateSamplingRule
  - CreateGroup
  - TagResource
  - UntagResource
- AWS CLI — Siehe xray in der AWS CLI Befehlsreferenz
- SDKs – Siehe die jeweilige SDK-Dokumentation auf der Seite für die AWS -Dokumentation

> **Note**
>
> Wenn Sie einer X-Ray-Ressource keine Tags hinzufügen oder ändern können oder wenn Sie keine Ressource mit bestimmten Tags hinzufügen können, verfügen Sie möglicherweise nicht über die erforderlichen Berechtigungen, um diesen Vorgang auszuführen. Um Zugriff zu beantragen, wenden Sie sich an einen AWS Benutzer in Ihrem Unternehmen, der über Administratorrechte in X-Ray verfügt.

Themen



## Tag-Einschränkungen

Für Tags gelten die folgenden Einschränkungen.

- Maximale Anzahl von Tags pro Ressource: 50
- Maximale Schlüssellänge – 128 Unicode-Zeichen.
- Maximale Wertlänge – 256 Unicode-Zeichen.
- Gültige Werte für Schlüssel- und Wert sind a-z, A-Z, 0-9, Leerzeichen und die folgenden Zeichen: _ . : / = + - und @.
- Bei Tag-Schlüsseln und -Werten muss die Groß- und Kleinschreibung beachtet werden.
- Verwenden Sie es nicht `aws:` als Präfix für Schlüssel; es ist für die AWS Verwendung reserviert.

> **Note**
>
> Sie können Systemtags nicht bearbeiten oder löschen.

# Tags in der Konsole verwalten

Sie können optionale Tags hinzufügen, wenn Sie eine X-Ray-Gruppe oder eine Probenahmeregel erstellen. Tags können auch später in der Konsole geändert oder gelöscht werden.

Die folgenden Verfahren erklären, wie Sie Tags für Ihre Gruppen und Sampling-Regeln in der X-Ray-Konsole hinzufügen, bearbeiten und löschen.

Themen



## Fügen Sie Tags zu einer neuen Gruppe (Konsole) hinzu

Wenn Sie eine neue X-Ray-Gruppe erstellen, können Sie optionale Tags auf der Seite Gruppe erstellen hinzufügen.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die X-Ray-Konsole unter https://console.aws.amazon.com/xray/home.
2. Erweitern Sie im Navigationsbereich die Option Konfiguration und wählen Sie Gruppen aus.
3. Wählen Sie Create group (Gruppe erstellen) aus.
4. Geben Sie auf der Seite Gruppe erstellen einen Namen und einen Filterausdruck für die Gruppe an. Weitere Informationen zu diesen Eigenschaften finden Sie unter Gruppen konfigurieren.
5. Geben Sie im Feld Tags einen Tag-Schlüssel und optional einen Tag-Wert ein. Sie können beispielsweise den Tag-Schlüssel und den Tag-Wert von **`Stage`** eingeben**`Production`**, um anzugeben, dass diese Gruppe für Produktionszwecke bestimmt ist. Wenn Sie ein Tag hinzufügen, wird eine neue Zeile angezeigt, in der Sie bei Bedarf ein weiteres Tag hinzufügen können. Tag-EinschränkungenIn diesem Thema finden Sie Einschränkungen für Tags.
6. Wenn Sie mit dem Hinzufügen von Tags fertig sind, wählen Sie Gruppe erstellen.

## Fügen Sie Stichwörter zu einer neuen Stichprobenregel hinzu (Konsole)

Wenn Sie eine neue Röntgenprobenregel erstellen, können Sie auf der Seite Probenahmeregel erstellen Tags hinzufügen.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die X-Ray-Konsole unter https://console.aws.amazon.com/xray/home.
2. Erweitern Sie im Navigationsbereich die Option Konfiguration und wählen Sie Sampling aus.
3. Wählen Sie Stichprobenregel erstellen aus.
4. Geben Sie auf der Seite Stichprobenregel erstellen einen Namen, eine Priorität, Grenzwerte, Übereinstimmungskriterien und passende Attribute an. Weitere Informationen zu diesen Eigenschaften finden Sie unter Konfigurieren Sie Stichprobenregeln.
5. Geben Sie im Feld Tags einen Tag-Schlüssel und optional einen Tag-Wert ein. Sie können beispielsweise den Tag-Schlüssel und den Tag-Wert von eingeben**`Stage`**, um anzugeben**`Production`**, dass diese Stichprobenregel für Produktionszwecke bestimmt ist. Wenn Sie ein Tag hinzufügen, wird eine neue Zeile angezeigt, in der Sie bei Bedarf ein weiteres Tag hinzufügen können. Tag-EinschränkungenIn diesem Thema finden Sie Einschränkungen für Tags.
6. Wenn Sie mit dem Hinzufügen von Tags fertig sind, wählen Sie Stichprobenregel erstellen aus.

## Bearbeiten oder löschen Sie Tags für eine Gruppe (Konsole)

Sie können Tags in einer X-Ray-Gruppe auf der Seite Gruppe bearbeiten ändern oder löschen.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die X-Ray-Konsole unter https://console.aws.amazon.com/xray/home.
2. Erweitern Sie im Navigationsbereich die Option Konfiguration und wählen Sie Gruppen aus.
3. Wählen Sie in der Tabelle Gruppen den Namen einer Gruppe aus.
4. Bearbeiten Sie auf der Seite Gruppe bearbeiten unter Tags die Tag-Schlüssel und -Werte. Sie können keine doppelten Tag-Schlüssel haben. Tag-Werte sind optional; Sie können Werte löschen, falls gewünscht. Weitere Informationen zu anderen Eigenschaften auf der Seite Gruppe bearbeiten finden Sie unterGruppen konfigurieren. Tag-EinschränkungenIn diesem Thema finden Sie Einschränkungen für Tags.
5. Um ein Tag zu löschen, wählen Sie X rechts neben dem Tag aus.
6. Wenn Sie mit dem Bearbeiten oder Löschen von Tags fertig sind, wählen Sie Gruppe aktualisieren.

### Bearbeiten oder löschen Sie Tags für eine Stichprobenregel (Konsole)

Sie können Tags in einer Röntgen-Sampling-Regel auf der Seite Sampling-Regel bearbeiten ändern oder löschen.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die X-Ray-Konsole unter https://console.aws.amazon.com/xray/home.
2. Erweitern Sie im Navigationsbereich die Option Konfiguration und wählen Sie Sampling aus.
3. Wählen Sie in der Tabelle mit den Stichprobenregeln den Namen einer Stichprobenregel aus.
4. Bearbeiten Sie unter Tags die Tag-Schlüssel und -Werte. Sie können keine doppelten Tag-Schlüssel haben. Tag-Werte sind optional; Sie können Werte löschen, falls gewünscht. Weitere Informationen zu anderen Eigenschaften auf der Seite Stichprobenregel bearbeiten finden Sie unterKonfigurieren Sie Stichprobenregeln. Tag-EinschränkungenIn diesem Thema finden Sie Einschränkungen für Tags.
5. Um ein Tag zu löschen, wählen Sie X rechts neben dem Tag aus.
6. Wenn Sie mit dem Bearbeiten oder Löschen von Tags fertig sind, wählen Sie Stichprobenregel aktualisieren.

## Verwaltung von Stichwörtern in AWS CLI

Sie können Tags hinzufügen, wenn Sie eine X-Ray-Gruppe oder eine Probenahmeregel erstellen. Sie können die auch verwenden AWS CLI , um Tags zu erstellen und zu verwalten. Verwenden Sie die AWS X-Ray Konsole oder die UntagResourceAPIs, um Tags für eine bestehende Gruppen- oder Stichprobenregel zu aktualisieren. TagResource

Themen

- Hinzufügen von Tags zu einer neuen X-Ray-Gruppe oder Sampling-Regel (CLI)
- Hinzufügen von Tags zu einer vorhandenen Ressource (CLI)
- Tags auf einer Ressource auflisten (CLI)
- Tags auf einer Ressource löschen (CLI)

### Hinzufügen von Tags zu einer neuen X-Ray-Gruppe oder Sampling-Regel (CLI)

Verwenden Sie einen der folgenden Befehle, um beim Erstellen einer neuen X-Ray-Gruppe oder Sampling-Regel optionale Tags hinzuzufügen.

- *Um einer neuen Gruppe Tags hinzuzufügen, führen Sie den folgenden Befehl aus und ersetzen Sie group_name durch den Namen Ihrer Gruppe, mydomain.com durch den Endpunkt Ihres Dienstes, key_name durch einen Tag-Schlüssel und optional value durch einen Tag-Wert.* Weitere Informationen zum Erstellen einer Gruppe finden Sie unter. Konfiguration von Sampling-, Gruppen- und Verschlüsselungseinstellungen mit der X-Ray-API

```
aws xray create-group \
   --group-name "group_name" \
   --filter-expression "service(\"mydomain.com\") {fault OR error}" \
   --tags [{"Key": "key_name","Value": "value"},{"Key": "key_name","Value": "value"}]
```

Im Folgenden wird ein Beispiel gezeigt.

```
aws xray create-group \
   --group-name "AdminGroup" \
   --filter-expression "service(\"mydomain.com\") {fault OR error}" \
   --tags [{"Key": "Stage","Value": "Prod"},{"Key": "Department","Value": "QA"}]
```

- Um einer neuen Stichprobenregel Tags hinzuzufügen, führen Sie den folgenden Befehl aus. Ersetzen Sie *dabei key_name* durch einen Tag-Schlüssel und optional *value* durch einen Tag-Wert. Dieser Befehl gibt die Werte im `--sampling-rule` Parameter als JSON-Datei an. Weitere Informationen zum Erstellen einer Stichprobenregel finden Sie unterKonfiguration von Sampling-, Gruppen- und Verschlüsselungseinstellungen mit der X-Ray-API.

```
aws xray create-sampling-rule \
   --cli-input-json file://file_name.json
```

Im Folgenden finden Sie den Inhalt der JSON-Datei *file_name.json*, die durch den Parameter angegeben wird. `--cli-input-json`

```
{
    "SamplingRule": {
        "RuleName": "rule_name",
        "RuleARN": "string",
        "ResourceARN": "string",
        "Priority": integer,
        "FixedRate": double,
        "ReservoirSize": integer,
        "ServiceName": "string",
```

```
        "ServiceType": "string",
        "Host": "string",
        "HTTPMethod": "string",
        "URLPath": "string",
        "Version": integer,
        "Attributes": {"attribute_name": "value","attribute_name": "value"...}
    }
    "Tags": [
           {
               "Key":"key_name",
               "Value":"value"
           },
           {
               "Key":"key_name",
               "Value":"value"
           }
          ]
}
```

Nachfolgend finden Sie einen Beispielbefehl.

```
aws xray create-sampling-rule \
   --cli-input-json file://9000-base-scorekeep.json
```

Im Folgenden finden Sie den Inhalt der durch den Parameter angegebenen `9000-base-scorekeep.json` Beispieldatei. `--cli-input-json`

```
{
    "SamplingRule": {
        "RuleName": "base-scorekeep",
        "ResourceARN": "*",
        "Priority": 9000,
        "FixedRate": 0.1,
        "ReservoirSize": 5,
        "ServiceName": "Scorekeep",
        "ServiceType": "*",
        "Host": "*",
        "HTTPMethod": "*",
        "URLPath": "*",
        "Version": 1
    }
    "Tags": [
```

```
            {
                "Key":"Stage",
                "Value":"Prod"
            },
            {
                "Key":"Department",
                "Value":"QA"
            }
           ]
}
```

## Hinzufügen von Tags zu einer vorhandenen Ressource (CLI)

Sie können den `tag-resource` Befehl ausführen, um einer vorhandenen X-Ray-Gruppe oder einer Stichprobenregel Tags hinzuzufügen. Diese Methode ist möglicherweise einfacher als das Hinzufügen von Tags durch Ausführen von `update-group` oder`update-sampling-rule`.

Um einer Gruppe oder einer Stichprobenregel Tags hinzuzufügen, führen Sie den folgenden Befehl aus. Ersetzen Sie dabei den ARN durch den ARN der Ressource und geben Sie die Schlüssel und optionalen Werte der Tags an, die Sie hinzufügen möchten.

```
aws xray tag-resource \
   --resource-arn "ARN" \
   --tag-keys [{"Key":"key_name","Value":"value"}, {"Key":"key_name","Value":"value"}]
```

Im Folgenden wird ein Beispiel gezeigt.

```
aws xray tag-resource \
   --resource-arn "arn:aws:xray:us-east-2:01234567890:group/AdminGroup" \
   --tag-keys [{"Key": "Stage","Value": "Prod"},{"Key": "Department","Value": "QA"}]
```

## Tags auf einer Ressource auflisten (CLI)

Sie können den `list-tags-for-resource` Befehl ausführen, um die Tags einer X-Ray-Gruppe oder einer Stichprobenregel aufzulisten.

Um die Tags aufzulisten, die einer Gruppe oder einer Stichprobenregel zugeordnet sind, führen Sie den folgenden Befehl aus und ersetzen Sie den ARN durch den ARN der Ressource.

```
aws xray list-tags-for-resource \
```

```
   --resource-arn "ARN"
```

Im Folgenden wird ein Beispiel gezeigt.

```
aws xray list-tags-for-resource \
   --resource-arn "arn:aws:xray:us-east-2:01234567890:group/AdminGroup"
```

### Tags auf einer Ressource löschen (CLI)

Sie können den `untag-resource` Befehl ausführen, um Tags aus einer X-Ray-Gruppe oder einer Stichprobenregel zu entfernen.

Um Tags aus einer Gruppe oder einer Stichprobenregel zu entfernen, führen Sie den folgenden Befehl aus. Ersetzen Sie dabei den ARN durch den ARN der Ressource und geben Sie die Schlüssel der Tags an, die Sie entfernen möchten.

Mit dem `untag-resource` Befehl können Sie nur ganze Tags entfernen. Um Tag-Werte zu entfernen, verwenden Sie die X-Ray-Konsole oder löschen Sie Tags und fügen Sie neue Tags mit denselben Schlüsseln, aber unterschiedlichen oder leeren Werten hinzu.

```
aws xray untag-resource \
   --resource-arn "ARN" \
   --tag-keys ["key_name","key_name"]
```

Im Folgenden wird ein Beispiel gezeigt.

```
aws xray untag-resource \
   --resource-arn "arn:aws:xray:us-east-2:01234567890:group/group_name" \
   --tag-keys ["Stage","Department"]
```

## Steuern Sie den Zugriff auf X-Ray-Ressourcen anhand von Tags

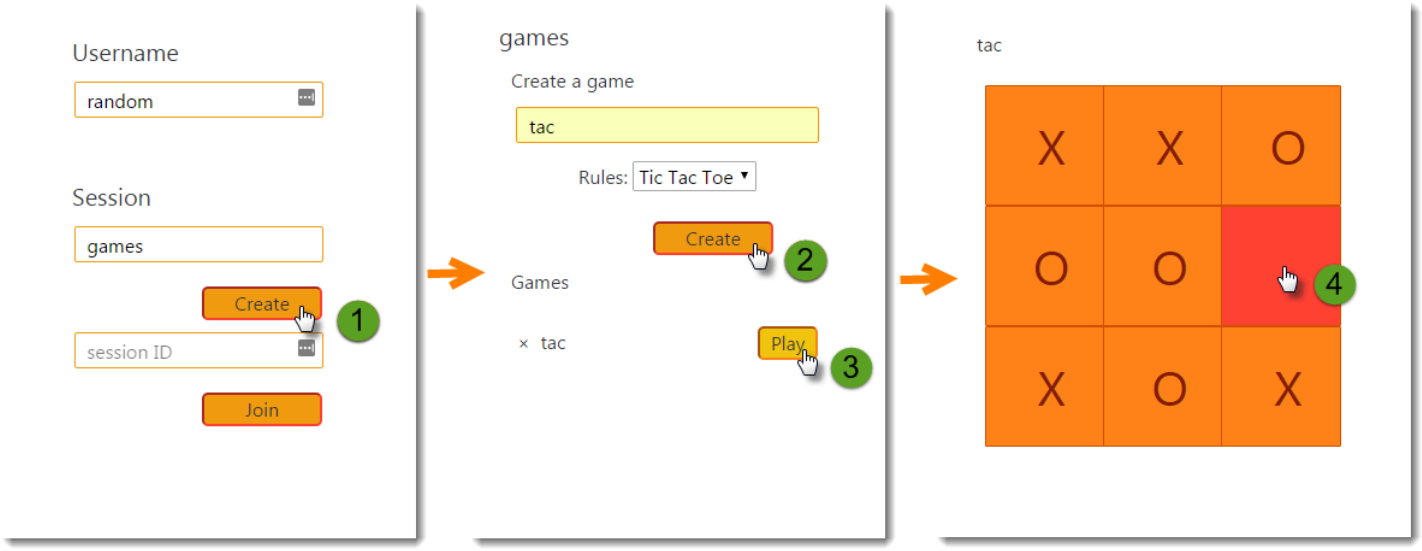
Sie können Tags an X-Ray-Gruppen oder Sampling-Regeln anhängen oder Tags in einer Anfrage an X-Ray übergeben. Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an, indem Sie die Schlüssel `xray:ResourceTag/key-name`, `aws:RequestTag/key-name`, oder Bedingung `aws:TagKeys` verwenden. Weitere Informationen zu diesen Bedingungsschlüsseln finden Sie unter Steuern des Zugriffs auf AWS Ressourcen mithilfe von Ressourcen-Tags.

Ein Beispiel für eine identitätsbasierte Richtlinie zur Einschränkung des Zugriffs auf eine Ressource auf der Grundlage der Markierungen dieser Ressource finden Sie unter Verwaltung des Zugriffs auf X-Ray-Gruppen und Stichprobenregeln auf der Grundlage von Tags.

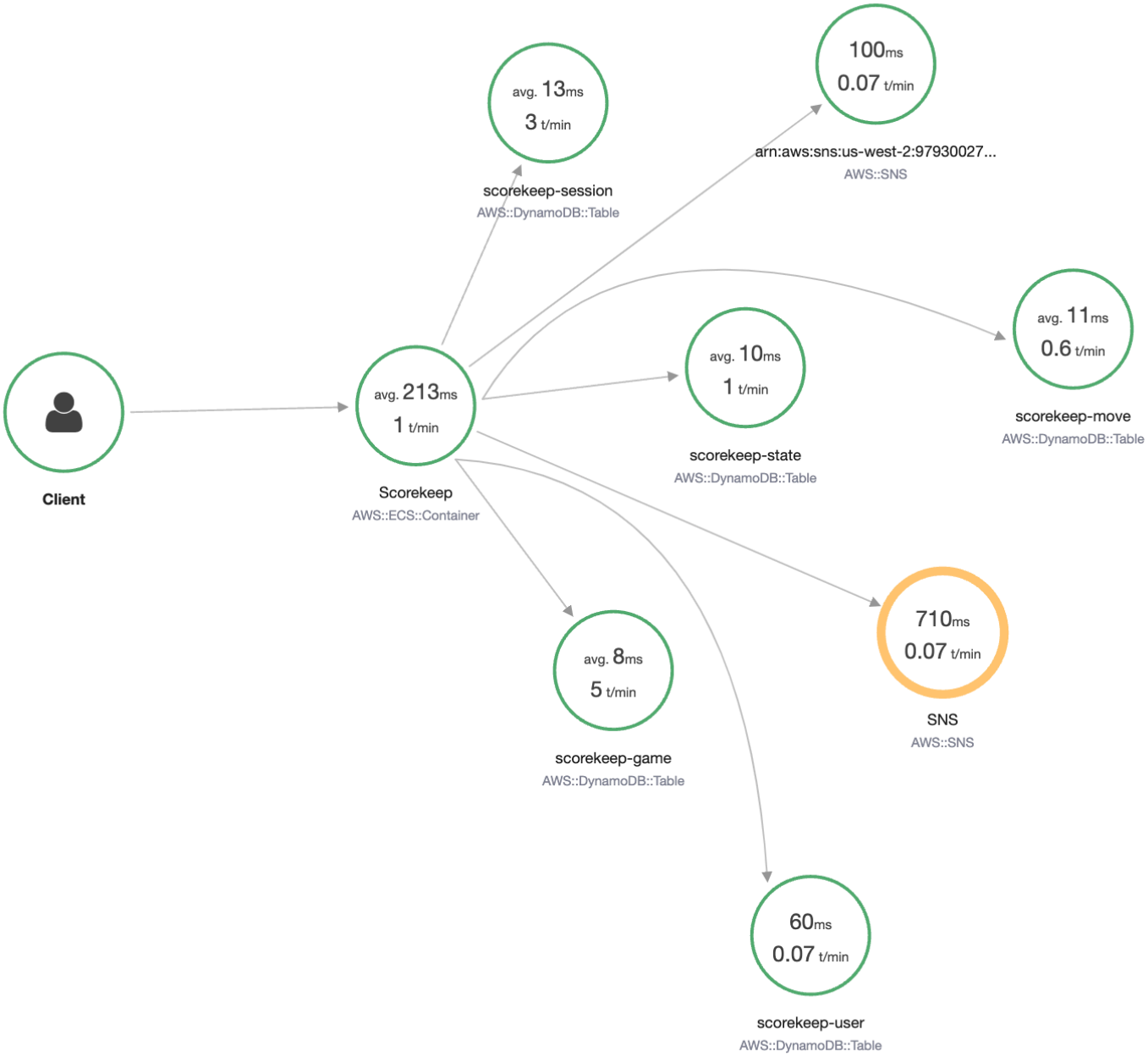
# AWS X-Ray Beispielanwendung

Die AWS X-Ray-eb-java-scorekeepBeispiel-App, die auf verfügbar ist GitHub, zeigt die Verwendung des AWS X-Ray-SDK zum Instrumentieren eingehender HTTP-Aufrufe, DynamoDB-SDK-Clients und HTTP-Clients. Die Beispiel-App verwendet , AWS CloudFormation um DynamoDB-Tabellen zu erstellen, Java-Code auf der Instance zu kompilieren und den X-Ray-Daemon ohne zusätzliche Konfiguration auszuführen.

Weitere Informationen zum Installieren und Verwenden einer instrumentierten Beispielanwendung mithilfe der AWS Management Console oder der finden Sie im Scorekeep-Tutorial AWS CLI.




Das Beispiel enthält eine Frontend-Web-App, die aufgerufene API und die DynamoDB-Tabellen, die zum Speichern von Daten verwendet werden. Grundlegende Instrumentierung mit Filtern , Plug-Ins und instrumentierten AWS SDK-Clients wird im `xray-gettingstarted` Zweig des Projekts angezeigt. Dies ist die Verzweigung, die Sie im Tutorial "Erste Schritte" bereitstellen. Da diese Verzweigung nur die Grundlagen beinhaltet, können Sie einen diff-Vorgang mit der `master`-Verzweigung durchführen, um schnell die Grundlagen zu erfassen.




Die Beispielanwendung veranschaulicht die grundlegende Instrumentierung in folgenden Dateien:

- HTTP-Anforderungsfilter – `WebConfig.java`
- AWS SDK-Client-Instrumentierung – `build.gradle`

Der `xray` Zweig der Anwendung umfasst die Verwendung von HTTPClient , Annotationen , SQL-Abfragen , benutzerdefinierte Untersegmente , eine instrumentierte AWS Lambda Funktion sowie instrumentierten Initialisierungscode und Skripts .

Um die Benutzeranmeldung und - AWS SDK for JavaScript autorisierung zu unterstützen, fügt die `xray-cognito` Verzweigung Amazon Cognito hinzu, um die Benutzerauthentifizierung und -autorisierung zu unterstützen. Mit Anmeldeinformationen, die von Amazon Cognito abgerufen wurden, sendet die Webanwendung auch Ablaufverfolgungsdaten an X-Ray, um Anforderungsinformationen aus der Sicht des Clients aufzuzeichnen. Der Browser-Client wird als eigener Knoten auf der Ablaufverfolgungskarte angezeigt und zeichnet zusätzliche Informationen auf, einschließlich der URL der Seite, die der Benutzer anzeigt, und der Benutzer-ID.

Schließlich fügt der `xray-worker` Zweig eine instrumentierte Python-Lambda-Funktion hinzu, die unabhängig ausgeführt wird und Elemente aus einer Amazon SQS-Warteschlange verarbeitet. Scorekeep fügt ein Element zur Warteschlange hinzu, wenn ein Spiel endet. Der durch CloudWatch Ereignisse ausgelöste Lambda-Worker ruft alle paar Minuten Elemente aus der Warteschlange ab und verarbeitet sie, um Spieldatensätze in Amazon S3 zur Analyse zu speichern.

Themen

- Erste Schritte mit der Scorekeep-Beispielanwendung
- Manuelles Instrumentieren von AWS SDK-Clients
- Erstellen zusätzlicher Untersegmente
- Aufzeichnung von Anmerkungen, Metadaten und Benutzer-IDs
- Instrumentieren von ausgehenden HTTP-Aufrufen
- Instrumentieren von Aufrufen einer PostgreSQL-Datenbank
- AWS Lambda Instrumentierungsfunktionen
- Instrumentieren von Startup-Code
- Instrumentieren von Skripten
- Instrumentieren eines Web-App-Clients
- Verwenden instrumentierter Clients in Auftragnehmer-Threads

## Erste Schritte mit der Scorekeep-Beispielanwendung

In diesem Tutorial wird der `xray-gettingstarted` Zweig der Scorekeep-Beispielanwendung verwendet, die verwendet, AWS CloudFormation um die Ressourcen zu erstellen und zu

konfigurieren, die die Beispielanwendung und den X-Ray-Daemon auf Amazon ECS ausführen. Die Anwendung verwendet das Spring-Framework, um eine JSON-Web-API zu implementieren, und das , AWS SDK for Java um Daten in Amazon DynamoDB zu speichern. Ein Servlet-Filter in der Anwendung instrumentiert alle eingehenden Anfragen, die von der Anwendung bedient werden, und ein Anfrage-Handler auf dem AWS SDK-Client instrumentiert Downstream-Aufrufe an DynamoDB .

Sie können diesem Tutorial entweder mit der AWS Management Console oder der folgen AWS CLI.

Sections



## Voraussetzungen

In diesem Tutorial wird verwendet, AWS CloudFormation um die Ressourcen zu erstellen und zu konfigurieren, die die Beispielanwendung und den X-Ray-Daemon ausführen. Die folgenden Voraussetzungen sind erforderlich, um das Tutorial zu installieren und auszuführen:

1. Wenn Sie einen IAM-Benutzer mit eingeschränkten Berechtigungen verwenden, fügen Sie die folgenden Benutzerrichtlinien in der IAM-Konsole hinzu:

   - `AWSCloudFormationFullAccess` – für den Zugriff auf und die Verwendung von CloudFormation
   - `AmazonS3FullAccess` – zum Hochladen einer Vorlagendatei in CloudFormation mithilfe der AWS Management Console
   - `IAMFullAccess` – zum Erstellen der Amazon-ECS- und Amazon EC2-Instance-Rollen
   - `AmazonEC2FullAccess` – zum Erstellen der Amazon EC2-Ressourcen
   - `AmazonDynamoDBFullAccess` – zum Erstellen der DynamoDB-Tabellen

- `AmazonECS_FullAccess` – zum Erstellen von Amazon-ECS-Ressourcen
- `AmazonSNSFullAccess` – zum Erstellen des Amazon SNS-Themas
- `AWSXrayReadOnlyAccess` – für die Berechtigung zum Anzeigen der Ablaufverfolgungszuordnung und Ablaufverfolgungen in der X-Ray-Konsole

2. Um das Tutorial mit der auszuführen AWS CLI, installieren Sie die CLI-Version 2.7.9 oder höher und konfigurieren Sie die CLI mit dem Benutzer aus dem vorherigen Schritt. Stellen Sie sicher, dass die Region konfiguriert ist, wenn Sie die AWS CLI mit dem Benutzer konfigurieren. Wenn keine Region konfiguriert ist, müssen Sie an jeden CLI-Befehl `--region` *`AWS-REGION`* anhängen.
3. Stellen Sie sicher, dass Git installiert ist, um das Beispielanwendungs-Repo zu klonen.
4. Verwenden Sie das folgende Codebeispiel, um den `xray-gettingstarted` Zweig des Scorekeep-Repositorys zu klonen:

```
git clone https://github.com/aws-samples/eb-java-scorekeep.git xray-scorekeep -b
 xray-gettingstarted
```

## Installieren der Scorekeep-Anwendung mit CloudFormation

AWS Management Console

Installieren der Beispielanwendung mithilfe der AWS Management Console

1. Öffnen Sie die CloudFormation-Konsole.
2. Wählen Sie Stack erstellen und dann im Dropdown-Menü Mit neuen Ressourcen aus.
3. Wählen Sie im Abschnitt Specify template (Vorlage angeben) die Option Upload a template file (Vorlagendatei hochladen) aus.
4. Wählen Sie Datei auswählen, navigieren Sie zu dem `xray-scorekeep/cloudformation` Ordner, der beim Klonen des Git-Repo erstellt wurde, und wählen Sie die `cf-resources.yaml` Datei aus.
5. Wählen Sie Next (Weiter), um fortzufahren.
6. Geben Sie `scorekeep` in das Textfeld Stack-Name ein und wählen Sie dann unten auf der Seite Weiter aus, um fortzufahren. Beachten Sie, dass der Rest dieses Tutorials davon ausgeht, dass der Stack den Namen hat`scorekeep`.

7. Scrollen Sie nach unten auf der Seite Stack-Optionen konfigurieren und wählen Sie Weiter, um fortzufahren.
8. Scrollen Sie nach unten auf der Seite Überprüfen, wählen Sie die Bestätigung des Kontrollkästchens aus, das IAM-Ressourcen mit benutzerdefinierten Namen erstellen CloudFormation kann, und wählen Sie Stack erstellen aus.
9. Der CloudFormation Stack wird jetzt erstellt. Der Stack-Status beträgt etwa fünf Minuten`CREATE_IN_PROGRESS`, bevor Sie zu wechseln`CREATE_COMPLETE`. Der Status wird regelmäßig aktualisiert, oder Sie können die Seite aktualisieren.

AWS CLI

Installieren der Beispielanwendung mithilfe der AWS CLI

1. Navigieren Sie zum `cloudformation` Ordner des `xray-scorekeep`Repositorys, das Sie zuvor im Tutorial geklont haben:

```
cd xray-scorekeep/cloudformation/
```

2. Geben Sie den folgenden AWS CLI Befehl ein, um den CloudFormation Stack zu erstellen:

```
aws cloudformation create-stack --stack-name scorekeep --capabilities
 "CAPABILITY_NAMED_IAM" --template-body file://cf-resources.yaml
```

3. Warten Sie, bis der CloudFormation Stack-Status lautet`CREATE_COMPLETE`, was etwa fünf Minuten dauert. Verwenden Sie den folgenden AWS CLI Befehl, um den Status zu überprüfen:

```
aws cloudformation describe-stacks --stack-name scorekeep --query
 "Stacks[0].StackStatus"
```

## Generieren von Ablaufverfolgungsdaten

Die Beispielanwendung enthält eine Front-End-Webanwendung. Verwenden Sie die Web-App, um Datenverkehr an die API zu generieren und Ablaufverfolgungsdaten an X-Ray zu senden. Rufen Sie zunächst die URL der Web-App mit der AWS Management Console oder der ab AWS CLI:

AWS Management Console

Suchen der Anwendungs-URL mithilfe der AWS Management Console

1. Öffnen Sie die CloudFormation-Konsole.
2. Wählen Sie den `scorekeep`Stack aus der Liste aus.
3. Wählen Sie auf der `scorekeep`Stack-Seite die Registerkarte Outputs und dann den `LoadBalancerUrl` URL-Link aus, um die Webanwendung zu öffnen.

AWS CLI

Suchen der Anwendungs-URL mithilfe der AWS CLI

1. Verwenden Sie den folgenden Befehl, um die URL der Webanwendung anzuzeigen:

```
aws cloudformation describe-stacks --stack-name scorekeep --query
 "Stacks[0].Outputs[0].OutputValue"
```

2. Kopieren Sie diese URL und öffnen Sie in einem Browser, um die Scorekeep-Webanwendung anzuzeigen.

Verwenden der Webanwendung zum Generieren von Ablaufverfolgungsdaten

1. Wählen Sie Create aus, um einen Benutzer und eine Sitzung zu erstellen.
2. Geben Sie einen Game Name ein, legen Sie die Rules auf Tic Tac Toe fest und wählen Sie anschließend Create aus, um ein Spiel zu erstellen.
3. Wählen Sie Play, um das Spiel zu starten.
4. Wählen Sie eine Kachel, um einen Spielzug zu machen, und ändern Sie den Spielestatus.

Jeder dieser Schritte generiert HTTP-Anfragen an die API und nachgelagerte Aufrufe an DynamoDB, um Benutzer-, Sitzungs-, Spiel-, Verschiebungs- und Zustandsdaten zu lesen und zu schreiben.

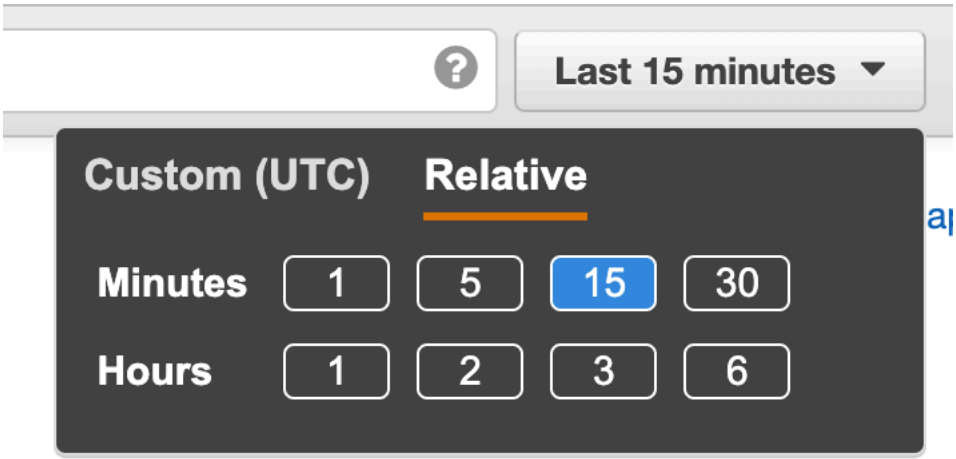
## Anzeigen der Trace-Übersicht in der AWS Management Console

Sie können die Trace-Übersicht und die von der Beispielanwendung generierten Traces in den X-Ray- und - CloudWatch Konsolen sehen.

X-Ray console

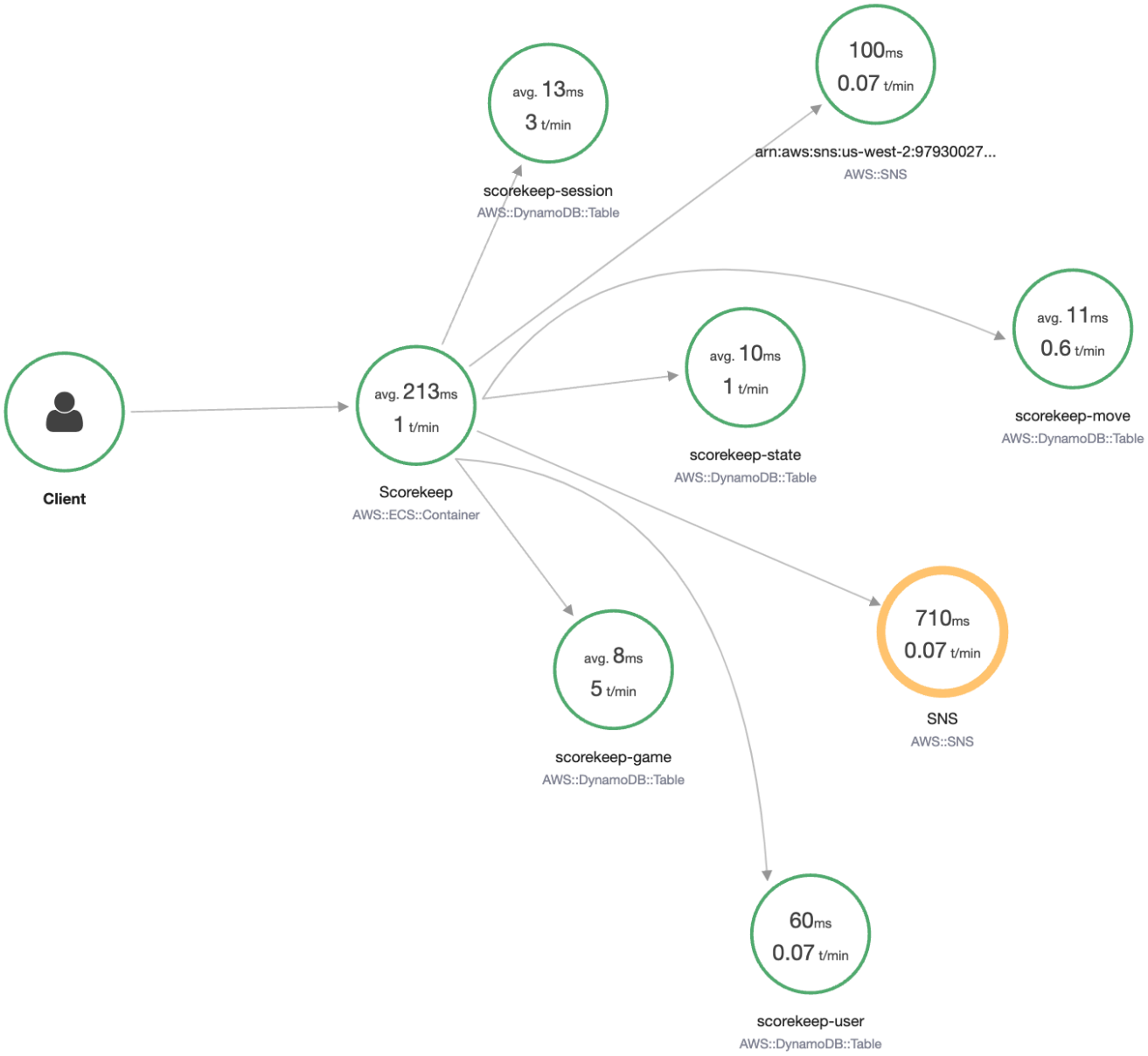
Verwenden der X-Ray-Konsole

1. Öffnen Sie die Seite Trace Map der X-Ray-Konsole .
2. Die Konsole zeigt eine Darstellung des Servicediagramms, das X-Ray aus den von der Anwendung gesendeten Ablaufverfolgungsdaten generiert. Passen Sie den Zeitraum der Trace-Übersicht bei Bedarf an, um sicherzustellen, dass alle Traces seit dem ersten Start der Webanwendung angezeigt werden.




Die Ablaufverfolgungszuordnung zeigt den Web-App-Client, die in Amazon ECS ausgeführte API und jede DynamoDB-Tabelle, die die Anwendung verwendet. Jede Anforderung an die Anwendung, bis zu einer konfigurierbaren Höchstanzahl von Anforderungen pro Sekunde, wird verfolgt, und zwar wie sie auf die API trifft, Anforderungen an nachgelagerte Services generiert und abgeschlossen wird.

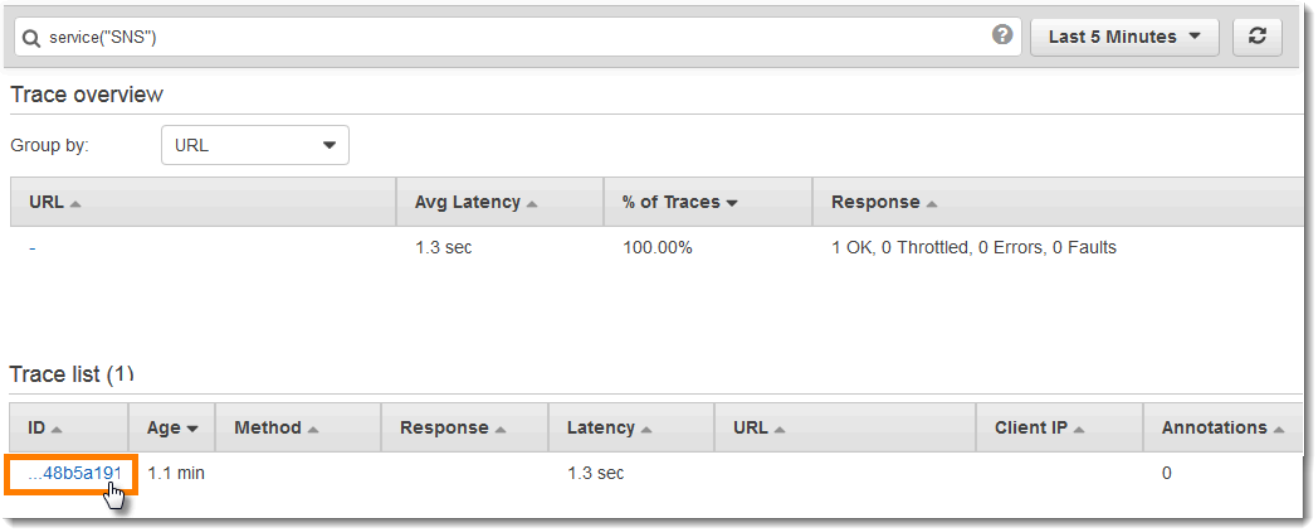
Sie können jeden Knoten in der Service-Grafik wählen, um Ablaufverfolgungen für Anforderungen anzusehen, die Datenverkehr zu diesem Knoten generiert haben. Derzeit ist der Amazon SNS-Knoten gelb. Zeigen Sie die Details an, um herauszufinden, warum.



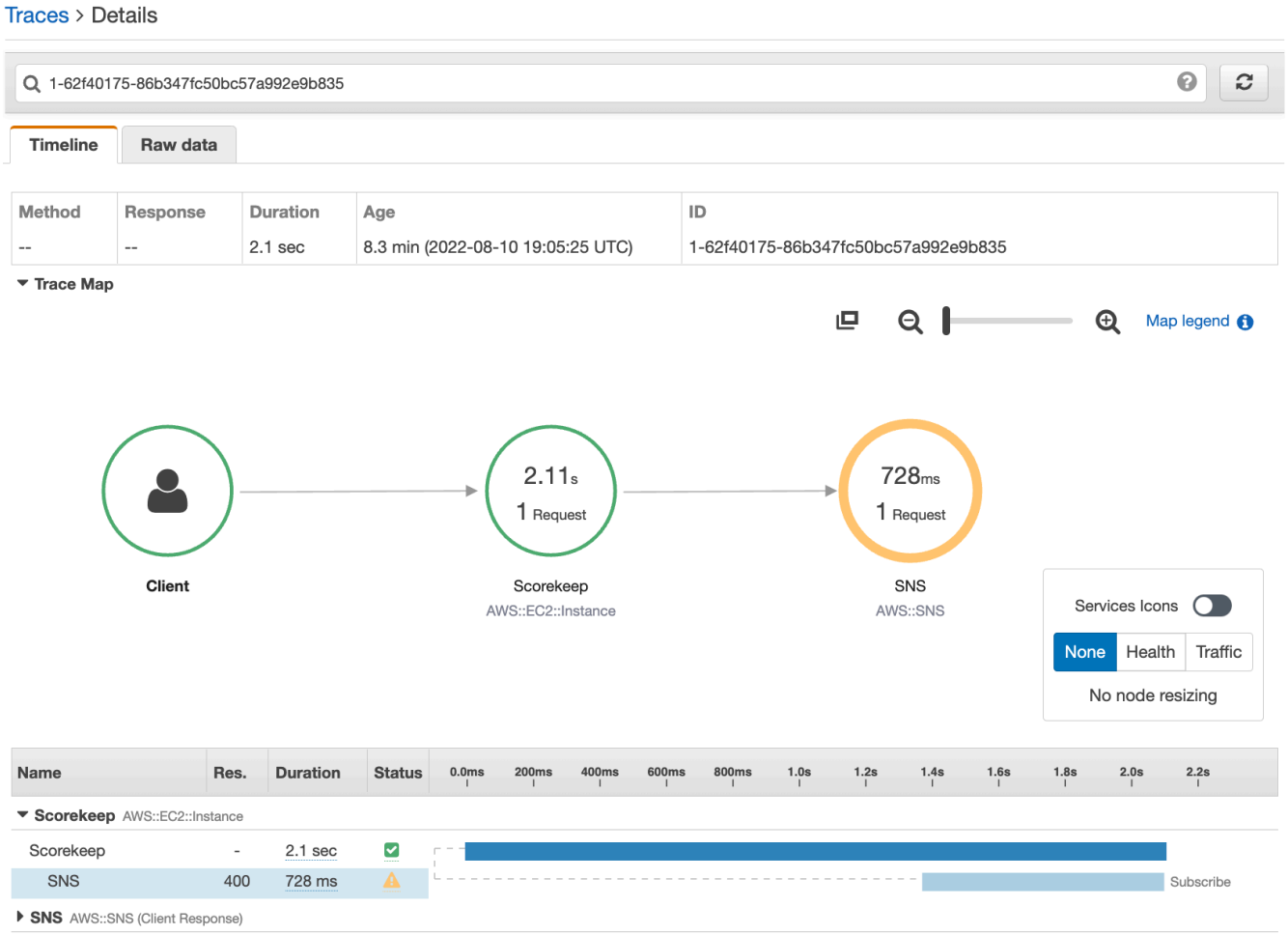

So finden Sie die Ursache des Fehlers

1. Wählen Sie den Knoten mit dem Namen SNS. Der Bereich mit den Knotendetails wird angezeigt.
2. Wählen Sie View traces (Ablaufverfolgungen anzeigen), um auf den Bildschirm Trace overview (Ablaufverfolgungsübersicht) zuzugreifen.

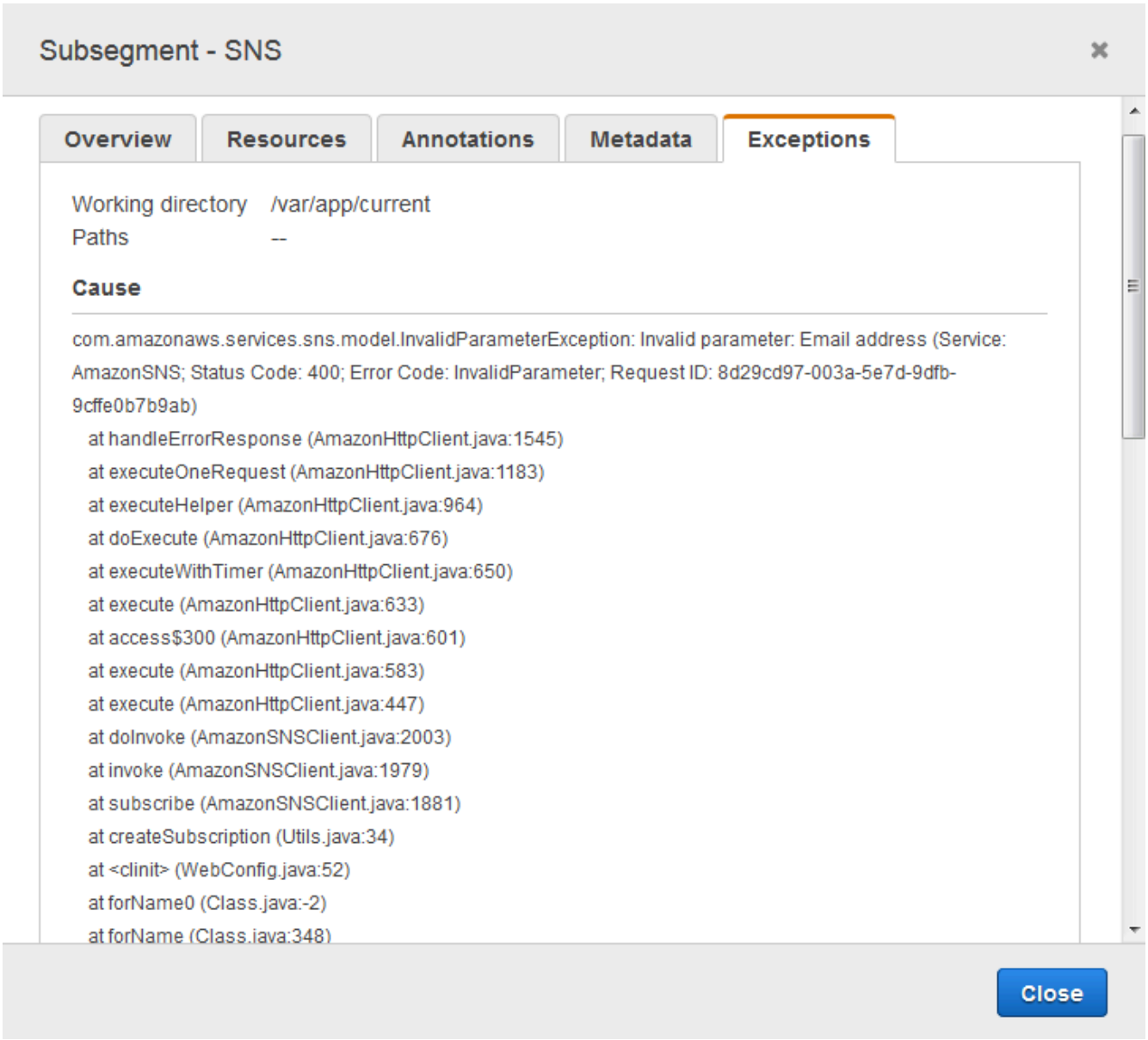
3. Wählen Sie die Nachverfolgung aus der Trace list. Diese Ablaufverfolgung verfügt über keine Methode oder URL, da sie während des Startups und nicht als Reaktion auf eine eingehende Anforderung aufgezeichnet wurde.




4. Wählen Sie das Fehlerstatussymbol im Amazon SNS-Segment unten auf der Seite, um die Seite Ausnahmen für das SNS-Teilsegment zu öffnen.



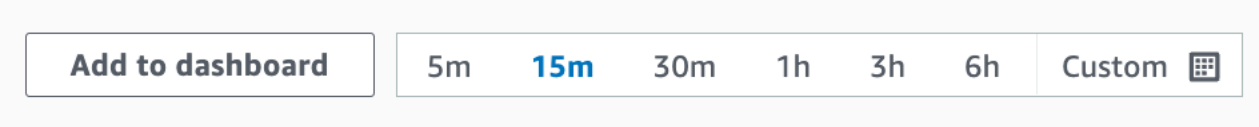

5. Das X-Ray-SDK erfasst automatisch Ausnahmen, die von instrumentierten AWS SDK-Clients ausgelöst werden, und zeichnet die Stack-Ablaufverfolgung auf.




CloudWatch console

Verwenden der CloudWatch Konsole

1. Öffnen Sie die X-Ray-Trace-Übersichtsseite der - CloudWatch Konsole.
2. Die Konsole zeigt eine Darstellung des Servicediagramms, das X-Ray aus den von der Anwendung gesendeten Ablaufverfolgungsdaten generiert. Passen Sie den Zeitraum der Trace-Übersicht bei Bedarf an, um sicherzustellen, dass alle Traces seit dem ersten Start der Webanwendung angezeigt werden.



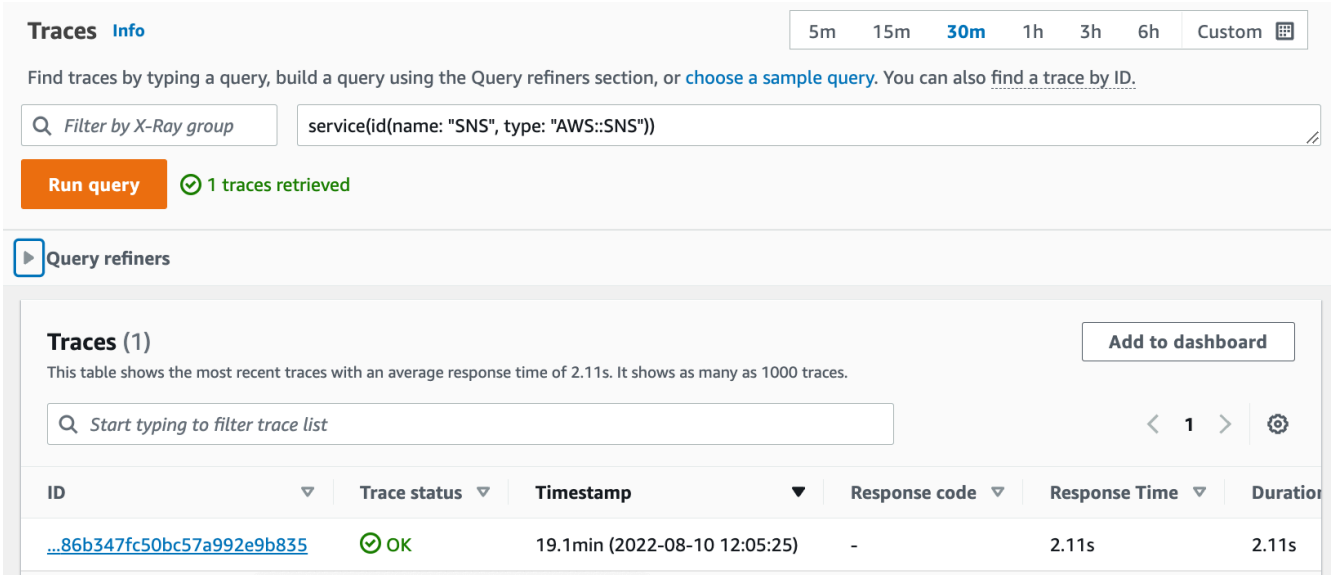

Die Ablaufverfolgungszuordnung zeigt den Web-App-Client, die in Amazon EC2 ausgeführte API und jede DynamoDB-Tabelle, die die Anwendung verwendet. Jede Anforderung an die Anwendung, bis zu einer konfigurierbaren Höchstanzahl von Anforderungen pro Sekunde, wird verfolgt, und zwar wie sie auf die API trifft, Anforderungen an nachgelagerte Services generiert und abgeschlossen wird.

Sie können jeden Knoten in der Service-Grafik wählen, um Ablaufverfolgungen für Anforderungen anzusehen, die Datenverkehr zu diesem Knoten generiert haben. Derzeit ist der Amazon SNS-Knoten orange. Zeigen Sie die Details an, um herauszufinden, warum.

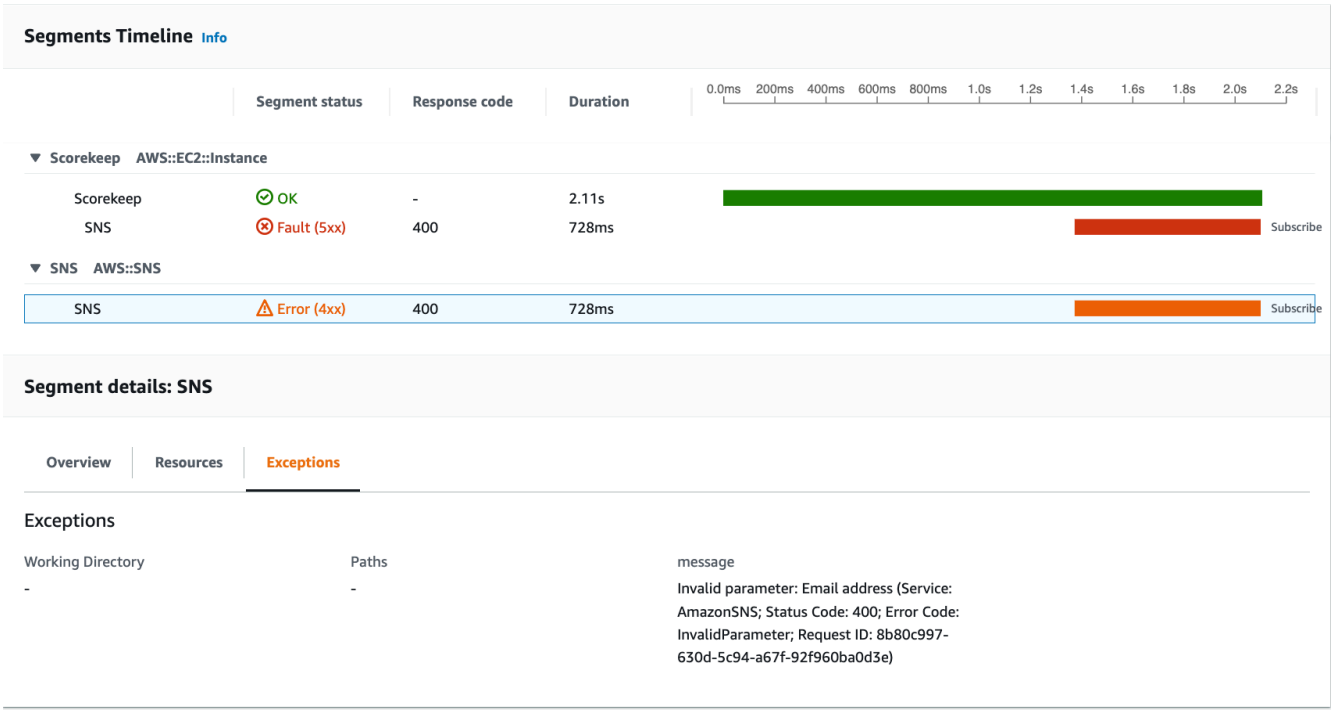

scorekeep-state
DynamoDB Table
scorekeep-game
DynamoDB Table
scorekeep-move
DynamoDB Table
Client
Scorekeep
ECS Container
SNS
SNS
scorekeep-session
DynamoDB Table

So finden Sie die Ursache des Fehlers

1. Wählen Sie den Knoten mit dem Namen SNS. Das Detailfenster des SNS-Knotens wird unter der Karte angezeigt.
2. Wählen Sie Traces anzeigen, um auf die Seite Traces zuzugreifen.
3. Fügen Sie unten auf der Seite den Trace aus der Liste Traces hinzu. Diese Ablaufverfolgung verfügt über keine Methode oder URL, da sie während des Startups und nicht als Reaktion auf eine eingehende Anforderung aufgezeichnet wurde.




4. Wählen Sie das Amazon SNS-Teilsegment unten in der Segmentzeitleiste und dann die Registerkarte Ausnahmen für das SNS-Teilsegment aus, um die Ausnahmedetails anzuzeigen.




Die Ursache gibt an, dass die E-Mail-Adresse, die in einem Aufruf von `createSubscription` in der Klasse `WebConfig` angegeben wurde, ungültig ist. Im nächsten Abschnitt beheben wir das.

## Konfigurieren von Amazon-SNS-Benachrichtigungen

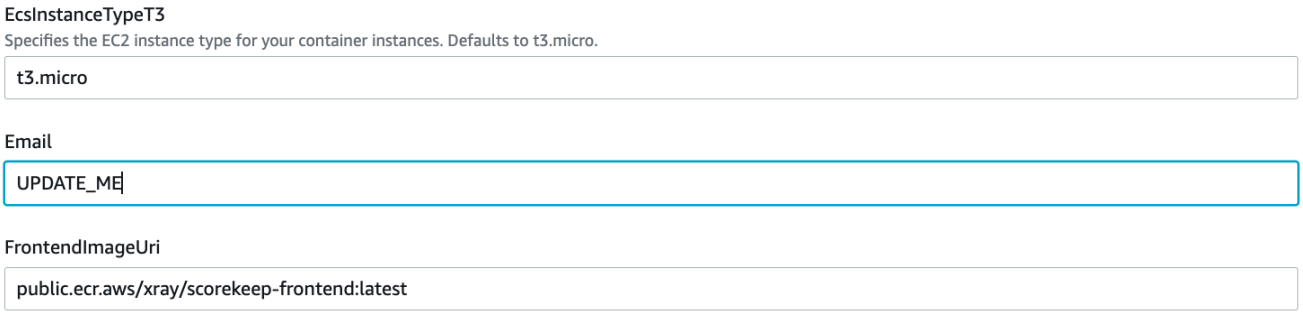
Scorekeep verwendet Amazon SNS, um Benachrichtigungen zu senden, wenn Benutzer ein Spiel beenden. Wenn die Anwendung gestartet wird, versucht sie, ein Abonnement für eine E-Mail-Adresse zu erstellen, die in einem CloudFormation Stack-Parameter definiert ist. Dieser Aufruf schlägt derzeit fehl. Konfigurieren Sie eine Benachrichtigungs-E-Mail, um Benachrichtigungen zu aktivieren und die in der Ablaufverfolgungszuordnung hervorgehobenen Fehler zu beheben.

AWS Management Console

So konfigurieren Sie Amazon SNS-Benachrichtigungen mit der AWS Management Console

1. Öffnen Sie die CloudFormation-Konsole.
2. Wählen Sie das Optionsfeld neben dem `scorekeep`Stack-Namen in der Liste und dann Aktualisieren aus.
3. Stellen Sie sicher, dass Aktuelle Vorlage verwenden ausgewählt ist, und klicken Sie dann auf der Seite Stack aktualisieren auf Weiter.

4. Suchen Sie den Parameter E-Mail in der Liste und ersetzen Sie den Standardwert durch eine gültige E-Mail-Adresse.




5. Scrollen Sie ans Seitenende und wählen Sie Next (Weiter).
6. Scrollen Sie nach unten auf der Seite Überprüfen, wählen Sie das Kontrollkästchen Bestätigung, das IAM-Ressourcen mit benutzerdefinierten Namen erstellen CloudFormation kann, und wählen Sie Stack aktualisieren aus.
7. Der CloudFormation Stack wird jetzt aktualisiert. Der Stack-Status beträgt etwa fünf MinutenUPDATE_IN_PROGRESS, bevor Sie zu wechselnUPDATE_COMPLETE. Der Status wird regelmäßig aktualisiert, oder Sie können die Seite aktualisieren.

AWS CLI

So konfigurieren Sie Amazon SNS-Benachrichtigungen mit der AWS CLI

1. Navigieren Sie zu dem `xray-scorekeep/cloudformation/` Ordner, den Sie zuvor erstellt haben, und öffnen Sie die `cf-resources.yaml` Datei in einem Texteditor.
2. Suchen Sie den `Default` Wert im Parameter E-Mail und ändern Sie ihn von *`UPDATE_ME`* in eine gültige E-Mail-Adresse.

```
Parameters:
  Email:
    Type: String
    Default: UPDATE_ME # <- change to a valid abc@def.xyz email address
```

3. Aktualisieren Sie den CloudFormation Stack im `cloudformation` Ordner mit dem folgenden AWS CLI Befehl:

```
aws cloudformation update-stack --stack-name scorekeep --capabilities
 "CAPABILITY_NAMED_IAM" --template-body file://cf-resources.yaml
```

4. Warten Sie, bis der CloudFormation Stack-Status lautet`UPDATE_COMPLETE`, was einige Minuten dauert. Verwenden Sie den folgenden AWS CLI Befehl, um den Status zu überprüfen:

```
aws cloudformation describe-stacks --stack-name scorekeep --query
 "Stacks[0].StackStatus"
```

Wenn die Aktualisierung abgeschlossen ist, startet Scorekeep neu und erstellt ein Abonnement für das SNS-Thema. Überprüfen Sie Ihre E-Mail-Adresse und bestätigen Sie das Abonnement, um Aktualisierungen anzuzeigen, wenn Sie ein Spiel abgeschlossen haben. Öffnen Sie die Ablaufverfolgungszuordnung, um zu überprüfen, ob die Aufrufe an SNS nicht mehr fehlschlagen.

## Erkunden der Beispielanwendung

Die Beispielanwendung ist eine HTTP-Web-API in Java, die für die Verwendung des X-Ray SDK for Java konfiguriert ist. Wenn Sie die Anwendung mit der CloudFormation Vorlage bereitstellen, werden die DynamoDB-Tabellen, der Amazon-ECS-Cluster und andere -Services erstellt, die zum Ausführen von Scorekeep auf ECS erforderlich sind. Eine Aufgabendefinitionsdatei für ECS wird über erstellt CloudFormation. Diese Datei definiert die Container-Images, die pro Aufgabe in einem ECS-Cluster verwendet werden. Diese Images erhalten Sie von der offiziellen öffentlichen X-Ray-ECR. Das scorekeep-API-Container-Image enthält die mit Gradle kompilierte API. Das Container-Image des Scorekeep-Frontend-Containers bedient das Frontend mithilfe des nginx-Proxy-Servers. Dieser Server leitet Anfragen an Pfade weiter, die mit /api beginnen, an die API.

Zum Instrumentieren eingehender HTTP-Anforderungen fügt die Anwendung den vom SDK bereitgestellten `TracingFilter` hinzu.

Example src/main/java/scorekeep/WebConfig.java – Servlet-Filter

```
import javax.servlet.Filter;
import com.amazonaws.xray.javax.servlet.AWSXRayServletFilter;
...

@Configuration
public class WebConfig {

  @Bean
  public Filter TracingFilter() {
    return new AWSXRayServletFilter("Scorekeep");
```

```
  }
...
```

Dieser Filter sendet Ablaufverfolgungsdaten über alle eingehenden Anforderungen, die die Anwendung verarbeitet, einschließlich Anforderungs-URL, Methode, Antwortstatus sowie Start- und Endzeit.

Die Anwendung führt auch Downstream-Aufrufe an DynamoDB über die durch AWS SDK for Java. Um diese Aufrufe zu instrumentieren, verwendet die Anwendung einfach die AWS SDK-bezogenen Submodule als Abhängigkeiten, und das X-Ray-SDK für Java instrumentiert automatisch alle AWS SDK-Clients.

Die Anwendung verwendet `Docker`, um den Quellcode auf der Instance mit der zu erstellen, `Gradle Docker Image` und die -`Scorekeep API Dockerfile`Datei, um das ausführbare JAR auszuführen, das Gradle in seiner generiert`ENTRYPOINT`.

Example Verwendung von Docker zum Erstellen über Gradle Docker Image

```
docker run --rm -v /PATH/TO/SCOREKEEP_REPO/home/gradle/project -w /home/gradle/project
 gradle:4.3 gradle build
```

Example Dockerfile-Eintragspunkt

```
ENTRYPOINT [ "sh", "-c", "java -Dserver.port=5000 -jar scorekeep-api-1.0.0.jar" ]
```

Die `build.gradle`-Datei lädt die SDK-Untermodule von Maven während der Kompilierung herunter, indem sie zu Abhängigkeiten erklärt werden.

Example build.gradle – Abhängigkeiten

```
...
dependencies {
    compile("org.springframework.boot:spring-boot-starter-web")
    testCompile('org.springframework.boot:spring-boot-starter-test')
    compile('com.amazonaws:aws-java-sdk-dynamodb')
    compile("com.amazonaws:aws-xray-recorder-sdk-core")
    compile("com.amazonaws:aws-xray-recorder-sdk-aws-sdk")
    compile("com.amazonaws:aws-xray-recorder-sdk-aws-sdk-instrumentor")
    ...
}
```

```
dependencyManagement {
    imports {
        mavenBom("com.amazonaws:aws-java-sdk-bom:1.11.67")
        mavenBom("com.amazonaws:aws-xray-recorder-sdk-bom:2.11.0")
    }
}
```

Die Core-, AWS SDK- und AWS SDK-Instrumentor-Submodule sind alles, was erforderlich ist, um alle Downstream-Aufrufe, die mit dem AWS SDK getätigt werden, automatisch zu instrumentieren.

Um die Rohsegmentdaten an die X-Ray-API weiterzuleiten, muss der X-Ray-Daemon auf Datenverkehr auf UDP-Port 2000 achten. Dazu wird der X-Ray-Daemon in einem Container ausgeführt, der zusammen mit der Scorekeep-Anwendung auf ECS als Sidecar-Container bereitgestellt wird. Weitere Informationen finden Sie im Thema X-Ray-Daemon.

Example X-Ray-Daemon-Containerdefinition in einer ECS-Aufgabendefinition

```
...
Resources:
  ScorekeepTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ContainerDefinitions:
      ...

      - Cpu: '256'
        Essential: true
        Image: amazon/aws-xray-daemon
        MemoryReservation: '128'
        Name: xray-daemon
        PortMappings:
          - ContainerPort: '2000'
            HostPort: '2000'
            Protocol: udp
      ...
```

Das X-Ray SDK for Java bietet eine Klasse mit dem Namen `AWSXRay`, die den globalen Recorder bereitstellt, einen `TracingHandler` , mit dem Sie Ihren Code instrumentieren können. Sie können die globale Aufzeichnung so konfigurieren, dass der `AWSXRayServletFilter`, der Segmente für eingehende HTTP-Aufrufe erstellt, angepasst wird. Das Beispiel enthält einen statischen Block in der `WebConfig`-Klasse, der die globale Aufzeichnung mit Plugins und Samplingregeln konfiguriert.

Example src/main/java/scorekeep/WebConfig.java – Recorder

```
import com.amazonaws.xray.AWSXRay;
import com.amazonaws.xray.AWSXRayRecorderBuilder;
import com.amazonaws.xray.javax.servlet.AWSXRayServletFilter;
import com.amazonaws.xray.plugins.ECSPlugin;
import com.amazonaws.xray.plugins.EC2Plugin;
import com.amazonaws.xray.strategy.sampling.LocalizedSamplingStrategy;
...

@Configuration
public class WebConfig {
  ...

  static {
    AWSXRayRecorderBuilder builder = AWSXRayRecorderBuilder.standard().withPlugin(new
 ECSPlugin()).withPlugin(new EC2Plugin());

    URL ruleFile = WebConfig.class.getResource("/sampling-rules.json");
    builder.withSamplingStrategy(new LocalizedSamplingStrategy(ruleFile));

    AWSXRay.setGlobalRecorder(builder.build());
    ...

  }
}
```

In diesem Beispiel wird der Builder zum Laden von Samplingregeln aus einer Datei namens `sampling-rules.json` verwendet. Samplingregeln bestimmen die Rate, mit der das SDK Segmente für eingehende Anforderungen aufzeichnet.

Example src/main/java/resources/sampling-rules.json

```
{
  "version": 1,
  "rules": [
    {
      "description": "Resource creation.",
      "service_name": "*",
      "http_method": "POST",
      "url_path": "/api/*",
      "fixed_target": 1,
      "rate": 1.0
```

```
    },
    {
      "description": "Session polling.",
      "service_name": "*",
      "http_method": "GET",
      "url_path": "/api/session/*",
      "fixed_target": 0,
      "rate": 0.05
    },
    {
      "description": "Game polling.",
      "service_name": "*",
      "http_method": "GET",
      "url_path": "/api/game/*/*",
      "fixed_target": 0,
      "rate": 0.05
    },
    {
      "description": "State polling.",
      "service_name": "*",
      "http_method": "GET",
      "url_path": "/api/state/*/*/*",
      "fixed_target": 0,
      "rate": 0.05
    }
  ],
  "default": {
    "fixed_target": 1,
    "rate": 0.1
  }
}
```

Die Samplingregeldatei definiert vier benutzerdefinierte Samplingregeln und die Standardregel. Für jede eingehende Anforderung wertet das SDK die benutzerdefinierten Regeln in der Reihenfolge aus, in der sie definiert sind. Das SDK wendet die erste Regel an, die der Methode, dem Pfad und dem Service-Namen der Anforderung entspricht. Bei Scorekeep fängt die erste Regel alle POST-Anforderungen (Aufrufe zum Erstellen von Ressourcen) ab, indem pro Sekunde ein festes Ziel einer einzelnen Anfrage und eine Rate von 1.0 angewendet wird bzw. 100 % der Anforderungen nach dem festen Ziel erfüllt sind.

Die übrigen drei benutzerdefinierte Regeln wenden eine Rate von fünf Prozent ohne festes Ziel auf Sitzungs-, Spiel- und Statuslesevorgänge (GET-Anfragen) an. Dies minimiert die Anzahl der

Ablaufverfolgungen für regelmäßige Aufrufe, die das Front-End automatisch alle paar Sekunden ausführt, um sicherzustellen, dass die Inhalte auf dem neuesten Stand sind. Für alle übrigen Anforderungen definiert die Datei eine Standardrate von einer einzelnen Anfrage pro Sekunde und einer Rate von 10 Prozent.

Die Beispielanwendung zeigt auch, wie Sie erweiterte Funktionen wie die manuelle SDK-Client-Instrumentierung verwenden sowie zusätzliche Untersegmente und ausgehende HTTP-Aufrufe erstellen. Weitere Informationen finden Sie unter AWS X-Ray Beispielanwendung.

## Optional: Richtlinie für geringste Rechte

Die Scorekeep-ECS-Container greifen mithilfe von Vollzugriffsrichtlinien wie `AmazonSNSFullAccess` und auf Ressourcen zu`AmazonDynamoDBFullAccess`. Die Verwendung von Vollzugriffsrichtlinien ist keine bewährte Methode für Produktionsanwendungen. Im folgenden Beispiel wird die DynamoDB-IAM-Richtlinie aktualisiert, um die Sicherheit der Anwendung zu verbessern. Weitere Informationen zu bewährten Sicherheitsmethoden in IAM-Richtlinien finden Sie unter Identity and Access Management für AWS X-Ray.

Example ECS-TaskRole Definition für cf-resources.yaml-Vorlage

```
ECSTaskRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          -
            Effect: "Allow"
            Principal:
              Service:
                - "ecs-tasks.amazonaws.com"
            Action:
              - "sts:AssumeRole"
      ManagedPolicyArns:
        - "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess"
        - "arn:aws:iam::aws:policy/AmazonSNSFullAccess"
        - "arn:aws:iam::aws:policy/AWSXrayFullAccess"
      RoleName: "scorekeepRole"
```

Um Ihre Richtlinie zu aktualisieren, identifizieren Sie zunächst den ARN Ihrer DynamoDB-Ressourcen. Anschließend verwenden Sie den ARN in einer benutzerdefinierten IAM-Richtlinie. Schließlich wenden Sie diese Richtlinie auf Ihr Instance-Profil an.

So identifizieren Sie den ARN Ihrer DynamoDB-Ressource:

1. Öffnen Sie die DynamoDB-Konsole.
2. Wählen Sie in der linken Navigationsleiste Tabellen aus.
3. Wählen Sie eine der aus`scorekeep-*`, um die Tabellendetailseite anzuzeigen.
4. Wählen Sie auf der Registerkarte Übersicht die Option Zusätzliche Informationen aus, um den Abschnitt zu erweitern und den Amazon-Ressourcennamen (ARN) anzuzeigen. Kopieren Sie diesen Wert.
5. Fügen Sie den ARN in die folgende IAM-Richtlinie ein und ersetzen Sie die `AWS_ACCOUNT_ID` Werte `AWS_REGION` und durch Ihre spezifische Region und Konto-ID. Diese neue Richtlinie erlaubt nur die angegebenen Aktionen anstelle der `AmazonDynamoDBFullAccess` Richtlinie, die jede Aktion zulässt.

   Example

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ScorekeepDynamoDB",
            "Effect": "Allow",
            "Action": [
                "dynamodb:PutItem",
                "dynamodb:UpdateItem",
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:Scan",
                "dynamodb:Query"
            ],
            "Resource": "arn:aws:dynamodb:<AWS_REGION>:<AWS_ACCOUNT_ID>:table/
scorekeep-*"
        }
    ]
}
```

Die Tabellen, die über die Anwendung erstellt werden, folgen einer einheitlichen Namenskonvention. Sie können das `scorekeep-*` Format verwenden, um alle Scorekeep-Tabellen anzugeben.

Ändern Ihrer IAM-Richtlinie

1. Öffnen Sie die Scorekeep-Aufgabenrolle (scorekeepRole) in der IAM-Konsole.
2. Aktivieren Sie das Kontrollkästchen neben der `AmazonDynamoDBFullAccess` Richtlinie und wählen Sie Entfernen, um diese Richtlinie zu entfernen.
3. Wählen Sie Berechtigungen hinzufügen und dann Richtlinien anfügen und schließlich Richtlinie erstellen aus.
4. Wählen Sie die Registerkarte JSON und fügen Sie die oben erstellte Richtlinie ein.
5. Wählen Sie unten auf der Seite Weiter: Tags aus.
6. Wählen Sie unten auf der Seite Weiter: Überprüfen aus.
7. Weisen Sie für Name einen Namen für die Richtlinie zu.
8. Wählen Sie unten auf der Seite Richtlinie erstellen aus.
9. Fügen Sie die neu erstellte Richtlinie an die `scorekeepRole` Rolle an. Es kann einige Minuten dauern, bis die angehängte Richtlinie wirksam wird.

Wenn Sie die neue Richtlinie an die `scorekeepRole` Rolle angehängt haben, müssen Sie sie vor dem Löschen des CloudFormation Stacks trennen, da diese angehängte Richtlinie das Löschen des Stacks verhindert. Die Richtlinie kann durch Löschen der Richtlinie automatisch getrennt werden.

Entfernen Ihrer benutzerdefinierten IAM-Richtlinie

1. Öffnen Sie die IAM-Konsole.
2. Wählen Sie in der linken Navigationsleiste Richtlinien aus.
3. Suchen Sie nach dem benutzerdefinierten Richtliniennamen, den Sie zuvor in diesem Abschnitt erstellt haben, und wählen Sie das Optionsfeld neben dem Richtliniennamen aus, um ihn hervorzuheben.
4. Wählen Sie das Dropdown-Menü Aktionen und dann Löschen aus.
5. Geben Sie den Namen der benutzerdefinierten Richtlinie ein und wählen Sie dann Löschen, um das Löschen zu bestätigen. Dadurch wird die Richtlinie automatisch von der `scorekeepRole` Rolle getrennt.

# Bereinigen

Gehen Sie wie folgt vor, um die Ressourcen der Scorekeep-Anwendung zu löschen:

> **Note**
>
> Wenn Sie benutzerdefinierte Richtlinien im vorherigen Abschnitt dieses Tutorials erstellt und angehängt haben, müssen Sie die Richtlinie aus dem entfernen, `scorekeepRole` bevor Sie den CloudFormation Stack löschen.

AWS Management Console

Löschen der Beispielanwendung mithilfe der AWS Management Console

1. Öffnen Sie die CloudFormation-Konsole.
2. Wählen Sie das Optionsfeld neben dem `scorekeep`Stack-Namen in der Liste und dann Löschen aus.
3. Der CloudFormation Stack wird jetzt gelöscht. Der Stack-Status lautet `DELETE_IN_PROGRESS` einige Minuten, bis alle Ressourcen gelöscht werden. Der Status wird regelmäßig aktualisiert, oder Sie können die Seite aktualisieren.

AWS CLI

Löschen der Beispielanwendung mithilfe der AWS CLI

1. Geben Sie den folgenden AWS CLI Befehl ein, um den CloudFormation Stack zu löschen:

   ```
   aws cloudformation delete-stack --stack-name scorekeep
   ```

2. Warten Sie, bis der CloudFormation Stack nicht mehr existiert, was etwa fünf Minuten dauert. Verwenden Sie den folgenden AWS CLI Befehl, um den Status zu überprüfen:

   ```
   aws cloudformation describe-stacks --stack-name scorekeep --query
    "Stacks[0].StackStatus"
   ```

## Nächste Schritte

Weitere Informationen zu X-Ray finden Sie im nächsten Kapitel, Konzepte.

Um Ihre eigene App zu instrumentieren, erfahren Sie mehr über das X-Ray SDK for Java oder eines der anderen X-Ray SDKs:

- X-Ray SDK für Java – AWS X-Ray SDK für Java
- X-Ray SDK für Node.js – AWS X-Ray-SDK für Node.js
- X-Ray SDK für .NET – AWS X-Ray SDK für .NET

Informationen zum lokalen Ausführen des X-Ray-Daemons oder auf AWS finden Sie unter AWS X-Ray Dämon.

Informationen zum Beitrag zur Beispielanwendung in GitHubfinden Sie unter eb-java-scorekeep.

# Manuelles Instrumentieren von AWS SDK-Clients

Das X-Ray SDK for Java instrumentiert automatisch alle AWS SDK-Clients, wenn Sie das AWS SDK Instrumentor-Submodul in Ihre Build-Abhängigkeiten aufnehmen.

Sie können die automatische Client-Instrumentierung durch Löschen des Instrumentor-Untermoduls deaktivieren. Auf diese Weise können Sie einige Clients manuell instrumentieren und andere ignorieren oder verschiedene Tracing-Handler auf unterschiedlichen Clients anwenden.

Um die Unterstützung für die Instrumentierung bestimmter AWS SDK-Clients zu veranschaulichen, übergibt die Anwendung einen Ablaufverfolgungs-Handler `AmazonDynamoDBClientBuilder` an als Anforderungs-Handler im Benutzer-, Spiel- und Sitzungsmodell. Diese Codeänderung weist das SDK an, alle Aufrufe an DynamoDB mit diesen Clients zu instrumentieren.

Example **`src/main/java/scorekeep/SessionModel.java`** – Manuelle AWS SDK-Client-Instrumentierung

```
import com.amazonaws.xray.AWSXRay;
import com.amazonaws.xray.handlers.TracingHandler;

public class SessionModel {
  private AmazonDynamoDB client = AmazonDynamoDBClientBuilder.standard()
        .withRegion(Constants.REGION)
        .withRequestHandlers(new TracingHandler(AWSXRay.getGlobalRecorder()))
```

```
        .build();
  private DynamoDBMapper mapper = new DynamoDBMapper(client);
```

Wenn Sie das AWS SDK-Instrumentor-Submodul aus Projektabhängigkeiten entfernen, werden nur die manuell instrumentierten AWS SDK-Clients in der Ablaufverfolgungszuordnung angezeigt.

## Erstellen zusätzlicher Untersegmente

In der Benutzermodellklasse erstellt die Anwendung manuell Untersegmente, um alle nachgelagerten Aufrufe, die innerhalb der `saveUser`-Funktion vorgenommen wurden, zu gruppieren, und fügt Metadaten hinzu.

Example **src/main/java/scorekeep/UserModel.java** – Benutzerdefinierte Untersegmente

```
import com.amazonaws.xray.AWSXRay;
import com.amazonaws.xray.entities.Subsegment;
...
    public void saveUser(User user) {
    // Wrap in subsegment
    Subsegment subsegment = AWSXRay.beginSubsegment("## UserModel.saveUser");
    try {
      mapper.save(user);
    } catch (Exception e) {
      subsegment.addException(e);
      throw e;
    } finally {
      AWSXRay.endSubsegment();
    }
  }
```

## Aufzeichnung von Anmerkungen, Metadaten und Benutzer-IDs

In der Spielmodellklasse zeichnet die Anwendung `Game` jedes Mal Objekte in einem Metadatenblock auf, wenn sie ein Spiel in DynamoDB speichert. Unabhängig davon erfasst die Anwendung Spiel-IDs in Anmerkungen zur Verwendung mit Filterausdrücken.

Example **src/main/java/scorekeep/GameModel.java** – Anmerkungen und Metadaten

```
import com.amazonaws.xray.AWSXRay;
import com.amazonaws.xray.entities.Segment;
import com.amazonaws.xray.entities.Subsegment;
```

```
...
  public void saveGame(Game game) throws SessionNotFoundException {
    // wrap in subsegment
    Subsegment subsegment = AWSXRay.beginSubsegment("## GameModel.saveGame");
    try {
      // check session
      String sessionId = game.getSession();
      if (sessionModel.loadSession(sessionId) == null ) {
        throw new SessionNotFoundException(sessionId);
      }
      Segment segment = AWSXRay.getCurrentSegment();
      subsegment.putMetadata("resources", "game", game);
      segment.putAnnotation("gameid", game.getId());
      mapper.save(game);
    } catch (Exception e) {
      subsegment.addException(e);
      throw e;
    } finally {
      AWSXRay.endSubsegment();
    }
  }
```

Im Bewegungs-Controller erfasst die Anwendung Benutzer-IDs mit `setUser`. Benutzer-IDs werden in einem eigenen Feld aufgezeichnet und zur Suche indiziert.

Example src/main/java/scorekeep/MoveController.java – Benutzer-ID

```
import com.amazonaws.xray.AWSXRay;
...
  @RequestMapping(value="/{userId}", method=RequestMethod.POST)
  public Move newMove(@PathVariable String sessionId, @PathVariable String
 gameId, @PathVariable String userId, @RequestBody String move) throws
 SessionNotFoundException, GameNotFoundException, StateNotFoundException,
 RulesException {
    AWSXRay.getCurrentSegment().setUser(userId);
    return moveFactory.newMove(sessionId, gameId, userId, move);
  }
```

## Instrumentieren von ausgehenden HTTP-Aufrufen

Die User Factory-Klasse zeigt, wie die Anwendung die Version von X-Ray SDK for Java verwendet`HTTPClientBuilder`, um ausgehende HTTP-Aufrufe zu instrumentieren.

Example **src/main/java/scorekeep/UserFactory.java** – HTTPClient-Instrumentierung

```
import com.amazonaws.xray.proxies.apache.http.HttpClientBuilder;

  public String randomName() throws IOException {
    CloseableHttpClient httpclient = HttpClientBuilder.create().build();
    HttpGet httpGet = new HttpGet("http://uinames.com/api/");
    CloseableHttpResponse response = httpclient.execute(httpGet);
    try {
      HttpEntity entity = response.getEntity();
      InputStream inputStream = entity.getContent();
      ObjectMapper mapper = new ObjectMapper();
      Map<String, String> jsonMap = mapper.readValue(inputStream, Map.class);
      String name = jsonMap.get("name");
      EntityUtils.consume(entity);
      return name;
    } finally {
      response.close();
    }
  }
```

Wenn Sie derzeit `org.apache.http.impl.client.HttpClientBuilder` verwenden, können Sie einfach die Import-Anweisung für die Klasse mit einer Anweisung für `com.amazonaws.xray.proxies.apache.http.HttpClientBuilder` austauschen.

## Instrumentieren von Aufrufen einer PostgreSQL-Datenbank

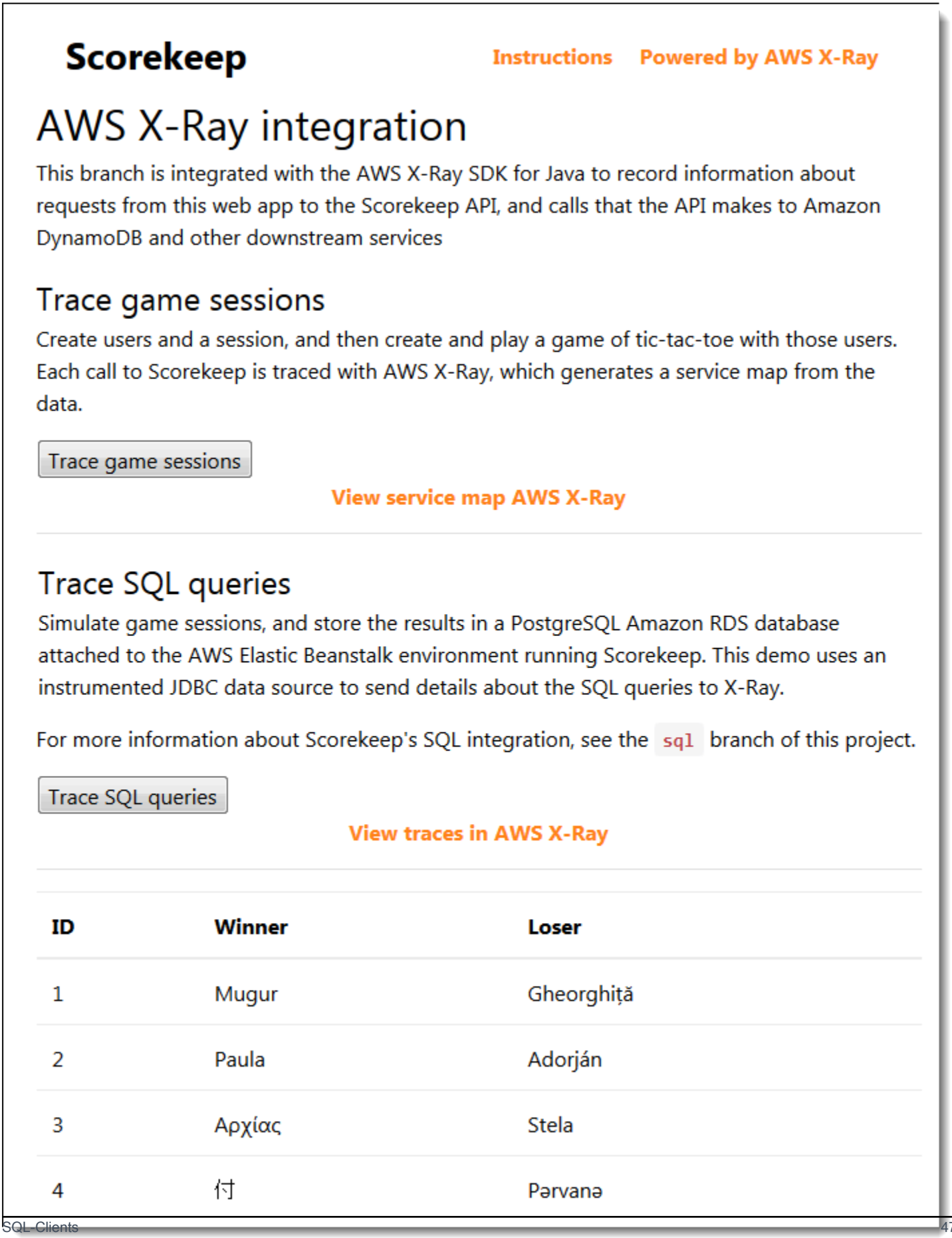
Die`application-pgsql.properties`Die Datei fügt der in erstellen Datenquelle einen X-Ray PostgreSQL-Ablaufverfolgungs-Interceptor hinzu`RdsWebConfig.java`aus.

Example **application-pgsql.properties**— PostgreSQL-Datenbank-Instrumentierung

```
spring.datasource.continue-on-error=true
spring.jpa.show-sql=false
spring.jpa.hibernate.ddl-auto=create-drop
spring.datasource.jdbc-interceptors=com.amazonaws.xray.sql.postgres.TracingInterceptor
spring.jpa.database-platform=org.hibernate.dialect.PostgreSQL94Dialect
```

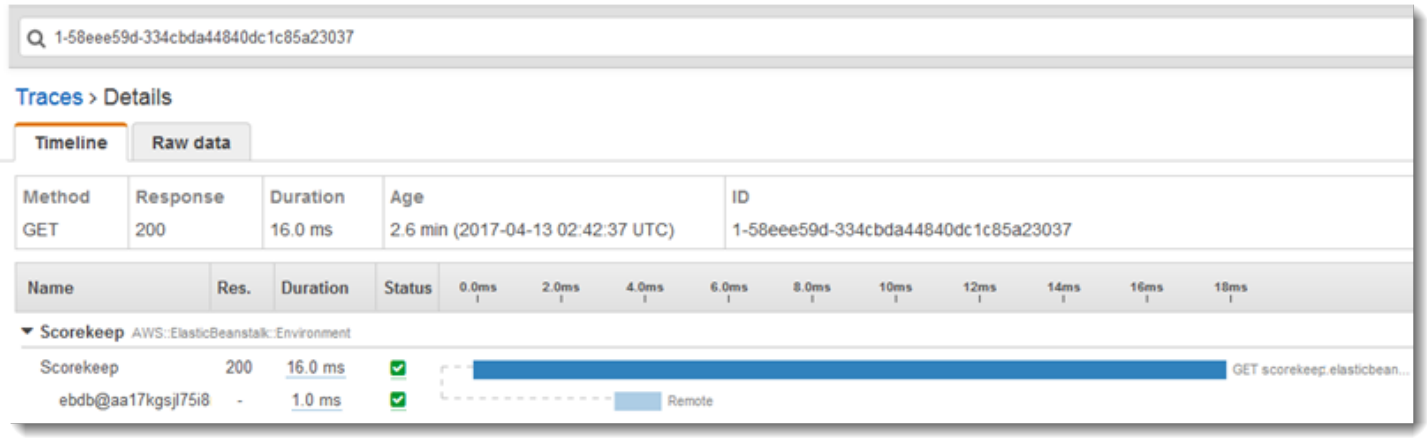
> **Note**
>
> Weitere Informationen zum Hinzufügen einer PostgreSQL-Datenbank zum Anwendungsumfeld finden Sie unter Konfiguration von Datenbanken mit Elastic Beanstalk im AWS Elastic Beanstalk-Entwicklerhandbuch.

Die X-Ray -Demo-Seite im`xray`-Bereich umfasst eine Demonstration, die instrumentierte Datenquellen nutzt, um Ablaufverfolgungen zu generieren, die Informationen zu den erzeugten SQL-Abfragen anzeigt. Navigieren Sie zu dem `/#/xray`-Pfad in der laufenden Anwendung oder wählen Sie in der Navigationsleiste Powered by AWS X-Ray aus, um die Demo-Seite anzusehen.


Scorekeep
Instructions
Powered by AWS X-Ray
AWS X-Ray integration
This branch is integrated with the AWS X-Ray SDK for Java to record information about requests from this web app to the Scorekeep API, and calls that the API makes to Amazon DynamoDB and other downstream services
Trace game sessions
Create users and a session, and then create and play a game of tic-tac-toe with those users. Each call to Scorekeep is traced with AWS X-Ray, which generates a service map from the data.
Trace game sessions
View service map AWS X-Ray
Trace SQL queries
Simulate game sessions, and store the results in a PostgreSQL Amazon RDS database attached to the AWS Elastic Beanstalk environment running Scorekeep. This demo uses an instrumented JDBC data source to send details about the SQL queries to X-Ray.
For more information about Scorekeep's SQL integration, see the sql branch of this project.
Trace SQL queries
View traces in AWS X-Ray
ID
Winner
Loser
1
Mugur
Gheorghiță
2
Paula
Adorján
3
Αρχίας
Stela
4
付
Pərvanə

Wählen Sie SQL-Abfragen nachverfolgen aus, um Spielsitzungen zu simulieren und die Ergebnisse in der zugehörigen Datenbank zu speichern. Dann wähleAnzeigen von AblaufverfolgungenAWSX-Rayum eine gefilterte Liste von Ablaufverfolgungen innerhalb der API anzusehen`/api/history`-Route.

Wählen Sie eine der Ablaufverfolgungen aus der Liste aus, um die Zeitleiste, einschließlich der SQL-Abfrage, ansehen zu können.




## AWS Lambda Instrumentierungsfunktionen

Scorekeep verwendet zwei AWS Lambda Funktionen. Bei der ersten handelt es sich um eine Node.js-Funktion der `lambda`-Verzweigung, die zufällige Namen für neue Benutzer generiert. Wenn ein Benutzer eine Sitzung erstellt, ohne einen Namen einzugeben, ruft die Anwendung eine Funktion namens `random-name` mit dem AWS SDK for Java auf. Das X-Ray SDK for Java zeichnet Informationen über den Aufruf an Lambda in einem Untersegment wie jeder andere Aufruf auf, der mit einem instrumentierten AWS SDK-Client getätigt wurde.

> **Note**
>
> Das Ausführen der `random-name` Lambda-Funktion erfordert die Erstellung zusätzlicher Ressourcen außerhalb der Elastic Beanstalk-Umgebung. Weitere Informationen und Anweisungen finden Sie in der Readme: AWS Lambda Integration .

Die zweite Funktion, `scorekeep-worker`, ist eine Python-Funktion, die unabhängig von der Scorekeep-API ausgeführt wird. Wenn ein Spiel endet, schreibt die API die Sitzungs- und Spiele-ID in eine SQS-Warteschlange. Die Worker-Funktion liest Elemente aus der Warteschlange und ruft die

Scorekeep-API auf, um vollständige Datensätze jeder Spielsitzung für die Speicherung in Amazon S3 zu erstellen.

Scorekeep enthält AWS CloudFormation Vorlagen und Skripts zum Erstellen beider Funktionen. Da Sie das X-Ray-SDK mit dem Funktionscode bündeln müssen, erstellen die Vorlagen die Funktionen ohne Code. Wenn Sie Scorekeep bereitstellen, erstellt eine im Ordner `.ebextensions` enthaltene Konfigurationsdatei ein Quell-Bundle, in dem das SDK enthalten ist, und aktualisiert den Funktionscode und die Konfiguration mit der AWS Command Line Interface.

Funktionen

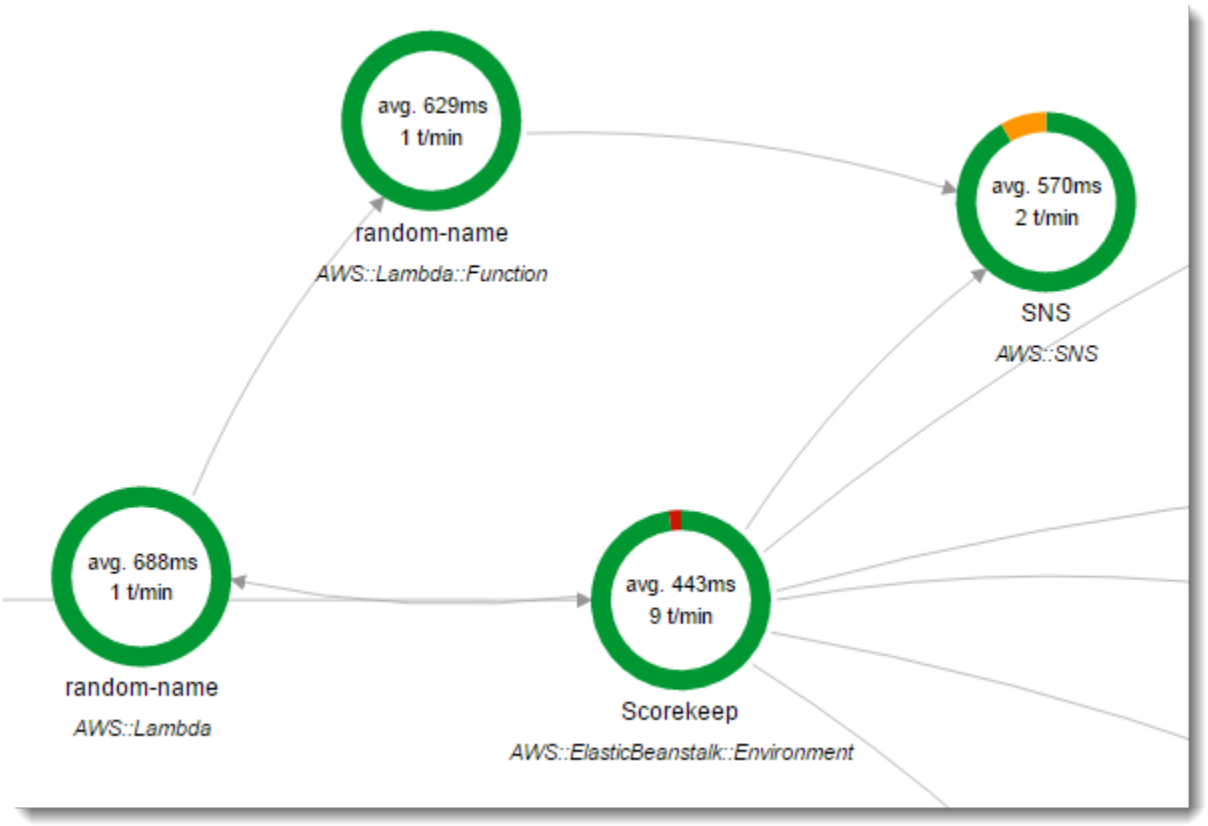
- Zufälliger Name
- Worker

## Zufälliger Name

Scorekeep ruft die Funktion für zufällige Namen auf, wenn ein Benutzer eine Spielesitzung beginnt, ohne dass er sich anmeldet oder einen Benutzernamen angibt. Wenn Lambda den Aufruf an verarbeitet`random-name`, liest es den Ablaufverfolgungs-Header , der die Ablaufverfolgungs-ID und die Sampling-Entscheidung enthält, die vom X-Ray SDK for Java geschrieben wurden.

Für jede Stichprobenanforderung führt Lambda den X-Ray-Daemon aus und schreibt zwei Segmente. Das erste Segment zeichnet Informationen über den Aufruf an Lambda auf, der die Funktion aufruft. Dieses Segment enthält dieselben Informationen wie das von Scorekeep aufgezeichnete Teilsegment, jedoch aus Sicht von Lambda. Das zweite Segment repräsentiert die von der Funktion durchgeführte Arbeit.

Lambda übergibt das Funktionssegment über den Funktionskontext an das X-Ray-SDK. Wenn Sie eine Lambda-Funktion instrumentieren, verwenden Sie das SDK nicht, um ein Segment für eingehende Anforderungen zu erstellen. Lambda stellt das Segment bereit, und Sie verwenden das SDK, um Clients zu instrumentieren und Untersegmente zu schreiben.




Die `random-name`-Funktion wird in Node.js implementiert. Es verwendet das SDK für JavaScript in Node.js, um Benachrichtigungen mit Amazon SNS zu senden, und das X-Ray SDK für Node.js, um den AWS SDK-Client zu instrumentieren. Zum Schreiben von Anmerkungen erstellt die Funktion ein benutzerdefiniertes Untersegment mit `AWSXRay.captureFunc` und schreibt die Anmerkungen in die instrumentierte Funktion. In Lambda können Sie keine Anmerkungen direkt in das Funktionssegment schreiben, sondern nur in ein von Ihnen erstelltes Untersegment.

Example **`function/index.js`** – Zufallsname-Lambda-Funktion

```
var AWSXRay = require('aws-xray-sdk-core');
var AWS = AWSXRay.captureAWS(require('aws-sdk'));

AWS.config.update({region: process.env.AWS_REGION});
var Chance = require('chance');

var myFunction = function(event, context, callback) {
  var sns = new AWS.SNS();
  var chance = new Chance();
  var userid = event.userid;
  var name = chance.first();
```

```
  AWSXRay.captureFunc('annotations', function(subsegment){
    subsegment.addAnnotation('Name', name);
    subsegment.addAnnotation('UserID', event.userid);
  });

  // Notify
  var params = {
    Message: 'Created randon name "' + name + '"" for user "' + userid + '".',
    Subject: 'New user: ' + name,
    TopicArn: process.env.TOPIC_ARN
  };
  sns.publish(params, function(err, data) {
    if (err) {
      console.log(err, err.stack);
      callback(err);
    }
    else {
      console.log(data);
      callback(null, {"name": name});
    }
  });
};

exports.handler = myFunction;
```

Diese Funktion wird automatisch erstellt, wenn Sie die Beispielanwendung für Elastic Beanstalk bereitstellen. Der `xray` Zweig enthält ein Skript zum Erstellen einer leeren Lambda-Funktion. Konfigurationsdateien im `.ebextensions` Ordner erstellen das Funktionspaket mit `npm install` während der Bereitstellung und aktualisieren dann die Lambda-Funktion mit der AWS CLI.

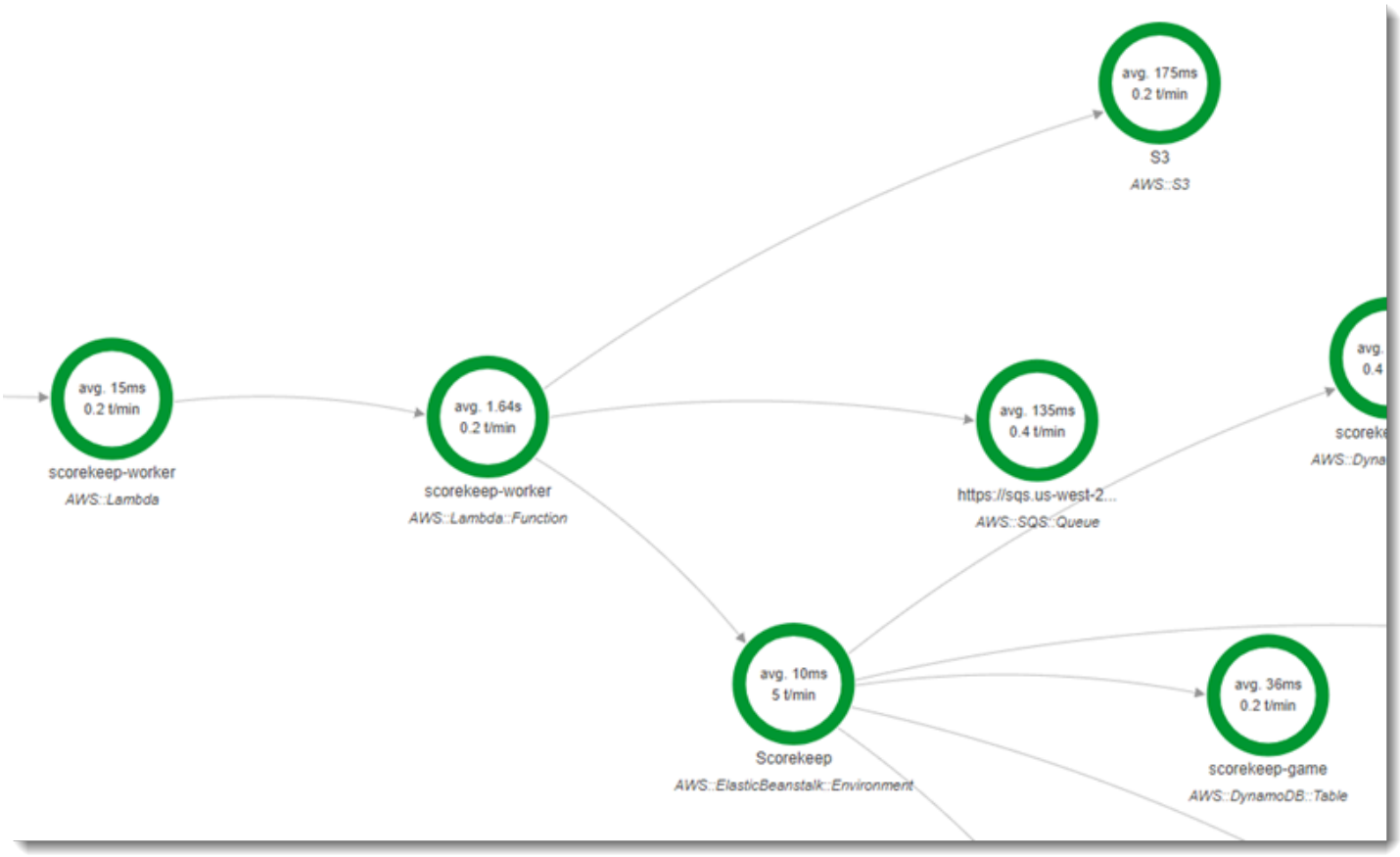
## Worker

Die instrumentierten Worker-Funktion wird in einer eigenen Verzweigung, `xray-worker`, bereitgestellt, da sie nur ausgeführt werden kann, wenn Sie die Worker-Funktion und verwandte Ressourcen zuerst erstellen. Detaillierte Anweisungen finden Sie in der Readme-Datei zur Verzweigung.

Die Funktion wird alle 5 Minuten durch ein gebündeltes Amazon CloudWatch Events-Ereignis ausgelöst. Wenn es ausgeführt wird, ruft die Funktion ein Element aus einer Amazon SQS-Warteschlange ab, die Scorekeep verwaltet. Jede Nachricht enthält Informationen zu einem abgeschlossenen Spiel.

Der Worker ruft den Spieledatensatz und die Spieledokumente von anderen Tabellen ab, auf die der Spieledatensatz verweist. Der Spieledatensatz in DynamoDB enthält beispielsweise eine Liste von Schritten, die während des Spiels ausgeführt wurden. Die Liste enthält nicht die Züge selbst, sondern die IDs der Züge, die in einer separaten Tabelle gespeichert werden.

Sitzungen und Status werden ebenfalls als Referenzen gespeichert. Auf diese Weise wird verhindert, dass die Einträge in der Spieletabelle zu groß werden. Allerdings sind zusätzliche Aufrufe notwendig, um alle Informationen zum Spiel zu erhalten. Der Auftragnehmer dereferenziert all diese Einträge und erstellt eine vollständige Aufzeichnung des Spiels als einzelnes Dokument in Amazon S3. Wenn Sie Analysen der Daten durchführen möchten, können Sie Abfragen direkt in Amazon S3 mit Amazon Athena ausführen, ohne leseintensive Datenmigrationen durchzuführen, um Ihre Daten aus DynamoDB zu entfernen.




Für die Worker-Funktion ist die aktive Nachverfolgung in der Konfiguration in AWS Lambda aktiviert. Im Gegensatz zur Zufallsnamenfunktion erhält der Worker keine Anforderung von einer instrumentierten Anwendung und AWS Lambda daher keinen Ablaufverfolgungs-Header. Bei aktiver Nachverfolgung erstellt Lambda die Nachverfolgungs-ID und trifft Sampling-Entscheidungen.

Das X-Ray SDK für Python befindet sich nur wenige Zeilen oben in der Funktion, die das SDK importiert und seine `patch_all` Funktion ausführt, um die HTTclients AWS SDK for Python (Boto) und zu patchen, die zum Aufrufen von Amazon SQS und Amazon S3 verwendet werden. Wenn der Auftragnehmer die Scorekeep-API aufruft, fügt das SDK den Ablaufverfolgungs-Header zur Anforderung hinzu, um Aufrufe über die API nachzuverfolgen.

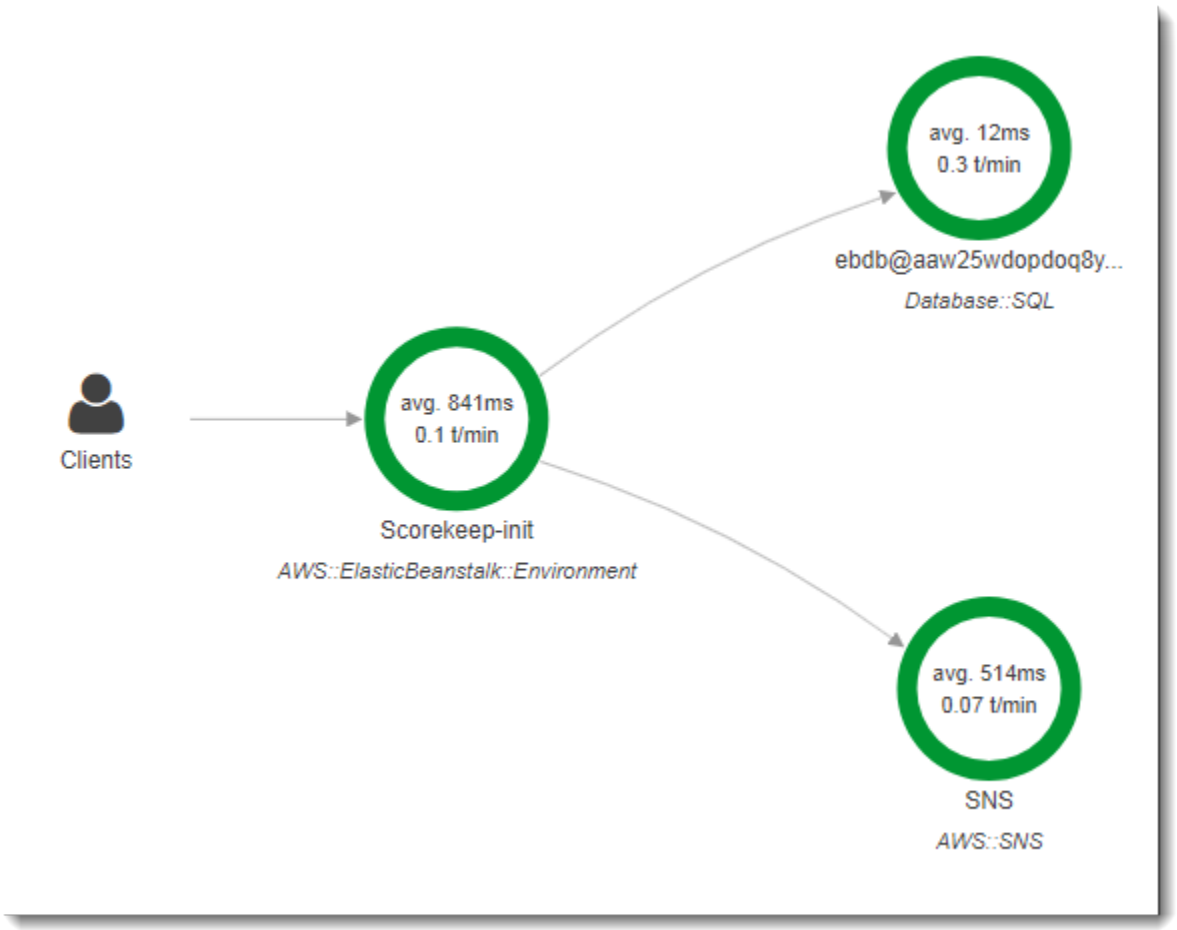
Example **`_lambda/scorekeep-worker/scorekeep-worker.py`** – Worker Lambda-Funktion

```
import os
import boto3
import json
import requests
import time
from aws_xray_sdk.core import xray_recorder
from aws_xray_sdk.core import patch_all

patch_all()
queue_url = os.environ['WORKER_QUEUE']

def lambda_handler(event, context):
    # Create SQS client
    sqs = boto3.client('sqs')
    s3client = boto3.client('s3')

    # Receive message from SQS queue
    response = sqs.receive_message(
        QueueUrl=queue_url,
        AttributeNames=[
            'SentTimestamp'
        ],
        MaxNumberOfMessages=1,
        MessageAttributeNames=[
            'All'
        ],
        VisibilityTimeout=0,
        WaitTimeSeconds=0
    )
   ...
```

# Instrumentieren von Startup-Code

Das X-Ray-SDK SDK for Java erstellt automatisch Segmente für eingehende Anfragen. Wenn eine Anfrage im Leistungsumfang enthalten ist, können Sie instrumentierte Clients verwenden und Untersegmente ohne Probleme aufzeichnen. Wenn Sie jedoch versuchen, einen instrumentierten Client im Startup-Code zu verwenden, erhalten Sie eine SegmentNotFoundException.

Startup-Code wird außerhalb des standardmäßigen Anfrage-/Antwort-Ablaufs einer Webanwendung ausgeführt, sodass Sie für eine Instrumentierung Segmente manuell erstellen müssen. Scorekeep stellt die Instrumentierung von Startup-Code in den `WebConfig`-Dateien dar. Scorekeep ruft während des Starts eine SQL-Datenbank und Amazon SNS auf.




Der`WebConfig`-Klasse erstellt ein Amazon SNS SNS-Abonnement für Benachrichtigungen. Um ein Segment bereitzustellen, auf das das X-Ray-SDK schreiben kann, wenn der Amazon SNS SNS--Client verwendet wird, ruft Scorekeep`beginSegment`und`endSegment`auf dem globalen Rekorder.

Example **src/main/java/scorekeep/WebConfig.java**— InstrumentiertAWSSDK-Client im Startup-Code

```
AWSXRay.beginSegment("Scorekeep-init");
if ( System.getenv("NOTIFICATION_EMAIL") != null ){
  try { Sns.createSubscription(); }
  catch (Exception e ) {
    logger.warn("Failed to create subscription for email "+
 System.getenv("NOTIFICATION_EMAIL"));
  }
}
AWSXRay.endSegment();
```

In :RdsWebConfigIn, das Scorekeep verwendet, wenn eine Amazon RDS-Datenbank verbunden ist, erstellt die Konfiguration außerdem ein Segment für den SQL-Client, das Hibernate verwendet, wenn es das Datenbankschema während des Starts anwendet.

Example **src/main/java/scorekeep/RdsWebConfig.java**— Instrumentierter SQL-Datenbank-Client im Startup-Code

```
@PostConstruct
public void schemaExport() {
  EntityManagerFactoryImpl entityManagerFactoryImpl = (EntityManagerFactoryImpl)
 localContainerEntityManagerFactoryBean.getNativeEntityManagerFactory();
  SessionFactoryImplementor sessionFactoryImplementor =
 entityManagerFactoryImpl.getSessionFactory();
  StandardServiceRegistry standardServiceRegistry =
 sessionFactoryImplementor.getSessionFactoryOptions().getServiceRegistry();
  MetadataSources metadataSources = new MetadataSources(new
 BootstrapServiceRegistryBuilder().build());
  metadataSources.addAnnotatedClass(GameHistory.class);
  MetadataImplementor metadataImplementor = (MetadataImplementor)
 metadataSources.buildMetadata(standardServiceRegistry);
  SchemaExport schemaExport = new SchemaExport(standardServiceRegistry,
 metadataImplementor);

  AWSXRay.beginSegment("Scorekeep-init");
  schemaExport.create(true, true);
  AWSXRay.endSegment();
}
```

`SchemaExport` wird automatisch ausgeführt und verwendet einen SQL-Client. Da der Client instrumentiert ist, muss Scorekeep die Standardimplementierung überschreiben und ein Segment bereitstellen, das das SDK verwenden kann, wenn der Client aufgerufen wird.

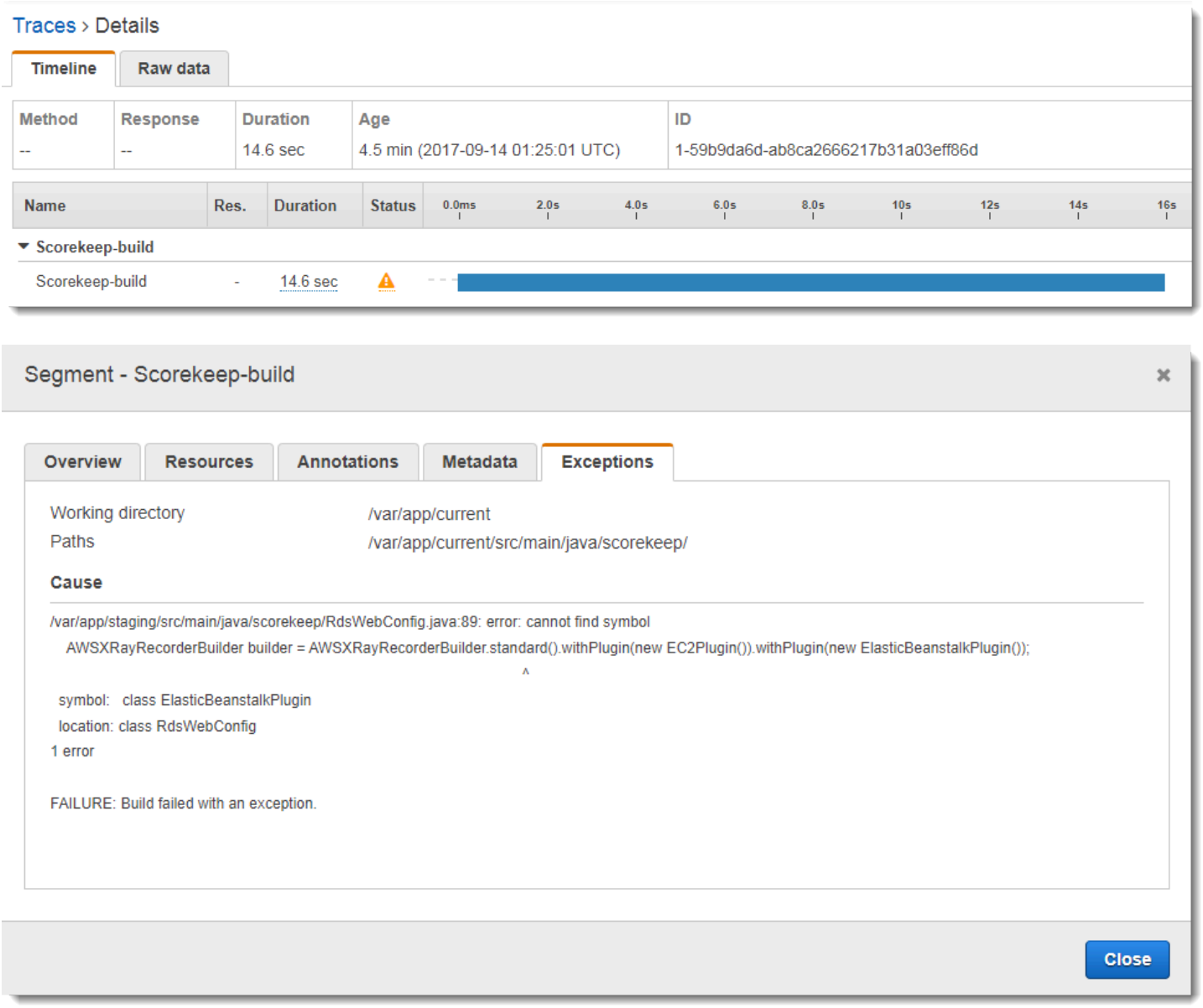
## Instrumentieren von Skripten

Sie können auch Code instrumentieren, der nicht Teil Ihrer Anwendung ist. Wenn der X-Ray-Daemon ausgeführt wird, leitet er alle empfangenen Segmente an X-Ray weiter, auch wenn sie nicht vom X-Ray-SDK generiert werden. Scorekeep nutzt seine eigenen Skripte zur Instrumentierung des Builds, der die Anwendung während der Bereitstellung kompiliert.

Example **`bin/build.sh`** – Instrumentiertes Build-Skript

```
SEGMENT=$(python bin/xray_start.py)
gradle build --quiet --stacktrace &> /var/log/gradle.log; GRADLE_RETURN=$?
if (( GRADLE_RETURN != 0 )); then
  echo "Gradle failed with exit status $GRADLE_RETURN" >&2
  python bin/xray_error.py "$SEGMENT" "$(cat /var/log/gradle.log)"
  exit 1
fi
python bin/xray_success.py "$SEGMENT"
```

`xray_start.py`, `xray_error.py` und `xray_success.py` sind einfache Python-Skripte, die Segmentobjekte erstellen, diese in JSON-Dokumente konvertieren und über UDP an den Daemon senden. Wenn der Gradle-Build fehlschlägt, finden Sie die Fehlermeldung, indem Sie auf den scorekeep-build-Knoten in der Ablaufverfolgungszuordnung der X-Ray-Konsole klicken.




# Instrumentieren eines Web-App-Clients

In der `xray-cognito` Verzweigung verwendet Scorekeep Amazon Cognito, um Benutzern zu ermöglichen, ein Konto zu erstellen und sich damit anzumelden, um ihre Benutzerinformationen aus einem Amazon Cognito-Benutzerpool abzurufen. Wenn sich ein Benutzer anmeldet, verwendet Scorekeep einen Amazon Cognito-Identitätspool, um temporäre AWS Anmeldeinformationen für die Verwendung mit der abzurufen AWS SDK for JavaScript.

Der Identitätenpool ist so konfiguriert, dass angemeldete Benutzer Ablaufverfolgungsdaten in AWS X-Ray schreiben können. Die Web-App nutzt diese Anmeldeinformationen, um die Benutzer-ID des

angemeldeten Benutzers, den Browserpfad und die Client-Ansicht von Aufrufen der Scorekeep-API aufzuzeichnen.

Der Großteil der Vorgänge wird in einer Service-Klasse mit dem Namen `xray` ausgeführt. Diese Serviceklasse bietet Methoden zum Generieren der erforderlichen Kennungen, zum Erstellen von Segmenten in Bearbeitung, zum Abschließen von Segmenten und zum Senden von Segmentdokumenten an die X-Ray-API.

Example **public/xray.js** – Aufzeichnen und Hochladen von Segmenten

```
...
  service.beginSegment = function() {
    var segment = {};
    var traceId = '1-' + service.getHexTime() + '-' + service.getHexId(24);

    var id = service.getHexId(16);
    var startTime = service.getEpochTime();

    segment.trace_id = traceId;
    segment.id = id;
    segment.start_time = startTime;
    segment.name = 'Scorekeep-client';
    segment.in_progress = true;
    segment.user =  sessionStorage['userid'];
    segment.http = {
      request: {
        url: window.location.href
      }
    };

    var documents = [];
    documents[0] = JSON.stringify(segment);
    service.putDocuments(documents);
    return segment;
  }

  service.endSegment = function(segment) {
    var endTime = service.getEpochTime();
    segment.end_time = endTime;
    segment.in_progress = false;
    var documents = [];
    documents[0] = JSON.stringify(segment);
    service.putDocuments(documents);
```

```
  }

  service.putDocuments = function(documents) {
    var xray = new AWS.XRay();
    var params = {
      TraceSegmentDocuments: documents
    };
    xray.putTraceSegments(params, function(err, data) {
      if (err) {
        console.log(err, err.stack);
      } else {
        console.log(data);
      }
    })
  }
```

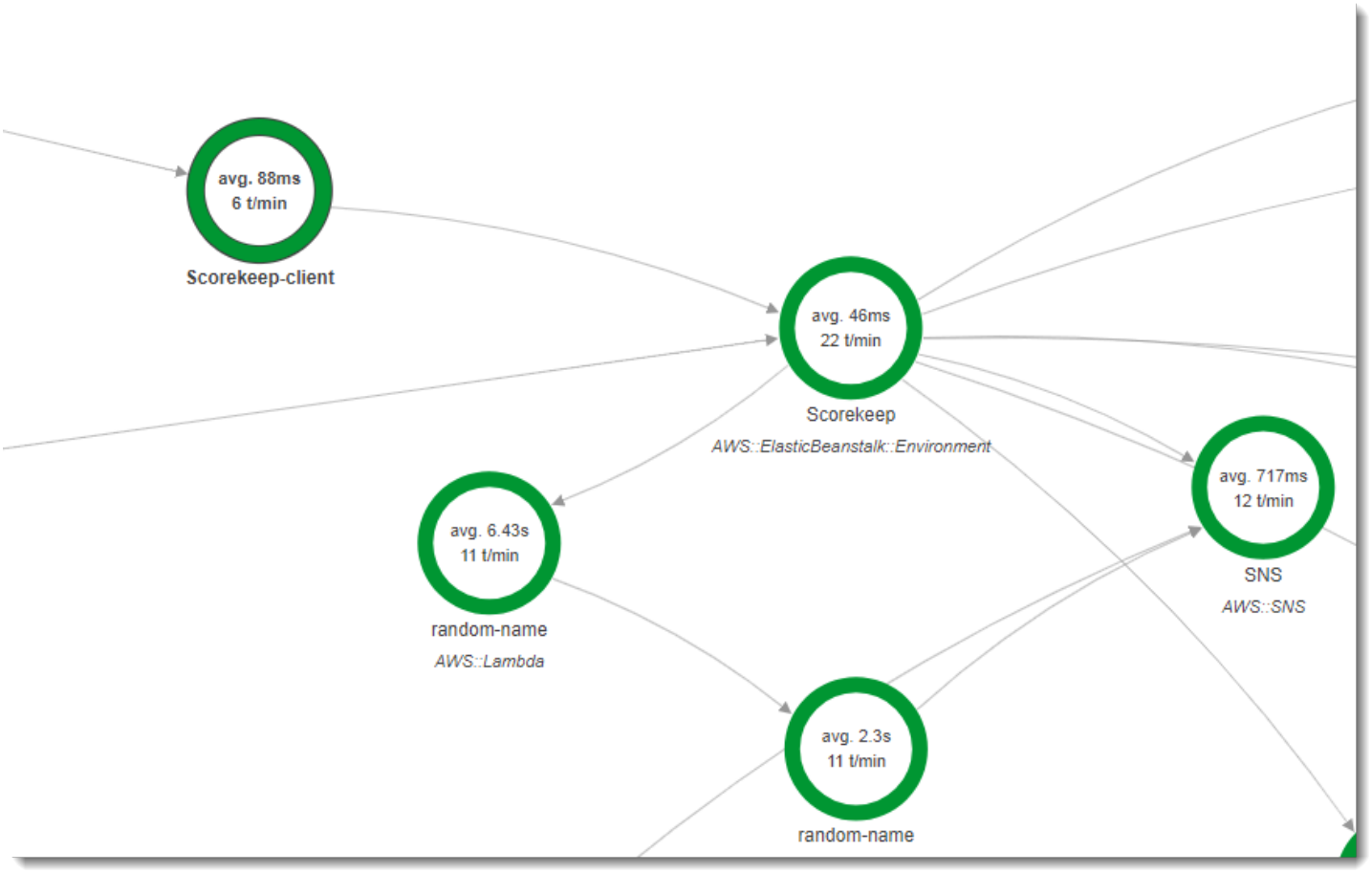
Diese Methoden werden im Header und in `transformResponse`-Funktionen in den Ressourcen-Services aufgerufen, die die Web-App zum Aufrufen der Scorekeep-API verwendet. Um das Client-Segment in dieselbe Ablaufverfolgung wie das Segment aufzunehmen, das die API generiert, muss die Webanwendung die Ablaufverfolgungs-ID und die Segment-ID in einen Ablaufverfolgungs-Header (`X-Amzn-Trace-Id`) aufnehmen, den das X-Ray-SDK lesen kann. Wenn die instrumentierte Java-Anwendung eine Anfrage mit diesem Header erhält, verwendet das X-Ray SDK for Java dieselbe Ablaufverfolgungs-ID und macht das Segment vom Web-App-Client zum übergeordneten Element seines Segments.

Example **`public/app/services.js`** – Aufzeichnen von Segmenten für Winkel-Ressourcenaufrufe und Schreiben von Ablaufverfolgungs-Headern

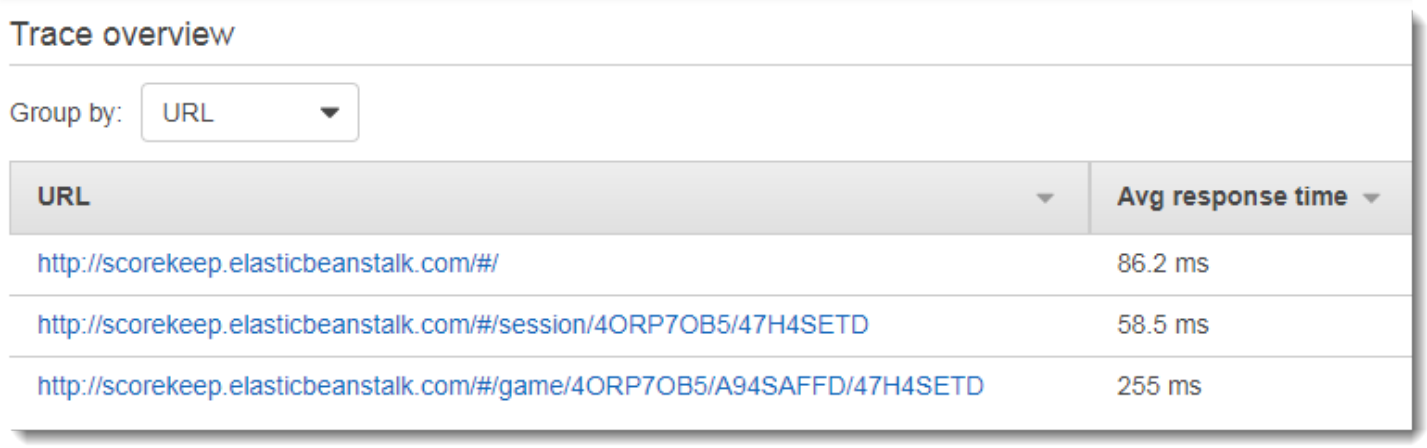
```
var module = angular.module('scorekeep');
module.factory('SessionService', function($resource, api, XRay) {
  return $resource(api + 'session/:id', { id: '@_id' }, {
    segment: {},
    get: {
      method: 'GET',
      headers: {
        'X-Amzn-Trace-Id': function(config) {
          segment = XRay.beginSegment();
          return XRay.getTraceHeader(segment);
        }
      },
      transformResponse: function(data) {
        XRay.endSegment(segment);
```

```
        return angular.fromJson(data);
      },
    },
...
```

Die resultierende Ablaufverfolgungszuordnung enthält einen Knoten für den Web-App-Client.




Ablaufverfolgungen, die Segmente aus der Web-App einschließen, zeigen die URL an, die der Benutzer im Browser sieht (Pfad beginnend mit `/#/`). Ohne Client-Instrumentierung erhalten Sie nur die URL der API-Ressource, die die Web-App aufruft (Pfade beginnend mit `/api/`).




## Verwenden instrumentierter Clients in Auftragnehmer-Threads

Scorekeep verwendet einen Auftragnehmer-Thread, um eine Benachrichtigung in Amazon SNS zu veröffentlichen, wenn ein Benutzer ein Spiel gewinnt. Die Veröffentlichung der Benachrichtigung dauert länger als alle übrigen Anfragevorgänge zusammen und wirkt sich nicht auf den Client oder Benutzer aus. Die Aufgabe asynchron auszuführen ist daher eine gute Möglichkeit, die Reaktionszeit zu verbessern.

Das X-Ray SDK for Java erkennt jedoch nicht, welches Segment aktiv war, wenn der Thread erstellt wird. Wenn Sie also versuchen, den instrumentierten AWS SDK for Java-Client innerhalb des Threads zu verwenden, wird eine `SegmentNotFoundException` ausgegeben und der Thread stürzt ab.

Example Web-1.error.log

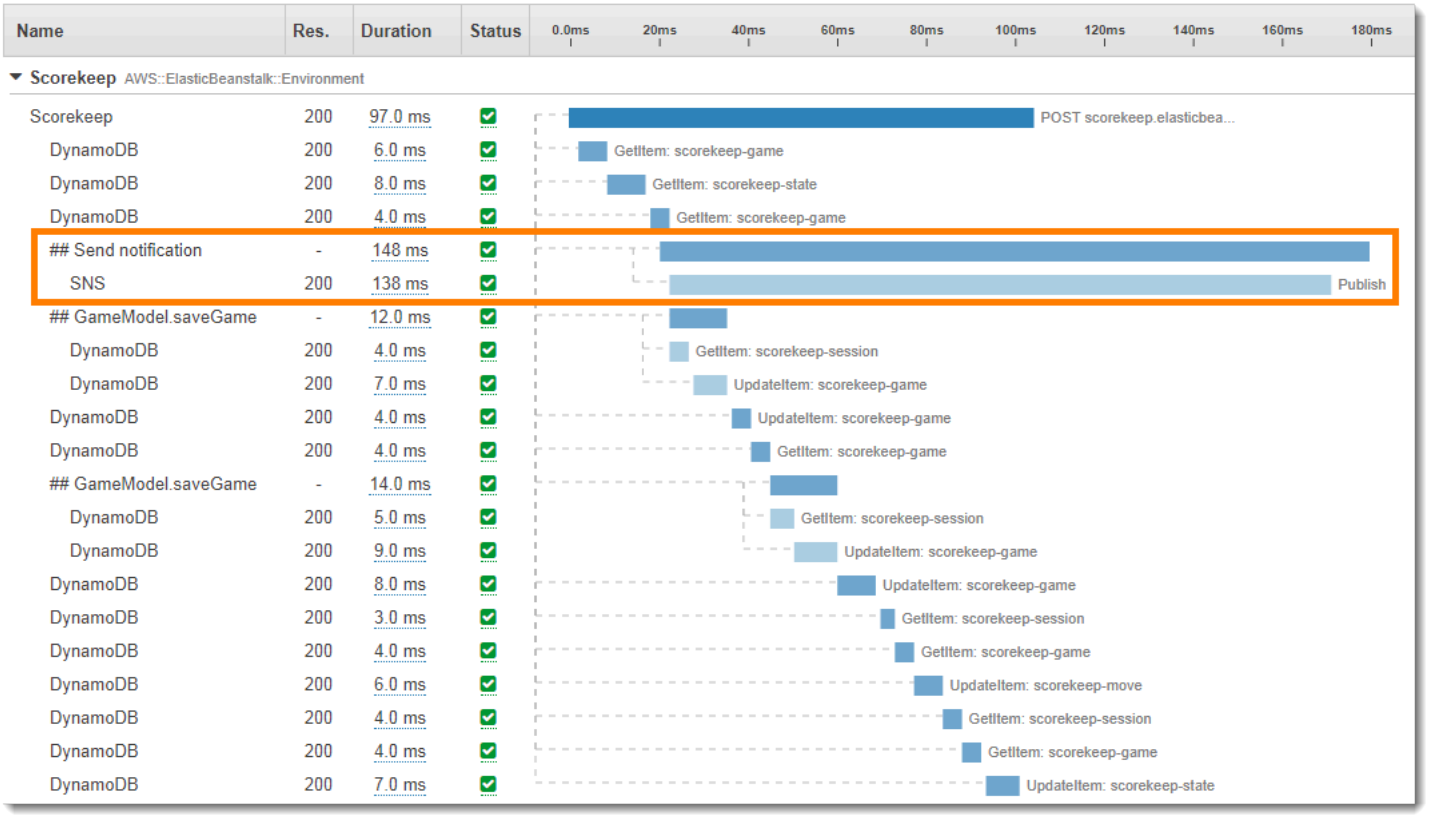
```
Exception in thread "Thread-2" com.amazonaws.xray.exceptions.SegmentNotFoundException:
 Failed to begin subsegment named 'AmazonSNS': segment cannot be found.
        at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
        at
 sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl.java:62)
        at
 sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.ja
...
```

Um dies zu beheben, verwendet die Anwendung`GetTraceEntity`um einen Verweis auf das Segment im Hauptthread zu erhalten, und`Entity.run()`um den Worker-Thread-Code mit Zugriff auf den Kontext des Segments sicher auszuführen.

Example **src/main/java/scorekeep/MoveFactory.java**- Übergeben von Übergeben von Ablaufverfolgungskontext an

```
import com.amazonaws.xray.AWSXRay;
import com.amazonaws.xray.AWSXRayRecorder;
import com.amazonaws.xray.entities.Entity;
import com.amazonaws.xray.entities.Segment;
import com.amazonaws.xray.entities.Subsegment;
...
      Entity segment = recorder.getTraceEntity();
      Thread comm = new Thread() {
        public void run() {
         segment.run(() -> {
            Subsegment subsegment = AWSXRay.beginSubsegment("## Send notification");
            Sns.sendNotification("Scorekeep game completed", "Winner: " + userId);
            AWSXRay.endSubsegment();
          }
        }
```

Da die Anfrage jetzt vor dem Aufruf von Amazon SNS behoben wird, erstellt die Anwendung ein separates Untersegment für den Thread. Dadurch wird verhindert, dass das X-Ray -SDK das Segment schließt, bevor es die Antwort von Amazon SNS aufzeichnet. Wenn Scorekeep die Anfrage gelöst hat und kein Untersegment offen war, kann die Antwort von Amazon SNS verloren gehen.




Weitere Informationen zu Multithreading finden Sie unter Übermitteln von Segmentkontext zwischen Threads in einer Multithread-Anwendung.

# Fehlerbehebung AWS X-Ray

In diesem Thema werden häufige Fehler und Probleme aufgeführt, die bei der Verwendung der X-Ray-API, -Konsole oder -SDKs auftreten können. Wenn Sie auf ein Problem stoßen, das hier nicht aufgeführt ist, können Sie die Schaltfläche Feedback auf dieser Seite verwenden, um es zu melden.

Sections



## Seiten für X-Ray-Ablaufverfolgungszuordnungen und Ablaufverfolgungsdetails

Die folgenden Abschnitte können hilfreich sein, wenn Sie Probleme mit der X-Ray-Trace-Übersicht und der Seite mit den Trace-Details haben:

### Ich sehe nicht alle meine CloudWatch Protokolle

Wie Protokolle so konfiguriert werden, dass sie in der X-Ray-Ablaufverfolgungsübersicht und auf den Seiten mit den Ablaufverfolgungsdetails angezeigt werden, hängt vom Service ab.

- API-Gateway-Protokolle werden angezeigt, wenn die Protokollierung in API Gateway aktiviert ist.

Nicht alle Service-Übersichtsknoten unterstützen die Anzeige der zugehörigen Protokolle. Anzeigen von Protokollen für die folgenden Knotentypen:

- Lambda-Kontext
- Lambda-Funktion
- API Gateway-Stufe
- Amazon-ECS-Cluster
- Amazon-ECS-Instance

- Amazon-ECS-Service
- Amazon-ECS-Aufgaben
- Amazon-EKS-Cluster
- Amazon-EKS-Namespace
- Amazon-EKS-Knoten
- Amazon-EKS-Pod
- Amazon-EKS-Service

## Ich sehe nicht alle meine Alarme auf der X-Ray-Ablaufverfolgungskarte

Die X-Ray-Ablaufverfolgungszuordnung zeigt nur das Warnsymbol für einen Knoten an, wenn sich Alarme, die diesem Knoten zugeordnet sind, im ALARM-Status befinden.

Die Ablaufverfolgungszuordnung ordnet Alarme Knoten zu, die die folgende Logik verwenden:

- Wenn der Knoten einen - AWS Service darstellt, werden alle Alarme mit dem diesem Service zugeordneten Namespace dem Knoten zugeordnet. Beispielsweise `AWS::Kinesis` ist ein Knoten vom Typ mit allen Alarmen verknüpft, die auf Metriken im CloudWatch Namespace basieren`AWS/Kinesis`.
- Wenn der Knoten eine - AWS Ressource darstellt, werden die Alarme für diese spezifische Ressource verknüpft. Beispielsweise ist ein Knoten vom Typ `AWS::DynamoDB::Table` mit dem Namen „MyTable“ mit allen Alarmen verknüpft, die auf einer Metrik mit dem Namespace basieren `AWS/DynamoDB` und deren `TableName` Dimension auf festgelegt ist`MyTable`.
- Wenn der Knoten vom unbekannten Typ ist, der durch einen gestrichelten Rahmen um den Namen identifiziert wird, sind diesem Knoten keine Alarme zugeordnet.

## Ich sehe einige AWS Ressourcen nicht auf der Trace-Map

Nicht jede AWS Ressource wird durch einen dedizierten Knoten dargestellt. Einige AWS Services werden durch einen einzelnen Knoten für alle Anfragen an den Service dargestellt. Die folgenden Ressourcentypen werden mit einem Knoten pro Ressource angezeigt:

- `AWS::DynamoDB::Table`
- `AWS::Lambda::Function`

Lambda-Funktionen werden durch zwei Knoten dargestellt – einen für den Lambda-Container und einen für die Funktion. Dies hilft, Kaltstartprobleme mit Lambda-Funktionen zu identifizieren. Lambda-Container-Knoten werden auf dieselbe Weise mit Alarmen und Dashboards verknüpft wie Lambda-Funktionsknoten.

- `AWS::ApiGateway::Stage`
- `AWS::SQS::Queue`
- `AWS::SNS::Topic`

## Es gibt zu viele Knoten auf der Trace-Map

Verwenden Sie X-Ray-Gruppen, um Ihre Übersicht in mehrere Übersichten aufzuteilen. Weitere Informationen finden Sie unter Verwenden von Filterausdrücken mit Gruppen.

# X-Ray SDK für Java

Fehler: Ausnahme im Thread „Thread-1“ com.amazonaws.xray.exceptions.SegmentNotFoundException: Untersegment mit dem Namen „AmazonSNS ': segment kann nicht gefunden werden.

Dieser Fehler weist darauf hin, dass das X-Ray-SDK versucht hat, einen ausgehenden Aufruf an aufzuzeichnen AWS, aber kein offenes Segment finden konnte. Dies kann in folgenden Situationen auftreten:

- Ein Servlet-Filter ist nicht konfiguriert – Das X-Ray-SDK erstellt Segmente für eingehende Anforderungen mit einem Filter namens `AWSXRayServletFilter`. Konfigurieren Sie einen Servlet-Filter, um eingehende Anfragen zu instrumentieren.
- Sie verwenden instrumentierte Clients außerhalb des Servlet-Codes – Wenn Sie einen instrumentierten Client verwenden, um Aufrufe im Startcode oder in einem anderen Code durchzuführen, der nicht als Antwort auf eine eingehende Anfrage ausgeführt wird, müssen Sie ein Segment manuell erstellen. Beispiele finden Sie unter Instrumentieren von Startup-Code.
- Sie verwenden instrumentierte Clients in Worker-Threads – Wenn Sie einen neuen Thread erstellen, verliert der X-Ray-Recorder seinen Verweis auf das offene Segment. Sie können die Methoden `getTraceEntity` und `setTraceEntity` verwenden, um eine Referenz zum aktuellen Segment oder Untersegment zu erhalten (`Entity`) und sie innerhalb des Threads wieder an den Recorder zu übergeben. Ein Beispiel finden Sie unter Verwenden instrumentierter Clients in Auftragnehmer-Threads.

## X-Ray SDK für Node.js

Problem: CLS funktioniert nicht mit Sequelize

Übergeben Sie den Namespace des X-Ray-SDK für Node.js mit der `-cls`Methode an Sequenzierung.

```
var AWSXRay = require('aws-xray-sdk');
const Sequelize = require('sequelize');
Sequelize.cls = AWSXRay.getNamespace();
const sequelize = new Sequelize(...);
```

Problem: CLS funktioniert nicht mit Bluebird

Verwenden Sie `cls-bluebird`, damit Bluebird mit CLS funktioniert.

```
var AWSXRay = require('aws-xray-sdk');
var Promise = require('bluebird');
var clsBluebird = require('cls-bluebird');
clsBluebird(AWSXRay.getNamespace());
```

## Der X-Ray-Daemon

Problem: Der Daemon verwendet die falschen Anmeldeinformationen

Der Daemon verwendet das AWS SDK, um Anmeldeinformationen zu laden. Wenn Sie Anmeldeinformationen mit mehreren Methoden bereitstellen, wird die Methode mit der höchsten Priorität verwendet. Weitere Informationen finden Sie unter Ausführen des Daemons.

# Sicherheit in AWS X-Ray

Cloud-Sicherheit AWS hat höchste Priorität. Als AWS Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame Verantwortung von Ihnen AWS und Ihnen. Das Modell der geteilten Verantwortung beschreibt dies als Sicherheit der Cloud und Sicherheit in der Cloud:

- Sicherheit der Cloud — AWS ist verantwortlich für den Schutz der Infrastruktur, die AWS-Services in der läuft AWS Cloud. AWS bietet Ihnen auch Dienste, die Sie sicher nutzen können. Die Wirksamkeit unserer Sicherheitsfunktionen wird regelmäßig von externen Prüfern im Rahmen des AWS -Compliance-Programms getestet und überprüft. Weitere Informationen zu den Compliance-Programmen, die für X-Ray gelten, finden Sie unter AWS-Services Umfang nach Compliance-Programmen.
- Sicherheit in der Cloud — Ihre Verantwortung richtet sich nach dem AWS-Service , was Sie verwenden. In Ihre Verantwortung fallen außerdem weitere Faktoren, wie z. B. die Vertraulichkeit der Daten, die Anforderungen Ihrer Organisation sowie geltende Gesetze und Vorschriften.

Diese Dokumentation hilft Ihnen zu verstehen, wie Sie das Modell der gemeinsamen Verantwortung bei der Verwendung von X-Ray anwenden können. In den folgenden Themen erfahren Sie, wie Sie X-Ray konfigurieren, um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Sie erfahren auch, wie Sie andere verwenden können AWS-Services , die Ihnen helfen können, Ihre X-Ray-Ressourcen zu überwachen und zu sichern.

Themen

# Datenschutz in AWS X-Ray

AWS X-Ray verschlüsselt immer Spuren und zugehörige ruhende Daten. Wenn Sie Verschlüsselungsschlüssel aus Gründen der Einhaltung von Vorschriften oder internen Anforderungen überprüfen und deaktivieren müssen, können Sie X-Ray so konfigurieren, dass ein AWS Key Management Service (AWS KMS) -Schlüssel zum Verschlüsseln von Daten verwendet wird.

X-Ray bietet einen Von AWS verwalteter Schlüssel Namen`aws/xray`. Verwenden Sie diesen Schlüssel, wenn Sie die Schlüsselverwendung in AWS CloudTrail prüfen möchten, den eigentlichen Schlüssel aber nicht verwalten müssen. Wenn Sie den Zugriff auf den Schlüssel verwalten oder die Schlüsselrotation konfigurieren müssen, können Sie einen vom Kunden verwalteten Schlüssel erstellen.

Wenn Sie die Verschlüsselungseinstellungen ändern, verbringt X-Ray einige Zeit damit, Datenschlüssel zu generieren und zu übertragen. Während der neue Schlüssel verarbeitet wird, kann X-Ray Daten mit einer Kombination aus den neuen und alten Einstellungen verschlüsseln. Vorhandene Daten werden nicht erneut verschlüsselt, wenn Sie Verschlüsselungseinstellungen ändern.

> **Note**
>
> AWS KMSwird berechnet, wenn X-Ray einen KMS-Schlüssel zum Verschlüsseln oder Entschlüsseln von Trace-Daten verwendet.
>
> - Standardverschlüsselung — Kostenlos.
> - Von AWS verwalteter Schlüssel— Zahlen Sie für die Nutzung des Schlüssels.
> - vom Kunden verwalteter Schlüssel — Bezahlen Sie für die Speicherung und Nutzung des Schlüssels.
>
> Einzelheiten finden Sie unter AWS Key Management ServicePreise.

> **Note**
>
> X-Ray Insights-Benachrichtigungen senden Ereignisse an AmazonEventBridge, das derzeit keine von Kunden verwalteten Schlüssel unterstützt. Weitere Informationen finden Sie unter Datenschutz bei Amazon EventBridge.

Sie benötigen Zugriff auf Benutzerebene auf einen vom Kunden verwalteten Schlüssel, um X-Ray für seine Verwendung zu konfigurieren und anschließend verschlüsselte Spuren anzeigen zu können. Weitere Informationen finden Sie unter Benutzerberechtigungen für die Verschlüsselung.

CloudWatch console

So konfigurieren Sie X-Ray für die Verwendung eines KMS-Schlüssels für die Verschlüsselung mithilfe der Konsole CloudWatch

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im linken Navigationsbereich die Option Einstellungen aus.
3. Wählen Sie im Bereich X-Ray Traces unter Verschlüsselung die Option Einstellungen anzeigen.
4. Wählen Sie im Abschnitt Verschlüsselungskonfiguration die Option Bearbeiten.
5. Wählen Sie Einen KMS-Schlüssel verwenden aus.
6. Wählen Sie einen Schlüssel aus dem Dropdown-Menü aus:
   - aws/xray — Benutze die. Von AWS verwalteter Schlüssel
   - Schlüsselalias — Verwenden Sie einen vom Kunden verwalteten Schlüssel in Ihrem Konto.
   - Geben Sie die Schlüssel-ARN manuell ein — Verwenden Sie einen vom Kunden verwalteten Schlüssel in einem anderen -Konto. Geben Sie den vollständigen Amazon-Ressourcennamen (ARN) des Schlüssels in das entsprechende Feld ein.
7. Wählen Sie Verschlüsselung aktualisieren.

X-Ray console

So konfigurieren Sie X-Ray für die Verwendung eines KMS-Schlüssels für die Verschlüsselung mithilfe der X-Ray-Konsole

1. Öffnen Sie die X-Ray-Konsole.
2. Wählen Sie Encryption (Verschlüsselung) aus.
3. Wählen Sie Einen KMS-Schlüssel verwenden aus.
4. Wählen Sie einen Schlüssel aus dem Dropdown-Menü aus:
   - aws/xray — Benutze die. Von AWS verwalteter Schlüssel
   - Schlüsselalias — Verwenden Sie einen vom Kunden verwalteten Schlüssel in Ihrem Konto.
   - Geben Sie die Schlüssel-ARN manuell ein — Verwenden Sie einen vom Kunden verwalteten Schlüssel in einem anderen -Konto. Geben Sie den vollständigen Amazon-Ressourcennamen (ARN) des Schlüssels in das entsprechende Feld ein.
5. Wählen Sie Apply (Anwenden) aus.

> **Note**
>
> X-Ray-Schlüssel werden von asymmetrischen KMS-Schlüsseln nicht unterstützt.

Wenn X-Ray nicht auf Ihren Verschlüsselungscode zugreifen kann, werden keine Daten mehr gespeichert. Dies kann passieren, wenn Ihr Benutzer den Zugriff auf den KMS-Schlüssel verliert oder wenn Sie einen Schlüssel deaktivieren, der gerade verwendet wird. In diesem Fall zeigt X-Ray eine Benachrichtigung in der Navigationsleiste an.

Informationen zum Konfigurieren von Verschlüsselungseinstellungen mit der X-Ray-API finden Sie unterKonfiguration von Sampling-, Gruppen- und Verschlüsselungseinstellungen mit der X-Ray-API.

# Identity and Access Management für AWS X-Ray

AWS Identity and Access Management (IAM) ist ein AWS-Service , mit dem ein Administrator den Zugriff auf - AWS Ressourcen sicher steuern kann. IAM-Administratoren steuern, wer für die Nutzung von X-Ray-Ressourcen authentifiziert (angemeldet) und autorisiert (im Besitz von Berechtigungen) werden kann. IAM ist ein AWS-Service , den Sie ohne zusätzliche Kosten nutzen können.

Themen



# Zielgruppe

Wie Sie AWS Identity and Access Management (IAM) verwenden, unterscheidet sich je nach Ihrer Arbeit in X-Ray.

Service-Benutzer – Wenn Sie den X-Ray-Service zur Ausführung von Aufgaben verwenden, stellt Ihnen Ihr Administrator die Anmeldeinformationen und Berechtigungen bereit, die Sie benötigen. Wenn Sie für Ihre Arbeit weitere X-Ray-Funktionen ausführen, benötigen Sie möglicherweise zusätzliche Berechtigungen. Wenn Sie die Fuktionsweise der Zugriffskontrolle nachvollziehen, wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anzufordern müssen. Wenn Sie nicht auf ein Feature in X-Ray zugreifen können, finden Sie weitere Informationen unter Fehlerbehebung für AWS X-Ray-Identität und -Zugriff.

Service-Administrator – Wenn Sie in Ihrem Unternehmen für X-Ray-Ressourcen verantwortlich sind, haben Sie wahrscheinlich vollständigen Zugriff auf X-Ray. Ihre Aufgabe besteht darin, zu bestimmen, auf welche X-Ray-Funktionen und -Ressourcen Ihre Service-Benutzer zugreifen sollen. Sie müssen dann Anträge an Ihren IAM-Administrator stellen, um die Berechtigungen Ihrer Servicenutzer zu ändern. Lesen Sie die Informationen auf dieser Seite, um die Grundkonzepte von IAM nachzuvollziehen. Weitere Informationen dazu, wie Ihr Unternehmen IAM mit X-Ray verwenden kann, finden Sie unter Wie AWS X-Ray funktioniert mit IAM.

IAM-Administrator – Wenn Sie als IAM-Administrator fungieren, sollten Sie Einzelheiten dazu kennen, wie Sie Richtlinien zur Verwaltung des Zugriffs auf X-Ray verfassen können. Beispiele für identitätsbasierte X-Ray-Richtlinien, die Sie in IAM verwenden können, finden Sie unter AWS X-Ray Beispiele für identitätsbasierte Politik.

# Authentifizierung mit Identitäten

Die Authentifizierung ist die Art und Weise, wie Sie sich AWS mit Ihren Identitätsdaten bei anmelden. Sie müssen als Root-Benutzer des AWS-Kontos, als IAM-Benutzer oder durch Übernahme einer IAM-Rolle authentifiziert (bei angemeldet AWS) sein.

Sie können sich bei AWS als Verbundidentität anmelden, indem Sie Anmeldeinformationen verwenden, die über eine Identitätsquelle bereitgestellt werden. AWS IAM Identity Center (IAM Identity Center)-Benutzer, die Single-Sign-On-Authentifizierung Ihres Unternehmens und Ihre Google- oder Facebook-Anmeldeinformationen sind Beispiele für Verbundidentitäten. Wenn Sie sich als Verbundidentität anmelden, hat der Administrator vorher mithilfe von IAM-Rollen einen Identitätsverbund eingerichtet. Wenn Sie AWS über einen Verbund auf zugreifen, übernehmen Sie indirekt eine Rolle.

Je nachdem, um welchen Benutzertyp es sich handelt, können Sie sich bei der AWS Management Console oder im - AWS Zugriffsportal anmelden. Weitere Informationen zur Anmeldung bei AWS finden Sie unter So melden Sie sich bei Ihrem an AWS-Konto im AWS-Anmeldung - Benutzerhandbuch.

Wenn Sie AWS programmgesteuert auf zugreifen, AWS stellt ein Software Development Kit (SDK) und eine Befehlszeilenschnittstelle (Command Line Interface, CLI) bereit, um Ihre Anforderungen mithilfe Ihrer Anmeldeinformationen kryptografisch zu signieren. Wenn Sie keine AWS Tools verwenden, müssen Sie Anforderungen selbst signieren. Weitere Informationen zur Verwendung der empfohlenen Methode zum eigenständigen Signieren von Anforderungen finden Sie unter Signieren von AWS API-Anforderungen im IAM-Benutzerhandbuch.

Unabhängig von der verwendeten Authentifizierungsmethode müssen Sie möglicherweise zusätzliche Sicherheitsinformationen angeben. empfiehlt beispielsweise, AWS die Multi-Faktor-Authentifizierung (MFA) zu verwenden, um die Sicherheit Ihres Kontos zu erhöhen. Weitere Informationen finden Sie unter Multi-Faktor-Authentifizierung im AWS IAM Identity Center - Benutzerhandbuch und Verwenden der Multi-Faktor-Authentifizierung (MFA) in AWS im IAM-Benutzerhandbuch.

## AWS-Konto Root-Benutzer

Wenn Sie ein erstellen AWS-Konto, beginnen Sie mit einer Anmeldeidentität, die vollständigen Zugriff auf alle AWS-Services und Ressourcen im Konto hat. Diese Identität wird als AWS-Konto Root-Benutzer bezeichnet und Sie melden sich mit der E-Mail-Adresse und dem Passwort an, mit denen Sie das Konto erstellt haben. Wir raten ausdrücklich davon ab, den Root-Benutzer für

Alltagsaufgaben zu verwenden. Schützen Sie Ihre Root-Benutzer-Anmeldeinformationen und verwenden Sie diese, um die Aufgaben auszuführen, die nur der Root-Benutzer ausführen kann. Eine vollständige Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie unter Aufgaben, die Root-Benutzer-Anmeldeinformationen erfordern im IAM-Benutzerhandbuch.

### IAM-Benutzer und -Gruppen

Ein IAM-Benutzer ist eine Identität in Ihrem AWS-Konto mit bestimmten Berechtigungen für eine einzelne Person oder Anwendung. Wenn möglich, empfehlen wir, temporäre Anmeldeinformationen zu verwenden, anstatt IAM-Benutzer zu erstellen, die langfristige Anmeldeinformationen wie Passwörter und Zugriffsschlüssel haben. Bei speziellen Anwendungsfällen, die langfristige Anmeldeinformationen mit IAM-Benutzern erfordern, empfehlen wir jedoch, die Zugriffsschlüssel zu rotieren. Weitere Informationen finden Sie unter Regelmäßiges Rotieren von Zugriffsschlüsseln für Anwendungsfälle, die langfristige Anmeldeinformationen erfordern im IAM-Benutzerhandbuch.

Eine IAM-Gruppe ist eine Identität, die eine Sammlung von IAM-Benutzern angibt. Sie können sich nicht als Gruppe anmelden. Mithilfe von Gruppen können Sie Berechtigungen für mehrere Benutzer gleichzeitig angeben. Gruppen vereinfachen die Verwaltung von Berechtigungen, wenn es zahlreiche Benutzer gibt. Sie könnten beispielsweise einer Gruppe mit dem Namen IAMAdmins Berechtigungen zum Verwalten von IAM-Ressourcen erteilen.

Benutzer unterscheiden sich von Rollen. Ein Benutzer ist einer einzigen Person oder Anwendung eindeutig zugeordnet. Eine Rolle kann von allen Personen angenommen werden, die sie benötigen. Benutzer besitzen dauerhafte Anmeldeinformationen. Rollen stellen temporäre Anmeldeinformationen bereit. Weitere Informationen finden Sie unter Erstellen eines IAM-Benutzers (anstatt einer Rolle) im IAM-Benutzerhandbuch.

### IAM-Rollen

Eine IAM-Rolle ist eine Identität in Ihrem AWS-Konto mit bestimmten Berechtigungen. Sie ist einem IAM-Benutzer vergleichbar, ist aber nicht mit einer bestimmten Person verknüpft. Sie können vorübergehend eine IAM-Rolle in der übernehmen, AWS Management Console indem Sie die Rollen wechseln. Sie können eine Rolle übernehmen, indem Sie eine AWS CLI - oder AWS -API-Operation aufrufen oder eine benutzerdefinierte URL verwenden. Weitere Informationen zu Methoden für die Verwendung von Rollen finden Sie unter Verwenden von IAM-Rollen im IAM-Benutzerhandbuch.

IAM-Rollen mit temporären Anmeldeinformationen sind in folgenden Situationen hilfreich:

- Verbundbenutzerzugriff – Um einer Verbundidentität Berechtigungen zuzuweisen, erstellen Sie eine Rolle und definieren Berechtigungen für die Rolle. Wird eine Verbundidentität authentifiziert,

so wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter Erstellen von Rollen für externe Identitätsanbieter im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM. Informationen zu Berechtigungssätzen finden Sie unter Berechtigungssätze im AWS IAM Identity Center -Benutzerhandbuch.

- Temporäre IAM-Benutzerberechtigungen – Ein IAM-Benutzer oder eine -Rolle kann eine IAM-Rolle übernehmen, um vorübergehend andere Berechtigungen für eine bestimmte Aufgabe zu erhalten.
- Kontoübergreifender Zugriff – Sie können eine IAM-Rolle verwenden, um einem vertrauenswürdigen Prinzipal in einem anderen Konto den Zugriff auf Ressourcen in Ihrem Konto zu ermöglichen. Rollen stellen die primäre Möglichkeit dar, um kontoübergreifendem Zugriff zu gewähren. Bei einigen können AWS-Services Sie jedoch eine Richtlinie direkt an eine Ressource anfügen (anstatt eine Rolle als Proxy zu verwenden). Informationen zu den Unterschieden zwischen Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter So unterscheiden sich IAM-Rollen von ressourcenbasierten Richtlinien im IAM-Benutzerhandbuch.
- Serviceübergreifender Zugriff – Einige AWS-Services verwenden Funktionen in anderen AWS-Services. Wenn Sie beispielsweise einen Aufruf in einem Service tätigen, führt dieser Service häufig Anwendungen in Amazon EC2 aus oder speichert Objekte in Amazon S3. Ein Dienst kann dies mit den Berechtigungen des aufrufenden Prinzipals mit einer Servicerolle oder mit einer serviceverknüpften Rolle tun.
  - Forward Access Sessions (FAS) – Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen in auszuführen AWS, gelten Sie als Prinzipal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service auslösen. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, in Kombination mit der Anforderung AWS-Service , Anfragen an nachgelagerte Services zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Service eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder -Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter Zugriffssitzungen weiterleiten.
  - Servicerolle: Eine Servicerolle ist eine IAM-Rolle, die ein Service übernimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.

  - Serviceverknüpfte Rolle – Eine serviceverknüpfte Rolle ist eine Art von Servicerolle, die mit einem verknüpft ist AWS-Service. Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Serviceverknüpfte Rollen werden in Ihrem angezeigt AWS-Konto und gehören dem Service. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.
- Anwendungen, die auf Amazon EC2 ausgeführt werden – Sie können eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer EC2-Instance ausgeführt werden und - AWS CLI oder AWS -API-Anforderungen stellen. Das ist eher zu empfehlen, als Zugriffsschlüssel innerhalb der EC2-Instance zu speichern. Um einer EC2-Instance eine - AWS Rolle zuzuweisen und sie für alle ihre Anwendungen verfügbar zu machen, erstellen Sie ein Instance-Profil, das an die Instance angehängt ist. Ein Instance-Profil enthält die Rolle und ermöglicht, dass Programme, die in der EC2-Instance ausgeführt werden, temporäre Anmeldeinformationen erhalten. Weitere Informationen finden Sie unter Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon EC2-Instances ausgeführt werden im IAM-Benutzerhandbuch.

Informationen dazu, wann Sie IAM-Rollen oder IAM-Benutzer verwenden sollten, finden Sie unter Erstellen einer IAM-Rolle (anstatt eines Benutzers) im IAM-Benutzerhandbuch.

## Verwalten des Zugriffs mit Richtlinien

Sie steuern den Zugriff in , AWS indem Sie Richtlinien erstellen und sie an AWS Identitäten oder Ressourcen anfügen. Eine Richtlinie ist ein Objekt in , AWS das, wenn es einer Identität oder Ressource zugeordnet wird, deren Berechtigungen definiert. AWS wertet diese Richtlinien aus, wenn ein Prinzipal (Benutzer, Root-Benutzer oder Rollensitzung) eine Anforderung stellt. Berechtigungen in den Richtlinien bestimmen, ob die Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden in AWS als JSON-Dokumente gespeichert. Weitere Informationen zu Struktur und Inhalten von JSON-Richtliniendokumenten finden Sie unter Übersicht über JSON-Richtlinien im IAM-Benutzerhandbuch.

Administratoren können AWS JSON-Richtlinien verwenden, um anzugeben, wer Zugriff auf was hat. Das bedeutet, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen

auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

IAM-Richtlinien definieren Berechtigungen für eine Aktion unabhängig von der Methode, die Sie zur Ausführung der Aktion verwenden. Angenommen, es gibt eine Richtlinie, die Berechtigungen für die `iam:GetRole`-Aktion erteilt. Ein Benutzer mit dieser Richtlinie kann Rolleninformationen von der AWS Management Console, der AWS CLI oder der AWS -API abrufen.

## Identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.

Identitätsbasierte Richtlinien können weiter als Inline-Richtlinien oder verwaltete Richtlinien kategorisiert werden. Inline-Richtlinien sind direkt in einen einzelnen Benutzer, eine einzelne Gruppe oder eine einzelne Rolle eingebettet. Verwaltete Richtlinien sind eigenständige Richtlinien, die Sie mehreren Benutzern, Gruppen und Rollen in Ihrem anfügen können AWS-Konto. Verwaltete Richtlinien umfassen - AWS verwaltete Richtlinien und vom Kunden verwaltete Richtlinien. Informationen dazu, wie Sie zwischen einer verwalteten Richtlinie und einer eingebundenen Richtlinie wählen, finden Sie unter Auswahl zwischen verwalteten und eingebundenen Richtlinien im IAM-Benutzerhandbuch.

## Ressourcenbasierte Richtlinien

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben. Prinzipale können Konten, Benutzer, Rollen, Verbundbenutzer oder umfassen AWS-Services.

Ressourcenbasierte Richtlinien sind Richtlinien innerhalb dieses Diensts. Sie können AWS verwaltete Richtlinien von IAM nicht in einer ressourcenbasierten Richtlinie verwenden.

## Zugriffssteuerungslisten (ACLs)

Zugriffssteuerungslisten (ACLs) steuern, welche Prinzipale (Kontomitglieder, Benutzer oder Rollen) auf eine Ressource zugreifen können. ACLs sind ähnlich wie ressourcenbasierte Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Amazon S3 und Amazon VPC sind Beispiele für Services AWS WAF, die ACLs unterstützen. Weitere Informationen zu ACLs finden Sie unter Zugriffssteuerungsliste (ACL) – Übersicht (Access Control List) im Amazon-Simple-Storage-Service-Entwicklerhandbuch.

## Weitere Richtlinientypen

AWS unterstützt zusätzliche, weniger häufig verwendete Richtlinientypen. Diese Richtlinientypen können die maximalen Berechtigungen festlegen, die Ihnen von den häufiger verwendeten Richtlinientypen erteilt werden können.

- Berechtigungsgrenzen – Eine Berechtigungsgrenze ist ein erweitertes Feature, mit der Sie die maximalen Berechtigungen festlegen können, die eine identitätsbasierte Richtlinie einer IAM-Entität (IAM-Benutzer oder -Rolle) erteilen kann. Sie können eine Berechtigungsgrenze für eine Entität festlegen. Die daraus resultierenden Berechtigungen sind der Schnittpunkt der identitätsbasierten Richtlinien einer Entität und ihrer Berechtigungsgrenzen. Ressourcenbasierte Richtlinien, die den Benutzer oder die Rolle im Feld `Principal` angeben, werden nicht durch Berechtigungsgrenzen eingeschränkt. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen über Berechtigungsgrenzen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten im IAM-Benutzerhandbuch.
- Service-Kontrollrichtlinien (SCPs )– SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OU) in angeben AWS Organizations. AWS Organizations ist ein Service zum Gruppieren und zentralen Verwalten mehrerer AWS-Konten , die Ihrem Unternehmen gehören. Wenn Sie innerhalb einer Organisation alle Features aktivieren, können Sie Service-Kontrollrichtlinien (SCPs) auf alle oder einzelne Ihrer Konten anwenden. Die SCP beschränkt Berechtigungen für Entitäten in Mitgliedskonten, einschließlich jeder Root-Benutzer des AWS-Kontos. Weitere Informationen zu Organizations und SCPs finden Sie unter Funktionsweise von SCPs im AWS Organizations -Benutzerhandbuch.
- Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen verbundenen Benutzer programmgesteuert erstellen. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Berechtigungen können auch aus einer ressourcenbasierten Richtlinie

stammen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen finden Sie unter Sitzungsrichtlinien im IAM-Benutzerhandbuch.

### Mehrere Richtlinientypen

Wenn mehrere auf eine Anforderung mehrere Richtlinientypen angewendet werden können, sind die entsprechenden Berechtigungen komplizierter. Wie AWS bestimmt, ob eine Anforderung zugelassen werden soll, wenn mehrere Richtlinientypen beteiligt sind, erfahren Sie unter Logik zur Richtlinienbewertung im IAM-Benutzerhandbuch.

## Wie AWS X-Ray funktioniert mit IAM

Bevor Sie IAM verwenden, um den Zugriff auf X-Ray zu verwalten, sollten Sie wissen, welche IAM-Funktionen für die Verwendung mit X-Ray verfügbar sind. Einen allgemeinen Überblick darüber, wie X-Ray und andere Produkte mit IAM AWS-Services funktionieren AWS-Services , finden Sie unter That Work with IAM im IAM-Benutzerhandbuch.

Sie können AWS Identity and Access Management (IAM) verwenden, um Benutzern X-Ray-Berechtigungen zu gewähren und Ressourcen in Ihrem Konto zu berechnen. IAM steuert den Zugriff auf den X-Ray-Dienst auf API-Ebene, um Berechtigungen einheitlich durchzusetzen, unabhängig davon, welchen Client (Konsole, AWS SDK AWS CLI) Ihre Benutzer verwenden.

Um die X-Ray-Konsole zum Anzeigen von Trace-Maps und Segmenten zu verwenden, benötigen Sie lediglich Leseberechtigungen. Um den Zugriff auf die Konsole zu ermöglichen, fügen Sie Ihrem IAM-Benutzer die `AWSXrayReadOnlyAccess` verwaltete Richtlinie hinzu.

Erstellen Sie für lokale Entwicklungen und Tests eine IAM-Rolle mit Lese- und Schreibberechtigungen. Übernehmen Sie die Rolle und speichern Sie temporäre Anmeldeinformationen für die Rolle. Sie können diese Anmeldeinformationen mit dem X-Ray-Daemon AWS CLI, dem und dem AWS SDK verwenden. Weitere Informationen finden Sie unter Verwenden temporärer Sicherheitsanmeldedaten mit dem AWS CLI.

Um Ihre instrumentierte App bereitzustellen AWS, erstellen Sie eine IAM-Rolle mit Schreibberechtigungen und weisen Sie sie den Ressourcen zu, auf denen Ihre Anwendung ausgeführt wird. `AWSXRayDaemonWriteAccess`beinhaltet die Erlaubnis, Traces hochzuladen, und einige Leseberechtigungen, um die Verwendung von Sampling-Regeln zu unterstützen. Weitere Informationen finden Sie unter Konfigurieren Sie Stichprobenregeln.

Die Lese- und Schreibrichtlinien enthalten keine Berechtigung zum Konfigurieren von Einstellungen für den Verschlüsselungsschlüssel und Samplingregeln. Verwenden Sie `AWSXrayFullAccess` für den Zugriff auf diese Einstellungen oder fügen Sie Konfigurations-APIs in einer benutzerdefinierten Richtlinie hinzu. Für Verschlüsselung und Entschlüsselung mit einem von Ihnen erstellten kundenverwalteten Schlüssel benötigen Sie auch die Berechtigung zur Verwendung des Schlüssels.

Themen

- Identitätsbasierte X-Ray-Richtlinien
- Ressourcenbasierte X-Ray-Richtlinien
- Autorisierung auf Basis von X-Ray-Tags
- Lokale Ausführung Ihrer Anwendung
- Ihre Anwendung wird ausgeführt in AWS
- Benutzerberechtigungen für die Verschlüsselung

## Identitätsbasierte X-Ray-Richtlinien

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen zugelassen oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter denen Aktionen zugelassen oder abgelehnt werden. X-Ray unterstützt bestimmte Aktionen, Ressourcen und Bedingungsschlüssel. Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der IAM-Referenz für JSON-Richtlinienelemente im IAM-Benutzerhandbuch.

Aktionen

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Action` einer JSON-Richtlinie beschreibt die Aktionen, mit denen Sie den Zugriff in einer Richtlinie zulassen oder verweigern können. Richtlinienaktionen haben normalerweise denselben Namen wie der zugehörige AWS API-Vorgang. Es gibt einige Ausnahmen, z. B. Aktionen, die nur mit Genehmigung durchgeführt werden können und für die es keinen passenden API-Vorgang gibt. Es gibt auch einige Operationen, die mehrere Aktionen in einer Richtlinie erfordern. Diese zusätzlichen Aktionen werden als abhängige Aktionen bezeichnet.

Schließen Sie Aktionen in eine Richtlinie ein, um Berechtigungen zur Durchführung der zugeordneten Operation zu erteilen.

Richtlinienaktionen in X-Ray verwenden das folgende Präfix vor der Aktion:`xray:`. Um beispielsweise jemandem die Erlaubnis zu erteilen, Gruppenressourcendetails mit dem `GetGroup` X-Ray-API-Vorgang abzurufen, nehmen Sie die `xray:GetGroup` Aktion in seine Richtlinie auf. Richtlinienanweisungen müssen entweder ein – `Action`oder ein `NotAction`-Element enthalten. X-Ray definiert eigene Aktionen, die Aufgaben beschreiben, die Sie mit diesem Dienst ausführen können.

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie wie folgt durch Kommata:

```
"Action": [
      "xray:action1",
      "xray:action2"
```

Sie können auch Platzhalter verwenden, um mehrere Aktionen anzugeben. Beispielsweise können Sie alle Aktionen festlegen, die mit dem Wort `Get` beginnen, einschließlich der folgenden Aktion:

```
"Action": "xray:Get*"
```

Eine Liste der X-Ray-Aktionen finden Sie unter Definierte Aktionen von AWS X-Ray im IAM-Benutzerhandbuch.

Ressourcen

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das bedeutet die Festlegung, welcher Prinzipal Aktionen für welche Ressourcen unter welchen Bedingungen ausführen kann.

Das JSON-Richtlinienelement `Resource` gibt die Objekte an, auf welche die Aktion angewendet wird. Anweisungen müssen entweder ein – `Resource`oder ein `NotResource`-Element enthalten. Als bewährte Methode geben Sie eine Ressource mit dem zugehörigen Amazon-Ressourcennamen (ARN) an. Sie können dies für Aktionen tun, die einen bestimmten Ressourcentyp unterstützen, der als Berechtigungen auf Ressourcenebene bezeichnet wird.

Verwenden Sie für Aktionen, die keine Berechtigungen auf Ressourcenebene unterstützen, z. B. Auflistungsoperationen, einen Platzhalter (*), um anzugeben, dass die Anweisung für alle Ressourcen gilt.

```
"Resource": "*"
```

Sie können den Zugriff auf Ressourcen mithilfe einer IAM-Richtlinie steuern. Für Aktionen, die Berechtigungen auf Ressourcenebene unterstützen, verwenden Sie einen Amazon-Ressourcennamen (ARN), um die Ressource zu identifizieren, für die die Richtlinie gilt.

Alle X-Ray-Aktionen können in einer IAM-Richtlinie verwendet werden, um Benutzern die Erlaubnis zur Verwendung dieser Aktion zu erteilen oder zu verweigern. Allerdings unterstützen nicht alle X-Ray-Aktionen Berechtigungen auf Ressourcenebene, mit denen Sie angeben können, für welche Ressourcen eine Aktion ausgeführt werden kann.

Für Aktionen, die Berechtigungen auf Ressourcenebene nicht unterstützen, muss „*“ als Ressource verwendet werden.

Die folgenden X-Ray-Aktionen unterstützen Berechtigungen auf Ressourcenebene:

- `CreateGroup`
- `GetGroup`
- `UpdateGroup`
- `DeleteGroup`
- `CreateSamplingRule`
- `UpdateSamplingRule`
- `DeleteSamplingRule`

Nachstehend finden Sie ein Beispiel für eine identitätsbasierte Berechtigungsrichtlinie für eine `CreateGroup`-Aktion: Das Beispiel zeigt die Verwendung eines ARN in Bezug auf den Gruppennamen `local-users` mit der eindeutigen ID als Platzhalter. Die eindeutige ID wird generiert, wenn die Gruppe erstellt wird, und kann daher in der Richtlinie nicht im Voraus vorhergesehen werden. Bei der Verwendung von `GetGroup`, `UpdateGroup` oder `DeleteGroup` können Sie diese entweder als Platzhalter oder als den genauen ARN definieren, einschließlich ID.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "xray:CreateGroup"
```

```
            ],
            "Resource": [
                "arn:aws:xray:eu-west-1:123456789012:group/local-users/*"
            ]
        }
    ]
}
```

Nachstehend finden Sie ein Beispiel für eine identitätsbasierte Berechtigungsrichtlinie für eine `CreateSamplingRule`-Aktion:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "xray:CreateSamplingRule"
            ],
            "Resource": [
                "arn:aws:xray:eu-west-1:123456789012:sampling-rule/base-scorekeep"
            ]
        }
    ]
}
```

> **Note**
>
> Der ARN einer Sampling-Regel definiert sich anhand ihres Namens. Im Gegensatz zu Gruppen-ARNs weisen Samplingregeln keine eindeutig generierte ID auf.

Eine Liste der X-Ray-Ressourcentypen und ihrer ARNs finden Sie unter Resources Defined by AWS X-Ray im IAM-Benutzerhandbuch. Informationen zu den Aktionen, mit denen Sie den ARN einzelner Ressourcen angeben können, finden Sie unter Von AWS X-Ray definierte Aktionen.

Bedingungsschlüssel

X-Ray stellt keine dienstspezifischen Bedingungsschlüssel bereit, unterstützt aber die Verwendung einiger globaler Bedingungsschlüssel. Eine Übersicht aller AWS globalen Bedingungsschlüssel finden Sie unter AWS Globale Bedingungskontextschlüssel im IAM-Benutzerhandbuch.

### Beispiele

Beispiele für identitätsbasierte X-Ray-Richtlinien finden Sie unter. AWS X-Ray Beispiele für identitätsbasierte Politik

## Ressourcenbasierte X-Ray-Richtlinien

X-Ray unterstützt ressourcenbasierte Richtlinien für die aktuelle und future AWS-Service Integration, wie Amazon SNS Active Tracing. Ressourcenbasierte X-Ray-Richtlinien können durch andere AWS Management Console s oder über das AWS SDK oder die CLI aktualisiert werden. Die Amazon SNS SNS-Konsole versucht beispielsweise, automatisch ressourcenbasierte Richtlinien für das Senden von Traces an X-Ray zu konfigurieren. Das folgende Richtliniendokument enthält ein Beispiel für die manuelle Konfiguration ressourcenbasierter X-Ray-Richtlinien.

Example Beispiel für eine ressourcenbasierte X-Ray-Richtlinie für Amazon SNS Active Tracing

Dieses Beispielrichtliniendokument spezifiziert die Berechtigungen, die Amazon SNS benötigt, um Trace-Daten an X-Ray zu senden:

```
{
    Version: "2012-10-17",
    Statement: [
      {
        Sid: "SNSAccess",
        Effect: Allow,
        Principal: {
          Service: "sns.amazonaws.com",
        },
        Action: [
          "xray:PutTraceSegments",
          "xray:GetSamplingRules",
          "xray:GetSamplingTargets"
        ],
        Resource: "*",
        Condition: {
          StringEquals: {
            "aws:SourceAccount": "account-id"
          },
          StringLike: {
            "aws:SourceArn": "arn:partition:sns:region:account-id:topic-name"
          }
        }
```

```
      }
    ]
  }
```

Verwenden Sie die CLI, um eine ressourcenbasierte Richtlinie zu erstellen, die Amazon SNS Berechtigungen zum Senden von Trace-Daten an X-Ray erteilt:

```
aws xray put-resource-policy --policy-name MyResourcePolicy --policy-document
 '{ "Version": "2012-10-17", "Statement": [ { "Sid": "SNSAccess", "Effect": "Allow",
 "Principal": { "Service": "sns.amazonaws.com" }, "Action": [ "xray:PutTraceSegments",
 "xray:GetSamplingRules", "xray:GetSamplingTargets" ], "Resource": "*",
 "Condition": { "StringEquals": { "aws:SourceAccount": "account-id" }, "StringLike":
 { "aws:SourceArn": "arn:partition:sns:region:account-id:topic-name" } } } ] }'
```

Um diese Beispiele zu verwenden, ersetzen Sie *partition*, *regionaccount-id*, und *topic-name*durch Ihre spezifische AWS Partition, Region, Konto-ID und Ihren Amazon SNS SNS-Themennamen. Um allen Amazon SNS SNS-Themen die Erlaubnis zu geben, Trace-Daten an X-Ray zu senden, ersetzen Sie den Themennamen durch. *

## Autorisierung auf Basis von X-Ray-Tags

Sie können Tags an X-Ray-Gruppen oder Sampling-Regeln anhängen oder Tags in einer Anfrage an X-Ray übergeben. Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an, indem Sie die Schlüssel `xray:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, oder Bedingung `aws:TagKeys` verwenden. Weitere Informationen zum Taggen von X-Ray-Ressourcen finden Sie unterKennzeichnen von Regeln und Gruppen für die Röntgenprobenahme.

Ein Beispiel für eine identitätsbasierte Richtlinie zur Einschränkung des Zugriffs auf eine Ressource auf der Grundlage der Markierungen dieser Ressource finden Sie unter Verwaltung des Zugriffs auf X-Ray-Gruppen und Stichprobenregeln auf der Grundlage von Tags.

## Lokale Ausführung Ihrer Anwendung

Ihre instrumentierte Anwendung sendet Trace-Daten an den X-Ray-Daemon. Der Daemon puffert segmentierte Dokumente und lädt sie stapelweise in den X-Ray-Dienst hoch. Der Daemon benötigt Schreibberechtigungen, um Trace-Daten und Telemetrie in den X-Ray-Dienst hochzuladen.

Wenn Sie den Daemon lokal ausführen, erstellen Sie eine IAM-Rolle, übernehmen Sie die Rolle und speichern Sie temporäre Anmeldeinformationen in Umgebungsvariablen oder in einer Datei,

die in einem Ordner benannt `credentials` ist, der `.aws` in Ihrem Benutzerordner heißt. Weitere Informationen finden Sie unter Verwenden temporärer Sicherheitsanmeldedaten mit dem AWS CLI.

Example ~/.aws/credentials

```
[default]
aws_access_key_id={access key ID}
aws_secret_access_key={access key}
aws_session_token={AWS session token}
```

Wenn Sie bereits Anmeldeinformationen für die Verwendung mit dem AWS SDK oder konfiguriert haben AWS CLI, kann der Daemon diese verwenden. Wenn mehrere Profile verfügbar sind, verwendet der Daemon das Standard-Profil.

### Ihre Anwendung wird ausgeführt in AWS

Wenn Sie Ihre Anwendung auf ausführen AWS, verwenden Sie eine Rolle, um der Amazon EC2 EC2-Instance oder Lambda-Funktion, die den Daemon ausführt, Berechtigungen zu erteilen.

- Amazon Elastic Compute Cloud (Amazon EC2) — Erstellen Sie eine IAM-Rolle und fügen Sie sie der EC2-Instance als Instance-Profil hinzu.
- Amazon Elastic Container Service (Amazon ECS) — Erstellen Sie eine IAM-Rolle und fügen Sie sie als Container-Instance-IAM-Rolle an Container-Instances an.
- AWS Elastic Beanstalk (Elastic Beanstalk) — Elastic Beanstalk enthält X-Ray-Berechtigungen in seinem Standard-Instanzprofil. Sie können das Standard-Instance-Profil verwenden oder einem benutzerdefinierten Instance-Profil Schreibberechtigungen hinzufügen.
- AWS Lambda (Lambda) — Fügen Sie der Ausführungsrolle Ihrer Funktion Schreibberechtigungen hinzu.

Um eine Rolle für die Verwendung mit X-Ray zu erstellen

1. Öffnen Sie die IAM-Konsole.
2. Wählen Sie Roles.
3. Klicken Sie auf Create New Role.
4. Geben Sie für Role Name (Name der Rolle) **`xray-application`** ein. Wählen Sie Next Step (Weiter) aus.
5. Wählen Sie für Rollentyp Amazon EC2.

6. Fügen Sie die folgende verwaltete Richtlinie hinzu, auf die Ihre Anwendung Zugriff hat AWS-Services:

   - AWSXRayDaemonWriteAccess— Erlaubt dem X-Ray-Daemon die Erlaubnis, Trace-Daten hochzuladen.

   Wenn Ihre Anwendung das AWS SDK für den Zugriff auf andere Dienste verwendet, fügen Sie Richtlinien hinzu, die den Zugriff auf diese Dienste gewähren.
7. Wählen Sie Next Step (Weiter) aus.
8. Wählen Sie Create Role (Rolle erstellen) aus.

## Benutzerberechtigungen für die Verschlüsselung

X-Ray verschlüsselt standardmäßig alle Trace-Daten, und Sie können es so konfigurieren, dass ein von Ihnen verwalteter Schlüssel verwendet wird. Wenn Sie sich für einen vom AWS Key Management Service Kunden verwalteten Schlüssel entscheiden, müssen Sie sicherstellen, dass die Zugriffsrichtlinie des Schlüssels es Ihnen ermöglicht, X-Ray die Erlaubnis zu erteilen, ihn zum Verschlüsseln zu verwenden. Andere Benutzer in Ihrem Konto benötigen ebenfalls Zugriff auf den Schlüssel, um verschlüsselte Trace-Daten in der X-Ray-Konsole anzeigen zu können.

Für einen vom Kunden verwalteten Schlüssel konfigurieren Sie Ihren Schlüssel mit einer Zugriffsrichtlinie, die die folgenden Aktionen ermöglicht:

- Der Benutzer, der den Schlüssel in X-Ray konfiguriert, hat die Berechtigung, `kms:CreateGrant` und `kms:DescribeKey` anzurufen.
- Benutzer, die auf verschlüsselte Ablaufverfolgungsdaten zugreifen können, besitzen die Berechtigung zum Aufruf von `kms:Decrypt`.

Wenn Sie der Gruppe Hauptbenutzer im Bereich Schlüsselkonfiguration der IAM-Konsole einen Benutzer hinzufügen, hat er die erforderlichen Rechte für beide Operationen. Die Berechtigungen müssen nur für die Schlüsselrichtlinie festgelegt werden, sodass Sie keine AWS KMS Berechtigungen für Ihre Benutzer, Gruppen oder Rollen benötigen. Weitere Informationen finden Sie unter Verwenden wichtiger Richtlinien im AWS KMS Entwicklerhandbuch.

Bei der Standardverschlüsselung oder wenn Sie das AWS verwaltete CMK (`aws/xray`) wählen, hängt die Berechtigung davon ab, wer Zugriff auf die X-Ray-APIs hat. Jeder Benutzer mit Zugriff auf `PutEncryptionConfig`, enthalten in `AWSXrayFullAccess`,

kann die Verschlüsselungskonfiguration ändern. Um zu verhindern, dass ein Benutzer den Verschlüsselungsschlüssel ändert, gewähren Sie ihm nicht die Berechtigung zur Verwendung von `PutEncryptionConfig`.

## AWS X-Ray Beispiele für identitätsbasierte Politik

Standardmäßig sind Benutzer und Rollen nicht berechtigt, X-Ray-Ressourcen zu erstellen oder zu ändern. Sie können auch keine Aufgaben mit der AWS Management Console AWS CLI, oder AWS API ausführen. Ein Administrator muss IAM-Richtlinien erstellen, die Benutzern und Rollen die Berechtigung zum Ausführen bestimmter API-Operationen für die angegebenen Ressourcen gewähren, die diese benötigen. Der Administrator muss diese Richtlinien anschließend den -Benutzern oder -Gruppen anfügen, die diese Berechtigungen benötigen.

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter Erstellen von Richtlinien auf der JSON-Registerkarte im IAM-Benutzerhandbuch.

Themen

- Bewährte Methoden für Richtlinien
- Verwenden der X-Ray-Konsole
- Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer
- Verwaltung des Zugriffs auf X-Ray-Gruppen und Stichprobenregeln auf der Grundlage von Tags
- Von IAM verwaltete Richtlinien für X-Ray
- X-Ray-Updates für AWS verwaltete Richtlinien
- Angabe einer Ressource innerhalb einer IAM-Richtlinie

### Bewährte Methoden für Richtlinien

Identitätsbasierte Richtlinien legen fest, ob jemand X-Ray-Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder sie löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen AWS-Konto. Befolgen Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Anleitungen und Empfehlungen:

- Beginnen Sie mit AWS verwalteten Richtlinien und wechseln Sie zu Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um damit zu beginnen, Ihren Benutzern und Workloads

Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter AWS -verwaltete Richtlinien oder AWS -verwaltete Richtlinien für Auftrags-Funktionen im IAM-Benutzerhandbuch.

- Anwendung von Berechtigungen mit den geringsten Rechten – Wenn Sie mit IAM-Richtlinien Berechtigungen festlegen, gewähren Sie nur die Berechtigungen, die für die Durchführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung von IAM zum Anwenden von Berechtigungen finden Sie unter Richtlinien und Berechtigungen in IAM im IAM-Benutzerhandbuch.
- Verwenden von Bedingungen in IAM-Richtlinien zur weiteren Einschränkung des Zugriffs – Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und Ressourcen zu beschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Serviceaktionen zu gewähren, wenn diese für einen bestimmten Zweck verwendet werden AWS-Service, z. AWS CloudFormation B. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingung im IAM-Benutzerhandbuch.
- Verwenden von IAM Access Analyzer zur Validierung Ihrer IAM-Richtlinien, um sichere und funktionale Berechtigungen zu gewährleisten – IAM Access Analyzer validiert neue und vorhandene Richtlinien, damit die Richtlinien der IAM-Richtliniensprache (JSON) und den bewährten IAM-Methoden entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie unter Richtlinienvalidierung zum IAM Access Analyzer im IAM-Benutzerhandbuch.
- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem System erfordert AWS-Konto, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren Richtlinien MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter Konfigurieren eines MFA-geschützten API-Zugriffs im IAM-Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden in IAM finden Sie unter Bewährte Methoden für die Sicherheit in IAM im IAM-Benutzerhandbuch.

## Verwenden der X-Ray-Konsole

Um auf die AWS X-Ray Konsole zugreifen zu können, benötigen Sie ein Mindestmaß an Berechtigungen. Diese Berechtigungen müssen es Ihnen ermöglichen, Details zu den X-Ray-Ressourcen in Ihrem aufzulisten und anzuzeigen AWS-Konto. Wenn Sie eine identitätsbasierte Richtlinie erstellen, die strenger ist als die mindestens erforderlichen Berechtigungen, funktioniert die Konsole nicht wie vorgesehen für Entitäten (Benutzer oder Rollen) mit dieser Richtlinie.

Um sicherzustellen, dass diese Entitäten weiterhin die X-Ray-Konsole verwenden können, hängen Sie die `AWSXRayReadOnlyAccess` AWS verwaltete Richtlinie an die Entitäten an. Diese Richtlinie wird unter IAM-verwaltete Richtlinien für X-Ray ausführlicher beschrieben. Weitere Informationen finden Sie unter Hinzufügen von Berechtigungen zu einem Benutzer im IAM-Benutzerhandbuch.

Sie müssen Benutzern, die nur die API AWS CLI oder die AWS API aufrufen, keine Mindestberechtigungen für die Konsole gewähren. Stattdessen sollten Sie nur Zugriff auf die Aktionen zulassen, die den API-Operation entsprechen, die Sie ausführen möchten.

## Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie umfasst Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der API AWS CLI oder AWS .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
```

```
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

## Verwaltung des Zugriffs auf X-Ray-Gruppen und Stichprobenregeln auf der Grundlage von Tags

Sie können Bedingungen in Ihrer identitätsbasierten Richtlinie verwenden, um den Zugriff auf X-Ray-Gruppen und Stichprobenregeln auf der Grundlage von Stichwörtern zu steuern. Die folgende Beispielrichtlinie könnte verwendet werden, um einer Benutzerrolle die Berechtigungen zum Erstellen, Löschen oder Aktualisieren von Gruppen mit den Tags `stage:prod` oder zu verweigern. `stage:preprod` Weitere Informationen zum Markieren von Regeln und Gruppen für die Röntgenabtastung finden Sie unterKennzeichnen von Regeln und Gruppen für die Röntgenprobenahme.

Um einem Benutzer den Zugriff auf das Erstellen, Aktualisieren oder Löschen einer Gruppe mit einem Tag `stage:prod` zu verweigern`stage:preprod`, weisen Sie dem Benutzer eine Rolle mit einer Richtlinie zu, die der folgenden ähnelt.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllXRay",
            "Effect": "Allow",
            "Action": "xray:*",
            "Resource": "*"
        },
        {
            "Sid": "DenyCreateGroupWithStage",
```

```
            "Effect": "Deny",
            "Action": [
                "xray:CreateGroup"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/stage": [
                        "preprod",
                        "prod"
                    ]
                }
            }
        },
        {
            "Sid": "DenyUpdateGroupWithStage",
            "Effect": "Deny",
            "Action": [
                "xray:UpdateGroup",
                "xray:DeleteGroup"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/stage": [
                        "preprod",
                        "prod"
                    ]
                }
            }
        }
    ]
}
```

Um die Erstellung einer Stichprobenregel `aws:RequestTag` zu verweigern, geben Sie damit Tags an, die nicht als Teil einer Erstellungsanfrage übergeben werden können. Um die Aktualisierung oder Löschung einer Stichprobenregel `aws:ResourceTag` zu verweigern, verwenden Sie „Ablehnen“ von Aktionen, die auf den Tags dieser Ressourcen basieren.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "AllowAllXRay",
            "Effect": "Allow",
            "Action": "xray:*",
            "Resource": "*"
        },
        {
            "Sid": "DenyCreateSamplingRuleWithStage",
            "Effect": "Deny",
            "Action": "xray:CreateSamplingRule",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/stage": [
                        "preprod",
                        "prod"
                    ]
                }
            }
        },
        {
            "Sid": "DenyUpdateSamplingRuleWithStage",
            "Effect": "Deny",
            "Action": [
                "xray:UpdateSamplingRule",
                "xray:DeleteSamplingRule"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/stage": [
                        "preprod",
                        "prod"
                    ]
                }
            }
        }
    ]
}
```

Sie können diese Richtlinien den Benutzern in Ihrem Konto zuordnen (oder sie zu einer einzigen Richtlinie zusammenfassen und dann die Richtlinie anhängen). Damit der Benutzer Änderungen an einer Gruppen- oder Stichprobenregel vornehmen kann, darf die Gruppen- oder Stichprobenregel nicht mit `stage=prepod` oder gekennzeichnet sein`stage=prod`. Der Tag-Schlüssel `Stage`

der Bedingung stimmt sowohl mit `Stage` als auch mit `stage` überein, da die Namen von Bedingungsschlüsseln nicht zwischen Groß- und Kleinschreibung unterscheiden. Weitere Informationen zum Bedingungsblock finden Sie unter IAM JSON Policy Elements: Condition im IAM-Benutzerhandbuch.

Ein Benutzer mit einer Rolle, der die folgende Richtlinie zugewiesen ist, kann das Tag nicht `role:admin` zu Ressourcen hinzufügen und keine Tags aus einer Ressource entfernen, die `role:admin` damit verknüpft ist.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllXRay",
            "Effect": "Allow",
            "Action": "xray:*",
            "Resource": "*"
        },
        {
            "Sid": "DenyRequestTagAdmin",
            "Effect": "Deny",
            "Action": "xray:TagResource",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/role": "admin"
                }
            }
        },
        {
            "Sid": "DenyResourceTagAdmin",
            "Effect": "Deny",
            "Action": "xray:UntagResource",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/role": "admin"
                }
            }
        }
    ]
}
```

## Von IAM verwaltete Richtlinien für X-Ray

Um die Erteilung von Berechtigungen zu vereinfachen, unterstützt IAM verwaltete Richtlinien für jeden Dienst. Ein Service kann diese verwalteten Richtlinien mit neuen Berechtigungen aktualisieren, wenn er neue APIs veröffentlicht. AWS X-Ray stellt verwaltete Richtlinien für nur Lese- und Schreibzugriff sowie für Administratoranwendungen bereit.

- `AWSXrayReadOnlyAccess`— Leseberechtigungen für die Verwendung der X-Ray-Konsole oder des AWS SDK zum Abrufen von Trace-Daten, Trace-Maps, Erkenntnissen und der X-Ray-Konfiguration von der X-Ray-API. AWS CLI Beinhaltet Observability Access Manager (OAM) `oam:ListSinks` und `oam:ListAttachedSinks` Berechtigungen, die es der Konsole ermöglichen, im Rahmen der CloudWatchkontenübergreifenden Observability geteilte Traces von Quellkonten einzusehen. Die Aktionen `BatchGetTraceSummaryById` und die `GetDistinctTraceGraphs` API sind nicht dafür vorgesehen, von Ihrem Code aufgerufen zu werden, und sie sind auch nicht in den SDKs und enthalten. AWS CLI AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "xray:GetSamplingRules",
                "xray:GetSamplingTargets",
                "xray:GetSamplingStatisticSummaries",
                "xray:BatchGetTraces",
                "xray:BatchGetTraceSummaryById",
                "xray:GetDistinctTraceGraphs",
                "xray:GetServiceGraph",
                "xray:GetTraceGraph",
                "xray:GetTraceSummaries",
                "xray:GetGroups",
                "xray:GetGroup",
                "xray:ListTagsForResource",
                "xray:ListResourcePolicies",
                "xray:GetTimeSeriesServiceStatistics",
                "xray:GetInsightSummaries",
                "xray:GetInsight",
                "xray:GetInsightEvents",
                "xray:GetInsightImpactGraph",
                "oam:ListSinks"
```

```
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "oam:ListAttachedLinks"
            ],
            "Resource": "arn:aws:oam:*:*:sink/*"
        }

}
```

- `AWSXRayDaemonWriteAccess`— Schreibberechtigungen für die Verwendung des X-Ray-Daemons oder AWS SDK zum Hochladen von Segmentdokumenten und Telemetrie in die X-Ray-API. AWS CLI Umfasst Leseberechtigungen zum Abrufen von Samplingregeln und zur Berichterstellung zu Samplingergebnissen. Weitere Informationen finden Sie unter Konfigurieren Sie Stichprobenregeln.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "xray:PutTraceSegments",
                "xray:PutTelemetryRecords",
                "xray:GetSamplingRules",
                "xray:GetSamplingTargets",
                "xray:GetSamplingStatisticSummaries"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

- `AWSXrayCrossAccountSharingConfiguration`— Erteilt Berechtigungen zum Erstellen, Verwalten und Anzeigen von Observability Access Manager-Links für die gemeinsame

Nutzung von X-Ray-Ressourcen zwischen Konten. Wird verwendet, um die CloudWatch kontenübergreifende Observability zwischen Quell- und Monitoring-Konten zu ermöglichen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "xray:Link",
                "oam:ListLinks"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "oam:DeleteLink",
                "oam:GetLink",
                "oam:TagResource"
            ],
            "Resource": "arn:aws:oam:*:*:link/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "oam:CreateLink",
                "oam:UpdateLink"
            ],
            "Resource": [
                "arn:aws:oam:*:*:link/*",
                "arn:aws:oam:*:*:sink/*"
            ]
        }
    ]

}
```

- `AWSXrayFullAccess`— Erlaubnis zur Verwendung aller X-Ray-APIs, einschließlich Lese- und Schreibberechtigungen sowie der Erlaubnis, Verschlüsselungsschlüsseleinstellungen und Sampling-Regeln zu konfigurieren. Beinhaltet Observability Access Manager (OAM) `oam:ListSinks` und `oam:ListAttachedSinks` Berechtigungen, die es der Konsole

ermöglichen, im Rahmen der CloudWatchkontenübergreifenden Observability geteilte Traces von Quellkonten einzusehen.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "xray:*",
                "oam:ListSinks"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "oam:ListAttachedLinks"
            ],
            "Resource": "arn:aws:oam:*:*:sink/*"
        }
    ]
}
```

So fügen Sie einem IAM-Benutzer, einer Gruppe oder einer Rolle eine veraltete Richtlinie hinzu:

1. Öffnen Sie die IAM-Konsole.
2. Öffnen Sie die Rolle, die Ihrem Instance-Profil, einem IAM-Benutzer oder einer IAM-Gruppe zugewiesen ist.
3. Fügen Sie unter Berechtigungen die verwaltete Richtlinie an.

## X-Ray-Updates für AWS verwaltete Richtlinien

Sehen Sie sich Details zu Aktualisierungen der AWS verwalteten Richtlinien für X-Ray an, seit dieser Dienst begonnen hat, diese Änderungen zu verfolgen. Um automatische Benachrichtigungen über Änderungen an dieser Seite zu erhalten, abonnieren Sie den RSS-Feed auf der X-Ray-Dokument-Verlaufsseite.

| Änderung | Beschreibung | Datum |
| --- | --- | --- |
| IAM-verwaltete Richtlinien für X-Ray — Neue `AWSXrayCrossAccountSharingConfiguration` `AWSXrayReadOnlyAccess` und aktualisierte `AWSXrayFullAccess` Richtlinien hinzugefügt. | X-Ray fügte Observability Access Manager (OAM) - Berechtigungen `oam:ListSinks` und `oam:ListAttachedSinks` zu diesen Richtlinien hinzu, sodass die Konsole im Rahmen der CloudWatch kontoüber greifenden Observability von Quellkonten geteilte Traces anzeigen kann. | 27. November 2022 |
| IAM-verwaltete Richtlinien für X-Ray — Aktualisierung der `AWSXrayReadOnlyAccess` Richtlinie. | X-Ray hat eine API-Aktion hinzugefügt,`ListResourcePolicies` . | 15. November 2022 |
| Verwenden der X-Ray-Konsole — Aktualisierung der `AWSXrayReadOnlyAccess` Richtlinie | X-Ray hat zwei neue API-Aktionen hinzugefügt, `BatchGetTraceSummaryById` und`GetDistinctTraceGraphs` .<br><br>Diese Aktionen sind nicht dafür vorgesehen, von Ihrem Code aufgerufen zu werden. Daher sind diese API-Aktionen nicht in den AWS SDKs AWS CLI und enthalten. | 11. November 2022 |

## Angabe einer Ressource innerhalb einer IAM-Richtlinie

Sie können den Zugriff auf Ressourcen mithilfe einer IAM-Richtlinie steuern. Für Aktionen, die Berechtigungen auf Ressourcenebene unterstützen, verwenden Sie einen Amazon-Ressourcennamen (ARN), um die Ressource zu identifizieren, für die die Richtlinie gilt.

Alle X-Ray-Aktionen können in einer IAM-Richtlinie verwendet werden, um Benutzern die Erlaubnis zur Verwendung dieser Aktion zu erteilen oder zu verweigern. Allerdings unterstützen nicht alle X-Ray-Aktionen Berechtigungen auf Ressourcenebene, mit denen Sie angeben können, für welche Ressourcen eine Aktion ausgeführt werden kann.

Für Aktionen, die Berechtigungen auf Ressourcenebene nicht unterstützen, muss „*“ als Ressource verwendet werden.

Die folgenden X-Ray-Aktionen unterstützen Berechtigungen auf Ressourcenebene:

- `CreateGroup`
- `GetGroup`
- `UpdateGroup`
- `DeleteGroup`
- `CreateSamplingRule`
- `UpdateSamplingRule`
- `DeleteSamplingRule`

Nachstehend finden Sie ein Beispiel für eine identitätsbasierte Berechtigungsrichtlinie für eine `CreateGroup`-Aktion: Das Beispiel zeigt die Verwendung eines ARN in Bezug auf den Gruppennamen `local-users` mit der eindeutigen ID als Platzhalter. Die eindeutige ID wird generiert, wenn die Gruppe erstellt wird, und kann daher in der Richtlinie nicht im Voraus vorhergesehen werden. Bei der Verwendung von `GetGroup`, `UpdateGroup` oder `DeleteGroup` können Sie diese entweder als Platzhalter oder als den genauen ARN definieren, einschließlich ID.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "xray:CreateGroup"
            ],
            "Resource": [
                "arn:aws:xray:eu-west-1:123456789012:group/local-users/*"
            ]
        }
    ]
```

```
}
```

Nachstehend finden Sie ein Beispiel für eine identitätsbasierte Berechtigungsrichtlinie für eine `CreateSamplingRule`-Aktion:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "xray:CreateSamplingRule"
            ],
            "Resource": [
                "arn:aws:xray:eu-west-1:123456789012:sampling-rule/base-scorekeep"
            ]
        }
    ]
}
```

> **Note**
>
> Der ARN einer Sampling-Regel definiert sich anhand ihres Namens. Im Gegensatz zu Gruppen-ARNs weisen Samplingregeln keine eindeutig generierte ID auf.

## Fehlerbehebung für AWS X-Ray-Identität und -Zugriff

Verwenden Sie die folgenden Informationen, um häufig auftretende Probleme zu diagnostizieren und zu beheben, die bei der Arbeit mit X-Ray und IAM auftreten können.

Themen

- Ich bin nicht berechtigt, eine Aktion in X-Ray durchzuführen
- Ich bin nicht berechtigt, iam durchzuführen:PassRole
- Ich bin Administrator und möchte anderen den Zugriff auf X-Ray ermöglichen
- Ich möchte Leute außerhalb meinesAWS-Kontoum auf meine X-Ray-Ressourcen zuzugreifen

## Ich bin nicht berechtigt, eine Aktion in X-Ray durchzuführen

Wenn die AWS Management Console Ihnen mitteilt, dass Sie nicht zur Ausführung einer Aktion autorisiert sind, müssen Sie sich an Ihren Administrator wenden, um Unterstützung zu erhalten. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Der folgende Beispielfehler tritt auf, wenn`mateojackson`Der Benutzer versucht, die Konsole zu verwenden, um Details zu einer Stichprobenregel anzuzeigen, hat aber keine`xray:GetSamplingRules`Berechtigungen.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to
 perform: xray:GetSamplingRules on resource: arn:${Partition}:xray:${Region}:
${Account}:sampling-rule/${SamplingRuleName}
```

In diesem Fall bittet Mateo seinen Administrator, seine Richtlinien zu aktualisieren, um ihm den Zugriff auf die Sampling-Regel-Ressource mit der `xray:GetSamplingRules`-Aktion zu ermöglichen.

## Ich bin nicht berechtigt, iam durchzuführen:PassRole

Wenn Sie eine Fehlermeldung erhalten, dass Sie nicht berechtigt sind, Folgendes auszuführen`iam:PassRole`Bei dieser Aktion müssen Ihre Richtlinien aktualisiert werden, damit Sie eine Rolle an X-Ray übergeben können.

Einige AWS-Services erlauben die Übergabe einer vorhandenen Rolle an diesen Service, sodass keine neue Servicerolle oder serviceverknüpfte Rolle erstellt werden muss. Hierzu benötigen Sie Berechtigungen für die Übergabe der Rolle an den Service.

Der folgende Beispielfehler tritt auf, wenn ein IAM-Benutzer mit dem Namen`marymajor`versucht, die Konsole zu verwenden, um eine Aktion in X-Ray auszuführen. Die Aktion erfordert jedoch, dass der Service über Berechtigungen verfügt, die durch eine Servicerolle gewährt werden. Mary besitzt keine Berechtigungen für die Übergabe der Rolle an den Service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In diesem Fall müssen die Richtlinien von Mary aktualisiert werden, um die Aktion `iam:PassRole` ausführen zu können.

Wenden Sie sich an Ihren AWS-Administrator, falls Sie weitere Unterstützung benötigen. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

## Ich bin Administrator und möchte anderen den Zugriff auf X-Ray ermöglichen

Um anderen den Zugriff auf X-Ray zu ermöglichen, müssen Sie eine IAM-Entität (Benutzer oder Rolle) für die Person oder Anwendung erstellen, die Zugriff benötigt. Sie werden die Anmeldeinformationen für diese Einrichtung verwenden, um auf AWS zuzugreifen. Anschließend müssen Sie der Entität eine Richtlinie hinzufügen, die ihnen die richtigen Berechtigungen in X-Ray gewährt.

Informationen zum Einstieg finden Sie unter Erstellen Ihrer ersten delegierten IAM-Benutzer und -Gruppen im IAM-Benutzerhandbuch.

## Ich möchte Leute außerhalb meinesAWS-Kontoum auf meine X-Ray-Ressourcen zuzugreifen

Sie können eine Rolle erstellen, die Benutzer in anderen Konten oder Personen außerhalb Ihrer Organisation für den Zugriff auf Ihre Ressourcen verwenden können. Sie können festlegen, wem die Übernahme der Rolle anvertraut wird. Im Fall von Services, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (Access Control Lists, ACLs) verwenden, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen darüber, ob X-Ray diese Funktionen unterstützt, finden Sie unterWie AWS X-Ray funktioniert mit IAM.
- Informationen zum Gewähren des Zugriffs auf Ihre Ressourcen für alle Ihre AWS-Konten finden Sie unter Gewähren des Zugriffs für einen IAM-Benutzer in einem anderen Ihrer AWS-Konto im IAM-Benutzerhandbuch.
- Informationen dazu, wie Sie AWS-Konten-Drittanbieter Zugriff auf Ihre Ressourcen bereitstellen, finden Sie unter Gewähren des Zugriffs auf AWS-Konten von externen Benutzern im IAM-Benutzerhandbuch.
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter Gewähren von Zugriff für extern authentifizierte Benutzer (Identitätsverbund) im IAM-Benutzerhandbuch.

- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontenübergreifenden Zugriff finden Sie unter So unterscheiden sich IAM-Rollen von ressourcenbasierten Richtlinien im IAM-Benutzerhandbuch.

# Protokollieren und Überwachen in AWS X-Ray

Die Überwachung ist ein wichtiger Teil der Aufrechterhaltung von Zuverlässigkeit, Verfügbarkeit und Performance Ihrer AWS-Lösungen. Sie sollten Überwachungsdaten aller Bestandteile IhrerAWS-Lösung, damit Sie Ausfälle, die sich über mehrere Punkte erstrecken, leichter debuggen können.AWSstellt mehrere Tools für die Überwachung Ihrer X-Ray-Ressourcen und zur Reaktion auf potenzielle Vorfälle bereit:

AWS CloudTrail-Protokolle

AWS X-Rayist in integriert.AWS CloudTrailum -API-Aktionen eines Benutzers, einer Rolle oder einesAWS-Dienst in X-Ray. Sie können mit CloudTrail X-Ray-API-Anfragen in Echtzeit überwachen und Protokolle in Amazon S3, Amazon CloudWatch Logs und Amazon CloudWatch Events speichern. Weitere Informationen finden Sie unter Protokollieren von X-Ray-API-Aufrufen mit AWS CloudTrail .

AWS Config-Nachverfolgung

AWS X-Rayist in integriert.AWS Configum Konfigurationsänderungen an Ihren X-Ray-Verschlüsselungsressourcen aufzuzeichnen. Sie könnenAWS Configum X-Ray-Verschlüsselungsressourcen durchzuführen, den X-Ray-Konfigurationsverlauf zu überprüfen und Benachrichtigungen basierend auf Ressourcenänderungen zu senden. Weitere Informationen finden Sie unter Verfolgung von Konfigurationsänderungen der X-Ray-Verschlüsselung mitAWS Config .

Überwachung von Amazon CloudWatch

Sie können das X-Ray-SDK SDK for Java verwenden, um unbeschwerte Amazon CloudWatch CloudWatch-Metriken aus Ihren gesammelten X-Ray-Segmenten zu veröffentlichen. Diese Metriken werden von der Start- und Endzeit des Segments sowie den Status-Flags für Fehler, Ausfall und Ablehnung abgeleitet. Mit diesen Trace-Metriken können Sie Wiederholungen und Abhängigkeitsprobleme in Teilsegmenten anzeigen. Weitere Informationen finden Sie unter AWS X-Ray Metriken für das X-Ray SDK for Java.

# Konformitätsvalidierung für AWS X-Ray

Informationen darüber, ob AWS-Service ein AWS-Services in den Geltungsbereich bestimmter Compliance-Programme fällt, finden Sie unter Umfang nach Compliance-Programm AWS-Services unter . Wählen Sie dort das Compliance-Programm aus, an dem Sie interessiert sind. Allgemeine Informationen finden Sie unter AWS Compliance-Programme AWS .

Sie können Prüfberichte von Drittanbietern unter herunterladen AWS Artifact. Weitere Informationen finden Sie unter Berichte herunterladen unter .

Ihre Verantwortung für die Einhaltung der Vorschriften bei der Nutzung AWS-Services hängt von der Vertraulichkeit Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften ab. AWS stellt die folgenden Ressourcen zur Verfügung, die Sie bei der Einhaltung der Vorschriften unterstützen:

- Schnellstartanleitungen zu Sicherheit und Compliance — In diesen Bereitstellungsleitfäden werden architektonische Überlegungen erörtert und Schritte für die Implementierung von Basisumgebungen beschrieben AWS , bei denen Sicherheit und Compliance im Mittelpunkt stehen.
- Architecting for HIPAA Security and Compliance on Amazon Web Services — In diesem Whitepaper wird beschrieben, wie Unternehmen HIPAA-fähige Anwendungen erstellen AWS können.

  > **Note**
  >
  > AWS-Services Nicht alle sind HIPAA-fähig. Weitere Informationen finden Sie in der Referenz für HIPAA-berechtigte Services.

- AWS Compliance-Ressourcen — Diese Sammlung von Arbeitsmappen und Leitfäden gilt möglicherweise für Ihre Branche und Ihren Standort.
- AWS Leitfäden zur Einhaltung von Vorschriften für Kunden — Verstehen Sie das Modell der gemeinsamen Verantwortung aus dem Blickwinkel der Einhaltung von Vorschriften. In den Leitfäden werden die bewährten Verfahren zur Sicherung zusammengefasst AWS-Services und die Leitlinien den Sicherheitskontrollen in verschiedenen Frameworks (einschließlich des National Institute of Standards and Technology (NIST), des Payment Card Industry Security Standards Council (PCI) und der International Organization for Standardization (ISO)) zugeordnet.
- Evaluierung von Ressourcen anhand von Regeln im AWS Config Entwicklerhandbuch — Der AWS Config Service bewertet, wie gut Ihre Ressourcenkonfigurationen den internen Praktiken, Branchenrichtlinien und Vorschriften entsprechen.

- AWS Security Hub— Auf diese AWS-Service Weise erhalten Sie einen umfassenden Überblick über Ihren internen Sicherheitsstatus. AWS Security Hub verwendet Sicherheitskontrollen, um Ihre AWS -Ressourcen zu bewerten und Ihre Einhaltung von Sicherheitsstandards und bewährten Methoden zu überprüfen. Eine Liste der unterstützten Services und Kontrollen finden Sie in der Security-Hub-Steuerungsreferenz.
- Amazon GuardDuty — Dies AWS-Service erkennt potenzielle Bedrohungen für Ihre Workloads AWS-Konten, Container und Daten, indem es Ihre Umgebung auf verdächtige und böswillige Aktivitäten überwacht. GuardDuty kann Ihnen helfen, verschiedene Compliance-Anforderungen wie PCI DSS zu erfüllen, indem es die in bestimmten Compliance-Frameworks vorgeschriebenen Anforderungen zur Erkennung von Eindringlingen erfüllt.
- AWS Audit Manager— Auf diese AWS-Service Weise können Sie Ihre AWS Nutzung kontinuierlich überprüfen, um das Risikomanagement und die Einhaltung von Vorschriften und Industriestandards zu vereinfachen.

## Ausfallsicherheit in AWS X-Ray

Die globale AWS-Infrastruktur ist um AWS-Regionen und Availability Zones herum aufgebaut. AWS-Regionen bieten mehrere physisch getrennte und isolierte Availability Zones, die mit einem Netzwerk mit geringer Latenz, hohem Durchsatz und hoher Redundanz verbunden sind. Mithilfe von Availability Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch Failover zwischen Availability Zones ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind besser hoch verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

Weitere Informationen über AWS-Regionen und Availability Zones finden Sie unter Globale AWS-Infrastruktur.

## Sicherheit der Infrastruktur in AWS X-Ray

Als verwalteter Service ist AWS X-Ray durch die globalen Verfahren zur Gewährleistung der Netzwerksicherheit von AWS geschützt. Informationen zu AWS-Sicherheitsdiensten und wie AWS die Infrastruktur schützt, finden Sie unter AWSCloud-Sicherheit. Informationen zum Entwerfen Ihrer AWS-Umgebung anhand der bewährten Methoden für die Infrastruktursicherheit finden Sie unter Infrastrukturschutz im Security Pillar AWS Well-Architected Framework.

Du benutztAWSveröffentlichte API-Aufrufe für den Zugriff auf X-Ray über das Netzwerk. Kunden müssen Folgendes unterstützen:

- Transport Layer Security (TLS). Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Verschlüsselungs-Suiten mit Perfect Forward Secrecy (PFS) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Die meisten modernen Systemen wie Java 7 und höher unterstützen diese Modi.

Außerdem müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der einem IAM-Prinzipal zugeordnet ist. Alternativ können Sie mit AWS Security Token Service (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

# Verwenden von AWS X-Ray mit VPC-Endpunkten

Wenn Sie Amazon Virtual Private Cloud (Amazon VPC) zum Hosten IhresAWSRessourcen, Sie können eine private Verbindung zwischen Ihrer VPC und X-Ray herstellen. Dadurch können Ressourcen in Ihrer Amazon VPC mit dem X-Ray-Service kommunizieren, ohne das öffentliche Internet nutzen zu müssen.

Amazon VPC ist einAWS-Servicedas Sie zum Starten verwenden könnenAWSRessourcen in einem virtuellen Netzwerk, das Sie definieren. Mit einer VPC haben Sie die Kontrolle über Ihre Netzwerkeinstellungen, wie IP-Adressbereich, Subnetze, Routing-Tabellen und Netzwerk-Gateways. Um Ihre VPC mit X-Ray zu verbinden, definieren Sie eineSchnittstelle VPC-Endpunkt. Der Endpunkt bietet zuverlässige, skalierbare Konnektivität zu X-Ray, ohne dass ein Internet-Gateway, eine NAT-Instanz (Network Address Translation) oder eine VPN-Verbindung erforderlich ist. Weitere Informationen finden Sie unter Was ist Amazon VPC im Benutzerhandbuch zu Amazon VPC.

Schnittstelle, über die VPC-Endpunkte betrieben werdenAWS PrivateLink, einAWSTechnologie, die private Kommunikation ermöglicht zwischenAWS-Servicesdurch die Verwendung einer elastischen Netzwerkschnittstelle mit privaten IP-Adressen. Weitere Informationen finden Sie imNeu —AWS PrivateLinkzumAWS-ServicesBlogbeitrag undErste Schrittein derAmazon VPC-Benutzerhandbuch.

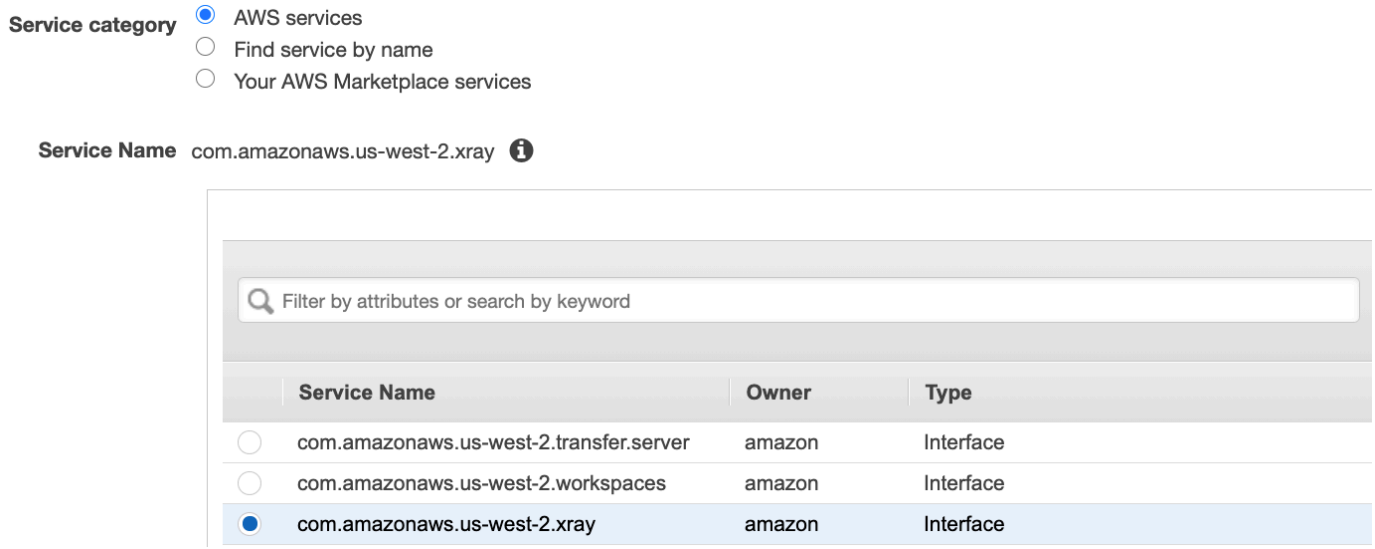
Um sicherzustellen, dass Sie in der von Ihnen ausgewählten Umgebung einen VPC-Endpunkt für X-Ray erstellen könnenAWS-Region, sieheUnterstützte Regionen.

## Einen VPC-Endpunkt für X-Ray erstellen

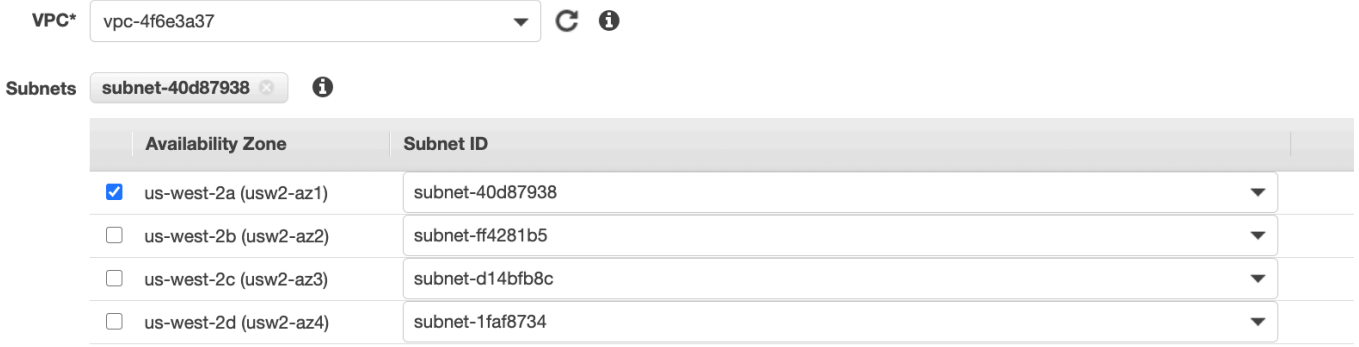
Um X-Ray mit Ihrer VPC zu verwenden, erstellen Sie einen VPC-Schnittstellen-Endpunkt für X-Ray.

1. Öffnen Sie die Amazon VPC-Konsole unter https://console.aws.amazon.com/vpc/.

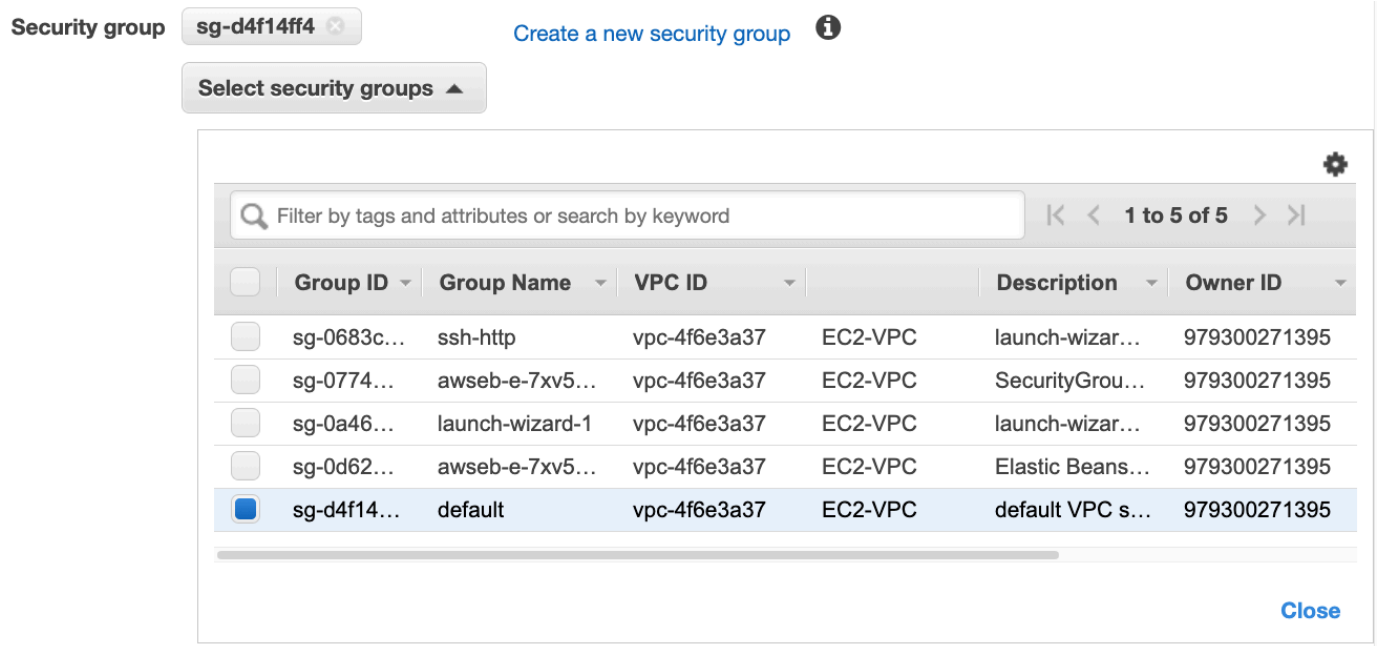
2. Navigieren Sie zuEndpunkteim Navigationsbereich und wählen SieEndpunkt erstellen.
3. Suchen Sie nach dem Namen desAWS X-RayDienst:`com.amazonaws.region.xray`.




4. Wählen Sie die gewünschte VPC und anschließend ein Subnetz in Ihrer VPC aus, um den Schnittstellenendpunkt zu verwenden. Im ausgewählten Subnetz wird eine Endpunkt-Netzwerkschnittstelle erstellt. Sie können mehrere Subnetze in unterschiedlichen Availability Zones angeben (wie vom Service unterstützt), um sicherzustellen, dass Ihr Schnittstellenendpunkt robust gegenüber Ausfällen von Availability Zone ist. Wenn Sie dies tun, wird in jedem von Ihnen angegebenen Subnetz eine Schnittstellen-Netzwerkschnittstelle erstellt.




5. (Optional) Private DNS ist standardmäßig für den Endpunkt aktiviert, sodass Sie Anfragen an X-Ray mit seinem Standard-DNS-Hostnamen stellen können. Sie können sich dafür entscheiden, es zu deaktivieren.
6. Geben Sie die Sicherheitsgruppen an, die der Endpunktnetzwerkschnittstelle zugeordnet werden sollen.




7. (Optional) Geben Sie eine benutzerdefinierte Richtlinie an, um die Zugriffsberechtigungen für den X-Ray-Dienst zu steuern. Standardmäßig ist Vollzugriff erlaubt.

## Steuern des Zugriffs auf Ihren X-Ray-VPC-Endpunkt

Eine VPC-Endpunktrichtlinie ist eine IAM-Ressourcenrichtlinie, die Sie einem Endpunkt beim Erstellen oder Ändern des Endpunkts zuordnen. Wenn Sie einem Endpunkt beim Erstellen keine Richtlinie zuordnen, ordnet Amazon VPC ihm eine Standardrichtlinie mit Vollzugriff auf den Service zu. IAM-Benutzerrichtlinien oder servicespezifische Richtlinien werden von einer Endpunktrichtlinie nicht überschrieben oder ersetzt. Endpunktrichtlinien steuern unabhängig vom Endpunkt den Zugriff auf den angegebenen Service. Endpunktrichtlinien müssen im JSON-Format erstellt werden. Weitere Informationen finden Sie unter Steuerung des Zugriffs auf Services mit VPC-Endpunkten im Amazon VPC User Guide.

Mit der VPC-Endpunktrichtlinie können Sie die Berechtigungen für verschiedene X-Ray-Aktionen kontrollieren. Sie können beispielsweise eine Richtlinie erstellen, die nur erlaubtPutTraceSegmentsund alle anderen Aktionen ablehnen. Dadurch werden Workloads und Dienste in der VPC darauf beschränkt, nur Trace-Daten an X-Ray zu senden und alle anderen Aktionen wie das Abrufen von Daten, das Ändern der Verschlüsselungskonfiguration oder das Erstellen/Aktualisieren von Gruppen zu verweigern.

Im Folgenden finden Sie ein Beispiel für eine Endpunktrichtlinie für X-Ray. Diese Richtlinie ermöglicht Benutzern, die über die VPC eine Verbindung zu X-Ray herstellen, Segmentdaten an X-Ray zu senden und verhindert außerdem, dass sie andere X-Ray-Aktionen ausführen.

```
{"Statement": [
    {"Sid": "Allow PutTraceSegments",
      "Principal": "*",
      "Action": [
        "xray:PutTraceSegments"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

Um die VPC-Endpunktrichtlinie für X-Ray zu bearbeiten

1. Öffnen Sie die Amazon VPC-Konsole unter https://console.aws.amazon.com/vpc/.
2. Wählen Sie im Navigationsbereich Endpunkte aus.
3. Wenn Sie den Endpunkt für X-Ray noch nicht erstellt haben, folgen Sie den Schritten unterEinen VPC-Endpunkt für X-Ray erstellen.
4. Wählen Sie diecom.amazonaws.***Region***.xrayEndpunkt, und wählen Sie dannRichtlinieRegisterkarte.
5. Klicken Sie auf Edit Policy (Richtlinie bearbeiten) und nehmen Sie Ihre Änderungen vor.

## Unterstützte Regionen

X-Ray unterstützt derzeit folgende VPC-EndpunkteAWS-Regionen:

- US East (Ohio)
- USA Ost (Nord-Virginia)
- USA West (Nordkalifornien)
- USA West (Oregon)
- Africa (Cape Town)
- Asia Pacific (Hong Kong)
- Asia Pacific (Mumbai)

- Asia Pacific (Osaka)
- Asia Pacific (Seoul)
- Asien-Pazifik (Singapur)
- Asien-Pazifik (Sydney)
- Asien-Pazifik (Tokio)
- Canada (Central)
- Europe (Frankfurt)
- Europa (Irland)
- Europe (London)
- Europa (Mailand)
- Europe (Paris)
- Europe (Stockholm)
- Middle East (Bahrain)
- Südamerika (São Paulo)
- AWS GovCloud(US-Ost)
- AWS GovCloud(US-West)

# Dokumentenverlauf für AWS X-Ray

In der folgenden Tabelle werden die wichtigen Änderungen an der Dokumentation für AWS X-Ray beschrieben. Um Benachrichtigungen über Aktualisierungen dieser Dokumentation zu erhalten, können Sie einen RSS-Feed abonnieren.

Letzte Aktualisierung der Dokumentation: 8. Februar 2023

| Änderung | Beschreibung | Datum |
| --- | --- | --- |
| Hinzugefügte Funktionalität | X-Ray protokolliert jetzt Datenereignisse `PutTraceSegments` , einschließlich `GetTraceSummaries` , und `BatchGetTraces` zu AWS CloudTrail. X-Ray protokolliert jetzt auch das `GetSamplingStatisticSummaries` Verwaltungsereignis unter CloudTrail. Weitere Informationen finden Sie unter Protokollieren von X-Ray-API-Aufrufen mit AWS CloudTrail. | 7. März 2024 |
| Hinzugefügte Funktionalität | X-Ray unterstützt jetzt Trace-IDs, die mit OpenTelemetry oder einem anderen Framework erstellt wurden, das der W3C Trace Context-Spezifikation entspricht. Weitere Informationen finden Sie unter Trace-Daten an X-Ray senden. | 25. Oktober 2023 |
| Hinzugefügte Funktionalität | Sie können jetzt das aktive Tracing von Amazon SNS | 8. Februar 2023 |

| | | |
|---|---|---|
| | konfigurieren, sodass Sie Anfragen verfolgen und analysieren können, während sie Ihre Amazon SNS SNS-Themen durchlaufen. Weitere Informationen finden Sie unter Amazon SNS und AWS X-Ray. | |
| Thema „X-Ray SDK für Node.js“ aktualisiert | Es wurden Details zur Instrumentierung von Clients hinzugefügt, die AWS SDK for JavaScript V3 verwenden . Einzelheiten finden Sie unter Nachverfolgen von AWS SDK-Aufrufen mit dem X-Ray-SDK für Node.js. | 07. Februar 2023 |
| Die Einzelheiten der von IAM verwalteten Richtlinie wurden aktualisiert | Den Richtlinien und verwaltet en Richtlinien wurde eine IAM-Berechtigung für kontoübergreifende Beobachtbarkeit hinzugefü gt. `AWSXRayReadOnlyAcc ess` `AWSXRayFullAccess` `AWSXrayCrossAccoun tSharingConfigurat ion` Einzelheiten finden Sie unter IAM-verwaltete Richtlini en für X-Ray. | 07. Februar 2023 |

| | | |
|---|---|---|
| Hinzugefügte Funktionalität | AWS X-Ray unterstützt jetzt kontenübergreifende Observability, sodass Sie Anwendungen überwachen und Fehler beheben können, die sich über mehrere Konten innerhalb eines AWS-Region Einzelheiten finden Sie unter Kontoübergreifende Ablaufver folgung. | 27. November 2022 |
| Hinzugefügte Funktionalität | Sie können jetzt verknüpfte Traces zwischen Nachricht enproduzenten, einer Amazon SQS SQS-Warteschlange und Verbrauchern anzeigen und so eine verbundene Ansicht der Traces bieten, die von ereignisgesteuerte n Anwendungen gesendet wurden. Weitere Informationen finden Sie unter Ablaufver folgung ereignisgesteuerter Anwendungen. | 20. November 2022 |
| Die Informationen zu den von IAM verwalteten Richtlinien wurden aktualisiert | Die IAM-Berechtigung zum Auflisten von Ressource nrichtlinien wurde zur `AWSXRayReadOnlyAcc ess` verwalteten Richtlini e hinzugefügt. Einzelheiten finden Sie unter IAM-verwa ltete Richtlinien für X-Ray. | 15. November 2022 |

| | | |
|---|---|---|
| Die Berechtigungen der IAM-Konsole und die Details der verwalteten Richtlinien wurden aktualisiert | Der Satz der IAM-Berechtigungen, die die X-Ray-Konsole verwendet, wurde zusammen mit der Beschreibung der `AWSXRayReadOnlyAccess` verwalteten Richtlinie aktualisiert. Einzelheiten finden Sie unter Verwenden der X-Ray-Konsole. | 11. November 2022 |
| AWS Distribution für OpenTelemetry Ruby hinzugefügt | AWS Distro for OpenTelemetry (ADOT) bietet einen einzigen Satz von Open-Source-APIs, -Bibliotheken und -Agenten zur Erfassung verteilter Traces und Metriken. ADOT Ruby ermöglicht es Ihnen, Ihre Ruby-Anwendung für X-Ray und andere Tracing-Backends zu instrumentieren. Weitere Informationen finden Sie unter AWS Distro for Ruby. OpenTelemetry | 7. Februar 2022 |
| Hinzugefügte Funktionalität | Sie können jetzt Traces anzeigen und X-Ray von der CloudWatch Konsole aus konfigurieren. Weitere Informationen finden Sie unter X-Ray-Konsole. | 24. Januar 2022 |

| | | |
|---|---|---|
| Integriertes CloudWatch RUM | Mit AWS X-Ray und CloudWatch RUM können Sie den Anforderungspfad analysieren und debuggen, angefangen bei den Endbenutzern Ihrer Anwendung bis hin zu nachgelagerten AWS Managed Services. Weitere Informationen finden Sie unter CloudWatch RUM und AWS X-Ray. | 3. Dezember 2021 |
| Integrierte AWS Distribution für OpenTelemetry | The AWS Distro for OpenTelemetry (ADOT) bietet einen einzigen Satz von Open-Source-APIs, -Bibliotheken und -Agenten zur Erfassung verteilter Traces und Metriken. ADOT ermöglicht es Ihnen, Ihre Anwendung für X-Ray und andere Tracing-Backends zu instrumentieren. Weitere Informationen finden Sie unter Instrumentierung Ihrer App. | 23. September 2021 |

| | | |
|---|---|---|
| Hinzugefügte Funktionalität | AWS X-Ray lässt sich jetzt in Amazon Virtual Private Cloud integrieren, sodass Ressourcen in Ihrer Amazon VPC mit dem X-Ray-Service kommunizieren können, ohne das öffentliche Internet nutzen zu müssen. Weitere Informationen finden Sie unter Verwenden AWS X-Ray mit VPC-Endpunkten. | 20. Mai 2021 |
| Hinzugefügte Funktionalität | AWS X-Ray lässt sich jetzt in integrieren AWS CloudFormation und ermöglicht Ihnen die Bereitstellung und Konfiguration von X-Ray-Ressourcen. Weitere Informationen finden Sie unter X-Ray-Ressourcen erstellen mit CloudFormation. | 6. Mai 2021 |
| Hinzugefügte Funktionalität | AWS X-Ray lässt sich jetzt in Amazon integrieren EventBridge , um Ereignisse nachzuverfolgen, die durchlaufen wurden EventBridge. Dies bietet Benutzern einen vollständigeren Überblick über ihr System. Weitere Informationen finden Sie unter Amazon EventBridge und AWS X-Ray. | 2. März 2021 |
| Daemon zu ECR hinzugefügt | Der Daemon kann jetzt von Amazon ECR heruntergeladen werden. Weitere Informationen finden Sie unter Den Daemon herunterladen. | 1. März 2021 |

| | | |
|---|---|---|
| Hinzugefügte Funktionalität | AWS X-Ray unterstützt jetzt Insights-bezogene Benachrichtigungen an Amazon EventBridge. Auf diese Weise können Sie mithilfe von Insights automatische Maßnahmen ergreifen EventBridge. Weitere Informationen finden Sie unter Aktivieren von Insights-Benachrichtigungen unter X-Ray Insights verwenden. | 15. Oktober 2020 |
| Herunterladbare Daemons hinzugefügt | AWS X-Ray führt einen Support-Daemon für Linux ARM64 ein. Weitere Informationen finden Sie unter daemonbrazil ws AWS X-Ray | 1. Oktober 2020 |
| Hinzugefügte Funktionalität | AWS X-Ray unterstützt jetzt die aktive Integration mit Amazon CloudWatch Synthetics. Auf diese Weise können Sie Details zu einem Synthetics Canary-Client-Knoten wie Antwortzeit und Status anzeigen. Sie können in der Analytics-Konsole auch Analysen auf der Grundlage von Informationen aus einem Canary-Clientknoten von Synthetics durchführen. Weitere Informationen finden Sie unter Debuggen CloudWatch synthetischer Kanarien mit X-Ray. | 24. September 2020 |

| | | |
|---|---|---|
| Hinzugefügte Funktionalität | AWS X-Ray unterstützt jetzt end-to-end Ablaufverfolgungs-Workflows für. AWS Step Functions Sie können die Komponenten Ihrer Zustandsmaschine visualisieren, Leistungsengpässe identifizieren und Anfragen, die zu einem Fehler geführt haben, beheben. Weitere Informationen finden Sie unter AWS Step Functions und AWS X-Ray. | 14. September 2020 |
| Hinzugefügte Funktionalität | AWS X-Ray bietet Einblicke zur kontinuierlichen Analyse von Trace-Daten in Ihrem Konto, um aufkommende Probleme in Ihren Anwendungen zu identifizieren. Insights zeichnet Vorfälle auf und verfolgt die Auswirkungen von Vorfällen bis zur Lösung. Weitere Informationen finden Sie unter Verwenden von X-Ray Insights. | 3. September 2020 |

| | | |
|---|---|---|
| Hinzugefügte Funktionalität | AWS X-Ray führt den Java-Agenten für automatische Instrumentierung ein, der es Kunden ermöglicht, Trace-Daten zu sammeln, ohne die bestehende Java-basierte Anwendung ändern zu müssen. Sie können jetzt Web- und Servlet-basierte Java-Anwendungen mit minimalen Konfigurationsänderungen und ohne Codeänderungen verfolgen. Weitere Informationen finden Sie unter AWS X-Ray Auto-Instrumentierungs-Agent für Java. | 3. September 2020 |
| Hinzugefügte Funktionalität | AWS X-Ray hat der X-Ray-Konsole eine neue Gruppenseite hinzugefügt, um die Erstellung und Verwaltung von Trace-Gruppen zu vereinfachen. Weitere Informationen finden Sie unter Gruppen in der X-Ray-Konsole konfigurieren. | 24. August 2020 |

| | | |
|---|---|---|
| Hinzugefügte Funktionalität | AWS X-Ray ermöglicht jetzt das Hinzufügen von Tags zu Gruppen und Sampling-Regeln. Sie können auch den Zugriff auf Gruppen und Stichprobenregeln auf der Grundlage von Stichwörtern steuern. Weitere Informationen finden Sie unter Kennzeichnen von Regeln und Gruppen für X-Ray und Verwaltung des Zugriffs auf Röntgengruppen und Stichprobenregeln auf der Grundlage von Tags. | 24. August 2020 |

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.